The Temporal Multi-Dimensional Join

Peter Sune Jørgensen (sunes@cs.auc.dk)
Department of Computer Science, Aalborg University
Fredrik Bajers Vej 7E, 9220 Aalborg, Denmark
June 6, 2003

Abstract. The *temporal multi-dimensional join* (TMDJ) is a simple parameterizable operator which offers a systematic and efficient implementation for a wide range of advanced temporal operators. We start out by formalizing point-based, interval-based and duplicate-aware temporal operators. These are crucial but often confused semantic properties of temporal operators. We show that these semantic properties can be determined via a parameterization of the TMDJ. Finally, we describe a lightweight implementation of the TMDJ and report experimental results which show the performance of advanced temporal operations is orders of magnitude better than the performance of equivalent SQL solutions.

1 Introduction

An essential aspect of a temporal data model is the semantic properties of its temporal operators. Widely acknowledged key properties are point-based, interval-based, and duplicate-aware semantics. Although these terms are used widely a consensus definition is still missing. We illustrate this by considering the temporal difference of P and Q, i.e., $P -^t Q$. Even this simple task turns out to be quite complex and it is surprising to notice the number of different results that have been proposed. Essentially, the different results can be traced to the choice of the three semantic properties: interval-based, point-based and duplicate-aware.

Table 1. Temporal bags P and Q

P

A	I
7	[1,10]
7	[11,20]
7	[21,30]
7	[28,30]

Q

A	I
7	[15,18]
7	[17,22]

Consider the temporal bags in Table 1 (We use the term "bag" rather than "relation" to emphasize the possible presence of duplicates). For a *point-based* operator the result is independent of the grouping of time points into intervals, and as a consequence it is possible to view temporal data as a time-indexed

sequence of non-temporal data i.e. an interval timestamp is simply a shorthand notation for a sequence of time points. However, for an *interval-based* operator the result depends on the grouping of time points into intervals, and the grouping of time points in the result must be derived from the grouping of time points in its argument. Thus, it is significant that the time points of P in Table 1 between 11 and 30 are grouped into the intervals $[11, 20]$ and $[21, 30]$. For a *duplicate-aware* operator the multiplicity of a fact matters, which, e.g., makes the last tuple in P non-redundant. Combining the three properties yields eight semantically different classes of operators:

$$\left\{\begin{array}{c}\text{duplicate-aware } (da)\\ \text{not duplicate-aware } (\overline{da})\end{array}\right\} \times \left\{\begin{array}{c}\text{point-based } (pb)\\ \text{not point-based } (\overline{pb})\end{array}\right\} \times \left\{\begin{array}{c}\text{interval-based } (ib)\\ \text{not interval-based } (\overline{ib})\end{array}\right\}$$

Eight different possible results of the temporal difference $P -^t Q$ are illustrated in Table 2, which correspond to the eight different types of semantics. Results R_1, R_2, R_3, and R_4 contain the time points which are in P and not in Q, these are point-based results since the time points in the results do not depend on how the time points were grouped into intervals in P and Q Results R_1, R_2, R_5, and R_6 are interval-based, since the grouping of time points in P are preserved and respected in the results. Finally results R_1, R_3, R_5, and R_7 are duplicate-aware, since the last tuple of P is considered non-redundant.

Table 2. Point-based (pb), interval-based (ib), and duplicate-aware (da) results

R_1: pb, ib, da

A	I
7	[1,10]
7	[11,14]
7	[23,30]
7	[28,30]

R_2: $pb, ib, \overline{da}$

A	I
7	[1,10]
7	[11,14]
7	[23,30]

R_3: pb, $\overline{ib}$, da

A	I
7	[1,14]
7	[23,30]
7	[28,30]

R_4: $pb, \overline{ib}, \overline{da}$

A	I
7	[1,14]
7	[23,30]

R_5: $\overline{pb}$, ib, da

A	I
7	[1,10]
7	[11,20]
7	[21,30]
7	[28,30]

R_6: $\overline{pb}, ib, \overline{da}$

A	I
7	[1,10]
7	[11,20]
7	[21,30]

R_7: $\overline{pb}, \overline{ib}, da$

A	I
7	[1,3]
7	[4,20]
7	[21,30]
7	[28,30]

R_8: $\overline{pb}, \overline{ib}, \overline{da}$

A	I
7	[1,3]
7	[4,20]
7	[21,30]

The *temporal multi-dimensional join* (TMDJ) is a simple parameterizable operator, which can be used to efficiently implement a range of temporal operators. Conceptually the TMDJ groups tuples together in a number of subsets, where each subset is evaluated independently of all other subsets, and the final result of the TMDJ consists of the tuples derived from each subset. The data structure used for grouping tuples is the *grouped temporal bag*, where time points of non-temporally equivalent tuples can be explicitly grouped together.

Essentially, grouping all time points when a fact is true together gives us point-based semantics, grouping duplicate time points separately gives us duplicate-aware semantics, and grouping time points according to the timestamp gives us interval-based semantics.

The main contributions of this paper are:

- A formal definition of duplicate-aware, point-based, and interval-based semantics.
- A formal definition of the TMDJ, including a simple and efficient evaluation algorithm.
- A formal definition of the grouped temporal bag, the core data structure of the TMDJ.
- A specification of the parameters which determine the temporal semantics of the TMDJ.
- A performance study of a lightweight TMDJ implementation.

The remainder of the paper is organized as follows. Section 3 introduces the temporal data model. Section 4 formalizes interval-based, point-based, and duplicate-aware temporal operator semantics. Section 5 introduces grouped temporal bags, the core data structure of the TMDJ, which is used for grouping time points together. Section 6 formalizes the temporal multi-dimensional join (TMDJ), and specifies the parameters which can determine the temporal semantics of the TMDJ. Section 7 shows how the TMDJ can be used for temporal difference, and temporal aggregation. Section 8 evaluates the performance of the TMDJ. Finally conclusions and future work are presented in Section 9.

2 Related Work

The research into temporal databases has led to the development of various temporal data models [Ari86,NA89], and several temporal query languages, e.g. TSQL2 [Sno95], ATSQL [BJ96], IXSQL [LM97], and TQUEL [Sno96]. Often the main difference between the various data models have been the way in which the temporal dimension is incorporated into the model [TCG+93]. A common characteristic is that each model argues (often strongly!) for its specific data model. The result is a set of (incompatible) data models that are good for some applications but fail for others. We choose a different approach where we isolate three key properties that account for the differences between the models, and make them available as parameters of the TMDJ algorithm.

Several temporal query processing algorithms have been proposed [BSS96] [PJ99,Sno99,YW01,BJ03]. In general, the proposals are based on translating temporal query language statements into SQL statements, which are processed by an underlying conventional DBMS [Sli01]. It has been shown that such an approach is limited and suffers from a poor performance. Particularly, the advanced temporal operations considered in this paper cannot be implemented efficiently using plain SQL.

The semantic properties introduced in this paper extend the notions in [BBJ98], while the TMDJ is a temporal generalization of the MD-join [MAK01] [AB03], which has been used to efficiently implement complex OLAP queries.

3 Preliminaries

3.1 Temporal Data Model

A data model $\mathcal{M} = (\mathcal{D}, \mathcal{O})$ is composed of a set of data structures $\mathcal{D}$ and a set of operations $\mathcal{O}$ defined on these data structures. For instance, the relational data model is composed of relations and relational operators.

A *temporal data model* $\mathcal{M}_T = (\mathcal{D}_T, \mathcal{O}_T)$ is composed of temporal data structures $\mathcal{D}_T$ and a set of temporal operators $\mathcal{O}_T$. An operator is temporal iff it returns a temporal bag when applied to temporal bags. A *temporal bag* R is an instance of a *temporal schema* $\mathcal{R} = (X_1, ..., X_n \| I)$, where X_i is a non-temporal attribute and I is the temporal attribute. We use the $\|$ to separate the non-temporal attributes from the temporal attribute, and use $\mathbf{X}$ as a shorthand for the non-temporal attributes X_1, ..., X_n. The temporal attribute I is a closed interval with start point I^+ and end point I^- (i.e., $I = [I^+, I^-]$), where $I^+ \leq I^-$. We write $p \in I$ to state that time point p is contained in the interval I, i.e., $I^+ \leq p \leq I^-$.

3.2 Bag Algebra

A bag is a collection of elements that may contain duplicates [GM93]. We use $\{\!\{...\}\!\}$ to denote a bag. An element *n-belongs* ($\in^n$) to a bag iff it occurs exactly n times in the bag. Assume the bag $R = \{\!\{c, c, d, d, d\}\!\}$, then element c 2-belongs to R and element d 3-belongs to R. Below we define the most common bag operations.

Duplicate elimination, $R' = \varepsilon(R)$**:** R' contains a single instance of each element in R: $y \in^1 R' \Leftrightarrow y \in^n R$.

Selection, $R' = \sigma[P](R)$**:** R' contains all elements in R that satisfy predicate P: $y \in^n R' \Leftrightarrow y \in^n R \wedge P(y)$.

Projection, $R' = \pi[\mathbf{Z}](R)$**:** R' contains all elements of R projected on $\mathbf{Z}$: $y \in^n R' \Leftrightarrow R = R_1 \uplus R_2 \wedge |R_1| = n \wedge \forall t \in R_1(t.\mathbf{Z} = y) \wedge \forall t \in R_2(t.\mathbf{Z} \neq y)$.

Additive union, $R' = R_1 \uplus R_2$**:** R' contains all elements in R_1 and R_2: $y \in^{p+q} R' \Leftrightarrow y \in^p R_1 \wedge y \in^q R_2$.

Difference, $R' = R_1 - R_2$**:** R' contains all elements in R_1 minus all elements in R_2: $y \in^n R' \Leftrightarrow y \in^p R_1 \wedge y \in^q R_2 \wedge n = \max(0, p - q)$.

Cartesian product, $R' = R_1 \times R_2$**:** R' contains each element of R_1 combined with each element of R_2: $y \circ z \in^{p*q} R' \Leftrightarrow y \in^p R_1 \wedge z \in^q R_2$.

In the remainder of the paper we use the tuple calculus [SKS96] over bags to define newly introduced concepts and operators.

4 Semantic Properties

In the introduction we argued that the different opinions about the intended outcome of temporal operators can be attributed to three properties of the operators: is it interval-based, is it point-based, and is it duplicate-aware. This section gives a formal definition of these properties. First we define the time domain.

Definition 1. *$\mathcal{T}^p = (\mathcal{T}, <)$ is a* time point domain *over the set $\mathcal{T}$ iff $<$ defines a total order on $\mathcal{T}$. Each element of $\mathcal{T}$ corresponds to a* time point *of $\mathcal{T}^p$.*

Definition 2. *A* time interval *I of $\mathcal{T}^p$ is a set of connected time points iff any time point between two time points in I are also in I i.e. ($p_1 \in I \wedge p_2 \in I \wedge p_3 \in \mathcal{T}^p \wedge p_1 \leq p_3 \leq p_2) \Rightarrow p_3 \in I$. If $\mathcal{I}$ is the set of all time intervals of $\mathcal{T}^p$, then $\mathcal{T}^i = (\mathcal{I}, \subset)$ is a* time interval domain *over the time point domain $\mathcal{T}^p$.*

Note that intervals are often utilized as a syntactic shorthand representation for time points, due to the impractical nature of recording all time points when a tuple is true individually. Thus, it is clear that the difference between a point-based and an interval-based operator cannot be determined from the timestamp syntax. The characterizing difference between point-based operators and interval-based operators is found in the way they treat an interval timestamp. A point-based operator treats an interval as a set of individual time points, while an interval-based operator treats an interval as a set of *connected* time points i.e. the interval-based operator differentiates between the interval $[1, 10]$, and the intervals $[1, 5]$ and $[6, 10]$, while a point-based does not.

4.1 Point-based Operators

A point-based operator considers an interval timestamp as a set of individual time points. Thus, a point-based operator treats two temporal bags as equivalent, if the time points associated with a fact in one bag is identical to the time points associated with the same fact in the other bag. This is referred to as snapshot equivalence, and is defined as follows.

Definition 3. *The* timeslice operator, *τ_p, extracts the snapshot of a temporal bag R at time point p: $\tau_p(R) = \{\!\{\langle t.\mathbf{X}\rangle | t \in R \wedge p \in t.I)\}\!\}$.*

Definition 4. *Two temporal bags R_1 and R_2 are* snapshot equivalent, *$R_1 =^p R_2$, iff their snapshots are pairwise identical: $R_1 =^p R_2$* iff *$\forall p(\tau_p(R_1) = \tau_p(R_2))$.*

A temporal operator $\mathcal{O}$ is point-based iff snapshot equivalent arguments yield snapshot equivalent results. We use $\mathbf{A}$ as a shorthand notation for a list of arguments bags $R_1, ..., R_n$, and $\mathbf{A}' \subset \mathbf{A}$ is a shorthand notation for $R_1' \subseteq R_1 \wedge ... \wedge R_n' \subseteq R_n \wedge \bigcup_{i=1}^n R_i' \subset \bigcup_{i=1}^n R_i$.

Definition 5. *A temporal operator $\mathcal{O}$ is* point-based *iff it preserves snapshot equivalence, i.e., $\forall \mathbf{A_1}, \mathbf{A_2}(\mathbf{A_1} =^p \mathbf{A_2} \Rightarrow \mathcal{O}(\mathbf{A_1}) =^p \mathcal{O}(\mathbf{A_2}))$*

Example 1. Consider the *coalesce* operator (coal) [BSS96] an operator similar to conventional duplicate elimination, which merges values-equivalent tuples if the union of their timestamp is an interval.

$$
\begin{aligned}
&coal(R) = R', \text{ iff} \\
&\quad \forall p \in \mathcal{T}^p(t \in \tau_p(R) \Leftrightarrow t \in^1 \tau_p(R')) \wedge \\
&\quad \forall t, t' \in R'(t \neq t' \wedge t.\mathbf{X} = t'.\mathbf{X} \Rightarrow \neg adjc(t.I, t'.I) \wedge \neg ovlp(t.I, t'.I))
\end{aligned}
$$

The predicates adjc and ovlp are defined as usual:

$$
\begin{aligned}
adjc([I^+, I^-], [J^+, J^-]) &= (I^+ = succ(J^-)) \vee (succ(I^-) = J^+) \\
ovlp([I^+, I^-], [J^+, J^-]) &= (I^+ \leq J^+ \leq I^-) \vee (I^+ \leq J^- \leq I^-)
\end{aligned}
$$

Let R_1 and R_2 be temporal bags, where $R_1 = \{\!\{\langle 5 \| [1,15] \rangle, \langle 5 \| [10,20] \rangle\}\!\}$, and $R_2 = \{\!\{\langle 5 \| [1,5] \rangle, \langle 5 \| [6,15] \rangle, \langle 5 \| [10,20] \rangle\}\!\}$, then $coal(R_1) = coal(R_2) = \{\!\{\langle 5 \| [1,20] \rangle\}\!\}$.

The coalesce operator defines a normal form for point-based models, which ensures independence of both the timestamp representation and multiplicity of a fact i.e. the number of times a fact occurs in a snapshot.

Lemma 1. *Coalesce is a point-based operator.*

$$R_1 =^p R_2 \Rightarrow coal(R_1) = coal(R_2)$$

Proof: Since the snapshots are identical and coalesce merges all adjacent time points the results must be identical i.e. also snapshot equivalent.

Lemma 2. *Coalescing the argument of a temporal operator $\mathcal{O}$ yields point-based semantics.*

Proof: Temporal bags which are snapshot equivalent are identical when coalesced.

$$\forall \mathbf{A_1}, \mathbf{A_2}(\mathbf{A_1} =^p \mathbf{A_2} \Rightarrow coal(\mathbf{A_1}) = coal(\mathbf{A_2}) \wedge \mathcal{O}(coal(\mathbf{A_1})) = \mathcal{O}(coal(\mathbf{A_2})))$$

4.2 Interval-based Operators

Intuitively, an operator is interval-based iff it respects the grouping of time points into intervals. The defining property of interval-based operators is that they preserve the original grouping of time points. The first step towards a definition of interval-based operators is the definition of the time points that shall be associated with a result fact. For each operator $\mathcal{O}$ we assume the explicit definition of $\mathcal{O}^p$, which defines the bag of resulting time points associated with a set of non-temporal attribute values.

Example 2. Consider the definition of $\mathcal{O}^p$ for the set of basic temporal relational algebra operators: Temporal selection, temporal projection, temporal additive union, temporal difference, temporal Cartesian product, and coalesce.

$$
\begin{aligned}
\mathcal{O}^p_{\sigma^t_C(R)} &= \{\!\{\langle t.\mathbf{X} \| p\rangle | t \in R \wedge C(t) \wedge p \in t.I\}\!\} \\
\mathcal{O}^p_{\pi^t_Z(R)} &= \{\!\{\langle t.\mathbf{Z} \| p\rangle | t \in R \wedge p \in t.I\}\!\} \\
\mathcal{O}^p_{P \uplus^t Q} &= \{\!\{\langle t.\mathbf{X} \| p\rangle | (t \in P \vee t \in U) \wedge p \in t.I\}\!\} \\
\mathcal{O}^p_{P -^t Q} &= \{\!\{\langle t.\mathbf{X} \| p\rangle | t \in P \wedge p \in t.I \wedge \forall s \in Q(s.\mathbf{X} = t.\mathbf{X} \Rightarrow p \notin s.I)\}\!\} \\
\mathcal{O}^p_{P \times^t Q} &= \{\!\{\langle t.\mathbf{X}, s.\mathbf{Y} \| p\rangle | t \in P \wedge s \in Q \wedge p \in s.I \cap t.I\}\!\} \\
\mathcal{O}^p_{coal(R)} &= \{\!\{\langle \mathbf{X} \| p\rangle | p \in \mathcal{T}^p \wedge \exists t \in \tau_p(R)(\mathbf{X} = t.\mathbf{X})\}\!\}
\end{aligned}
$$

As an example let us consider the definition of $\mathcal{O}^p$ for temporal selection σ^t_C, where the condition C is $(X = 7)$ and the temporal bag $R = \{\!\{ \langle 7 \| [1,3]\rangle, \langle 10 \| [1,10]\rangle, \langle 7 \| [2,5]\rangle \}\!\}$ with the schema R(X‖I), then $\mathcal{O}^p_{\sigma^t_C(R)} = \{\!\{ \langle 7 \| 1\rangle, \langle 7 \| 2\rangle, \langle 7 \| 2\rangle, \langle 7 \| 3\rangle, \langle 7 \| 3\rangle, \langle 7 \| 4\rangle, \langle 7 \| 5\rangle \}\!\}$.

Definition 6. *Let $\mathcal{O}$ be a temporal operator, then $\mathcal{O}$ is* interval-based, *iff forall result tuples $\langle \mathbf{X} \| I\rangle$*

$$
\begin{aligned}
&\langle \mathbf{X} \| I\rangle \in \mathcal{O}(\mathbf{A}) \Leftrightarrow \\
&\quad \exists \mathbf{A}' \subseteq \mathbf{A}(\mathcal{O}^p(\mathbf{A}') \subseteq \mathcal{O}^p(\mathbf{A}) \wedge \forall p \in I(\langle \mathbf{X} \| p\rangle \in \mathcal{O}^p(\mathbf{A}')) \wedge && (1) \\
&\qquad \forall \mathbf{B_1}, ..., \mathbf{B_n}(\mathbf{B_1} \uplus ... \uplus \mathbf{B_n} = \mathbf{A}' \wedge \mathbf{B_1} \neq \emptyset \wedge ... \wedge \mathbf{B_n} \neq \emptyset \Rightarrow \\
&\qquad \mathcal{O}^p(\mathbf{B_1}) \uplus ... \uplus \mathcal{O}^p(\mathbf{B_n}) \neq \mathcal{O}^p(\mathbf{A}')) \wedge && (2) \\
&\quad \langle \mathbf{X} \| pred(I^-)\rangle \notin \mathcal{O}^p(\mathbf{A}') \wedge \langle \mathbf{X} \| succ(I^+)\rangle \notin \mathcal{O}^p(\mathbf{A}')) && (3)
\end{aligned}
$$

Thus, a result tuple, $\langle \mathbf{x} \| I\rangle$, must be derivable from a subset $\mathbf{A}'$ of the argument bags (1), this subset must be minimal (2), and the subset may not permit the derivation of larger result intervals (3).

Example 3. Consider the temporal projection $\pi^t_Z(R)$, where $R = \{\!\{\langle 5, 10 \| [1,3]\rangle, \langle 5, 10 \| [3,5]\rangle \}\!\}$ is an instance of the schema $R(X, Z \| I)$, and $\mathcal{O}^p_{\pi^t_Z(R)} = \{\!\{\langle 10 \| 1\rangle, \langle 10 \| 2\rangle, \langle 10 \| 3\rangle, \langle 10 \| 3\rangle, \langle 10 \| 4\rangle, \langle 10 \| 5\rangle\}\!\}$. Then there are two minimal subsets of R from which result tuples are derivable: $R_1 = \{\!\{\langle 5, 10 \| [1,3]\rangle \}\!\}$, and $R_2 = \{\!\{\langle 5, 10 \| [3,5]\rangle \}\!\}$, where $\mathcal{O}^p_{\pi^t_Z(R_1)} = \{\!\{\langle 10 \| 1\rangle, \langle 10 \| 2\rangle, \langle 10 \| 3\rangle\}\!\}$ and $\mathcal{O}^p_{\pi^t_Z(R_2)} = \{\!\{\langle 10 \| 3\rangle, \langle 10 \| 4\rangle, \langle 10 \| 5\rangle\}\!\}$. Thus, deriving the largest possible result intervals, the result of the interval-based temporal projection is $\pi^t_Z(R) = \{\!\{\langle 10 \| [1,3]\rangle, \langle 10 \| [3,5]\rangle\}\!\}$.

Example 4. Consider the coalesce operator $coal(R)$, where $R = \{\!\{ \langle 4 \| [1,4]\rangle, \langle 4 \| [5,8]\rangle\}\!\}$, and $\mathcal{O}^p_{coal(R)} = \{\!\{\langle 4 \| 1\rangle, \langle 4 \| 2\rangle, \langle 4 \| 3\rangle, \langle 4 \| 4\rangle, \langle 4 \| 5\rangle, \langle 4 \| 6\rangle, \langle 4 \| 7\rangle, \langle 4 \| 8\rangle\}\!\}$. There are two minimal subsets of R from which result tuples are derivable: $R_1 = \{\!\{\langle 4 \| [1,4]\rangle\}\!\}$, and $R_2 = \{\!\{\langle 4 \| [5,8]\rangle\}\!\}$, where $\mathcal{O}^p_{coal(R_1)} = \{\!\{\langle 4 \| 1\rangle, \langle 4 \| 2\rangle, \langle 4 \| 3\rangle, \langle 4 \| 4\rangle\}\!\}$ and $\mathcal{O}^p_{coal(R_2)} = \{\!\{\langle 4 \| 5\rangle, \langle 4 \| 6\rangle, \langle 4 \| 7\rangle, \langle 4 \| 8\rangle\}\!\}$. Deriving the largest possible result intervals yield $\langle 4 \| [1,4]\rangle$ and $\langle 4 \| [5,8]\rangle$, which does not match with the desired result of $\{\!\{\langle 4 \| [1,8]\rangle$. Thus, coalesce is not interval-based. Which is also intuitively correct, since coalesce merges intervals of overlapping and adjacent tuples i.e. it does not respect the grouping of time points into intervals.

4.3 Duplicate-aware Operators

With a temporal data model it is not a priori clear what a duplicate is. We say that a temporal bag contains duplicates iff one of its snapshots contains duplicates.

Definition 7. *A temporal bag R contains* temporal duplicates *iff a tuple t occurs multiple times in at least one of its snapshots.*

$$duplicates(R) = \exists p \in \mathcal{T}^p(t \in^n \tau_p(R) \wedge n > 1)$$

Definition 8. *An operator $\mathcal{O}$ is* duplicate-aware *iff (1) it is sensitive to duplicates and (2) the number of duplicates in each snapshot is consistent with the definition of $\mathcal{O}^p$:*

$$\exists \mathbf{A_1}, \mathbf{A_2}(\forall p \in \mathcal{T}^p(\varepsilon(\tau_p(\mathbf{A_2})) \subseteq \varepsilon(\tau_p(\mathbf{A_1}))) \ \wedge \mathcal{O}^p(\mathbf{A_1}) \neq \mathcal{O}^p(\mathbf{A_1} \uplus \mathbf{A_2})) \wedge \quad (1)$$
$$\forall \mathbf{A_1}, p \in \mathcal{T}^p(\tau_p(\mathcal{O}(\mathbf{A_1})) = \tau_p(\mathcal{O}^p(\mathbf{A_1}))) \quad (2)$$

Intuitively, (1) requires that the result changes if duplicates are added to the argument relations. (2) requires that the number of duplicates returned by $\mathcal{O}$ is correct at each point in time, i.e., consistent with the definition of $\mathcal{O}^p$.

Example 5. Consider the temporal additive union $P_1 \uplus^t Q_1$, let the temporal bags $P_1 = \{\!\{\langle 10 \| [7,9] \rangle\}\!\}$, $P_2 = \{\!\{\langle 10 \| [8,9] \rangle\}\!\}$, and $Q_1 = \{\!\{\langle 10 \| [5,6] \rangle\}\!\}$ be instances of the schema $R(X \| I)$, where $\mathcal{O}^p_{P_1 \uplus^t Q_1} = \{\!\{\langle 10 \| 5 \rangle, \langle 10 \| 6 \rangle, \langle 10 \| 7 \rangle, \langle 10 \| 8 \rangle, \langle 10 \| 9 \rangle\}\!\}$, and $\mathcal{O}^p_{(P_1 \uplus P_2) \uplus^t Q_1} = \{\!\{\langle 10 \| 5 \rangle, \langle 10 \| 6 \rangle, \langle 10 \| 7 \rangle, \langle 10 \| 8 \rangle, \langle 10 \| 8 \rangle, \langle 10 \| 9 \rangle, \langle 10 \| 9 \rangle\}\!\}$. If the results of the temporal additive union respectively are: $P_1 \uplus^t Q_1 = \{\!\{\langle 10 \| [7,9] \rangle, \langle 10 \| [5,6] \rangle\}\!\}$, and $(P_1 \uplus P_2) \uplus^t Q_1 = \{\!\{\langle 10 \| [7,9] \rangle, \langle 10 \| [5,6] \rangle, \langle 10 \| [8,9] \rangle\}\!\}$. Then the operator is duplicate-aware, since this means the temporal additive union is both sensitive to duplicates, and the number of duplicates is consistent with the definition of $\mathcal{O}^p_{P_1 \uplus^t Q_1}$.

5 Grouped Temporal Bags

A *grouped temporal bag* is a temporal data structure, where temporal tuples, which are non-temporally equivalent can be explicitly grouped together in *temporal groups*.

5.1 Structure

A *grouped temporal bag* G has the schema $(X_1, ..., X_n \| \mathbf{TC})$, where X_i is a non-temporal attribute, **TC** is a bag of temporal compounds, and $\|$ separates non-temporal attributes from the temporal compounds. A *temporal compound* TC is a tuple consisting of a time interval and m non-temporal attribute values (m can be 0). Table 3 shows the structure of a grouped temporal bag.The elements, $g \in G$, of a grouped temporal bag G are referred to as *temporal groups*. Note

Table 3. Structure of the Grouped Temporal Bag G

G

X_1 ... X_n	**TC**
$x_{1,1}$ ... $x_{1,n}$	$\{\!\{\langle t_{1,1}, a_{1,1,1}, ..., a_{1,1,m}\rangle, ..., \langle t_{1,y}, a_{1,y,1}, ..., a_{1,y,m}\rangle\}\!\}$
... ...	...
$x_{q,1}$ ... $x_{q,n}$	$\{\!\{\langle t_{q,1}, a_{q,1,1}, ..., a_{q,1,m}\rangle, ..., \langle t_{q,u}, a_{q,u,1}, ..., a_{q,u,m}\rangle\}\!\}$

that the cardinality of a temporal group, $|g.\mathbf{TC}|$, is not necessarily the same for each temporal group.

A grouped temporal bag G is *normalized* (closely related to coalesce for temporal bags cf. Section 4.1) if it does not contain temporally overlapping or adjacent temporal compounds with identical non-temporal attribute values, i.e., $\forall g \in G$ the following must hold:

$$\forall C_1, C_2 \in g.\mathbf{TC}(C_1 \neq C_2 \wedge C_1.\mathbf{X} = C_2.\mathbf{X} \Rightarrow \\ \neg adjc(C_1.I, C_2.I) \wedge \neg ovlp(C_1.I, C_2.I))$$

In the remainder of this paper we exclusively consider normalized grouped temporal bags. Thus, whenever we refer to a grouped temporal bag we always assume a normalized grouped temporal bag.

5.2 Grouping Strategies

A temporal group can model a number of temporal tuples Let g be a temporal group and R be a temporal bag, then g and R are *group equivalent* ($=^g$), iff g models the tuples that are in R.

$$g =^g R \text{ iff} \\ R = \{\!\{\langle g.\mathbf{X}, A_1, ..., A_m \| I\rangle | \langle I, A_1, ..., A_m\rangle \in g.\mathbf{TC}\}\!\}$$

Clearly, a grouped temporal bag can model a temporal bag in several distinct ways. For example, each temporal tuple could be modeled by an individual temporal group or all temporal tuples with the same non-temporal values could be modeled by a single temporal group. The specific strategy that is used to model a temporal bag is called the *grouping* of the grouped temporal bag. Below we introduce *scattered*, *compact*, *composite* and *filtered* groupings. We use the temporal bag R in Table 4 to illustrate the groupings.

Definition 9. *A grouped temporal bag G is a* scattered *grouping of the temporal bag R, iff each temporal group g models exactly one temporal tuple.*

$$group(R, scattered) = G, \text{ iff} \\ \langle \mathbf{X}, \mathbf{Z} \| I\rangle \in R \Leftrightarrow \langle \mathbf{X} \| \{\!\{\langle I, \mathbf{Z}\rangle\}\!\}\rangle \in G$$

Table 4. A temporal bag R

A	B	sum(A)	count(B)	I
10	10	10	1	[5,24]
10	10	20	2	[25,30]
5	4	5	1	[1,4]
5	4	10	2	[5,10]
5	4	10	2	[5,10]

Table 5. A scattered grouping of R

A	B	**TC**
10	10	$\{\!\{\langle [5,24], 10, 1\rangle\}\!\}$
10	10	$\{\!\{\langle [25,30], 20, 2\rangle\}\!\}$
5	4	$\{\!\{\langle [1,4], 5, 1\rangle\}\!\}$
5	4	$\{\!\{\langle [5,10], 10, 2\rangle\}\!\}$
5	4	$\{\!\{\langle [5,10], 10, 2\rangle\}\!\}$

Table 5 shows the scattered grouping of the temporal bag in Table 4.

Definition 10. *A grouped temporal bag G is a* compact *grouping of the temporal bag R, iff all temporal groups are non-temporally distinct, and all tuples with the same non-temporal values as a temporal group are modeled by that group.*

$$
\begin{aligned}
&\mathit{group}(R,\mathit{compact}) = G, \text{ iff}\\
&\qquad \langle \mathbf{X}, \mathbf{Z} \| I\rangle \in R \wedge p \in I \Leftrightarrow \langle \mathbf{X} \| \mathbf{TC}\rangle \in G \wedge \langle I', \mathbf{Z}\rangle \in \mathbf{TC} \wedge p \in I' \wedge\\
&\qquad \forall g_1, g_2 \in G(g_1 \neq g_2 \Rightarrow g_1.\mathbf{X} \neq g_2.\mathbf{X})
\end{aligned}
$$

Table 6 shows the compact representation of the temporal bag in table 4. Notice that a compact grouping is equivalent to coalescing (Remember grouped temporal bags are normalized).

Table 6. A compact grouping of R

A	B	**TC**
10	10	$\{\!\{\langle [5,24], 10, 1\rangle, \langle [25,30], 20, 2\rangle\}\!\}$
5	4	$\{\!\{\langle [1,4], 5, 1\rangle, \langle [5,10], 10, 2\rangle\}\!\}$

Definition 11. *A grouped temporal bag G is a* composite *grouping of a temporal bag R, iff it can be partitioned into a number of compact grouped temporal bags*

G_1, ... G_n, where any fact which i-belongs to a snapshot of R, 1-belongs to the temporal bags G_1, ... , G_i i.e. $G =^g R'$, $R' =^p R$ and $G_1 =^g coal(R)$.

$$group(R,composite) = G_1 \uplus ... \uplus G_n, \text{ iff}$$
$$\langle \mathbf{X}, \mathbf{Z} \rangle \in^i \tau_p(R) \Leftrightarrow \langle \mathbf{X} \| \mathbf{TC} \rangle \in G_i \wedge \langle I, \mathbf{Z} \rangle \in \mathbf{TC} \wedge p \in I \wedge$$
$$\forall g_1, g_2 \in G_i (g_1 \neq g_2 \Rightarrow g_1.\mathbf{X} \neq g_2.\mathbf{X})$$

Table 7 shows the composite grouping of the temporal bag in Table 4. Notice that a composite grouping is equivalent to a duplicate preserving coalesce.

Table 7. A composite grouping of R

A	B	**TC**
10	10	$\{\!\{\langle [5,24], 10, 1\rangle, \langle [25,30], 20, 2\rangle\}\!\}$
5	4	$\{\!\{\langle [1,4], 5, 1\rangle, \langle [5,10], 10, 2\rangle\}\!\}$
5	4	$\{\!\{\langle [5,10], 10, 2\rangle\}\!\}$

Definition 12. *A grouped temporal bag G is a* filtered *grouping of a temporal bag R, iff each temporal group g models exactly one temporal tuple and there are no temporal duplicates.*

$$group(R,filtered) = G, \text{ iff}$$
$$\neg \exists p \in \mathcal{T}^p (t \in^n \tau_p(R) \wedge n > 1) \wedge \langle \mathbf{X}, \mathbf{Z} \| I \rangle \in R \Leftrightarrow \langle \mathbf{X} \| \{\!\{\langle I, \mathbf{Z} \rangle\}\!\} \rangle \in G$$

Table 8 shows the filtered grouping of the temporal bag in Table 4. Note the non-deterministic nature of removing duplicates. If two groups overlap then the overlapping time points are removed from only one of the groups.

Table 8. A filtered grouping of R

A	B	**TC**
10	10	$\{\!\{\langle [5,24], 10, 1\rangle\}\!\}$
10	10	$\{\!\{\langle [25,30], 20, 2\rangle\}\!\}$
5	4	$\{\!\{\langle [1,4], 5, 1\rangle\}\!\}$
5	4	$\{\!\{\langle [5,10], 10, 2\rangle\}\!\}$

6 The Temporal Multi-Dimensional Join

The TMDJ is a simple parameterizable operator, which takes four arguments: A temporal bag D, a grouped temporal bag G, a group operator $\mathcal{O}$, and a condition Θ that references non-temporal attributes of D and G. The condition Θ is evaluated for each temporal tuple of D and each group of G. If a temporal tuple in D and a group in G satisfy the condition, then the group is updated according to the group operator $\mathcal{O}$.

A *group operator* $\mathcal{O}$ is an operator, which takes two operands: A temporal tuple t and a temporal group g, and it returns the temporal group g', where $g'.\mathbf{X} = g.\mathbf{X}$.

Definition 13. *Let D be a temporal bag, G be a grouped temporal bag, $\mathcal{O}$ be a group operator and θ a condition with attributes from D and G.*

$$\boldsymbol{TMDJ}(G, D, \mathcal{O}, \Theta) = \{\!\{g' | g \in G \wedge R = \{\!\{t | t \in D \wedge \Theta(t, g)\}\!\} \wedge g' = Apply(\mathcal{O}, R, g)\}\!\}$$

$$Apply(\mathcal{O}, R, g) = \begin{cases} g & \text{iff } R = \emptyset \\ Apply(\mathcal{O}, R', \mathcal{O}(t, g)) & \text{iff } R = \{\!\{t\}\!\} \uplus R' \end{cases}$$

A key property of the TMDJ is the existence of a simple and efficient evaluation algorithm. The parameters of the algorithm are: The temporal bags R_1 and R_2, the Θ condition, the group operator $\mathcal{O}$, and a grouping parameter.

```
TMDJ Algorithm
        IN: R1, R2, O, Θ, grouping
      Body: Initialize D = R1
            Initialize G = group(R2, grouping)
            For each temporal tuple t of D {
                For each group g of bucket[search-key(t)] {
                    If Θ(t, g) == TRUE Then {
                        g = O(t, g)
                } } }
            Return All temporal tuples in G
```

The first step is the initialization of the grouped temporal bag G as a grouping of R_2 (note, when initializing G attributes which appear in the Θ condition should not appear in a temporal compound). The initialization includes the construction of a hash index for the temporal groups in G. All groups with an identical search-key are hashed to the same bucket. The search-key is the summation of the binary representations of the non-temporal attributes. In the main loop each tuple of D is applied to all qualifying groups.

6.1 Semantics

We formalized point-based, interval-based and duplicate-aware semantics in Section 4. In Section 5 we introduced four grouping strategies: scattered, compact,

filtered and composite. In this section we specify how each grouping strategy determines the temporal semantics of the TMDJ.

The basic semantics of the TMDJ are determined by the specific group operator, which is applied to the groups in the grouped temporal bag G. The temporal semantics of the TMDJ, however, are determined by both the group operator and the grouping strategy, where the grouping strategy decides the grouping of time points and the group operator decides how they are processed.

Conceptually the TMDJ performs the temporal operation $Apply$ on each group g of the grouped temporal bag G, where $Apply$ is defined by a subset of the temporal bag D and a group operator $\mathcal{O}$. The result of the TMDJ is a temporal bag of tuples R, which consists of all the tuples modeled by each temporal group g'_i.

$$R = R_1 \uplus ... \uplus R_n, \text{ where } R_i =^g g'_i$$

This means that each bag of tuples R_i is derived from a temporal group g_i, and the temporal semantics of the deriving operation depends on how the time points are initially grouped into g_i i.e. the grouping of the grouped temporal bag G:

Scattered (ib, da): Each temporal group g initially corresponds to exactly one argument tuple t, $g =^g \{\!\{t\}\!\}$. Thus, all tuples derived from g are derived from the minimal subset t, and normalizing ensures that all derived intervals are maximal i.e. interval-based semantics. Additionally, since duplicates are grouped separately they are processed independently, which means the multiplicity of a fact is bound by the multiplicity of the fact from which it is derived from i.e. duplicate-aware semantics.

Filtered (ib): A filtered grouping is equivalent to a scattered grouping, except it does not recognize temporal duplicates i.e. this grouping yields interval-based semantics.

Compact (pb): A compact grouping of a temporal bag R is equal to coalescing R i.e. $G =^g coal(R)$. From Lemma 2 we know that this gives us point-based semantics.

Composite (pb, da): A composite grouping of a temporal bag R defines a point-based normal form similar to coalescing, except it deals correctly with temporal duplicates:

$$R_1 =^p R_2 \Leftrightarrow group(R_1, composite) = group(R_2, composite) \\ \wedge\, group(R_1, composite) =^g R'_1 \wedge R'_1 =^p R_1$$

This grouping yields point-based and duplicate-aware semantics.

7 Temporal Operators

The TMDJ can be used to implement a wide range of temporal operators. We have used it to implement temporal aggregation and temporal difference, as these

are difficult to implement using current database technology and are often not even supported.

We use the following auxiliary interval operations: $-^L$ returns the left interval of an interval subtraction, $-^R$ returns the right interval of an interval subtraction, and $\cap$ returns the intersection of two intervals.

$$\begin{array}{lll} I_1 -^L I_2 = [I_1^-, min(I_1^+, pred(I_2^-))] & \text{if } I_1^- < I_2^- \\ I_1 -^R I_2 = [max(I_1^-, succ(J^+)), I_1^+] & \text{if } I_2^+ < I_1^+ \\ I_1 \cap I_2 = [max(I_1^+, I_2^+), min(I_1^-, I_2^-)] & \text{if } ovlp(I_1, I_2) \end{array}$$

If the condition on the right is not satisfied the respective operator does not return a result interval.

7.1 Temporal Difference

The temporal difference $P -^t Q$ can be expressed as a TMDJ, where the grouped temporal bag G is a grouping of P, the temporal bag D is equal to Q, the Θ condition is non-temporal equivalence, $D.\mathbf{X} = G.\mathbf{X}$, and the group operator $\mathcal{O}$ is *subtract*.

$$P -^t Q = TMDJ(P, Q, subtract, D.\mathbf{X} = G.\mathbf{X})$$

Definition 14. *The group operator* subtract *removes time points that are in the temporal tuple t from the time points that are in the temporal group g.*

$$\begin{array}{l} subtract(t, g) = g', \text{ iff } g'.\mathbf{X} = g.\mathbf{X} \\ \qquad \wedge\, g'.\mathbf{TC} = \{\!\{I | C \in g.\boldsymbol{TC} \wedge I \in \{C.I -^L t.I, C.I -^R, t.I\} \wedge I \neq \emptyset\}\!\} \end{array}$$

Example 6. Consider applying subtract to the tuple $t = \langle 7 \| \ [5, 35] \ \rangle$ and the temporal group $g_1 = \langle \ 7 \ \| \{\!\{ \ \langle [3, 10] \rangle, \langle [15, 25] \rangle, \langle [30, 40] \rangle \ \}\!\} \ \rangle$, and subsequently to the temporal group $g_2 = \langle \ 7 \ \| \{\!\{ \ \langle [1, 40] \rangle, \langle [50, 60] \rangle \ \}\!\} \ \rangle$

$$\begin{array}{l} subtract(t, g_1) = \langle 7 \| \{\!\{ \langle [3, 4] \rangle, \langle [36, 40] \rangle \}\!\} \rangle \\ subtract(t, g_2) = \langle 7 \| \{\!\{ \langle [1, 4] \rangle, \langle [36, 40] \rangle, \langle [50, 60] \rangle \}\!\} \rangle \end{array}$$

Example 7. To illustrate the temporal difference $P -^t Q$ we use the temporal bags P and Q in Table 9. The first step is to initialize the temporal bag D as Q, and the grouped temporal bag G as P grouped respectively scattered, filtered, compact or composite as illustrated in Table 10. Subsequent to the initialization each tuple of D is processed tuple-by-tuple, if a group of G satisfies the Θ condition ($G.\mathbf{X} = D.\mathbf{X}$) with regard to the tuple currently being processed, then the tuple is subtracted from the group, as illustrated for each grouping in Table 11 (where qualifying groups are marked by $\Rightarrow$).

Table 9. Temporal bags P and Q

P

A	I
10	[1,10]
10	[11,20]
5	[21,40]
5	[21,40]

Q

A	I
10	[1,5]
10	[15,20]
5	[21,25]
5	[35,40]

Table 10. G grouped as P

A	**TC**
10	$\{\!\{\langle [1,10]\rangle\}\!\}$
10	$\{\!\{\langle [11,20]\rangle\}\!\}$
5	$\{\!\{\langle [21,40]\rangle\}\!\}$
5	$\{\!\{\langle [21,40]\rangle\}\!\}$

Scattered

A	**TC**
10	$\{\!\{\langle [1,10]\rangle\}\!\}$
10	$\{\!\{\langle [11,20]\rangle\}\!\}$
5	$\{\!\{\langle [21,30]\rangle\}\!\}$

Filtered

A	**TC**
10	$\{\!\{\langle [1,20]\rangle\}\!\}$
5	$\{\!\{\langle [21,40]\rangle\}\!\}$

Compact

A	**TC**
10	$\{\!\{\langle [1,20]\rangle\}\!\}$
5	$\{\!\{\langle [21,40]\rangle\}\!\}$
5	$\{\!\{\langle [21,40]\rangle\}\!\}$

Composite

7.2 Temporal Aggregation

The temporal aggregation ${}_{G_1,...G_m}\mathcal{G}_{f_1(A_1),...f_n(A_n)}(P)$ can be expressed as a TMDJ, where the grouped temporal bag G is a grouping of P, the temporal bag D is equal to P, the Θ condition is non-temporal equivalence, and the group operator $\mathcal{O}$ is *split*.

$$
\begin{gathered}
{}_{G_1,...G_m}\mathcal{G}_{f_1(A_1),...f_n(A_n)}(P) = \\
TMDJ(P, P, split(f_1(A_1), ...f_n(A_n)), D.\mathbf{X} = G.\mathbf{X})
\end{gathered}
$$

Definition 15. *The group operator* split *splits the time points of the temporal group g into a set containing the time points of g, that are also in the temporal tuple t, and a set which contains the time points of g that are not in t. Additionally the aggregate values of the first set are updated according to the set of aggregate functions $f_1, ..., f_n$.*

$$
\begin{aligned}
split(t, g, f_1(A_1), ..., f_n(A_n)) = g',\ \text{iff}\ g'.\boldsymbol{X} = g.\boldsymbol{X} \wedge g'.\boldsymbol{TC} = \\
\{\!\{\langle I \| f_1(C.A_1, t), ..., f_n(C.A_n, t)\rangle | C \in g.\boldsymbol{TC} \wedge \\
I \in \{C.I -^L t.I, C.I -^R t.I\} \wedge I \neq \emptyset\}\!\} \\
\uplus \{\!\{\langle I \| C.A_1, ..., C.A_n\rangle | C \in g.\boldsymbol{TC} \wedge I = C.I \cap t.I \wedge I \neq \emptyset\}\!\}
\end{aligned}
$$

Example 8. Consider applying split to the tuple $t = \langle 10, 5 \| [20, 30] \rangle$ from a temporal bag with the schema $(A, B \| I)$ and the temporal group $g = \langle 10 \| \{\!\{ \langle [1, 40] \rangle \}\!\} \rangle$, and counting the attribute B.

$$
split(t, g, count(B)) = \langle 10 \| \{\!\{\langle [1,19]\rangle, \langle [20,30], 1\rangle, \langle [31,40]\rangle\}\!\}\rangle
$$

Example 9. To illustrate the temporal aggregation ${}_A\mathcal{G}_{count(B)}(P)$ we use the temporal bag P in Table 12. The first step is to initialize the temporal bag D as P, and the grouped temporal bag G as P grouped respectively scattered, filtered,

Table 11. Temporal difference.

	A	TC
⇒	10	{{⟨[6, 10]⟩}}
⇒	10	{{⟨[11, 20]⟩}}
	5	{{⟨[21, 40]⟩}}
	5	{{⟨[21, 40]⟩}}

	A	TC
⇒	10	{{⟨[6, 10]⟩}}
⇒	10	{{⟨[11, 14]⟩}}
	5	{{⟨[21, 40]⟩}}
	5	{{⟨[21, 40]⟩}}

	A	TC
	10	{{⟨[6, 10]⟩}}
	10	{{⟨[11, 14]⟩}}
⇒	5	{{⟨[26, 40]⟩}}
⇒	5	{{⟨[26, 40]⟩}}

	A	TC
	10	{{⟨[6, 10]⟩}}
	10	{{⟨[11, 14]⟩}}
⇒	5	{{⟨[26, 34]⟩}}
⇒	5	{{⟨[26, 34]⟩}}

(a) Scattered grouping.

	A	TC
⇒	10	{{⟨[6, 10]⟩}}
⇒	10	{{⟨[11, 20]⟩}}
	5	{{⟨[21, 40]⟩}}

	A	TC
⇒	10	{{⟨[6, 10]⟩}}
⇒	10	{{⟨[11, 14]⟩}}
	5	{{⟨[21, 40]⟩}}

	A	TC
	10	{{⟨[6, 10]⟩}}
	10	{{⟨[11, 14]⟩}}
⇒	5	{{⟨[26, 40]⟩}}

	A	TC
	10	{{⟨[6, 10]⟩}}
	10	{{⟨[11, 14]⟩}}
⇒	5	{{⟨[26, 34]⟩}}

(b) Filtered grouping.

	A	TC
⇒	10	{{⟨[6, 20]⟩}}
	5	{{⟨[21, 40]⟩}}

	A	TC
⇒	10	{{⟨[6, 14]⟩}}
	5	{{⟨[21, 40]⟩}}

	A	TC
	10	{{⟨[6, 14]⟩}}
⇒	5	{{⟨[26, 40]⟩}}

	A	TC
	10	{{⟨[6, 14]⟩}}
⇒	5	{{⟨[26, 34]⟩}}

(c) Compact grouping

	A	TC
⇒	10	{{⟨[6, 20]⟩}}
	5	{{⟨[21, 40]⟩}}
	5	{{⟨[21, 40]⟩}}

	A	TC
⇒	10	{{⟨[6, 14]⟩}}
	5	{{⟨[21, 40]⟩}}
	5	{{⟨[21, 40]⟩}}

	A	TC
	10	{{⟨[6, 14]⟩}}
⇒	5	{{⟨[26, 40]⟩}}
⇒	5	{{⟨[26, 40]⟩}}

	A	TC
	10	{{⟨[6, 14]⟩}}
⇒	5	{{⟨[26, 34]⟩}}
⇒	5	{{⟨[26, 34]⟩}}

(d) Composite grouping

Table 12. Temporal bags P

P

A	B	I
10	10	[1,5]
10	5	[6,20]
5	10	[1,30]
5	5	[1,20]

Table 13. G grouped as P

A	**TC**
10	$\{\!\{\langle [1,5], \emptyset\rangle\}\!\}$
10	$\{\!\{\langle [6,20], \emptyset\rangle\}\!\}$
5	$\{\!\{\langle [1,30], \emptyset\rangle\}\!\}$
5	$\{\!\{\langle [1,20], \emptyset\rangle\}\!\}$

Scattered

A	**TC**
10	$\{\!\{\langle [1,5], \emptyset\rangle\}\!\}$
10	$\{\!\{\langle [6,20], \emptyset\rangle\}\!\}$
5	$\{\!\{\langle [1,30], \emptyset\rangle\}\!\}$

Filtered

A	**TC**
10	$\{\!\{\langle [1,20], \emptyset\rangle\}\!\}$
5	$\{\!\{\langle [1,30], \emptyset\rangle\}\!\}$

Compact

A	**TC**
10	$\{\!\{\langle [1,20], \emptyset\rangle\}\!\}$
5	$\{\!\{\langle [1,30], \emptyset\rangle\}\!\}$
5	$\{\!\{\langle [1,20], \emptyset\rangle\}\!\}$

Composite

compact or composite as illustrated in Table 13. Subsequent to the initialization each tuple of D is processed tuple-by-tuple, if a group of G satisfies the Θ condition ($G.\mathbf{X} = D.\mathbf{X}$) with regard to the tuple currently being processed, then the group is *split*, as illustrated for each grouping in Table 11 (where qualifying groups are marked by $\Rightarrow$).

8 Performance Evaluation

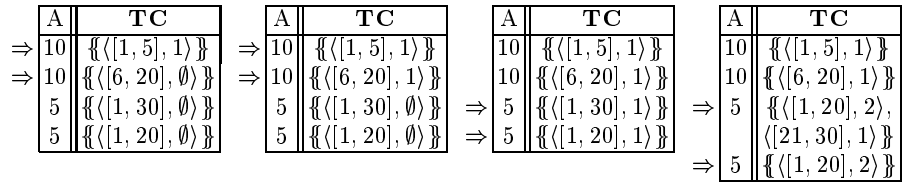
In this section we report the results of three test sets, where we measure the performance of a TMDJ implementation of temporal difference, temporal aggregation, and the initial grouping of the grouped temporal bag.

We use two test databases: One consisting of non-temporally distinct tuples, and one consisting of a chain of overlapping tuples. In the first set of tests we measure the performance of the TMDJ on the non-temporally distinct tuples, in the second test set we measure the performance of the TMDJ on the chain of overlapping tuples, and finally in the third test set we compare the performance of the TMDJ implementation of temporal difference and coalesce with equivalent SQL solutions, where coalesce is simply a compact grouping.

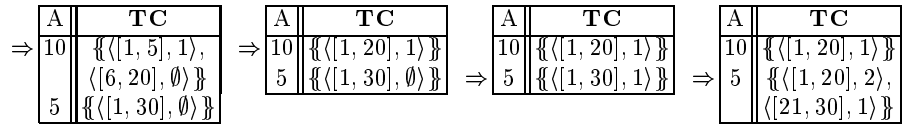
8.1 Implementation

We implemented a lightweight version of the TMDJ evaluation algorithm on top of Oracle9i with a few simple optimizations. The hash index is implemented as an array of buckets, where each bucket is implemented as a linked list to prevent bucket overflows. Each temporal group of the grouped temporal bag G is implemented as two linked lists: One list for the non-temporal attributes, and one list for the temporal compounds. This allows us to quickly determine if the Θ condition is satisfied, and subsequently flexible manipulation of the temporal compounds as required by a group operator.

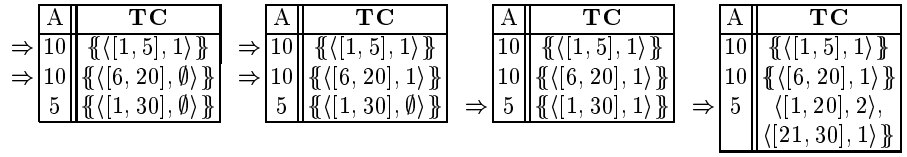
Table 14. Temporal aggregation.



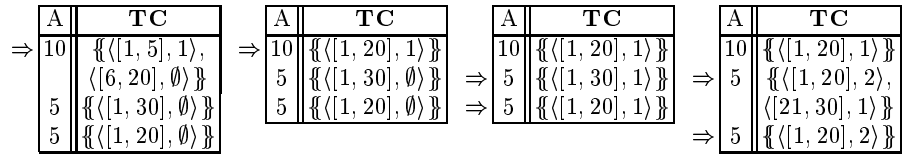
(a) Scattered grouping.



(b) Compact grouping.



(c) Filtered grouping.

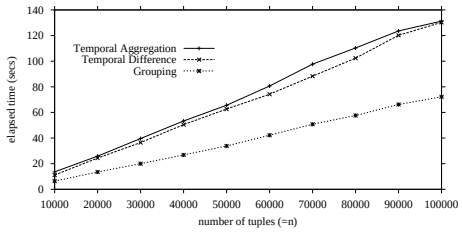


(d) Composite grouping.

8.2 Experimental Results

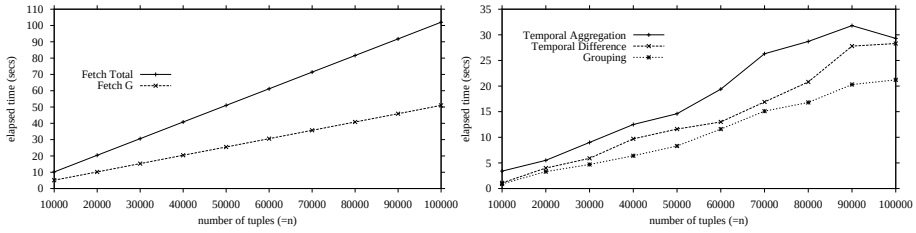
Test Set #1 (Non-temporally Distinct Tuples): In the first set of tests we measured the performance of the TMDJ on a test database which contained all non-temporally distinct tuples. This test provides a performance reference point, since the grouped temporal bag is identical for all groupings, and the hash index should work perfectly.

Fig. 1. Reference point.



The results were almost completely identical for all groupings, and a reference point for temporal difference, temporal aggregation, and the initial grouping of the grouped temporal bag is illustrated in Figure 1. The result includes the time it takes to fetch the argument tuples, as illustrated in Figure 2 (left) the amount of time spent fetching takes up quite a large percentage of the total performance cost (up to 95%!). Excluding the fetch time from the reference point yields results around 20 to 30 seconds of processing time for 100.000 tuples as illustrated in Figure 2. This also shows that the time it takes to create the initial grouping, and to compute both temporal difference and temporal aggregation is almost identical within a few seconds of each other. This is interesting since the initial grouping is little more than a scan of the temporal bag which is used to initialize the grouped temporal bag.

Fig. 2. Fetch (left) and reference point for all groupings excluding fetch (right).



Test Set #2 (Overlapping Tuples): In the second set of tests we measured the performance of the TMDJ on a test database, which contained only overlapping tuples. The results for the initial grouping of the grouped temporal bag are illustrated in Figure 3.

The results show that compact and scattered groupings perform best and quite close to the reference point, while composite and filtered groupings perform far worse than the reference point at a cost approximately 5 times the reference. This may be explained by the fact that composite and filtered groupings are variations of scattered and compact, which require special attention to temporal duplicates. However, it is interesting to note that the special attention goes in opposite directions i.e. composite preserves duplicates, where filtered removes duplicates.

Fig. 3. Grouping initialization for overlapping tuples.

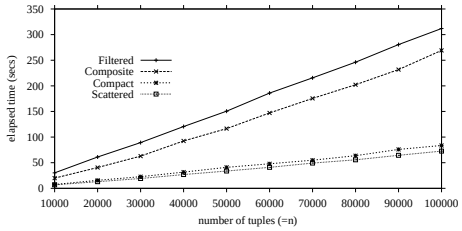
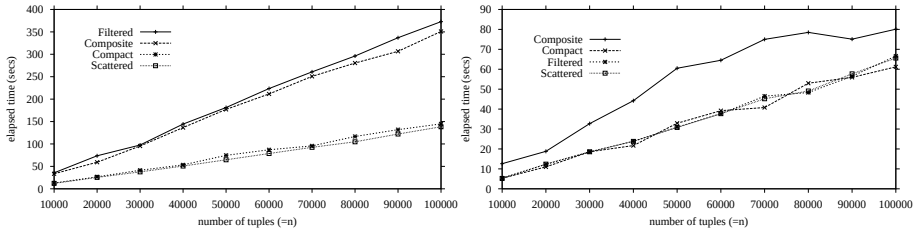


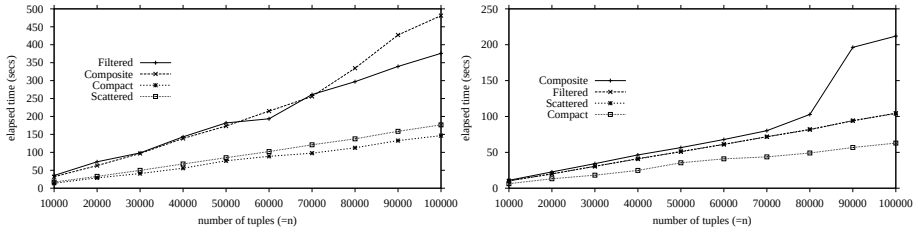
Fig. 4. Temporal aggregation results for overlapping tuples, including (left) and excluding (right) grouping initialization.



The test results for temporal aggregation on overlapping tuples are illustrated in Figure 4, including (left) and excluding (right) the grouping initialization. The test results for temporal difference on overlapping tuples are illustrated in Figure 5, including (left) and excluding (right) the grouping initialization.

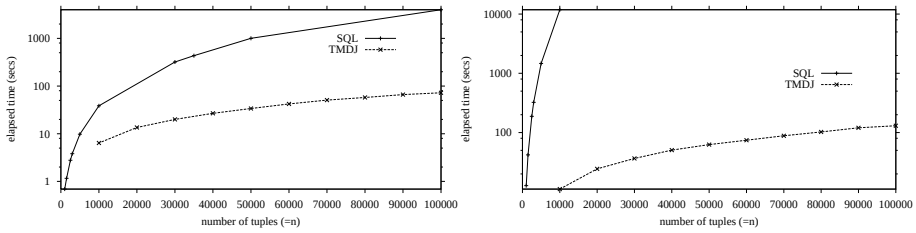
The results for temporal difference and temporal aggregation are very similar, as previously compact and scattered perform best. However, if we exclude the grouping time we get a slightly different view of the performance. For temporal aggregation we see scattered, filtered and compact groupings perform the same, while the composite grouping is quite expensive compared with the others. This result is similar for temporal difference where the composite grouping deterio-

Fig. 5. Temporal difference results for overlapping tuples, including (left) and excluding (right) grouping initialization.



rates at approximately 80.000 tuples. However, the compact grouping performs significantly better than both the scattered and filtered grouping. The reason that the composite grouping performs worse may be because it is effectively dealing with all the temporal duplicates of all the overlapping test tuples. While the reverse holds for the compact grouping, which is effectively dealing with a lot less tuples than the other groupings.

Fig. 6. Coalesce (left) and temporal difference (right) as performed by the TMDJ and equivalent SQL solutions.



Test Set #3 (SQL): In the third set of tests we compared temporal difference and coalesce as performed by the TMDJ with equivalent SQL solutions. The results are summarized in Figure 6. The SQL solutions quickly become impracticable as illustrated for temporal difference at 10.000 tuples, which takes the SQL solution 10.000 seconds while it takes approximately 10 seconds for the equivalent TMDJ.

8.3 Evaluation

The test results show that the TMDJ overall performs at a linear cost, and a high percentage of this cost is spent fetching tuples. It is likely that integrating

the TMDJ into the underlying DBMS would provide significant performance improvement.

The tests also showed that introducing temporal duplicates into the argument bags lowers the performance. Specifically temporal duplicates influence the performance of composite and filtered groupings, where composite preserves the temporal duplicates and filtered removes temporal duplicates compared respectively with compact and scattered groupings. With regards to the semantics this means point-based semantics perform at a cost similar to semantics which are both interval-based and duplicate-aware, while interval-based semantics perform at a cost near the cost of semantics, which are both point-based and duplicate-aware. Thus, if we want point-based semantics it is expensive to also have duplicate-aware semantics, where if we want interval-based semantics it is inexpensive to have duplicate-aware semantics.

Overall the test results show that temporal difference and temporal aggregation can very elegantly be reduced to a TMDJ, which exhibits a linear performance, and is orders of magnitude better than equivalent SQL solutions.

9 Conclusion and Future Work

In this paper we identified and formalized point-based, interval-based and duplicate-aware semantics. Point-based operators are defined as operators, where the time points in the result is independent of how time points are grouped in the argument bags. Interval-based operators are defined as operators, which respect and preserve the interval grouping of time points. Duplicate-aware operators are defined as operators, which are sensitive to temporal duplicates in the argument, and yield results with a clearly defined number of temporal duplicates.

Next, we formalized the temporal multi-dimensional join (TMDJ), and the grouped temporal bag, the core data structure of the TMDJ. Then we specified how grouping time points in the grouped temporal bag determines the temporal semantics of the TMDJ: Grouping all time points of a fact together yields point-based semantics, grouping duplicate time points separately yields duplicate-aware semantics, and finally grouping time points of a fact according to the timestamps yields interval-based semantics.

Finally, we studied the performance of the TMDJ in a series of tests, which concluded that the main performance issue is how to process temporal duplicates depending on the desired semantics. Where preserving duplicate is expensive for point-based semantics, and removing duplicates is expensive for interval-based semantics. Additionally, tests showed that the performance of the TMDJ is orders of magnitude better than equivalent SQL solutions for temporal difference and coalesce.

Future work includes a further formalization of the parameters to provide an orthogonal and complete framework for determining the semantic properties. Several other research directions may also prove to be interesting, such as the role of the TMDJ in complex temporal OLAP queries.

Acknowledgement I would like to thank my advisor Michael H. Böhlen for his support and guidance throughout the preparation of this paper.

References

[AB03] Michael O. Akinde and Michael H. Böhlen. Efficient computation of subqueries in complex OLAP. ICDE, 2003.

[Ari86] Gad Ariav. A temporally oriented data model. *ACM Transactions on Database Systems (TODS)*, 11(4):499–527, 1986.

[BBJ98] Michael H. Böhlen, R. Busatto, and Christian S. Jensen. Point-versus interval-based temporal data models. In *Proceedings of the Fourteenth International Conference on Data Engineering, February 23-27, 1998, Orlando, Florida, USA*, pages 192–200. IEEE Computer Society, 1998.

[BJ96] Michael H. Böhlen and Christian S. Jensen. Seamless integration of time into sql. University of Aalborg, 1996.

[BJ03] Michael H. Böhlen and Christian S. Jensen. The Tiger Prototype System in architecture and implementation of spatio-temporal DBMS (chapter 7). In *Spatiotemporal Databases: The Chorochronos Approach*, To appear in 2003.

[BSS96] M. Böhlen, R. Snodgrass, and M. Soo. Coalescing in temporal databases. In *In International Conference On Very Large Data Bases*, pages 180–191, 1996.

[GM93] Stéphane Grumbach and Tova Milo. Towards tractable algebras for bags. pages 49–58, 1993.

[LM97] N. A. Lorentzos and Y. G. Mitsopoulos. Sql exetension for interval data. *IEEE TKDE*, 9(1):480–490, May 1997.

[MAK01] Theodore Johnson Michael Akinde, Damianos Chatziantoniou and Samuel Kim. The MD-join: An operator for complex OLAP. 2209:52–67, 2001.

[NA89] S. B. Navathe and R. Ahmed. A temporal relational model and a query language. *Information Sciences: an International Journal*, 49(1-3):147–175, 1989.

[PJ99] Dieter Pfoser and Christian S. Jensen. Incremental join of time-oriented data. In *Statistical and Scientific Database Management*, pages 232–243, 1999.

[SKS96] A. Silberschatz, H. F. Korth, and S. Sudarshan. *Database System Concepts*. McGraw-Hill, 3rd edition, 1996.

[Sli01] Giedrius Slivinskas. A middleware approach to temporal query processing, pd.d. thesis, 2001.

[Sno95] R. Snodgrass. The TSQL2 temporal query language. 1995.

[Sno96] R. Snodgrass. The temporal query language tquel. *ACM Transactions on Database Systems*, 12(2):247–298, June 1996.

[Sno99] R. T. Snodgrass. *Developing Time-Oriented Database Applications in SQL*. Morgan Kaufmann, San Francisco, CA, 1999.

[TCG+93] A. Tansel, J. Clifford, S. Gadia, S. Jajodia, A. Segev, and R. T. Snodgrass. *Temporal Databases: Theory, Design, and Implementation*. Benjamin/-Cummings Publishing Company, Inc., Redwood City, California, 1993.

[YW01] Jun Yang and Jennifer Widom. Incremental computation and maintenance of temporal aggregates. In *ICDE*, pages 51–60, 2001.