ON STABLE BEHAVIOR OF COOPERATING LOGIC PROGRAMS [1]

Michael I. Dekhtyar
Dept. of S, Tver St. Univ.
Tver, Russia, 170000

Alexander Ja. Diko sky
Keldysh Inst. for Appl. Math.
Moscow, Russia, 125047

Mars K. Valie
Keldysh Inst. for Appl. Math.
Moscow, Russia, 125047

ABSTRACT

Systems of cooperating logic programs are considered which generalize dynamic deductive databases (DDDBs) from [1,2]. The notion of stable behavior is generalized from DDDBs to systems of cooperating logic programs. Decision problems for some variants of stability of cooperating productional logic programs are investigated. It is shown that these problems are reducible to satisfiability problem for propositional temporal logic of branching time (and, in one case, to that for linear time). It allows to transfer upper bounds of complexity of stability problems for DDDBs to cooperating productional logic programs.

Keywords: cooperating logic programs, stability problem, propositional temporal logic, decision complexity.

1 Introduction

In this paper we consider some logical approach to mathematical analysis of the behavior of interactive discrete dynamic systems. A state of a dynamic system is represented by a data base state (*DB state*), i.e. a finite set of

[1]This work was sponsored by the Russian Fundamental Studies Foundation (Grants 95-01-01321, 96-01-00395, 97-01-00136).

facts. Behavior of the system is determined by actions of a set of logical programs which update DB states. These actions generate a set of possible trajectories of the system, i.e. sequences of DB states. Different requirements on the system behavior can be defined in terms of conditions which should be satisfied by the set of trajectories. Here we investigate only one of interesting kinds of behavior properties which can be expressed in such terms, namely, the *stability* property of the system. Moreover, we limit ourselves by consideration only the case when systems consist of two (in general, nondeterministic) logic programs, master LP_1 and slave LP_2, which work over a (finite dynamic) database $\mathcal{E}$, updating states of $\mathcal{E}$ in turn. This notion of cooperating (symbiotic) logic programs generalizes the notion of dynamic deductive databases with external updates from [1,2]. In terms of [1,2] the master corresponds to the internal program managing $\mathcal{E}$, and the slave corresponds to an external medium which can update $\mathcal{E}$.

The binary relation induced by the updates of the logic program LP on DB states we denote by $\vdash_{LP}$. Then local behavior of the system in the current state $\mathcal{E}_0$ is described as one interaction of LP_1 and LP_2 in this state, i.e. as the sequence of two updates $\mathcal{E}_0 \vdash_{LP_1} \mathcal{E}_1' \vdash_{LP_2} \mathcal{E}_1$. Normally, one should distinguish between acceptable and not acceptable interactions, depending on a criterion of admissibility of system states. Each acceptable interaction applies to an admissible state $\mathcal{E}_0$ and yields an admissible state $\mathcal{E}_1$. However, the intermediate state $\mathcal{E}_1'$ may in general be inadmissible, in which case the reaction of LP_2 compensates for the ruinous action of LP_1. We represent the admissibility criterion by some *integrity constraint (IC)* expressed by a formula Φ over DB states. In terms of the IC the acceptability of the interaction is expressed as follows: the interaction of the form above is *acceptable* if $\mathcal{E}_0 \models \Phi$ and $\mathcal{E}_1 \models \Phi$. Thus, the interactive discrete dynamic system is represented by the system $\mathcal{B} =< LP_1, LP_2, \Phi >$, and its local behavior is expressed in terms of acceptable interactions.

Global behavior of the system in current state $\mathcal{E}_0$ is represented by sequences of interactions starting in $\mathcal{E}_0$ which we call trajectories

$$\mathcal{E}_0 \vdash_{LP_1} \mathcal{E}_1' \vdash_{LP_2} \mathcal{E}_1 \vdash_{LP_1} \mathcal{E}_2' \vdash_{LP_2} \mathcal{E}_2 \ ...$$

The infinite acceptable trajectories represent the stable behavior of the system: any possible ruinous actions of the program LP_1 is compensated by some action of the program (medium) LP_2 along all the trajectory. Such trajectories are called *stable*.

Trajectories of the system $\mathcal{B}$ form a tree $T(\mathcal{E}_0)$ with the root $\mathcal{E}_0$. A

number of natural properties of interactive behavior of $\mathcal{B}$ in a given DB state can be formalized in terms of this tree, in particular, different kinds of stability.

Definition 1 *Let* $Q_1, Q_2 \in \{\forall, \exists\}$. $\mathcal{B}$ *is* Q_1Q_2-stable in DB state $\mathcal{E}_0$ *if in the tree* $T(\mathcal{E}_0)$ *there is a* Q_1Q_2-*subtree in which all branches are infinite stable trajectories.*

On the other side, another kind of smooth behavior of the system was also considered in [1,2] when the master program restores IC violated by the slave. This kind of behavior was called homeostaticity. One of natural questions connected with these notions is to consider algorithmic decidability of problems of stability and homeostaticity. Of course, in general case these problems are undecidable. In [1,2] some classes of logic programs were distinguished for which these problems are decidable. Moreover, rather exact estimates for complexity of these problems were established. Note, however, that in [1,2] it was assumed that the second relation $\vdash_{LP_2}$ which describes the external medium actions is much more simple than $\vdash_{LP_1}$ defined by the internal program LP_1 (roughly speaking, this relation was defined not by a real logic program, but by a rather simple set of possible updates). Here we consider more general case when LP_2 may be in the same class as LP_1. Here we present only results for the case when programs LP_1 and LP_2 belong to the class GPROD of ground productional logic programs with updates. We show that for cooperative systems of this class decision problems for stability and homeostaticity are decidable and have the same decision complexity as for corresponding problems in [1,2]. In fact, for cooperative systems considered here decision problems for homeostaticity and stability coincide since it becomes unimportant which of LP_1 or LP_2 is a master or a slave.

To prove our results we show that variants of stability problem which are considered here are reducible to satisfiability problem for a variant of propositional logic of branching time. Moreover, for $\exists\exists$ -stability this reduction is simultaneously a reduction to satisfiability problem for logic of linear time. As a corollary we obtain that results on polynomial space and exponential time complexity of stability and homeostaticity problem for GPROD from [1,2] are generalized to systems of cooperating programs from GPROD. Moreover, we note that some lower bounds in [1,2] can be

improved. Namely, results on EXPTIME-hardness of the stability problem can be complemented by $2^{n/logn}$ lower bound of time complexity.

2 Basic Notions and Definitions

2.1 Productional logic programs

We consider productional logic programs with updates in a signature Σ consisting of a set of constants C and a set of predicate symbols Pr. Let $\mathbf{H}$ denote the Herbrand base over Σ. A productional logic program consists of clauses of following form:

$$\pi : \ Con_1 \& ... \& Con_k \Rightarrow Act_1, ..., Act_m$$

Each Con_i (*elementary condition*) is either a ground atom in $\mathbf{H}$ or its negation. Each Act_j (*action*) is one of elementary updates $insert(A)$, $delete(A)$ where $A \in \mathbf{H}$. For simplicity we assume that there are no conflicts in application of actions, i.e. there are simultaneously no $Con_j = insert(A)$ and $Con_l = delete(A)$.

A data base (DB) state $\mathcal{E}$ is a finite subset of the Herbrand base $\mathbf{H}$. The production π is *applicable* to a DB state $\mathcal{E}$ iff for every $1 \leq i \leq k$ $Con_i \in \mathcal{E}$ if Con_i is a ground atom and $Con_i \notin \mathcal{E}$ if Con_i is a negated ground atom. The result $\pi(\mathcal{E})$ of application of π to $\mathcal{E}$ is a DB state $\mathcal{E}_1$ obtained from $\mathcal{E}$ by adding all atoms A such that there is $Con_j = insert(A)$ and deleting all atoms A such that there is $Con_j = delete(A)$. So, the production π defines an update relation $\vdash_\pi$ on the set of all DB states: $\mathcal{E} \vdash_\pi \mathcal{E}_1$ iff $\mathcal{E}_1 = \pi(\mathcal{E})$. The update relation $\vdash_{LP}$ induced by a productional logic program LP is defined as

$$\vdash_{LP} \ = \ \cup_{\pi \in LP} \vdash_\pi \ .$$

We consider as integrity constraints (ICs) quantifier-free first order formulas over Σ. We say that a DB state $\mathcal{E}$ *satisfies* an IC Φ iff $\mathcal{E} \models \Phi$.

2.2 Propositional Temporal Logic

We use the following variant of propositional logic of branching time (BPTL) (for similar variants see [3]). Its formulas are constructed from propositional variables by using the Boolean connectives and temporal operators

$\forall X, \forall Y, \forall F,\ \forall G$ (operators $\exists X, \exists Y, \exists G$ and $\exists F$ are expressed as $\neg\forall X\neg$, $\neg\forall Y\neg$, $\neg\forall F\neg$ and $\neg\forall G\neg$, respectively).

Models of BPTL have the form $< T, \pi >$ where T is an infinite tree with branches of the height ω , and π assigns to any node s of T a set of propositional variables satisfied on s (as usual we write $s \models p$ instead of $p \in \pi(s)$). The relation $\models$ is extended to all the formulas of BPTL in the following way:

semantics of boolean connectives is defined as usual;

$s \models \forall X\ p$ iff $s' \models p$ for all sons s' of s;

$s \models \forall Y\ p$ iff $s' \models p$ if s is not the root of T and s' is the predecessor of s (if s is the root we can assume $s \models \forall Y\ p$ for any formula p);

$s \models \forall F p$ iff any forward path beginning with s contains a node s' such that $s' \models p$;

$s \models \forall G p$ iff $s' \models p$ for all nodes s' of the forward paths beginning with s .

According to the definition above $\exists Y p$ means "there exists the predecessor s' of s such that $s \models p$ " (the meaning of other operators is also clear).

The given above version of semantics for BPTL supposes that time structure is branching forwards and linear backwards. Another variant of semantics for BTPL assumes that time is linear forwards, too.

3 Reduction of Stability to BPTL

Let LP_1 be the productional logic program

$f_{11} -> upd_{11}$

...

$f_{1n_1} -> upd_{1n_1}$,

and LP_2 be the productional logic program

$f_{21} -> upd_{21}$

...

$f_{2n_2} -> upd_{2n_2}$,

where f_{ij} are conditions, upd_{ij} are updates.

Any DB state E can be described statically as conjunction $Conj(E)$ of (positive ground) atoms occurring in E . But to reflect changes of states caused by actions of LP_1 and LP_2 we should in following take into account

also some negative atoms. So, with any DB state E and logic programs LP_1, LP_2 we connect the formula $s(E)$ which is conjunction of $Conj(E)$ and negations of ground atoms occurring in LP_1 or LP_2 but not in E .

For any update upd which inserts $a_1, ..., a_k$, deletes $b_1, ..., b_l$ and leaves invariant $c_1, ..., c_m$ we introduce a formula UPD with the meaning:
$s \models UPD$ iff the following is true: for any E the formula $s(E)$ is satisfied in s iff there exists a DB state E' such that E is obtained by applying upd to E' and $s(E')$ is satisfied in the state s' of T previous to s.
$UPD = \bigwedge_{i=1}^{m} (c_i \equiv \exists Y c_i) \wedge \bigwedge_{i=1}^{k} a_i \wedge \bigwedge_{i=1}^{l} \neg b_i$.

Let Q be a new propositional variable. Then the formula $EVEN$ of the form

$$Q \wedge \forall G\ (Q \rightarrow (\forall X\ \neg\ Q \wedge \forall X\ \forall X\ Q))$$

expresses the property " Q is true exactly on the states in even levels of T ".

Let $Safety = Safety\ (LP_1, IC)$ denote the formula
$EVEN \wedge \forall G\ (Q \rightarrow\ (IC \wedge \bigvee_{i=1}^{n_1} f_{1i}))$.
It is obvious that if this formula is satisfied on the root of T then in all states at even levels of T integrity constraint IC is satisfied and at least one of rules of LP_1 is applicable.

Then the following assertions prove reducibility of stability problems for cooperating logic programs to satisfiability problem for BTPL:

$(\exists\exists):\ \ < LP_1, LP_2, IC >$ is $\exists\exists$-stable in E
iff the formula
$s(E)\ \wedge\ Safety\ \wedge$
$\forall G\ (Q \rightarrow (\ \bigvee_{i=1}^{n_1}\{f_{1i} \wedge \exists X[UPD_{1i} \wedge \bigvee_{j=1}^{n_2}(f_{2j} \wedge \exists X.UPD_{2j})]\}))$
is satisfiable.
Remark. For this formula linear and branching time satisfiability coincide.

$(\forall\exists):\ \ < LP_1, LP_2, IC >$ is $\forall\exists$-stable in E
iff the formula
$s(E) \wedge Safety \wedge \forall G\ (Q\ \rightarrow\ \bigwedge_{i=1}^{n_1}(f_{1i} \rightarrow \Phi_i))$ is satisfiable, where Φ_i denotes the subformula $\exists X[...]$ from the formula for $\exists\exists$ -stability.

$(\forall\forall):\ \ < LP_1, LP_2, IC >$ is $\forall\forall$-stable in E
iff the formula $\exists X[...]$ from the formula for

$s(E) \ \wedge \ Safety \ \wedge$
$\forall G \, (Q \ \rightarrow \ (\bigwedge_{i=1}^{n_1} \{ f_{1i} \rightarrow \exists X [UPD_{1i} \wedge (\bigvee_{i=1}^{n_1} f_{2i}) \wedge \bigwedge_{j=1}^{n_2} (f_{2j} \wedge \exists X.UPD_{2j})] \}))$
is satisfiable.

$(\exists\forall): \ \ < LP_1, LP_2, IC >$ is $\exists\forall$-stable in E
iff the formula
$s(E) \wedge Safety \wedge \ \forall G \, (Q \ \rightarrow \bigvee_{i=1}^{n_1} (f_{1i} \wedge \Psi_i))$ is satisfiable, where Ψ_i denotes the subformula $\exists X[...]$ from the formula for $\forall\forall$ -stability.

Note that all these reductions have polynomial complexity. So, using polynomial space and exponential time upper bounds for BPTL and its linear time counterpart we obtain the following

THEOREM. i) The Q_1Q_2 -stability problem for cooperating programs in $GPROD$ with quantifier-free integrity constraints is decidable in exponential time for any $Q_1, Q_2 \in \{\exists, \forall\}$;
ii) The $\exists\exists$ -stability problem for the same classes of programs and integrity constraints is decidable in polynomial space.

Note, moreover, that reductions above give only linear increasing of size of the formula in respect to the size of the source system of cooperating programs. It shows that some lower bounds in [1,2] can be improved. Namely, the results on EXPTIME-hardness of the stability problem can be complemented by $2^{n/logn}$ lower bound of time complexity.
We note that our consideration here was limited to $GPROD$ and quantifier-free integrity constraints, but some more general cases can be considered analogously.

References

[1] Dekhtyar M.I., Dikovsky A.Ja. Dynamic deductive data bases with steady behavior. In "Proc. of the 12 Intern. Conf. on Logic Programming", Ed. L.Sterling, The MIT Press, 1995, 183-197.
[2] Dekhtyar M.I., Dikovsky A.Ja. On homeostatic behavior of dynamic deductive data bases . In "Extended Abstracts of the Andrei Ershov 2nd Intern. Memorial Conf. Perspectives of Syst. Informatics", Novosibirsk, Russia, 1996, 196-201 (the full draft will be published in LNCS).

[3] Emerson E.A. Temporal and modal logic. In "Handbook of Theor. Comput. Sci.", Ed. J. van Leeuwen, Elsewier Sci. Publishers, 1990.

[4] Valiev M.K. On axiomatization of logic of discrete branching time. In "Modal and Intensional Logics. Proc. VIII Confer. on Logic and Method. of Science", Palanga, 1982 (in Russian).

[5] Valiev M.K. On temporal dependencies in databases. Trans. of Academy of Sci. Techn. Cybernetics, 1985, No.1, 106-115 (in Russian).