Action Theories in Temporal Databases

Nikos Papadakis and Dimitris Plexousakis

Department of Computer Science, University of Crete
P.O.Box 2208, Heraklion, Crete, GR-714 09 GREECE
(npapadak,dp)@csd.uch.gr

Abstract. Reasoning about action and change has been one of the main research themes of the knowledge representation and planning communities of the last 15 years. Action theories providing an axiomatic basis for managing change are applicable to a wide area of disciplines including software engineering, (cognitive) robotics and data/knowledge base systems. In this paper, we review action theories proposed for reasoning about the dynamics of database systems. We examine how these theories deal with the three infamous problems associated with this area, namely: (a) the frame problem, which refers to the identification of predicates or functions that remain unchanged as a result of action execution, (b) the ramification problem, which refers to determining the indirect effects of actions, and (c) the qualification problem, which refers to determining the preconditions which must hold prior to the execution of an action. We briefly describe the solutions which have been proposed for these problems and position these problems in a temporal database context. We also introduce an abstract solution based on the language of situation calculus. This solution extends the causality-based solution of ramification and qualification problems in conventional databases.

Keywords: action theories, database dynamics, frame problem, ramification problem, qualification problem, situation calculus, temporal databases.

Topic Area: Intelligent Systems, Databases and Information Retrieval

1 Introduction

The world represented in a database is not static. It changes continuously. The changes occur as results of database transactions. An atomic database transaction can be considered as an action. So, we can say that the changes in a database occur as results of actions. These actions change stored data in the database, and thus they may affect integrity constraints which determine the consistent states of the database. A database is consistent when all integrity constraints are satisfied.

An action may have direct and indirect effects. For example assume a database that stores information about the items in a room together with their position. Assume that there is a bokkcase with books and a lamp on table. When the bookcase changes position (as a direct effect of action "move bookcase"), the books change position too, as an indirect effect of the action "move bookcase". When the lamp changes position, the books and the bookcase do not change position. The truth value of any such proposition may change as the effect of any action's execution. The term *fluent* has been establish as a name for these propositions.

As we see from the above example, some items are affected by some actions while others are not. The problem of determining which predicates and functions are **not** affected when an action is executed is called the **frame problem** and was introduced by MacCarthy [1] in 1969.

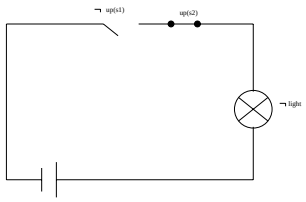

Fig. 1. A simple circuit

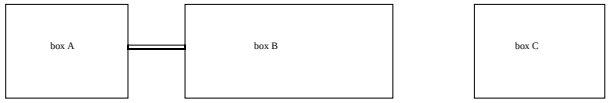

Fig. 2. Ramifications and qualifications

Another example is shown in figure 1. There are two switches and a lamp. There are three integrity constraints

$$up(s1) \wedge up(s2) \supset light \quad (1)$$
$$\neg up(s1) \supset \neg light \quad (2)$$
$$\neg up(s2) \supset \neg light \quad (3)$$

The first constraint says that when two switches are up, the lamp must be lit. The second and third constraint says that when a switch is down the lamp must not be lit. Assume that the circuit's situation is $S = \{\neg up(s_1), up(s_2), \neg light\}$. The action $toggle - switch(s_1)$ has the direct effect $up(s_1)$. The new situation is $S' = \{up(s_1), up(s_2), \neg light\}$. S' is inconsistent because it violates the integrity constraint (1). The indirect effect is $\neg up(s_2)$ or $light$. So the consistent situation is one of $S'' = \{up(s_1), \neg up(s_2), \neg light\}$ or $S'' = \{up(s_1), up(s_2), light\}$. Notice that the indirect effect $light$ is caused by the first constraint, while the effect $\neg up(s_2)$ is caused by the third constraint. All indirect effects are caused by the presence of constraints. The **ramification problem** refers to the concise description of the indirect effects of an action in the presence of constraints.

As an additional example, consider the case in which a driver wants to start his car. There are some preconditions which must hold before the driver can start his car. These are that there is no potato in the tail pipe, the tank is not empty, there is not an engine problem and the battery is not low. So, $\neg gas \supset \neg run$, $potato\ in\ tail\ pipe \supset \neg run$, $low - battery \supset \neg run$, $engine - problem \supset \neg run$.

The problem of determining the context in which an action is allowed to execute is the **qualification problem**.

As a last example (borrowed from [8]) consider the situation depicted in figure 2. There are three boxes A, B and C. The boxes A and B are connected. A box can move to a new position if the new position is clear.

$$poss(move(x, l')) \supset clear(l') . \quad (4)$$

Sentence (4) expresses that $clear(l')$ is a necessary condition of moving x to l'. This precondition is not always enough. For example, assume that the predicates $on(A, l_1)$, $on(B, l_2)$ and on (C, l_3) hold. The predicate $on(A, l_1)$, means that the item A is at position l_1. The action $move(A, l_2)$ can not be executed because $\neg clear(l_2)$ holds $(on(B, l_2))$. The action $move(A, l_2)$ has the indirect effect $move(B, l_4)$, $l_4 \neq l_2$

(because A and B are connected). So the predicate $clear(l_2)$ holds. Hence, the effect of the action move is expressed as:

$$move(A, l_2) \supset on(A, l_2) \wedge on(B, l_4)\,.$$

If the position l_4 is clear then we can execute the action $move(A, l_2)$ otherwise we cannot. So rule (4) must change to

$$on(x, l) \wedge y \neq x \wedge \neg connected(x, y) \wedge on(y, l') \supset disqualified(move(x, l')) \quad (5)$$

The predicate $disqualified(a)$ means that the action a cannot execute. As we can see, there are both ramification and qualification effects of an action. This problem is the **qualified ramification problem**.

The majority of approaches for ensuring consistency ignore the **frame** and **ramification** problem. A notable exception is the work described by Plexousakis in [23] and by Plexousakis and Mylopoulos [24] where the problem is addressed in a temporal database context.

The rest of paper is organized as follows: in section 2, we present the solutions which have been proposed for the frame, the ramification and the qualification problems. Also we examine the qualified ramification problem. The ramification and qualification problems in temporal databases are examined in section 3, and a solution is presented. The paper cocludes with a summary and directions for futher research.

2 Action Theories in Conventional Databases

2.1 The Frame Problem

The simplest solution is the *monotonic* approach [1]. This solution suggests two kinds of axioms, namely *action axioms* and *frame axioms*. The action axioms specify the fluents that hold after the execution of an action and frame axioms specify the fluents which do not change after an action. Some other solutions are the *default* approach [2] and the *STRIPS* approach [3].

For the solution of the frame problem, the use of the situation calculus [1] has been suggested. The situation calculus is a second-order language that represents the changes which occur in a domain of interest, as results of actions. One possible evolution of the world is a sequence of actions and is represented by a first-order term, called a situation. The initial situation S_0 is a situation at which no action has occurred yet. A binary function, $do(a, s)$ yields the situation resulting from the execution of an action a while in situation s.

Many solutions based on situation calculus have been suggested (Pednault [4], Hass [5] and Reiter [6]). The most important ones are those proposed by Reiter [6].

Reiter [6] suggests that a fluent f is true in a situation $S' = do(a, S)$ which came up after the execution of action a in a situation S, if f is false in S and the preconditions $(\gamma^+)_f$ (which make f true) hold in S', or f is true in S and the preconditions $(\gamma^-)_f$ (which make f false) do not hold in S'. An action a can execute in a situation S only if its preconditions (poss(a,S)) hold.

We explain this method with an example, (borrowed from [6]). Assume that there is a robot r which carries some items. An item breaks when the robot lets it fall or when a bomb explodes next to it. These are described by the follows axioms :

$$fragile(x, S) \supset broken(x, do(drop(r, x), S))$$
$$nextTo(b, x, S) \wedge bomb(b) \supset broken(x, do(explode(b), S))\,.$$

As we observe the above axioms have different assumptions but the same conclusion. So, we can join them in one

$$[(\exists r)\{a = drop(r,x) \wedge fragile(x,s)\}$$
$$\vee(\exists b)\{a = explode(b) \wedge nextTo(b,x,s)\}]$$
$$\supset broken(x, do(a,s))\,.$$

The above axiom contains two actions which, when executed, cause the fluent "broken" to become true. Before the execution of an action in situation S some preconditions must hold. The above axiom is rewritten as

$$Poss(a,s) \wedge [(\exists(r))\{a = drop(r,x) \wedge fragile(x,s)\}$$
$$\vee(\exists b)\{a = explode(b) \wedge nextTo(b,x,s)\}]$$
$$\supset broken(x, do(a,s))\,.$$

we set

$$(\gamma^+)_{broken}(a,s) \equiv\ [(\exists(r))\{a = drop(r,x) \wedge fragile(x,s)\}$$
$$\vee(\exists b)\{a = explode(b) \wedge nextTo(b,x,s)\}]\,,$$

where $(\gamma^+)_{broken}$ is the precondition which, when true, after the execution of action a, the fluent "broken" becomes true.Finally

$$Poss(a,s) \wedge (\gamma^+)_{broken}(a,s) \supset\ broken(x, do(a,s))\,. \quad (6)$$

If the robot repairs a broken item, then it is no longer broken. So we need to write a new axiom

$$Poss(a,s) \wedge [(\exists(r,x))a = repair(r,x)]$$
$$\supset \neg broken(x, do(a,s))\,.$$

we set

$$(\gamma^-)_{broken}(a,s) \equiv\ [(\exists(r,x))a = repair(r,x)]$$
$$\supset \neg broken(x, do(a,s))\,,$$

where $(\gamma^-)_{broken}$ is the precondition which, when true, then after the execution of action a the fluent "broken" becomes false. Finally

$$Poss(a,s) \wedge (\gamma^-)_{broken}(a,s) \supset\ \neg broken(x, do(a,s))\,. \quad (7)$$

From (6) and (7) we conclude that for each fluent an axiom is defined as follows

$$Poss(a,s) \supset f(do(a,S)) \equiv (\gamma^+)_f(a,s) \vee f(S) \wedge \neg(\gamma^-)_f(a,S)\,.$$

This solution needs $F + A$ axioms. One axiom for each fluent, plus one axiom for each action.

2.2 Ramification Problem

The ramification problem [7] refers to the indirect effects of an action and to the consistency of the database. For the ramification problem many solutions have been suggested. The simplest of them is the *minimal-change* approach [9]. The method suggests that, when an action occurs in a situation S, we try to find a consistent situation S' which has the fewer changes from the situation S. S' is the situation that is closer to S than any other situation.

Regarding the second example we described in the introduction, the minimal change approach gives two possible consistent situations $S_1 = \{up(s_1), up(s_2), light\}$ and $S_2 = \{up(s_1, \neg up(s_2), \neg light\}$. It is sensible to light the lamp, whereas downing the switch s_2 isn't. The minimal change approach cannot select one of them.

The above two solutions S_1 and S_2 cannot be distinguished unless we categorize the fluents [10,11,12]. The fluents are categorized in *primary* and *secondary*. A primary fluent can change only as a direct effect of an action, while a secondary one only as an indirect effect of an action. After an actioni takes place, we choose the situation with the fewer changes in primary fluents.

The categorization of fluents solves the ramification problem only if all fluents can be categorized. If some fluents are primary for some actions and secondary for some other, this solution is not satisfactory. A fluent could change or remain unchanged after an action. This depends on the context of a database. We need a context which allows us to produce the indirect effects of an action based on this dependence.

Causal relationships [13,14,15,16,17,18] capture this dependence between an action and an indirect effect. Each causal relationship consists two parts. The first part, called *context*, consists of one fluent formula which when true, the causal relationship can execute. The second part is the indirect effect of an action, called cause. A causal relationship has the form

$$\epsilon \ \ causes \ \ \rho \ \ if \ \ \Phi$$

where ϵ is an action, ρ is the indirect effect and Φ is the context. In the above example, there are four causal relationships: $up(s1)$ $causes$ $light$ if $up(s2)$, $up(s2)$ $causes$ $light$ if $up(s1)$, $\neg up(s1)$ $causes$ $\neg light$ if $\top$ and $\neg up(s2)$ $causes$ $\neg light$ if $\top$.

The causal relationships can be derived by using one of the algorithms in [16,17,18]

2.3 Qualification Problem

Several solutions have been proposed for the qualification problem. The most prominent ones are the *default approach* and *minimal-change* approach [8]. The default solution suggests that, for each action a, we must determine a formula F^a which, when true, prohibits action a from executing. The formula F^a has the form

$$F^a = \bigvee F_i \supset disq(a)\,,$$

where each F_i is a fluent. When any of the F_i is true, the action a can not execute. In the example of the driver and the car, the fluent formula of the fluent run is

$$F^{run} \equiv \exists x.in\ tail\ pipe(x) \vee tank-empty \vee low-battery \vee engine-problem \supset disq(ignite)\,.$$

Another solution is an extension of the minimal-change possible world approach that has been suggested for solving ramification problem. After each action a executes, we try to find a consistent

situation which contains all direct and indirect effects of a. If there is at least one such situation then the action can execute, otherwise it cannot.

As we have seen in the last example of the introduction, the ramification problem is difficult when there are qualifications. One solution is to extended the causal relationship in a way to include the qualifications. Each causal relationship changes from $\epsilon \ \ causes \ \rho \ if \ \ \Phi$ to $\epsilon \ \ causes \ \rho \ if \ \ \Phi \wedge \neg ab_C$, where ab_C is a fluent formula. When ab_C is true in a situation, then the causal relationship does not hold for this situation.

The ab_C is equivalence with the fluent $disq(\epsilon)$. The causal relationships can be derived by using an extension of the algorithms [16,17,18].

3 Temporal Databases

Most solutions of the frame, ramification and qualification problems in conventional database are based on situation calculus. For temporal, databases we need to incorporate time in the situation calculus. Pinto [19,20] and Fusaoka [22] have suggested some ways for inserting time in the situation calculus. Pinto suggested one correspondence between situation calculus and one linear time line. This correspondence is defined between real situations and time. This is not absolutely correct because situation calculus supports many parallel history of situations. The above weakness can be overcome by defining a correspondence between a branching structure time and situations. The branching time structure creates many parallel history of situations. One of these histories of situations, is the history which take place in real world. Each situation S which, occured in real world, make the predicate $actual(S)$ true (the predicate $actual$, shows if one situation occured (or not occured) in real world). As we show in the following, we extended this approach for solving the ramification and qualification problems, which are of great interst in temporal databases. We describe these problems with an example. Assume that a driver can not drive his car for four hours after drinking alcohol. This is expressed by a constraint of the form:

$$occurs(drink, t) \supset \neg occurs(drive, t_1) \ \wedge t_1 < t + 4h \, ,$$

where t and t_1 are temporal variablies and the predicate $occur(drink, t)$ means that the action drink is executed at time t. In a temporal database we need to describe the direct and indirect effects of an action not only to the next situation but possible for many future situations. In the above example, the action drink has the indirect effect that the driver can not drive during the following four hours. In these four hours, a number of other actions may execute leading to many different situations. In all these situations the action "drink alcohol" has the indirect effect $\neg drive$.

The causal relationships can not solve the ramification problem in temporal databases because they determine the direct and indirect effects only for the next situation. The same weakness characterizes all other solutions of ramification problem in conventional databases.

The above weakness can be alleviated by constructing a correspondence between situations and actions with time. Figure 3 depicts this correspondence. There are three parallel axes. The first is the situations axis, the second is the time axis and the third is the actions axis. We assume that all actions are instantaneous. When an action occurs, the database changes into a new situation. For example, at time t_1 when action a_1 occurs, the situation changes from S_0 to $S_1 = do(a_1, S_0)$. We introduce two predicates $occurs_{A,T}(a, t)$ and $occurs_{S,T}(S, t)$ which correspond to the actions and situations with a

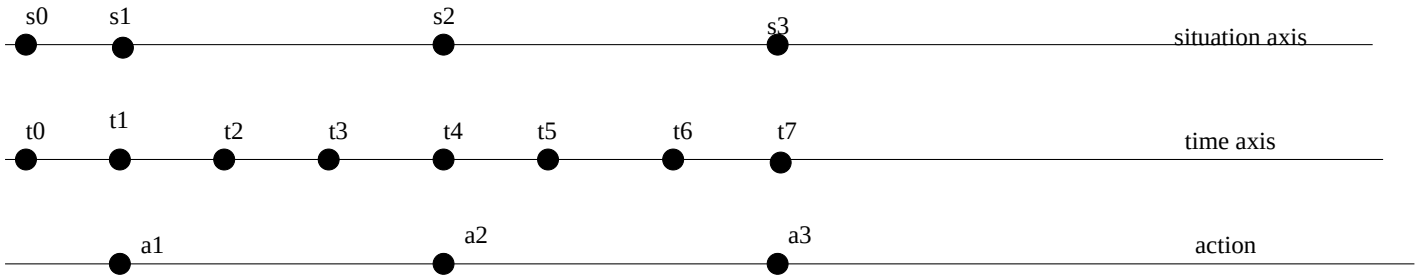

Fig. 3. The correspondce situations and actions with the time

specific time stamp. Also, we introduce two functions $start(S)$ and $end(S)$ which show the start and the end of situation S.

The ramification problem in temporal databases becomes more complex in case the effects (direct and indirect) of an action may change the effects of another action in the current and the future situations. We explain this with an example. Assume that someone that causes an accident cannot drive for the following five years, except if granted a pardon. In that case, he is allowed to drive one year after getting the pardon. At time t a driver x cannot drive if

$$occur_{A,T}(accident(x), t) \supset \neg\, occur_{A,T}(drive(x), t_1) \wedge t_1 \leq t + 5y\,,$$

holds. This condition is necessary but not enough to forbid driver x from driving. It is enough if x has been granted a pardon. So

$$occur_{A,T}(get - pardon(x), t) \wedge t_1 \geq t + 1y \;\; \supset candrive(x, t_1)\,.$$

As the above relations show there is a dependence between the actions drive, accident, get-pardon. We represent this dependence with the predicate $duration(a_1, a_2, t, t')$ which when true, it means that the action a_1 executes at time t and affects the action a_2 at time t'. The above relations take the form

$$\begin{aligned}\epsilon^{-}_{drive} &\equiv occur_{A,T}(accident(x), t) \;\; \wedge duration(accident(x), drive, t, t')\\ \epsilon^{+}_{drive} &\equiv occur_{A,T}(get - pardon(x), t) \;\; \wedge duration(get - pardon(x), drive, t, t')\,.\end{aligned}$$

The ϵ^{-}_{drive} expresses the preconditions which when true prohibit the action "drive" from executing at time t'. The ϵ^{+}_{drive} expresses the preconditions which when true permit the execution of action "drive" at time t'.

A driver can not drive at time t' if

$$E^{-}_{drive}(t') \equiv \epsilon^{-}_{drive}(t') \;\; \wedge \neg\epsilon^{+}_{drive}(t')\,,$$

holds. A driver can drive at time t' if

$$E^{+}_{drive}(t') \equiv \neg\epsilon^{-}_{drive}(t') \;\; \vee (\epsilon^{-}_{drive}(t') \wedge \;\; \epsilon^{+}_{drive}(t'))\,,$$

holds. This means that at the time t', someone can drive if the preconditions ϵ^{-}_{drive} do not hold or they do hold but also the preconditions ϵ^{+}_{drive} hold.

Generally for every fluent f the E_f^+ and E_f^- are the formulas which must hold, for it to become true or false at time t', respectively. Now we can write the causal relationships

$$a(t) \quad causes \quad f(t') \quad if \quad E_f^+ \qquad (8)$$
$$a(t) \quad causes \quad \neg f(t') \quad if \quad E_f^- \qquad (9)\,.$$

We solve both the ramification and qualification problem. The formulas E_f^+ and E_f^- capture the ramification and qualification effects. The causal relationships can be derived by using an extension of the algorithms [16,17,18]. The predicate *duration* must be defined for all actions which have dependences. In the above example assume additionaally that someone that drinks cannot drive for the following five hours. In that case the predicate duration

$$duration(drink(x), drive, t, t') \equiv \ occurs_{A,T}(drink(x), t') \wedge (t - t' < 5h)$$
$$duration(accident(x), drive, t, t') \equiv \ occurs_{A,T}(accident(x), t') \wedge (t - t' < 5y)$$
$$duration(getpardon(x), drive, t, t') \equiv \ occurs_{A,T}(getpardon(x), t') \wedge (t - t' < 1y)\,.$$

The complexity for the definition of the predicate *duration* is $O(n^2)$, where n is the number of actions. This happens when each action depends on each other action. This means that $O(n^2)$ predicates must be defined. We can decrease this complexity if for each action we define a "duration predicate" which contains all dependences with all other actions. In this case the complexity is $O(n)$. In order to achieve this we change the defintion of predicate duration as $duration(a, t_i)$ which means that the action a can not execute for the following t_i time units. If we assume that the time unit is the hour, in the above example we have

$$occurs_{A,T}(drink(x), t') \wedge duration(drive, t_i) \wedge \ t_i < 5 \supset duration(drive, 5)$$
$$occurs_{A,T}(accident(x), t') \wedge duration(drive, t_i) \wedge \ t_i < (5 * 365 * 24) \supset duration(drive, 5 * 365 * 24)$$
$$occurs_{A,T}(getpardon(x), t') \wedge duration(drive, t_i) \wedge \ t_i > 1 * 365 * 24 \supset duration(drive, 365 * 24)\,.$$

The first axiom means that the action $drink(x)$ which occurs at time t' has as indirect effect to disqualify the action *drive* for the following 5 hours. This occurs, if the action *drive* is disqualified at time t' for a time-interval that is smaller than 5 hours, otherwise if the action *drive* is disqualified for time-interval which is bigger than 5 hours, the action *drink* has not any effect.

We can rewrite the three above axioms as follows:

$$occurs_{A,T}(drink(x), t') \ \supset (duration(drive, t_i) \wedge t_i < 5) \wedge duration(drive, 5)$$
$$occurs_{A,T}(accident(x), t') \supset (duration(drive, t_i) \wedge \ t_i < 5 * 365 * 24) \wedge duration(drive, 5 * 365 * 24)$$
$$occurs_{A,T}(getpardon(x), t') \supset (duration(drive, t_i) \wedge \ t_i > 1 * 365 * 24) \wedge duration(drive, 365 * 24)\,.$$

We can generalize for any action a

$$occurs_{A,T}(a, t') \supset \bigwedge_{a'} (duration(a', t_i) \wedge f(t_i, c, a') \ \wedge duration(a', t_{i'})\,,$$

where the function f determines if the action a effects the action a'. The $\bigwedge$ means that we take all actions which can effect for the action a. At each time unit we decrease by one time unit the second parameter of predicate *duration*, if it is greater than zero. If the second parameter of predicate duration becomes zero then the action can execute.

The maximum number of causal relationships that need to be defined is $O(2 * F * A)$, where F is the number of fluents and A the number of actions.

The above problem becomes more complex if actions have duration. In this case, the direct and indirect effects of an action must be determined with regards to the start and/or end of this action. Assume that in the above example, that the action drink occurs during a time-span. The driver must not drive from the start of action drink until four hours after the end of this action. We need two new predicates $start(a)$ and $end(a)$ which show the start and end of an action. Now the constraint is rewritten as follows

$$start(drink, t_1) \wedge end(drink, t_2) \supset \neg occurs_{T,A}(drive, t) \wedge (t \leq t_2 + 4h \vee (t \geq t_1 \wedge t \leq t_2))$$

Now the predicate *duration* must be definied with reference the start and the end of the action drink.

$$occurs_{A,T}(start(drink(x)), t') \supset duration(drive, \infty)$$
$$occurs_{A,T}(end(drink(x)), t') \supset duration(drive, 5)\,.$$

We assume an action with duration is equivalent to two instatenous actions: one for the start and one for the end of the action. The first axiom means that when someone starts to drink they are forbiden to drive. The ∞ shows that we do not khow when they stopped drinking. The second axiom maens that when someone stops to drink, they could drive after 5 hours. We need to define $O(2 * A)$ axioms for the predicate duration. Also the axioms (8) and (9) must be define for the start and for the end each action. So, the maximum number of causal relationships that need to be defined is $O(4 * F * A)$, where F is the number of fluents and A the number of actions.

The ramification and qualification problem in temporal databases becomes more complex if the indirect and direct effects refer only to the future situations (exluding the present situation). Assume that, in the above example, the action *drink* has as indirect effect that the driver cannot drive after one hour from the start of the action *drink*. The above solution cannot solve this problem because it assumes that the direct and indirect effect of an action start for the moment at which the action starts. We must change the definition of predicate *duration* in order to capture the fact that the indirect effects of an action refer only to some future situations. We define the predicate *duration* as $duration(a, t_i, t)$ which mean that the action a cannot execute for time t_i after the time moment t. Now the predicate *duration* is

$$occurs_{A,T}(start(drink(x)), t') \supset duration(drive, \infty, t_\prime)$$
$$occurs_{A,T}(end(drink(x)), t') \supset duration(drive, 5, t')\,.$$

The complexity does not change. We still need $O(2 * A)$ axioms for the predicate duration.

The ramification problem in temporal databases becomes even more complex if the indirect and direct effects refer to past situations. In this case the effects may be periodic recursive. We have not dealt with this case yet.

4 Conclusions

This paper reviewed research results on the solution of the frame, ramification and qualification problems in conventional databases and proposed an intial approach to these problem in temporal databases. In future work, we shall investigate the extension of the algorithms [16,17,18] in order to derive the above causal relationships. Also we want to improve the solutions which are presented in section 6. We shall attempt to reduce the complexity. Furthermore, we want to solve the ramification problem, even for the cases that the indirect and direct effects refer to past situations.

References

1. J. McCarthy and P.J. Hayes. Some philophical problem from the standpoint of artificial intelligence. In B. Meltzer and D. Mitchie, editors, Machine Intelligence 4, pages 463-502. American Elsevier, New York, 1969.
2. R. Reiter A logic for default reasoning. Artificial Intelligence, 13:81-132, 1980.
3. R. Fikes and N. J. Nilsson, STRIPS: A new approach to the application of theorem proving to problem solving. Artificial Intelligence, 2:189-208, 1971.
4. E. Pednault. ADL: Exploring the Middle Ground between STRIPS and the Situation Calculus. In R.J. Brachman, H. Levesque, and R. Reiter, editors, Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning (KR' 89), pages 324-332. Morgan Kaufmann Publishers, Inc., San Mateo, California, 1989.
5. A. Haas. The Case for Domain-Specific Frame Axioms. In F. Brown, editor. The frame problem in artificial intelligence. Proceedings of the 1987 workshop, pages 343-348, Los Altos, California.
6. R. Reiter The frame problem in the situation calculus: A simple solution (sometimes) and a Completeness Result for Goal Regression. 1991
7. M. Ginsberg and D. Smith. Reasoning about action I: A possible words approach. Artificial Intelligence, 35:165-195, 1988.
8. M. Ginsberg and D. Smith. Reasoning about action II: A possible words approach. Artificial Intelligence, 35:311-342, 1988.
9. M. Winslett. Reasoning about action using a possible models approach. In Proceeding of the AAAI National Conference on Artifical Intelligence, pages 89-93, Saint Paul, MN, August 1988.
10. V. Lifshitz. Towards a metatheory of action. In J.F. Allen, R. Fikes, and E. Sandewall, editors, Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning, pages 376-386, Cambridge, MA, 1991.
11. V. Lifshitz. Frames in the space of situations, Artificial Intelligence, 46:365-376, 1990.
12. V. Lifshitz. Restricted monotonicity. In Proceedings of the AAAI National Conference on Artifical Intelligence, pages 432-437, Washington DC, July 1993.
13. C. Elkan. Reasoning about action in first order logic. In Proceedings of the Conference of the Canadian Society for Comptutational Studies of Intelligence (CSCSI), pages 221-227, Vancouver, Canada, May 1992.
14. N. McCain and Hudson Turner. A causal theory of ramifications and qualifications. In C. S. Mellish, editor, Proceedings of the International Joint Conference on Artifical Intelligence (IJCAI), pages 1978-1984, Montreal, Canada, August 1995.
15. J. Gustafon. Extending Temporal Action Logic for Ramification and Concurency, Thesis No 719 of Linkoping Studies in Science and Technology, 1998.
16. M. Thielscher. Ramification and causality. Artifical Intelligence, 89(1-2):317-364, 1997.
17. M. Thielscher. Reasoning about actions: Steady versus stabilizing state constraints. Artifical Intelligence, 104:339-355, 1988.
18. M. Thielscher.Nondeterministic actions in the fluent calculus: Disjunctive state update axioms. In S. Holldobler, editor, Intellectics and Computational Logic. Kluwer Academic, 1999.
19. J. Pinto. Temporal Reasoning in the Situation Calculus. Ph.D. Thesis, Dept. of Computer Science, Univ. of Toronto, Jan. 1994.
20. J. Pinto and R. Reiter. Temporal Reasoning in Logic Programming: A Case for the Situation Calculus. Proc. 10th Int. Conf. on Logic Programming, Budapest, Hungary, June 21-24, 1993.
21. M. Thielscher. Qualified ramifications. In B. Kuipers and B.Wbber, editors, Proceedings of the AAAI National Conference on Artificial Intelligence, pages 466-471, 1997
22. A. Fusaoka. Situation Calculus on a Dense Flow of Time. Proceedings of the AAAI National Conference on Artificial Intelligence, pages 633-638, 1996
23. Dimitris Plexousakis. Maintenance of Integrity Constraints in Temporal Deductive Knowledge Bases. Phd Thesis, Dept. of Computer Science, Univ. of Toronto, Jan. 1996.
24. Dimitris Plexousakis, John Mylopoulos: Accomodating Integrity Constraints During Database Design. Proceedings of EDBT 1996, pages 497-513