A Temporal Relational Algebra as a Basis for Temporal Relational Completeness

Alexander Tuzhilin and James Clifford

Information Systems Department
Stern School of Business
New York University

Abstract

We define a temporal algebra that is applicable to *any* temporal relational data model supporting discrete linear bounded time. This algebra has the five basic relational algebra operators extended to the temporal domain and an operator of *linear recursion*. We show that this algebra has the expressive power of a safe temporal calculus based on the predicate temporal logic with the **until** and **since** temporal operators. In [CrCl89], a historical calculus was proposed as a basis for historical relational completeness. We propose the temporal algebra defined in this paper and the equivalent temporal calculus as an alternative basis for temporal relational completeness.

1 Introduction

There have been several data models proposed in the literature for handling the "historical" or "temporal" dimension of data, together with various query languages and algebras defined for these models. A few representative examples of this work are [Ari86, ClWar83, ClCr87, Gad88, LoJo88, NaAhm88, Snod87, Tan86] (we provide this list to illustrate the scope of the work in this area and do not make any claims for its completeness). Because of the different approaches taken by these authors, it has been difficult to compare the query languages, whether calculi or algebras, introduced in their work. In light of this, Croker and Clifford [CrCl89] proposed the concept of ***historical relational completeness*** as a standard for comparison among various temporal query languages and algebras. A temporal query language or algebra is said to be historically relationally complete if it is at least as powerful as the historical calculus L_h introduced in [CrCl89].

In [Tuzh89], a temporal calculus was proposed for an arbitrary temporal relational data model. This calculus was based on a predicate temporal logic with **until / since** temporal operators [Kamp68]. In this calculus, time is referenced only implicitly; it is "encapsulated" in temporal operators, and the calculus does not have temporal constants or variables. Alternatively, we can use a two-sorted first order logic with one sort being time as a basis for a temporal calculus. In this calculus, some predicates can have a ***single*** temporal attribute, and arbitrary quantifications are allowed over temporal variables. Similar calculi with explicit references to time were proposed in [CrCl89] and in [KSW90]; they will be described in Section 4. It follows from [Kamp68] that the two calculi (based on temporal logic and on the first-order logic with explicit time references) have the same expressive power for time modeled with real numbers or integers[1].

In this paper, we define a temporal relational algebra equivalent to the two calculi for certain models of time. Initially, we define two versions of a temporal relational algebra. Both versions have the five operators of the relational algebra, i.e. select, project, Cartesian product, set difference and union, extended naturally to the temporal domain. In addition, each version has different temporal operators that have no equivalents in the relational algebra. We show in the paper that the first version of the temporal relational algebra is equivalent to the restricted version of the temporal calculus based on the temporal operators of *necessity* ($\Box$), *possibility* ($\diamond$), and *next* ($\circ$). We also show that the second version of the algebra is equivalent to the unrestricted version of the temporal calculus (based on the **until** and **since** temporal operators). This implies that the second version of the temporal relational algebra has

Proceedings of the 16th VLDB Conference
Brisbane, Australia 1990

[1] However, for time modeled with rationals, the two calculi have different expressive powers [Kamp68].

more expressive power than the first one. Therefore, we will consider only the second version when we will refer to the temporal relational algebra.

We propose the temporal relational algebra and the two temporal calculi as an alternative basis for temporal relational completeness because of the following reasons. First, the temporal calculi have a sound and well-studied theoretical basis since they are based on first-order logic and on temporal logic. Second, both the calculi and the algebra are very simple. Essentially, one temporal calculus is based on the first-order logic and another one is obtained from the first-order logic by adding a temporal operator **until** and its "mirror" image **since**. The temporal algebra is obtained from the relational algebra by extending its five basic operators and by adding a single additional temporal operator. Third, the two calculi and the algebra are equivalent for certain models of time, i.e. besides being simple and "natural," the two approaches have the same expressive power. This suggests that they capture an important class of temporal queries. Fourth, the temporal algebra and the two calculi are reduced to the relational algebra and calculus in the degenerate case when the time set consists of only one instance. Fifth, the temporal calculi are *independent* of a specific temporal relational data model, and the temporal algebra is independent of any data model based on the discrete bounded model of time. Some of the query languages and algebras proposed in the literature are *tailored* to a specific temporal data model. That is, operators of these languages take into account specific constructs of the underlying temporal data model. For example, the constructs **overlap, begin of** and **end of** of TQuel [Snod87] assume that the temporal data are grouped into *intervals*. There are no model-specific operators in the temporal calculus and in the algebra considered in this paper. This means that the temporal calculus can be applied to *any* temporal relational data model and the temporal algebra to any temporal relational data model supporting discrete bounded time.

Note that the standard relational calculus and algebra also possess the first three properties described above, and these properties account for the popularity of the relational model. Although the last two properties are not applicable to the standard relational case, we believe that they provide good additional justification for the temporal relational algebra and the two temporal calculi to be accepted as a basis for temporal relational completeness.

As will be explained in Section 4, the temporal calculus L_h proposed in [CrCl89] is different from our algebra and the two calculi. This means that this paper and [CrCl89] provide two alternative approaches to defining temporal (historical) relational completeness.

The rest of the paper is organized as follows. To make the paper self-contained, we review the temporal logic and the temporal calculus based on it in Section 2. In Section 3, we define a temporal relational algebra and show that it is equivalent to the temporal calculus. In Section 4, we propose the algebra and the calculus as the basis for the temporal relational completeness.

2 Overview of Temporal Logic and of the Calculus Based on It

In this section, we review a temporal calculus based on the predicate temporal logic as proposed in [Tuzh89] and [TuKe89].

Since temporal logic deals with time, we have to specify the model of time that we will be working with. The most general model represents time as an arbitrary set with a partial order imposed on it. By specifying additional axioms, we can introduce other models of time, e.g. time can be discrete or dense, bounded or unbounded, linear or branching [vBen83]. Although the temporal calculus can be defined for an arbitrary model of time (since it is based on the predicate temporal logic), we consider the discrete linear bounded model of time in this paper because the algebra **TA** defined in Section 3 is based on that model.

The syntax of a predicate temporal logic is obtained from the first-order logic by adding various temporal operators. We consider two predicate temporal logics in this paper. The first one, **TL**$'$, has the operators necessity $\Box$, possibility $\diamond$, and "next" $\circ$, as well as their past "mirror" images. $\Box A$ is true at time t if A holds at all times $t' \geq t$; $\diamond A$ is true at time t if A holds at some time $t' \geq t$; $\circ A$ is true at time t if A holds at time $t+1$.

The second type of temporal logic, **TL**, has the binary temporal operator **until** and its past "mirror" image **since**. A **until** B is true if A holds at all the future time points up to a time point at which B holds. It is well-known [Krog87] that for the discrete bounded linear time, other binary future operators, such as **before**, **atnext**, **while**, and their corresponding past mirror images have the same expressive power as the **until** / **since** pair. In this paper, we will also be using the future temporal operator **atfirst**, which is very similar to **atnext**, and its past "mirror" image **atlast**. A **atfirst** B is true at time t if A is true at the time instance when B first becomes true in the future (at or after time t). If B never becomes true, then A **atfirst** B is false at time t.

It is also well-known [Krog87, ReUr71, Gab91] that for the discrete bounded model of time the operators $\Box$, $\Diamond$, and $\circ$ can be expressed in terms of **since** / **until** pair. Therefore, **TL** is at least as powerful as **TL'**. We will, when convenient, consider $\Box$, $\Diamond$ and $\circ$ as part of **TL**. Moreover, Kamp [Kamp68] showed that **TL** is strictly more powerful than **TL'** for the continuous unbounded model of time. We extend Kamp's result to the discrete bounded model of time in the following proposition.

Proposition 1 **TL** *is strictly more powerful than* **TL'** *for the discrete linear bounded model of time.*

Proof: See the Appendix for the proof. ∎

The semantics of a temporal logic formula is defined with a *temporal structure* [Krog87], which comprises the values of all its predicates at *all* the time instances. Formally, let $P_1, \ldots, P_k$ be a finite set of predicates considered in the predicate temporal language[2]. Then, a temporal structure is a mapping $K : T \rightarrow \mathcal{P}_1 \times \ldots \times \mathcal{P}_k$, where T is a partially ordered set of time instances, and $\mathcal{P}_i$ is the set of all the possible interpretations of predicate P_i. The mapping K assigns to each time instance an instance of each of the predicates $P_1, \ldots, P_k$ at that time. We will use K_t instead of $K(t)$ to denote the value of temporal structure K at time t. We make an assumption, natural in the database context, that the domains of predicates *do not change* over time.

From a database perspective, a temporal structure K is most naturally looked at as mapping of each moment of time t into a state of the database, i.e. into instances of each of the database relations at time t. Therefore, each predicate in a temporal structure determines a *temporal relation*, i.e. a relation that changes over time.

A temporal database represented in a certain temporal data model, such as the TQuel data model [Snod87], HRDM [ClCr87], or the homogeneous data model [Gad88], defines a temporal structure, i.e. the mapping K, although often implicitly. Therefore, a temporal structure represents a common base for different temporal data models. For instance, Figure 1 shows an example relation in HRDM representing employees and their department and salary over time. Its corresponding temporal structure is shown in Figure 2[3].

Given a temporal structure for temporal logic predicates, we can extend this temporal structure to arbitrary temporal logic formulas in the standard inductive way [Krog87]. For example, we can define $K_t(A \textbf{ atfirst } B)$ in terms of $K_t(A)$ and $K_t(B)$ as follows. $K_t(A \textbf{ atfirst } B)$ is true if there is $t' \geq t$ such that $K_{t'}(B)$ and $K_{t'}(A)$ are true, and for all t'' such that $t < t'' < t'$, $K_{t''}(B)$ is false.

HRDM [ClCr87] defines historical relations similarly to the way temporal structures are specified in temporal logic. However, HRDM assumes that values of individual attributes, not relation instances, change over time. Similarly, Segev and Shoshani [SeSho87] define time sequences as values of attributes changing over time. They also define a query language over the collections of these time sequences. Clearly, the two approaches have a strong similarity with temporal structures of the temporal logic; the only difference is that temporal structures specify the whole relations changing over time, and HRDM and time sequences specify values of individual attributes changing over time. As shown above in Figures 1 and 2, it is clear that there is a simple isomorphism between these two representation schemes.

In this paper, we make a restrictive assumption that all the temporal relations are considered over the same bounded temporal domain (*lifespan* as defined in [ClCr87]). This assumption can be relaxed by extending differing temporal domains to one common temporal domain and assuming that the relations over extended portions of temporal domains are null. However, we do not consider this extension in this paper.

We now define temporal safety. Intuitively, a temporal formula is safe if it can produce only bounded relations at all the time instances[4]. More precisely, we define a *safe* temporal formula exactly as it is done for the snapshot relational case [Ull88], except that, in addition, we assume that the temporal operators **atfirst**, **until**, **before**, necessity and possibility and their past "mirror" images produce safe formulas if operands of these operators constitute safe formulas. It is easy to see that, indeed, these temporal operators cannot produce infinite temporal relations if they operate on finite relations.

Definition 1 *A temporal calculus query is an expression of the form*

$$\{x_1, x_2, \ldots, x_n \mid \phi(x_1, x_2, \ldots, x_n)\}$$

[2] Since we are interested in database applications, we consider only a finite number of predicates corresponding to the set of relations in a database.

[3] The dots in the time column in the table *do not* represent intervals. They indicate the fact that the instance of EMPL did not change for the specified moments of time. We would like to stress it again that the temporal structure is independent of a specific temporal relational data model, i.e. it can be defined for *any* such model.

[4] "Boundedness" refers to the structural and not to the temporal domain because we have already assumed that the temporal domain is bounded.

NAME	*DEPT*	*SALARY*
$[0, now] \rightarrow$ Tom	$[0,10) \rightarrow$ Sales $[10, now] \rightarrow$ Mktg	$[0,7) \rightarrow$ 20K $[7,11) \rightarrow$ 30K $[11, now] \rightarrow$ 27K
$[5, now] \rightarrow$ Juni	$[5, now] \rightarrow$ Acctng	$[5, now] \rightarrow$ 28K
$[2,10) \rightarrow$ Ashley $[14, now] \rightarrow$ Ashley	$[2,6) \rightarrow$ Engrng $[6,10) \rightarrow$ Mktg $[14, now] \rightarrow$ Engrng	$[2,5) \rightarrow$ 27K $[5,10) \rightarrow$ 30K $[14, now] \rightarrow$ 35K

Figure 1: The Historical Relation **EMPL**

time	K_i**(EMPL)**
$i = 0 \dots 1$	EMPL(Tom, Sales, 20K)
$i = 2 \dots 4$	EMPL(Tom, Sales, 20K) EMPL(Ashley, Engrng, 27K)
$i = 5$	EMPL(Tom, Sales, 20K) EMPL(Juni, Acctng, 28K) EMPL(Ashley, Engrng, 30K)
$i = 6$	EMPL(Tom, Sales, 20K) EMPL(Juni, Acctng, 28K) EMPL(Ashley, Mktg, 30K)
$i = 7 \dots 9$	EMPL(Tom, Sales, 30K) EMPL(Juni, Acctng, 28K) EMPL(Ashley, Mktg, 30K)
$i = 10$	EMPL(Tom, Mktg, 30K) EMPL(Juni, Acctng, 28K)
$i = 11 \dots 13$	EMPL(Tom, Mktg, 27K) EMPL(Juni, Acctng, 28K)
$i = 14 \dots now$	EMPL(Tom, Mktg, 27K) EMPL(Juni, Acctng, 28K) EMPL(Ashley, Engrng, 35K)

Figure 2: The Temporal Structure for **EMPL**

where ϕ is a safe *predicate temporal logic formula and $x_1, x_2, \dots, x_n$ are* some *of the free variables in ϕ. If ϕ is a formula from* **TL** *then we denote the corresponding temporal calculus as* TC. *If ϕ is a formula from* TL′ *then we denote the corresponding calculus as* TC′.

Let T be a temporal domain for the predicates in ϕ. The answer *to this query is a temporal relation defined on T, such that for any t in T, its instance is* $\{x_1, x_2, \dots, x_n \mid K_t(\phi(x_1, x_2, \dots, x_n))\}$.

Note that a temporal calculus query operates on temporal relations and returns a temporal relation, i.e. it returns the same type of object as the type of its operands.

Example 1

Consider a database schema consisting of two relation schemas *EMPL(NAME, DEPTNO, SALARY)* and *DEPT(DEPTNO, MGR)* and two instances of temporal relations with these schema over some time domain. Then we can consider the following queries:

Q1: Find Jack's manager at the time of his last salary increase.

$$\{\ MGR \mid DEPT(DEPTNO, MGR)\ \mathbf{atlast}\ (\odot\ EMPL('JACK', DEPTNO, SAL1) \wedge EMPL('JACK', DEPTNO, SAL2) \wedge SAL2 > SAL1\ )\ \}$$

where $\odot$ is the "previous" temporal operator, i.e. the mirror image of the "next" operator, **atlast** is the mirror image of the **atfirst** operator, and $\odot$ has higher precedence than $\wedge$.

Observe that the second operand of the **atlast** operator, which is a temporal relation, determines departments and salaries at the times of Jack's salary increases ("yesterday" it was lower then it is "today"). This temporal relation (corresponding to the second operand of **atlast**) will be empty at all other times. Then, the **atlast** operator determines the instance of the *DEPT* relation at the time of the latest salary increase.

Note that the answer to this query is a temporal relation: Jack's manager at the time of his latest salary increase depends on the "current" time.

Q2: Find the name and department of each employee that has at some time received a cut in salary [CrCl89].

$$\{ \mathit{NAME, DEPTNO} \mid \star (\mathit{EMPL(NAME,DEPTNO,SAL1)} \wedge \star\, \mathit{EMPL(NAME,DEPTNO,SAL2)} \wedge \mathit{SAL2} > \mathit{SAL1}) \}$$

where $\star$ is the mirror image of $\diamond$, i.e. $\star\alpha$ is true if α is true at *some* time in the past.

Observe that, since the two possibility operators are nested, the outer operator determines some time t' and the inner determines some time t'' such that $t'' \leq t'$ and the salary at time t' is smaller than at time t''.

∎

3 Temporal Algebra

In Section 2, we defined a temporal calculus based on the predicate temporal logic. In this section, we define a temporal algebra that is equivalent to this temporal calculus.

We assume that the operators of this algebra are applied to the temporal relations introduced in Section 2. As was discussed in Section 2, we assume that all the temporal domains are equal.

Since we want the proposed temporal algebra to be reduced to the relational algebra in the degenerate case when the time domain is reduced only to one time instance, we start with the following five operations which constitute direct extensions of the relational algebra operations to the temporal domain.

Let $\mathbf{R} = \{R_t\}_{t \in T}$, $\mathbf{S} = \{S_t\}_{t \in T}$ and $\mathbf{Q} = \{Q_t\}_{t \in T}$ be temporal relations defined over a temporal domain (lifespan) $T = [t_1, t_n]$[5]. Using the relational algebra terminology, two temporal relations are ***union compatible*** if their schemas have the same sets of attributes. Then we consider the following temporal algebra operators:

[5] We would like to point out that the operators O1 - O8, to be introduced below, can generally be defined for *any* temporal domain. However, the operator O9 can be defined only for discrete bounded temporal domains.

O1: *Select*: $\mathbf{S} = \sigma_F(\mathbf{R})$ iff $S_t = \sigma_F(R_t)$ for all t in T, where F is the first-order (non-temporal) formula defined as for the standard relational case [Ull88].

O2: *Project*: $\mathbf{S} = \pi_{A_1,\ldots,A_k}(\mathbf{R})$ iff $S_t = \pi_{A_1,\ldots,A_k}(R_t)$ for all t in T, where $A_1, \ldots, A_k$ are some attributes in R.

O3: *Cartesian product*: $\mathbf{S} = \mathbf{R} \times \mathbf{Q}$ iff $S_t = R_t \times Q_t$ for all t in T.

O4: *Set difference*: $\mathbf{S} = \mathbf{R} - \mathbf{Q}$ iff $\mathbf{S}$ and $\mathbf{R}$ are union compatible and $S_t = R_t - Q_t$ for all t in T.

O5: *Union*: $\mathbf{S} = \mathbf{R} \cup \mathbf{Q}$ iff $\mathbf{S}$ and $\mathbf{R}$ are union compatible and $S_t = R_t \cup Q_t$ for all t in T.

A temporal join operation $\bowtie$ can be defined exactly as for the snapshot case in terms of the Cartesian product, select and project operators.

In addition to these five operators we add the following temporal operators that have no equivalents in the relational algebra:

O6: *Sequential union (future and past)*:

1. future: $\mathbf{S} = \mathbf{SU_F}(\mathbf{R})$ iff $S_t = \bigcup_{i=t}^{t_n} R_i$;
2. past: $\mathbf{S} = \mathbf{SU_P}(\mathbf{R})$ iff $S_t = \bigcup_{i=t_1}^{t} R_i$.

At any time t, $\mathbf{SU_P}$ gives the present instance and the past history, and $\mathbf{SU_F}$ the present instance and the future history of a relation "compressed" to time t. For example, Figure 3 presents the relation obtained from relation **EMPL** from Figure 2 after the application of the operator $\mathbf{SU_F}$.

O7: *Sequential intersection (future and past)*:

1. future: $\mathbf{S} = \mathbf{SI_F}(\mathbf{R})$ iff $S_t = \bigcap_{i=t}^{t_n} R_i$;
2. past: $\mathbf{S} = \mathbf{SI_P}(\mathbf{R})$ iff $S_t = \bigcap_{i=t_1}^{t} R_i$.

At any time t, $\mathbf{SI_P}$ gives those tuples in a relation which remained constant from the beginning of the lifespan until now, and $\mathbf{SI_F}$ gives those tuples that will remain constant in the future from now until the end of the lifespan. For example, Figure 4 presents the relation obtained from **EMPL** after the application of the operator $\mathbf{SI_F}$.

O8: *Shift (future and past)*:

1. future: $\mathbf{S} = \mathbf{SH_F}(\mathbf{R})$ iff $S_{t+1} = R_t$ for $t = t_1, \ldots t_n - 1$, $S_{t_1} = \emptyset$;
2. past: $\mathbf{S} = \mathbf{SH_P}(\mathbf{R})$ iff $S_{t-1} = R_t$ for $t = t_1 + 1, \ldots t_n$, $S_{t_n} = \emptyset$.

This operator shifts all "facts" represented in the relations one unit of time into the future for $\mathbf{SH_F}$ operator and into the past for $\mathbf{SH_P}$ operator. For example, Figure 5 presents the relation obtained from **EMPL** after the application of the operator $\mathbf{SH_F}$.

Denote the algebra based on operations **O1 - O8** as **TA**$'$.

Proposition 2
For any temporal relation **R**, $\mathbf{SU_F}$(**R**) *is equivalent to* $\diamond$**R**, $\mathbf{SI_F}$(**R**) *to* $\Box$**R**, *and* $\mathbf{SH_F}$(**R**) *to* $\circ$**R**. *Similar relationships hold between the past versions of* **SU**, **SI**, **SH** *operators and the "mirror" images of* $\diamond$, $\Box$, $\circ$ *operators.*

Proof: Follows from the definitions of operators **O6 - O8** and the corresponding temporal operators. ∎

The next corollary immediately follows from the above proposition.

Corollary 3 **TA**$'$ *has the same expressive power as* **TC**$'$.

It follows from Proposition 1 that **TC** has more expressive power than **TC**$'$. Corollary 3 says that **TA**$'$ has the same expressive power as **TC**$'$. Therefore, **TA**$'$ has less expressive power than **TC**, and we have to find a temporal algebra equivalent to the temporal calculus **TC**. We propose the following temporal operator instead of operators **O6 - O8**:

O9: *Linear recursive operator (future and past).* Let **A** and **B** be union compatible temporal relations. Let $\mathbf{C} = \{C_t\}_{t \in T}$ be a temporal relation defined over the temporal domain T.

1. The *future* linear recursive operator is defined as follows: $\mathbf{C} = \mathbf{L_F}(\mathbf{A}, \mathbf{B})$ iff $C_{t+1} = (A_t \cap C_t) \cup B_t$, $C_{t_1} = \emptyset$.
2. The past linear recursive operator is defined similarly: $\mathbf{C} = \mathbf{L_P}(\mathbf{A}, \mathbf{B})$ iff $C_{t-1} = (A_t \cap C_t) \cup B_t$, $C_{t_n} = \emptyset$.

Denote the temporal algebra defined by operators **O1 - O5** and **O9** as **TA**.

Example 2
Assume that time is measured in years. Consider temporal relation **UNEMPL**(NAME) specifying that a person is unemployed for most of the year. Figure 6 gives an example of such a relation. Temporal relation **TAXES**(NAME, TAX) specifies taxes a person paid in a certain year. Figure 7 gives an example of such a relation. We say that a person is a

time	K_i(EMPL)
$i = 0 \ldots 4$	EMPL(Tom, Sales, 20K) EMPL(Tom, Sales, 30K) EMPL(Tom, Mktg, 30K) EMPL(Tom, Mktg, 27K) EMPL(Juni, Acctng, 28K) EMPL(Ashley, Engrng, 27K) EMPL(Ashley, Engrng, 30K) EMPL(Ashley, Mktg, 30K) EMPL(Ashley, Engrng, 35K)
$i = 5$	EMPL(Tom, Sales, 20K) EMPL(Tom, Sales, 30K) EMPL(Tom, Mktg, 30K) EMPL(Tom, Mktg, 27K) EMPL(Juni, Acctng, 28K) EMPL(Ashley, Engrng, 30K) EMPL(Ashley, Mktg, 30K) EMPL(Ashley, Engrng, 35K)
$i = 6$	EMPL(Tom, Sales, 20K) EMPL(Tom, Sales, 30K) EMPL(Tom, Mktg, 30K) EMPL(Tom, Mktg, 27K) EMPL(Juni, Acctng, 28K) EMPL(Ashley, Mktg, 30K) EMPL(Ashley, Engrng, 35K)
$i = 7 \ldots 9$	EMPL(Tom, Sales, 30K) EMPL(Tom, Mktg, 30K) EMPL(Tom, Mktg, 27K) EMPL(Juni, Acctng, 28K) EMPL(Ashley, Mktg, 30K) EMPL(Ashley, Engrng, 35K)
$i = 10$	EMPL(Tom, Mktg, 30K) EMPL(Tom, Mktg, 27K) EMPL(Juni, Acctng, 28K) EMPL(Ashley, Engrng, 35K)
$i = 11 \ldots now$	EMPL(Tom, Mktg, 27K) EMPL(Juni, Acctng, 28K) EMPL(Ashley, Engrng, 35K)

Figure 3: The Future Sequential Union of **EMPL**

time	K_i(EMPL)
$i = 0 \ldots 4$	empty relation
$i = 5 \ldots 10$	EMPL(Juni, Acctng, 28K)
$i = 11 \ldots 13$	EMPL(Tom, Mktg, 27K) EMPL(Juni, Acctng, 28K)
$i = 14 \ldots now$	EMPL(Tom, Mktg, 27K) EMPL(Juni, Acctng, 28K) EMPL(Ashley, Engrng, 35K)

Figure 4: The Future Sequential Intersection of **EMPL**

time	K_i(EMPL)
$i = 0$	empty relation
$i = 1 \ldots 2$	EMPL(Tom, Sales, 20K)
$i = 3 \ldots 5$	EMPL(Tom, Sales, 20K) EMPL(Ashley, Engrng, 27K)
$i = 6$	EMPL(Tom, Sales, 20K) EMPL(Juni, Acctng, 28K) EMPL(Ashley, Engrng, 30K)
$i = 7$	EMPL(Tom, Sales, 20K) EMPL(Juni, Acctng, 28K) EMPL(Ashley, Mktg, 30K)
$i = 8 \ldots 10$	EMPL(Tom, Sales, 30K) EMPL(Juni, Acctng, 28K) EMPL(Ashley, Mktg, 30K)
$i = 11$	EMPL(Tom, Mktg, 30K) EMPL(Juni, Acctng, 28K)
$i = 12 \ldots 14$	EMPL(Tom, Mktg, 27K) EMPL(Juni, Acctng, 28K)
$i = 15 \ldots now$	EMPL(Tom, Mktg, 27K) EMPL(Juni, Acctng, 28K) EMPL(Ashley, Engrng, 35K)

Figure 5: The Future Shift Operator of **EMPL**

YEAR	NAME
1986	Susan Bill
1987	Susan Jack
1988	Jack
1989 - 1990	$\emptyset$

Figure 6: Temporal Relation **UNEMPL**

YEAR	NAME	TAX
1986	Jack	8400
1987	Bill	10400
1988	Bill Susan	10800 12000
1989	Bill Susan	11500 13200
1990	Bill Susan Jack	12800 13600 9200

Figure 7: Temporal Relation **TAXES**

"good citizen" if he or she always paid taxes during the period of his or her last employment, i.e. since the last time the person was unemployed. The relation **GOOD_CITIZEN**(NAME) can be computed with the following temporal calculus expression:

$$\textbf{GOOD\_CITIZEN} = \{ \text{NAME} \mid \textbf{TAXES}(\text{NAME,TAX}) \ \textbf{since} \ \textbf{UNEMPL}(\text{NAME}) \}$$

where **since** is the mirror image of **until**.

GOOD_CITIZEN can also be computed in **TA** as follows. Set **TAXES1** = π_{NAME} (**TAXES**). Then **GOOD_CITIZEN** = L_F(**TAXES1**,**UNEMPL**), i.e. GOOD_CITIZEN$_{k+1}$ = (GOOD_CITIZEN$_k \cap$ TAXES1$_k$) $\cup$ UNEMPL$_k$. The resulting relation **GOOD_CITIZEN** is shown in Figure 8. The last row constitute predictions who will be a good citizen in 1991.

∎

Note that the linear recursive operator can be defined only for discrete bounded models of time. This constitutes a limitation of the algebra **TA**.

To prove the main result of this section that **TA** is equivalent to **TC**, we need the following technical lemma.

YEAR	NAME
1986	$\emptyset$
1987	Susan Bill
1988	Bill Susan Jack
1989	Bill Susan Jack
1990	Bill Susan
1991	Bill Susan

Figure 8: Temporal Relation **GOOD_CITIZEN**

Lemma 4

$$\{\mathbf{x}, \mathbf{y} \mid A(\mathbf{x}) \ \mathbf{atfirst}\ B(\mathbf{y})\} \quad (1)$$

can be computed in **TA** *by applying operators* **O1 - O5** *and* **O9** *to the temporal relations A and B, where* **x**, **y** *are the vectors of free variables in A and B respectively.*

Proof: The possibility operator, $\diamond A = \{\mathbf{x} \mid \mathbf{true}\ \mathbf{atfirst}\ A(\mathbf{x})\}$ can be computed with recursion $C_{k-1}(\mathbf{x}) = C_k(\mathbf{x}) \cup A_k(\mathbf{x})$, $C_{t_n} = \emptyset$. Let $A' = A \bowtie \diamond B$ and $B' = \diamond A \bowtie B$, where $\bowtie$ is the temporal join operator as define above. It is easy to see that (1) and $\{\mathbf{x}, \mathbf{y} \mid A'(\mathbf{x}, \mathbf{y})\ \mathbf{atfirst}\ B'(\mathbf{x}, \mathbf{y})\}$ define the same query. The purpose of introducing A' and B' is to make predicates in (1) union compatible. Note that the query $\{\mathbf{x} \mid A(\mathbf{x})\ \mathbf{atfirst}\ B(\mathbf{x})\}$ can be expressed with the linear recursion $C_{k-1}(\mathbf{x}) = (C_k(\mathbf{x}) \cap \bar{B}_k(\mathbf{x})) \cup (A_k(\mathbf{x}) \cap B_k(\mathbf{x}))$. Therefore, (1) can be computed in **TA** as follows. First, compute A' and B'; then $D = A' \cap B'$; and finally $G = \mathbf{L_P}(\bar{B}, D)$. It follows from the previous arguments that G and (1) define the same mapping. ∎

Theorem 5 **TA** *and* **TC** *have the same expressive power.*

Proof: The proof that a safe **TC** query can be expressed in **TA** proceeds by induction on the number of operations in a query. All formulas without temporal operators can be expressed in **TA** in the manner similar to the purely relational case. In Lemma 4, we showed that **atfirst** can be expressed in **TA**. Similarly, other temporal operators, including **atlast** can be expressed in **TA**.

$$\begin{array}{ccc} \mathbf{TC} & = & \mathbf{TA} \\ \vee & & \vee \\ \mathbf{TC'} & = & \mathbf{TA'} \end{array}$$

Figure 9: Summary of Relationships Among **TA**, **TA′**, **TC**, and **TC′**.

The proof that a **TA** expression can be defined with a safe **TC** query proceeds by induction on the number of operators in the expression. Operators **O1 - O5** can be expressed in **TC** as in the snapshot relational case. The past linear recursive operator $\mathbf{C} = \mathbf{L_P}(\mathbf{A}, \mathbf{B})$ can be expressed in terms of the operator **unless** [Krog87] as $\mathbf{C} = \{\mathbf{x} \mid \mathbf{A}(\mathbf{x})\ \mathbf{unless}\ \mathbf{B}(\mathbf{x})\}$, where A **unless** B is equivalent to (A **until** B) $\vee \circ \Box B$, and, therefore, is safe. ∎

Corollary 6 **TA** *has more expressive power than* **TA′**.

Proof: Follows from Proposition 1, Corollary 3 and Theorem 5. ∎

The summary of relationships among the temporal algebras **TA**, **TA′** and the temporal calculi **TC** and **TC′** is shown in Figure 9, where = means the same and > means more expressive power.

Note that when the temporal domain is reduced to one time instance, both **TA** and **TA′** are reduced to the standard relational algebra.

Since **TA** has more expressive power than **TA′** and **TC** more expressive power than **TC′**, we will consider only the algebra **TA** and the equivalent calculus **TC** in the sequel.

In [SeSho87], Segev and Shoshani defined an algebra, i.e. a set of operators, over time sequence collections (TSCs). A TSC can be considered as a ternary relation, the first attribute being a surrogate (not changing over time), the second attribute being time, and the third attribute representing some value at the time specified by the second attribute. Our algebra **TA** differs from the algebra in [SeSho87] in the following ways. First, TSC operators are defined over TSCs, and **TA** operators over relations. Second, as in [CrCl89], Segev and Shoshani consider explicit references to time. For example, the *select* operator in [SeSho87] can refer to specific time instances. Third, [SeSho87] introduces additional operators, such as *aggregation*, and arithmetic functions to enhance the functionality of the basic set of operators. Fourth, Segev and Shoshani allow different time domains, i.e. an operator can return a time sequence over a time domain different from the time domain of its operands.

Despite these differences, **TA** (and **TA′**) and the TSC algebra have some common features. The most interesting among them is the observation that our future and past sequential unions and intersections correspond to Segev and Shoshani's *accumulation* operators GROUP TO BEGIN and GROUP TO END for logical AND and OR operators. However, they have no operator corresponding to our linear recursion operator. As it follows from Corollary 6, this operator enhances the expressive power of the algebra, and, therefore, is an important one.

4 Temporal Logic and Temporal Algebra as a Basis for Temporal Relational Completeness

There have been several temporal data models and corresponding query languages and algebras proposed in the past. Examples of this work comprise [Ari86, Snod87, Gad88, ClCr87, NaAhm88, LoJo88, Tan86]. There is a considerable diversity of approaches among these temporal models, algebras and query languages [CrCl89]. Snodgrass provides a comparison of various approaches in [Snod87].

Croker and Clifford [CrCl89] propose a notion of *historical (temporal) relational completeness* as a basis for specifying a minimum degree of expressive power for temporal query languages and algebras. A temporal query language or algebra is temporally relationally complete if it has at least the expressive power of a certain *historical relational calculus* L_h introduced in [CrCl89]. L_h is based on many-sorted first-order logic [End72] with time being explicitly supported as a sort (constants, variables and quantifiers are allowed over the time sort). Since *any* temporal data model, including the models proposed in the literature, can serve as a data model for L_h, it means that L_h is directly compatible with any temporal query language or algebra [CrCl89].

Notice that *exactly* the same kind of argument is applicable to the temporal calculus **TC**: it also can be applied to *any* temporal relational data model. Therefore, **TC** can be directly compared with any temporal query language or algebra. Similarly, **TA** can be compared with any temporal query language or algebra supporting a discrete bounded model of time. Therefore, temporal logic and its corresponding temporal algebra can also be used as a basis for temporal relational completeness.

Alternatively, we can use a two-sorted first order logic as a basis for a temporal calculus. In this calculus, time constitutes a separate sort with partial or total order imposed on it, some predicates can have a *single* temporal attribute, and arbitrary quantifications are allowed over time variables. Similar calculi with explicit references to time were proposed in [CrCl89] and in [KSW90]. The calculus L_h [CrCl89] is based on first-order logic with explicit references to time. However, it is more general than the calculus described above because it allows different lifespans (time domains) for different temporal relations. The calculus of [KSW90] differs from the calculus described above only in one respect. In the calculus of [KSW90], temporal predicates are defined over time *intervals*, i.e. they always take two temporal attributes, whereas in the calculus considered above, temporal predicates are defined over time instances, i.e. they always take only one temporal attribute. It was shown in [Kamp68] that for time modeled with real numbers and with integers, the first-order logic with explicit time references and the temporal logic based on **until** / **since** temporal operators provide the same expressive power. This means that for the discrete bounded model of time the temporal algebra, the temporal calculus **TA**, and the calculus with the explicit time references provide the same expressive power.

We propose the temporal algebra introduced in this paper and the two temporal calculi as an alternative basis for temporal relational completeness because of the reasons stated in the introduction. We briefly review them here. First, the temporal calculi have a sound and well-studied theoretical basis since they are based on first-order logic and on temporal logic. Second, both the calculi and the algebra are very simple. Third, the two calculi and the algebra are equivalent for the discrete bounded model of time. Fourth, the temporal algebra and the two calculi are reduced to the relational algebra and calculus in the degenerate case when the time set consists of only one instance. Fifth, the temporal calculi are *independent* of a specific temporal relational data model, and the temporal algebra is independent of any data model based on the discrete bounded model of time.

5 Conclusion

In this paper, we defined two temporal relational algebras for a discrete linear bounded model of time. Both of them have the five basic operators of the relational algebra extended to support time. The first version, **TA′**, has temporal operators of sequential union and intersection and the temporal shift operator (both the future and the past versions). The second version, **TA**, has a single operator of temporal linear recursion. We showed that **TA** has more expressive power than **TA**$'$. In addition, **TA** has the following properties. First, it is a simple extension of the relational algebra to the temporal domain. Second, it is equivalent to the temporal calculus **TC** based on the predicate temporal logic with the **since** and **until** operators and to the temporal calculus based on the first-order logic with the explicit support of time. Third, this algebra is applicable to *any* temporal relational data model supporting discrete bounded linear time.

Therefore, we propose the three formalisms, i.e. temporal algebra **TA**, temporal calculus **TC**, and the calculus based on the first order logic with explicit time support, as an alternative basis for *temporal (historical) relational completeness* as introduced in [CrCl89].

Appendix

Sketch of Proof of Proposition 1

We will prove that the temporal operator **atfirst** cannot be expressed in **TL**$'$ for the discrete linear bounded model of time. Without loss of generality, assume that **TL**$'$ is a propositional logic. In the proof, we will denote **true** with 1 and **false** with 0.

Assume that there is an expression in **TL**$'$ defined on atoms A and B that is equivalent to A **atfirst** B. Assume, this expression contains n operators of **TL**$'$, i.e. some of the operators $\Box$, $\diamond$, $\circ$, $\wedge$, $\vee$, and $\neg$, and the mirror images of $\Box$, $\diamond$, and $\circ$. Consider the following temporal structure K for atoms A and B. The time domain is $2(n+1)^2+1$ time units long. B has $2n+3$ 1's separated from each other by n 0's. A has $n+1$ 1's separated from each other by $2n+1$ 0's. Each 1 of A occurs at the same time as a 1 of B. The leftmost 1 is $n+1$ steps from the left end of the time interval, and the rightmost 1 is $n+1$ steps from the right end of the time interval. Observe that the answer to the A **atfirst** B query consists of the intervals of 1's of length n and the intervals of 0's of length n. These two types of intervals follow each other starting with 1's, i.e. first, there are n ones, then n zeroes, then n ones, etc. Clearly, there are $n+1$ intervals of 1's (each containing n 1's). Figure 10 shows the temporal structure of atoms A and B and of the formula A **atfirst** B for $n = 2$.

We claim that this answer cannot be obtained with n operators from **TL**$'$. Intuitively, only $\Box$ and $\diamond$ can "stretch" 1's across the block of 0's ($\circ$ cannot do this because there are only n operators and n zeros). Also, $\Box$ can "create" only one block of 1's at a time. Since there are $n+1$ blocks to be created and only n operators this leads to the contradiction.

Acknowledgements

We would like to thank Al Croker for his careful reading of the earlier drafts of this paper and Dov Gabbay for pointing to us the equivalence of the temporal calculus **TC** and the calculus based on the first-order logic with explicit reference to time for the discrete models of time.

References

[Ari86] G. Ariav. A temporally oriented data model. *TODS*, 11(4):499–527, 1986.

[ClCr87] J. Clifford and A. Croker. The historical data model (HRDM) and algebra based on lifespans. In *Proceedings of the International Conference on Data Engineering*, 1987. IEEE Computer Society.

[ClWar83] J. Clifford and D. S. Warren. Formal Semantics for Time in Databases. *TODS*, 6(2):214–254, 1983.

[CrCl89] A. Croker and J. Clifford. *On Completeness of Historical Relational Data Models.* Working Paper CRIS #192, GBA #89-2, Center for Research on Information Systems, Stern School of Business, New York University, 1989.

[End72] H. B. Enderton. *A Mathematical Introduction to Logic.* Academic Press, 1972. New York.

[Gab91] D. Gabbay. *Temporal Logic: Mathematical Foundations and Computational Aspects.* Oxford University Press, 1991, forthcoming.

[Gad88] S. K. Gadia. A homogeneous relational model and query languages for temporal databases. *TODS*, 13(4):418–448, 1988.

[KSW90] F. Kabanza, J.-M. Stevenne and P. Wolper. Handling infinite temporal data. In *Proceedings of the ACM Symposium on Principles of Database Systems*, pages 392–403, 1990.

[Kamp68] H. Kamp. *On the Tense Logic and the Theory of Order.* PhD thesis, UCLA, 1968.

[Krog87] F. Kroger. *Temporal Logic of Programs.* Springer-Verlag, 1987. EATCS Monographs on Theoretical Computer Science.

A:	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0
B:	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1
A **atfirst** B:	0	1	1	1	0	0	0	1	1	1	0	0	0	1	1	1	0	0	0

Figure 10: Temporal Structure for A, B, and A **atfirst** B

[LoJo88] N. A. Lorentzos and R. G. Johnson. TRA: a model for a temporal relational algebra. In C. Rolland, F. Bodart, and M. Leonard, editors, *Temporal Aspects in Information Systems*, pages 95–108, North-Holland, 1988.

[NaAhm88] S. B. Navathe and R. Ahmed. TSQL – a language interface for history databases. In C. Rolland, F. Bodart, and M. Leonard, editors, *Temporal Aspects in Information Systems*, pages 109–122, North-Holland, 1988.

[ReUr71] N. Rescher and A. Urquhart. *Temporal Logic*. Springer-Verlag, 1971.

[SeSho87] A. Segev and A. Shoshani. Logical modeling of temporal data. In *Proceedings of ACM SIGMOD Conference*, pages 454–466, 1987.

[Snod87] R. Snodgrass. The temporal query language TQuel. *TODS*, 12(2):247–298, 1987.

[Tan86] A. U. Tansel. Adding time dimension to relational model and extending relational algebra. *Information Systems*, 11:343–355, 1986.

[TuKe89] A. Tuzhilin and Z. M. Kedem. *Using Temporal Logic and Datalog to Query Databases Evolving in Time*. Technical Report 484, New York University, 1989.

[Tuzh89] A. Tuzhilin. *Using Relational Discrete Event Systems and Models for Prediction of Future Behavior of Databases*. PhD thesis, New York University, October 1989.

[Ull88] J. Ullman. *Principles of Database and Knowledge-Base Systems*. Volume 1, Computer Science Press, 1988.

[vBen83] J. van Benthem. *The Logic of Time*. D. Reidel Publishing Company, 1983.