Evaluating Temporal Graphs built from Texts via Transitive Reduction

Abstract

Temporal information has been the focus of recent attention in information extraction, leading to some standardization effort, in particular for the task of relating events in a text. Part of this effort addresses the ability to compare two annotations of a given text, while relations between events in a story are intrinsically interdependent and cannot be evaluated separately. A proper evaluation measure is also crucial in the context of a machine learning approach to the problem. Finding a common comparison referent at the text level is not an obvious endeavour, and we argue here in favor of a shift from event-based measures to measures on a unique textual object, a minimal underlying temporal graph, or more formally the transitive reduction of the graph of relations between event boundaries. We support it by an investigation of its properties on synthetic data and on a well-know temporal corpus.

1 Introduction

Temporal processing of texts is a somewhat recent field from a methodological point of view, even though temporal semantics has a long tradition, dating back at least to the 1940's [15]. While theoretical and formal linguistic approaches to temporal interpretation have been very active in the 1990s, empirical approaches were less frequent, and very few natural language processing systems were evaluated beyond a few instances.

Temporal information being essential to the interpretation of a text and thus crucial in applications such as summarization or information extraction, it has received growing attention in the 2000s [9] and has lead to some standardization effort through the TimeML initiative [17]. We address here a central part in this task, namely the extraction of the network of temporal relations between events described in a text. Since temporal information is not easily broken down into local bits of information, there are many equivalent ways to express the same ordering of events. Human annotation is thus notoriously difficult [19] and comparisons between annotations cannot rely on simple precision/recall-type measures. The given practice nowadays has been to compute some sort of transitive closure over the network/graph of constraints on temporal events (usually expressed in the well-known Allen algebra [2], or a sub-algebra), and then either compare the sets of simple temporal relations that are deduced from it, or measure the agreement between the whole graphs, including disjunctions of information [23]. This reasoning model is also used to help the building of representations of

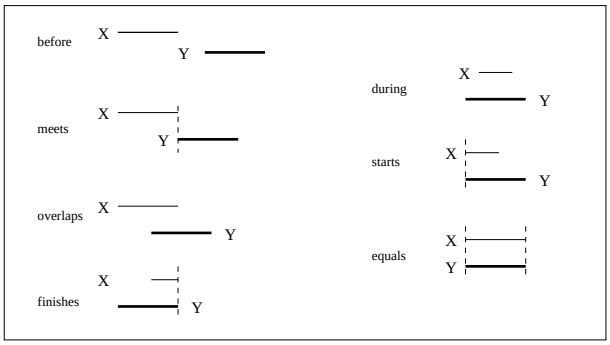


Figure 1: Allen relations. Each relation r has an inverse relation ri.

temporal situations by imposing global constraints on top of local decision problems [4, 22, 3].

We purpose to take a different route here, by extracting a single referent graph, a *minimal graph* of constraints. There are a number of ways of doing this and we argue for going after the graph of relations between event boundaries. We aim to accomplish two things by doing so: to find a graph that is easy to compute, and to eliminate a bias introduced by measures that do not take into account the combinatorial aspect of agreement on transitive closure graphs.

The next section presents in more detail the usual way of comparing annotation graphs between temporal entities extracted from a text, and the problems it raises. Then we argue for comparing event boundaries instead of events and define two new metrics that apply to that type of information. We focus on convex relations, a tractable sub-algebra of Allen relations, which covers human annotations. Finally, we present an empirical study of the behavior of these measures on generated data and on the TimeBank Corpus [14] to support our claim of the practicality of this methodology.

2 Comparing temporal constraint networks

Works on temporal annotation of texts strongly rely on Allen's interval algebra. Allen represents time and events as intervals, and states that 13 basic relations can hold between these intervals (see Figure 1 and Table 1). These binary relations, existing amongst all intervals of a collection (in our case, of a text), define a graph where nodes are the intervals and where edges are labeled with the set of relations which may hold between a pair of nodes.

We are interested in this paper in evaluating systems annotating texts by temporal relations holding between events or between temporal expressions and events. Evaluations are often not performed on graphs of relations between all events in a text, but on the subproblem of ordering pairs of successively described events [10, 23] or even same-sentence events [8][1]. The main reason of this choice is the difficulty of the task,

[1]Exceptions exist, as [11] and [12].

Relation	Meaning	Endpoint relations	Inverse relation
$I < J$	I before J	$I_2 < J_1$	$I > J$
$I\ m\ J$	I meets J	$I_2 = J_1$	$I\ mi\ J$
$I\ o\ J$	I overlaps J	$I_1 < J_1 \wedge I_2 < J_2 \wedge J_1 < I_2$	$I\ oi\ J$
$I\ s\ J$	I starts J	$I_1 = J_1 \wedge I_2 < J_2$	$I\ si\ J$
$I\ d\ J$	I during J	$J_1 < I_1 \wedge I_2 < J_2$	$I\ di\ J$
$I\ f\ J$	I finishes J	$J_1 < I_1 \wedge I_2 = J_2$	$I\ fi\ J$
$I = J$	I equals J	$I_1 = J_1 \wedge I_2 = J_2$	

Table 1: Allen relations. Each relation r has an inverse relation ri. An interval I starts at I_1 and ends at I_2.

even for human beings, of assigning temporal relations in a large text [19]. Another issue is that evaluation of full temporal graphs is still an open question, as will be further discussed in this section.

We detail now important notions concerning temporal networks and the comparison of these networks. All example relations given in this section are expressed in terms of Allen algebra, whose set of relations and their abbreviations are recalled in Table 1.

2.1 Temporal Closure

Temporal closure is an inferential closure mechanism that consists in composing known pairs of temporal relations in order to obtain new relations, up to a fixed point. *E.g.*: if $A < B$ and $C\,d\,B$, then $A < C$; the transition can lead to a disjunction of relations, for example if $A < B$ and $B\,d\,C$ then $A < C \vee A\,o\,C \vee A\,m\,C \vee A\,d\,C \vee A\,s\,C$.

A table of all composition rules in Allen algebra can be found in [2] or [16], and a sample for a few basic relations is given in Table 2. These new relations do not express new intrinsic constraints, but make the temporal situation more explicit. A constraint propagation algorithm ensures that all existing temporal relations are added to the network, labelling an inconsistency with $\emptyset$ [2]. This algorithm is sound, but not complete, as it does not detect all cases of inconsistency. See the simple version presented in Algorithm 1. More efficient versions for large, dense graphs have also been developped [25] (and we use one of them), but it is not our main focus here.

It is not possible to compare temporal graphs without performing a temporal closure on them. Indeed, there are several ways to encode the same temporal information in a graph, as shown in Figure 2. Only temporal closure makes explicit what is implicit and shows that two graphs are identical or different. But temporal closure also produces redundant information, which can lead to evaluation issues, as will be explained in Section 2.4.

In this paper, we call G^* the temporal closure of a graph G.

2.2 Time point algebra and convex relations

Interval graphs can be easily converted into graphs between points [25] (where an event is split into a beginning and an ending point; the mapping between Allen relations and

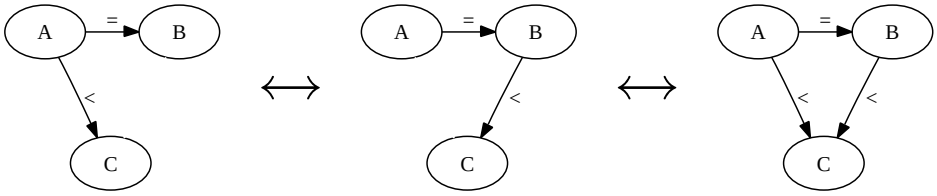


Figure 2: Three identical annotations. The last one is the result of temporal closure.

$\nearrow$	$<$	$>$	d	di
$<$	$<$	all	$<$ o m d s	$<$
$>$	all	$>$	$>$ oi mi d f	$>$
d	$<$	$>$	d	all
di	$<$ o m di fi	$>$ oi di mi si	o oi d s f di si fi =	di

Table 2: Composition between a few Allen relations.

Algorithm 1 Temporal closure

Let U = the disjunction of all 13 Allen relations,
$R_{m,n}$ = the current relation between nodes m and n

```
procedure CLOSURE(G)
    A=G.edges()
    N=G.vertices()
    changed = True
    while changed do
        changed = False
        for all pairs of nodes (i, j) ∈ N × N do
            for all k ∈ N such that ((i, k) ∈ A ∧ (k, j) ∈ A) do
                R1i,j = (Ri,k ∘ Rk,j)
                if no edge (a relation R2i,j) existed before between i and j then R2i,j = U
                end if
                Ri,j = R1i,j ∩ R2i,j                    ▷ intersect
                if Ri,j = ∅ then error                  ▷ inconsistency detected
                else if Ri,j = U then do nothing        ▷ no new information
                else
                    update edge (i,j)
                    changed = True
                end if
            end for
        end for
    end while
end procedure
```

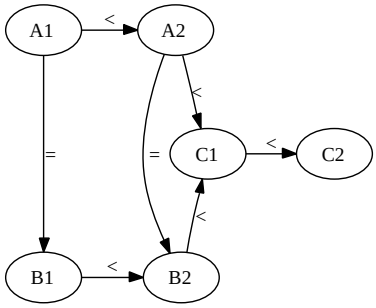


Figure 3: Endpoint graph (same temporal information as Figure 2).

point relations is given by Table 1). This leads to a smaller set of simple relations: equality (=) and precedence (< and >), and a simpler algebra, with only 7 consistent vectors ($\{<\}, \{<, =\}, \{=\}, \{<, =, >\}, \{>\}, \{>=\}, \{<, >\}$ where a set denotes a disjunction of relations). The point algebra is defined by the four relations r_1 to r_4 between beginning and end points of the two intervals I and J ($I_1\ r_1\ J_1$, $I_2\ r_2\ J_2$, $I_1\ r_3\ J_2$ and $I_2\ r_4\ J_1$). Converting an Allen graph into an endpoint graph is straightforward, see the correspondance in Table 1. Figure 3 shows a point graph equivalent to the interval graph of Figure 2.

As with the interval algebra, pairs of point relations can be combined and the temporal closure can be computed in the same way. The relation between two time points is *continuous* if the assigned set of simple relations is convex [25].

A so-called *convex relation* corresponds to cases where relations r_1 to r_4 are assigned to one of the 6 possible relations $\{<\}, \{<, =\}, \{=\}, \{<, =, >\}, \{>\}, \{>=\}$[2], considered as *conceptual neighbors* [6]. Using only these relations between endpoints restricts interval relations to sets of Allen relations that are *conceptual neighbors*, that is they encode relations that may be vague but in which intervals endpoints can only be in convex subsets of the timeline. Figure 4 shows which Allen relations are conceptual neighbors. Another useful way of seeing these conceptual neighbors is by considering continuous transformations of an interval endpoints on the timeline: when a relation r holds between two intervals I_1 and I_2, moving continuously their endpoints can only change the relation to a conceptual neighbor of r, for instance such a conceptual transformation cannot change a situation where I_1 starts I_2 to a situation where $I_1 < I_2$ without going through (at least) intermediary situations $I_1 \text{overlaps} I2$ and I_1 meets I_2.

Finally, instead of 2^{13} possible disjunctive relations in Allen algebra, the set of corresponding interval convex relations is reduced to 82. The corresponding sub-algebra is tractable, that is the problem of the satisfiability of a set of constraints has a sound and complete polynomial time algorithm. Moreover, it will ensure the uniqueness of minimal graphs, as will be defined and described in this paper.

See [18] for a more complete presentation within a natural language processing

[2]The 7 relations described above, except $\{<, >\}$, noted also $\neq$. These form a sub-algebra: compositions of disjunctions of these relations are disjunctions of these relations. Before OR after is not part of this sub-algebra.

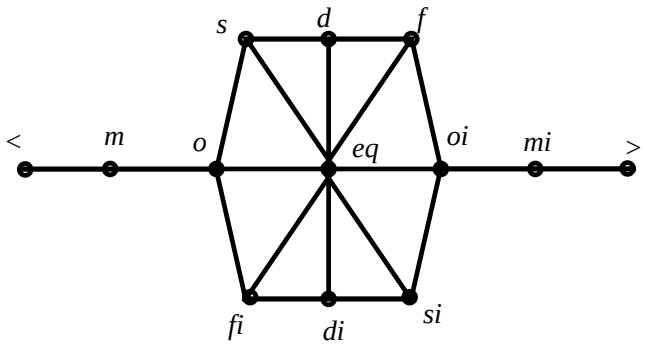


Figure 4: Temporal relations which are conceptual neighbors

perspective.

It is important to note that in a temporal graph built from text there are only convex relations since the graph is generated from a finite set of base relations, themselves mostly (if not always) convex relations, and the set of all convex relations forms a subalgebra. In the corpus we consider, we never met non-convex temporal annotations.

2.3 Strict and relaxed measures

In the general case, both humans and systems may assign disjunctions of atomic relations between two events (*i.e.* $A < B \vee A\,m\,B$). This is a way to reduce vagueness even if the exact relation is not known.

The presence of disjunctions raises the question of how to score relations that are only partly correct, like $A < B \vee A\,m\,B$ instead of $A < B$ or the reverse.

A *strict* measure only counts exact matching as success, and will for example score 0 for the latter example.

But we think that an evaluation measure should take better account of "close matches". For example, suppose that the gold standard relation between A and B is $A < B$. If the system chooses the disjunction $A < B \vee A\,m\,B$, it must be rewarded less than $A < B$ but more than $A > B$ or nothing. The system is vaguer but correct, as its annotation is a logical consequence of the standard annotation.

We proposed in [13] such a gradual measure that we might call "temporal" precision and recall. If $S_{i,j}$ is the (possibly disjunctive) relation between i and j given by the system and $K_{i,j}$ the (possibly disjunctive) gold standard, then:

$$P_{temp\ i,j} = \frac{Card(\,S_{i,j} \cap K_{i,j}\,)}{Card(\,K_{i,j}\,)} \qquad\qquad R_{temp} = \frac{Card(\,S_{i,j} \cap K_{i,j}\,)}{Card(\,S_{i,j}\,)}$$

With $Card(G_{i,j})$ = the number of atomic relations present in the disjunction. Thus, in our example, the system S_1 that answered *before* $\vee$ *overlaps* between i and j will get:

$$P_{temp\ i,j,S_1} = \frac{Card(\,(\textit{before} \vee \textit{overlaps}) \cap \textit{before}\,)}{Card(\,\textit{before}\,)} = 1$$

$$R_{temp\ i,j,S_1} = \frac{Card(\ (\textit{before} \vee \textit{overlaps}) \cap \textit{before}\ )}{Card(\ \textit{before} \vee \textit{overlaps}\ )} = \frac{1}{2}$$

While S_2 that answered *after* will get $P_{temp\ i,j,S_2} = R_{temp\ i,j,S_2} = 0$.

The final precision (resp. recall) is the average on the number of relations given by the system (resp. by the key):

$$P_{temp} = \frac{\sum_{i=1}^{i=n} \sum_{j=i+1}^{j=n} P_{temp\ i,j}}{Card(S)} \qquad R_{temp} = \frac{\sum_{i=1}^{i=n} \sum_{j=i+1}^{j=n} R_{temp\ i,j}}{Card(K)}$$

Similar measures have also been used during the TempEval evaluation campaign in 2007 [23] and were called *relaxed* recall and precision. We will use the words *strict* and *relaxed* to designate these two ways to score temporal relations.

2.4 Relative importance of relations

As shown above, temporal closure is necessary in order to able to compare properly two temporal graphs. But in a temporal graph, relations do not have all the same importance. Applying basic recall and precision scores (either strict or relaxed) on closed temporal graphs is not enough. Consider the very simple graph examples of Figure 5, in which the first graph K is the gold standard. S_1 contains only two relations, against six in K. But it seems unfair to consider a recall score of $\frac{2}{6}$, since adding only one relation ($B < C$) would be enough to infer all others. An intuitive recall would be around $\frac{2}{3}$.

Even if we suppose that we have a way to distinguish unambiguously "major" relations (bold lines in K) from "minor" (useless) ones (dotted lines), it would still not be enough. Indeed, graph S_2 finds the relation "$B < D$". This relation is minor in K, because it can be found by composing other relations; but in S_2, it is not the case, this relation actually carries a piece of information and must then be rewarded. However, even if the amount of temporal information brought by S_2 and S_3 seem equivalent, S_3 should get a higher score. Indeed, the amount of *missing* relations (to come to the full graph) is much lower in S_3 (only "$C < D$" is missing) than in S_2. Finally, S_4 should get a better recall than any other one. General cases involving all relations are obviously much more complex.

Precision is affected in a similar manner, since errors on major relations are likely to be propagated on inferred ones. The same kind of problem could be found in the co-reference task of MUC-6, and was addressed by considering a spanning tree of the graph of co-reference [24]. Any spanning tree is enough to encode equality networks since equality is an equivalence relation.

2.5 "Minimal" graphs

As said in previous section, a good, but insufficient way to deal with the relative importance of graph relations would be to work on what we called "major" relations, or "minimal graph". We consider that a temporal graph G_{min} is a "minimal graph" of a graph G if:

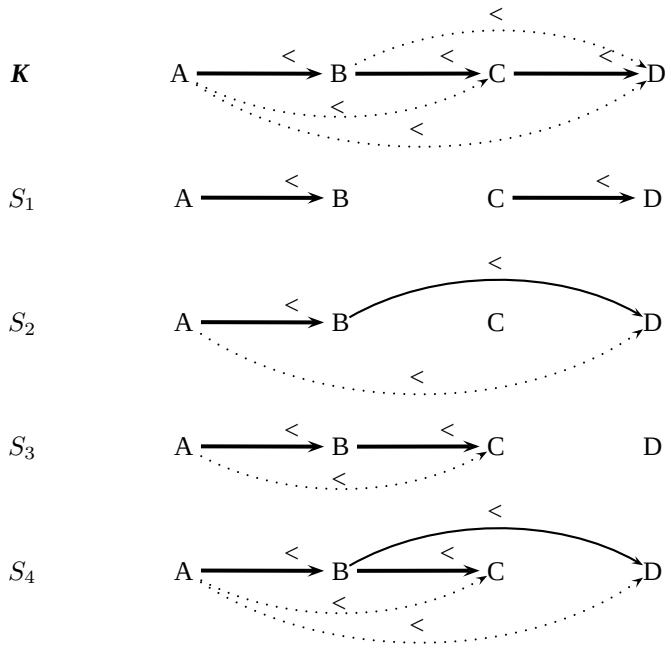


Figure 5: What a metric should deal with.

1. Its temporal closure leads to the same temporal information as G.
2. No relation can be removed from this graph without breaking the first property.

Unfortunately, a unique minimal graph does not exist in the general case, and in particular for Allen relations. [16] proposes a way to find all minimal graphs for a given temporal graph. Their algorithm first finds the *core* relations by intersecting all derivations and then computes all possible remaining combinations in order to find those composing a minimal graph.

For example, for the relation $R_{A,B}$ between A and B, derivations are $R_{A,C} \circ R_{C,B}$, $R_{A,D} \circ R_{D,B}$, $R_{A,E} \circ R_{E,B}$, etc. If the intersection of all these derived relations equals $R_{A,B}$, it means that $R_{A,B}$ is not a *core* relation, since it can be obtained by composing some other relations. Otherwise, the relation is a *core* relation, since removing it always leads to a loss of information. This operation is computationally feasible. The way this kernel is obtained ensures its uniqueness.

The second part of the procedure (compute remaining combinations) is computationally impractical for even medium-sized graphs, and the authors do not detail much their empirical investigations.

Turning back to the evaluation, [21] suggests the comparison of graphs through *core* relations, which are easy to compute and give a good idea of how important the relations are in a same graph. But core relations do not contain all the information provided by closed graphs, and measures on core graphs are only an approximation of what should be assessed. In this paper, we propose a mathod to obtain a unique graph respecting the two constraints above.

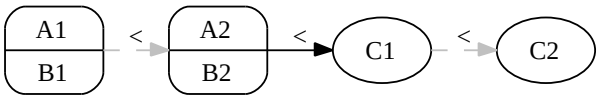


Figure 6: Endpoint graph with merges (same temporal information as Figure 3). Gray dashed arcs are trivial relations.

3 Proposed new metric

When confronting a graph to a gold standard, a similarity measure is necessary. Many similarity measures exist between two graphs once a many-to-many correspondence is found between the nodes of both graphs [20].

Node matching, which is a major problem in graph comparison in general, is not difficult in our case, since we consider that both graphs annotate the same events or expressions[3].

A traditional similarity function between two graphs is the following [20]:

$$sim(K, G) = \frac{f(K \sqcap_m G) - g(splits(m))}{f(K \cup G)}$$

where $K \sqcap_m G$ is the set of relations shared by both graphs according to the node matching function m, $K \cup G$ is the union between K and G relations, and $splits(m)$ the number of node splits imposed by the matching in map to obtain a graph to the other (see examples later). Functions f and g depend on the types of graphs and applications.

But this kind of metrics is not appropriate for temporal relations, because the transitivity of relations implies different features; also, these metrics are symmetrical, whereas two distinct recall- and precision-like values are more desirable. We adapted the general idea of two functions for split nodes and relation similarity, and arrived at the algorithm described below.

3.1 Transitive reduction of endpoint graph

To address the problem of finding minimal graphs and to take into account the relative importance of relations, we take inspiration from [5] in two main ideas. First, saturated graphs between events (intervals) are converted into endpoint graphs. Second, two nodes linked by an equality relation are merged together (this will help guarantee the unicity of the minimal graph, see below). Figure 6 presents the graph of Figure 2 after this transformation. The resulting point graph is saturated, by definition of the composition of event relations in Allen's algebra.

Recall that the graph we consider are built from "convex" annotations, i.e. there cannot be a "$<$ or $>$" relation between two points. We can keep relations $<$ and $\leq$ without loss of information, since all $>$ and $\geq$ can be obtained by symmetry.

With these specifications, the graph boils down to the directed graph of a transitive relation where an edge between two points x and y means $x \leq y$. A coherent graph will

[3] If this is not the case, creating fictitious unlinked nodes in one graph or both is enough.

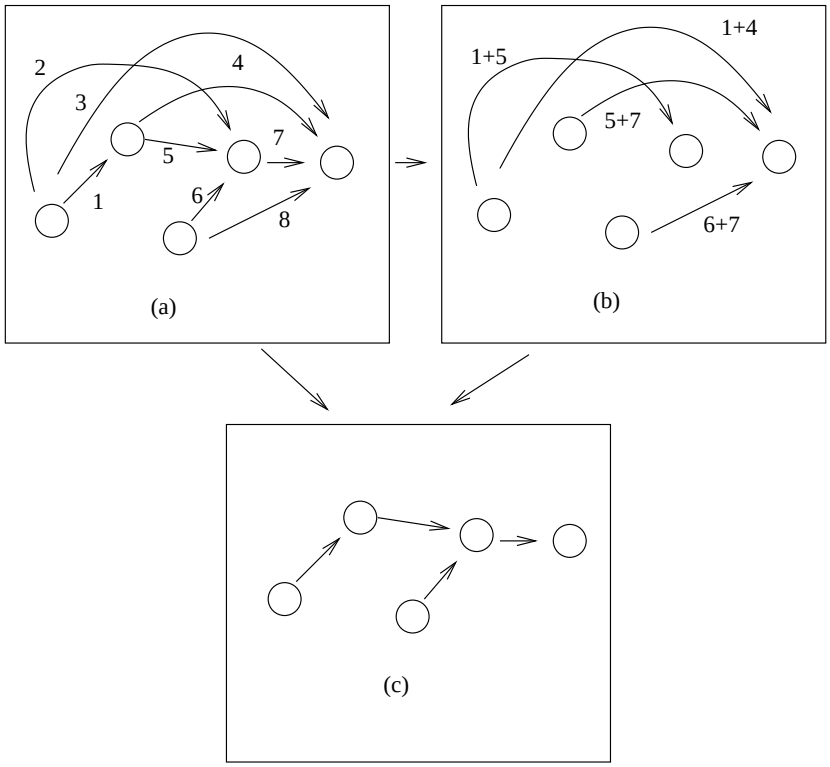


Figure 7: Transitive reduction of an acyclic graph; (a) is the initial closure of a transitive relation graph; (b) is the set of edges that can be obtained by composition of edges in (a), with examples of composition for each edge; (c) is the transitive reduction, the difference between (a) and (b).

thus be acyclic, since we collapse equal points into single nodes. It is important to note that as a consequence, no edge in the transitive closure can be labelled with the equality relation only. Thus we can see our problem as searching for the transitive reduction of a graph labelled with the transitive relation $\leq$ (but for which we keep the additional information that some edges can be more precisely labelled as $<$ instead of the disjunctive $\leq$). This is important because the minimal graph is the transitive reduction of the graph, and the transitive reduction of a directed acyclic graph is unique [1, 7]. The transitive reduction of a graph G is by definition the minimal set of edges that has the same transitive closure as G, i.e. the minimal graph G' such that G' is a subgraph of G and $G'^* = G*$ where G^* is the transitive closure of G. It is simply determined by $G^*/(G^* \circ G^*)$. Algorithm 2 details a simple computation of the transitive reduction while Figure 7 shows an illustration of the procedure on a simple transitive graph. Figure 8 shows the process from the initial endpoint graph with both $<$ and $\leq$ labels, to the minimal graph, via transitive reduction of the unlabelled graph.

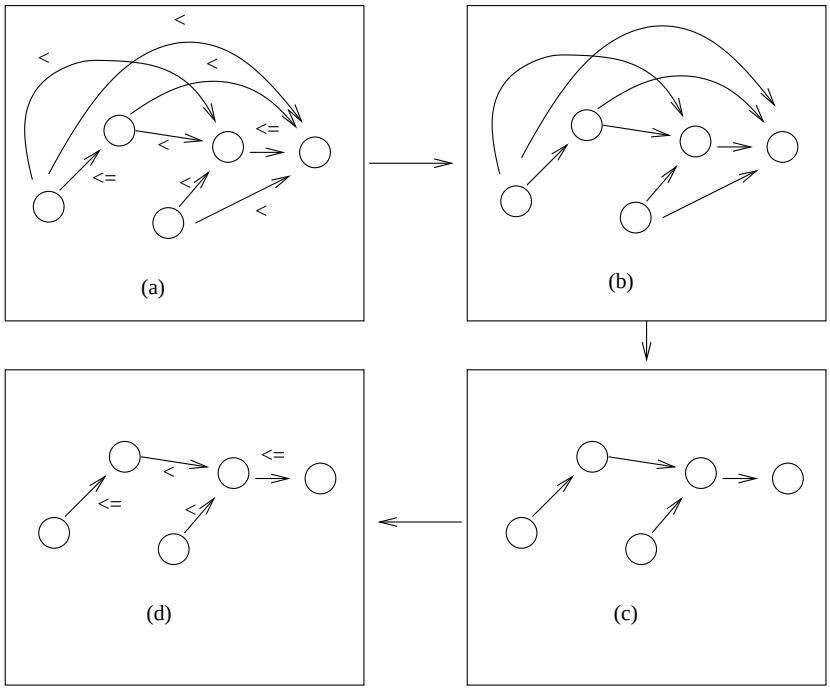


Figure 8: Transitive reduction of a point-based graph; (a) is the initial annotation transformed into a graph on events endpoints; (b) is the corresponding graph as if every label was $\leq$, (c) its transitive reduction and (d) the final minimal graph where the more precise initial information is reported.

Algorithm 2 Transitive reduction simple computation

```
procedure COMPOSE(G)                          ▷ find relations inferable from others
    newRels={}
    base_rels= { x for x in G.edges() if x.relation()==before or before_or_equals }

    for all one in base_rels do
        related = { x for x in G.edges() if x.source()=one.target() and x.relation() in
{before, before_or_equals} }

        for all other in related do
            relation = compose(one.relation(),other.relation())
            newRels.add(Edge(one.source(),other.target(),relation))
        end for
    end for
    return newRels
end procedure

procedure TRANSITIVE-REDUCTION(G)
    G = closure(G)
    non_min=compose(G)
    for all one in non_min do
        G.edges().remove(one)          ▷ remove relations deduced by composition
    end for
    for all one in G.edges() do

        if one!=before and one!=before_or_equals then
            G.edges().remove(one)    ▷ keep only <, ≤ and remove their symmetric
relations
        end if
    end for
end procedure
```

We call:

- *Major* relations, the relations of the transitive reduction, or G_{min}.
- *Minor* relations, the relations of temporal closure which are not present in the transitive reduction, *i.e.* $G^* - G_{min}$.

Formally:

Let $G = \{(x, y, R)/R \in \{<, \leq\}\}$, the temporal point graph, saturated with respect to the relation $<$ and $\leq$. So $G^* = G$

Let $E(G) = \{(x, y)/\exists R, (x, y, R) \in G\}$, the unlabelled corresponding graph. The function f which associates (x,y,R) in G to (x,y) in E(G) is an obvious bijection, as the original graph G has at most one relation holding between any two vertices. Here E(G)=f(G). Since G is closed, so is E(G).

Let $Proj(G', G) = \{(x, y, R) \in G/(x, y) \in G'\}$ the "projection" of an unlabelled graph into a labelled one. The function associating an edge (x, y) in G' to (x, y, R) in G is the inverse of f, f^{-1}. Proj(G',G)=f^{-1}(G'), and obviously $Proj(E(G), G)) = G$.

From [1], it is enough to prove that $E(G)$ (or G) is the graph of a transitive, acyclic relation to prove that $E(G)$ has a unique transitive reduction.

First, E(G) is transitive: let $(x, y) \in E(G)$ and $(y, z) \in E(G)$ then we have $(x, y, <)$ or $(x, y, \leq) \in G$ and $(y, z, <)$ or $(x, y, \leq)$ in G in any of the four possibilities we can infer either $(x, z, <)$ or $(x, z, \leq)$ is in G (because $<$ is transitive, $\leq$ is transitive and $x < y \leq z$ or $x \leq y < z$ both imply $x < z$); in other words any composition of $<$ and $\leq$ is $<$, i.e. $< \circ \leq = \leq \circ < = <$ so (x,z) is also in E(G) and E(G) is the graph of a transitive relation. This is the reason why we said that keeping a record of all $<$ relations while considering G as a graph of $\leq$ does not change the graph property.

Second, G is acyclic since $<$ is intransitive, and $x < y \leq z$ implies $x < z$ so that the only way to have cycles in G is if there are paths such as $x \leq y \leq z ... \leq x$. But in that case we can infer that $x = y = z = ...$ and the nodes would have been fused beforehand. So E(G) is also acyclic (it has exactly the samed edges as G).

So, E(G) is acyclic and transitive and thus admits a unique transitive reduction $E(G)_{min}$.

Since the graphs G and E(G) have exactly the same edges, they necessarily have the same reductions, and thus both have a unique transitive reduction. We can project back the original relations of G on $E(G)_{min}$, $(Proj(E(G)_{min}, G))$, to have a properly labelled reduction for G

3.2 Temporal recall and precision

The idea is not to compare only minimal graphs. Temporal closure should be used as well. As we showed in Section 2.4, key minor relations should still be rewarded if they are not redundant in the evaluated graph. However, they must carry a lower weight.

Minor relations are only to be considered in temporal recall, and not in precision. The reason is that recall evaluates the proportion of key relations found by the system, and this system can find some minor relations without the major relations that produced them in the key (see $B < D$ in S_4 example, Figure 5). On the opposite, precision

evaluates the proportion of system relations that are in the key graph, and there minor relations found by the system are by definition redundant.

Our recall-like measure is then a combination of two values. Given K the key graph and G the evaluated graph:

- The major temporal recall is the rate of key major relations (K_{min}) found in G^*.
- The minor temporal recall is the rate of key minor relations ($K^* - K_{min}$) found in G_{min}.

In the first case, the temporal closure is applied to G, since there is no reason to restrain the search of good relations in the evaluated graph. In the second case, only the transitive reduction G_{min} is considered; key minor relations must be rewarded only if they are not minor in G (case of $B < D$ in example S_2, Figure 5). G minor relations have already been assessed through their major relations (case of $A < C$ in example S_4).

The final value of temporal recall is a weighted sum of the two figures.

The precision-like measure is a single value corresponding to the ratio between correct relations in G_{min} and its total number of relations. G minor relations must not be considered at all for precision, since they are all redundant.

In precision as well as in recall, a merge must be considered as a relation in some way, because it corresponds to an '=' relation.

In order to make the measure clearer, we will present the metric in details by developing a basic example with a transitive closure made only of simple relations (the 13 basic Allen relations), then we will turn to the more general case of a closure made only of *convex* temporal relations (disjunctions of neighboring Allen relations).

4 First simple case: non-disjunctive Allen relations

Consider the sample graph $K1$ made of Allen non-disjunctive relations (see also the graph in Figure 9):

$K1 =$

	A	B	C	D	E	F
A		s	bi	b	s	b
B			bi	m	s	m
C				b	b	b
D					f	e
E						fi

The conversion into relations between endpoints leads to the following graph, where an event A is split into A1 and A2. Edges are not labeled since only relation '<' is considered at this point.

Note that merging equal nodes is not equivalent to labeling arcs with '=', because in the latter case the minimal graph would not be unique any more. For example, the necessary choice between $A1 < A2$, $B1 < A2$ and $E1 < A2$ would lead to three equivalent but different graphs.

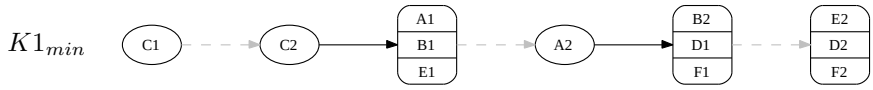


Figure 9: Key with simple relations, $K1_{min}$.

Consider now that the key $K1$ is compared to the following graph $G1$ (Figure 10; bold green edges are correct relations, thin red edges are wrong relations, gray dashed edges are trivial relations).

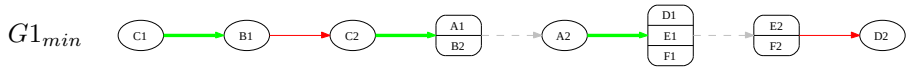


Figure 10: Evaluated graph with simple relations, $G1_{min}$.

4.1 Notations

- Li is the list of nodes for graph Gi_{min}:
  - for $K1_{min}$: $L1 = \{C1, C2, (A1, B1, E1), A2, (B2, D1, F1), (E2, D2, F2)\}$: 6 nodes
  - for $G1_{min}$: $L2 = \{C1, B1, C2, (A1, B2), A2, (D1, E1, F1), (E2, F2), D2\}$: 8 nodes
- Non-trivial relations are listed. Trivial relations are those involving two points of a same interval. For example, $C1 < C2$ is trivial, and thus not considered. In all figures, trivial relations are in dashed gray.
  - for $K1_{min}$: $R1 = \{[C2; (A1, B1, E1)], [A2; (B2, D1, F1)]\}$: 2 relations
  - for $G1_{min}$: $R2 = \{[C1; B1], [B1; C2], [C2; (A1, B2)], [A2; (D1, E1, F1)], [(E2, F2); D2]\}$: 5 relations

  These relations and the node list are enough to find the full graph.
- The temporal closures Gi^* are computed and listed below (bold relations are those added from the minimal graph, *i.e.* $K1^* - K1_{min}$ and $G1^* - G1_{min}$). Figures 11 and 12 also represent the closures.

$K1^* = \{[C2; (A1, B1, E1)], [A2; (B2, D1, F1)], [\mathbf{C1}; (\mathbf{A1}, \mathbf{B1}, \mathbf{E1})], [\mathbf{C1}; \mathbf{A2}], [\mathbf{C1}; (\mathbf{B2}, \mathbf{D1}, \mathbf{F1})], [\mathbf{C1}; (\mathbf{E2}, \mathbf{D2}, \mathbf{F2})], [\mathbf{C2}; \mathbf{A2}], [\mathbf{C2}; (\mathbf{B2}, \mathbf{D1}, \mathbf{F1})], [\mathbf{C2}; (\mathbf{E2}, \mathbf{D2}, \mathbf{F2})], [\mathbf{A2}; (\mathbf{E2}, \mathbf{D2}, \mathbf{F2})]\}$
$G1^* = \{[C1; B1], [B1; C2], [C2; (A1, B2)], [A2; (D1, E1, F1)], [(E2, F2); D2], [\mathbf{C1}; (\mathbf{A1}, \mathbf{B2})], [\mathbf{C1}; \mathbf{A2}], [\mathbf{C1}; (\mathbf{D1}, \mathbf{E1}, \mathbf{F1})], [\mathbf{C1}; (\mathbf{E2}, \mathbf{F2})], [\mathbf{C1}; \mathbf{D2}], [\mathbf{B1}; \mathbf{A2}], [\mathbf{B1}; (\mathbf{D1}, \mathbf{E1}, \mathbf{F1})], [\mathbf{B1}; (\mathbf{E2}, \mathbf{F2})], [\mathbf{B1}; \mathbf{D2}], [\mathbf{C2}; \mathbf{A2}], [\mathbf{C2}; (\mathbf{D1}, \mathbf{E1}, \mathbf{F1})], [\mathbf{C2}; (\mathbf{E2}, \mathbf{F2})], [\mathbf{C2}; \mathbf{D2}], [(\mathbf{A1}, \mathbf{B2}); (\mathbf{D1}, \mathbf{E1}, \mathbf{F1})], [(\mathbf{A1}, \mathbf{B2}); (\mathbf{E2}, \mathbf{F2})], [(\mathbf{A1}, \mathbf{B2}); \mathbf{D2}], [\mathbf{A2}; (\mathbf{E2}, \mathbf{F2})], [\mathbf{A2}; \mathbf{D2}]\}$

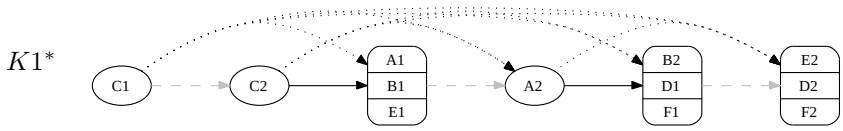


Figure 11: Temporal closure $K1^*$. Dotted relations represent $K1^* - K_{min}$.

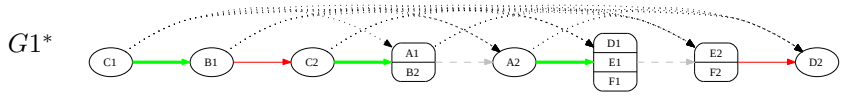


Figure 12: Temporal closure $G1^*$. Dotted relations represent $G1^* - G_{min}$.

- The pairing of nodes and the list of splits and merges needed to match both sets of nodes, is also computed (see also Figure 13):
  - $(A1, B1, E1)_{K1}$: 2 splits ("breaking" of 2 equality relations); $(B2, D1, F1)_{K1}$: 1 split ($D1$ and $F1$ stay together in $G1$); $(E2, D2, F2)_{K1}$: 1 split
  - $(A1, B2)_{G1}$: 1 merge (joining two nodes); $(D1, E1, F1)_{G1}$: 1 merge (joining $E1$ to the two others); $(E2, F2)_{G1}$: nothing (already together in $K1$)
  - Total: 4 splits and 2 merges.

A split is like a '=' relation provided by the key but not by the evaluated graph. It must then be penalized by the recall-like measure. On the opposite, a merge is a '=' relation given by mistake by the evaluated graph. The precision-like measure should penalize merges.

An edge in $G1$ is counted correct if the relation is correct for at least one pair of points from both nodes. For example, relation $\{A2\}_{G1} \rightarrow \{D1, E1, F1\}_{G1}$ is correct because $\{A2\}_{K1} \rightarrow \{B2, D1, F1\}_{K1}$, even if the sub-relation $A2 < E1$ is not true. This latter relation will be penalized anyway because a split is necessary for matching the graphs.

4.2 Temporal recall and precision

Our new metrics of temporal recall and precision rely on the notions defined above. With respect to the recall value, we have shown it was important to distinguish "major" relations, *i.e.* relations that belong to the minimal graph, from the other, "minor" relations. That is why we suggest to compute recall in two steps. Precision is not concerned by this issue.

4.2.1 Graph values

The *value* $v(G)$ of a graph G corresponds to the full number of relations. These relations can be provided by graph edges (relation value) or by merged nodes (node value, corresponding to '=' relations between endpoints):

$$v(G) = \text{node value} + \text{relation value} \quad (1)$$

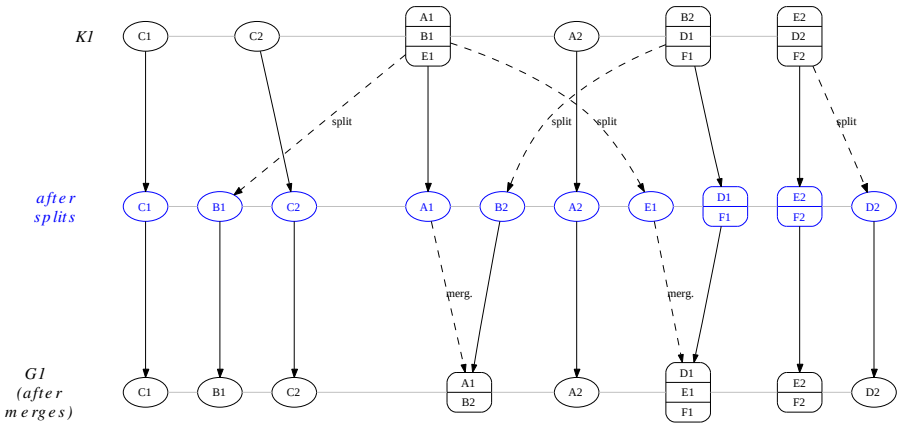


Figure 13: Split and merging operations between $K1$ and $G1$.

Relation value is the number of edges $|R|$ ('<' relations), while node value corresponds to the number of '=' relations:

$$\text{node value} = \sum_{n_i \in L} |m(n_i)| - 1 \tag{2}$$

where L is the set of nodes and $m(n_i)$ the number of points merged into a node (m points into a node corresponds to $m - 1$ merges). Node value can also be computed by:

$$\text{node value} = |E_G| \times 2 - |L| \tag{3}$$

where $|E_G|$ is the number of events[4].

In our example:

- $v(K1_{min}) = \text{node value} + \text{relation value} = 6 + 2 = 8$
- $v(G1_{min}) = 4 + 5 = 9$

The number of correct answers in an evaluated graph, or *correct value* v_c, is the number of correct '=' and '<' non-trivial relations. The number of correct '=' is "node value - merges", the number of correct '<' is "relation value - errors".

$$\begin{aligned} v_c(G) &= (\text{node value} - \text{merges}) + (\text{relation value} - \text{errors}) \\ &= v(G) - (\text{merges} + \text{errors}) \end{aligned} \tag{4}$$

$v_c(G1_{min}) = 9 - (2 + 2) = 5$

[4]'Events' are the intervals (6 in the example and $|E_{K1}| \times 2 = 12$ is the number of points), nodes are the actual vertices in the graph (points or sets of points: 6 in $K1_{min}$). The node value is thus the number of equality relations used to create sets of points.

4.2.2 Temporal recall and precision

A precision value deals with errors (incorrect relations) and merges through the correct value v_c. On the other hand, a recall value must take into account misses (key relations missed by the graph) and splits.

- *Major temporal recall* $R_t(G)$ is the number of key major relations found by G^*.

$$R(G1) = \frac{v(K_{min}) - (misses + splits)}{v(K_{min})} \tag{5}$$

  where $misses$ is the number of relations in K_{min} missed by G^* and $splits$ the number of splits (a split is a missed '=' relation).

  In our example: $R_t(G) = \frac{8-(0+4)}{8} = 0.5$

- *Minor temporal recall* $r_t(G)$ is the proportion of key minor relations ($K^* - K_{min}$) found by G_{min}. Minor relations are seeked in the minimal evaluated graph. Indeed, as we already said, comparing two relations from non-minimal graphs is redundant, since major relations that "produced" them have already been taken into account.

  In our example, $r_t(G1) = \frac{2}{8} = 0.25$ (this corresponds to $[C1; \{A1, B1, E1\}]$ and $[C2; (B2, D1, F1)]$).

- *Full temporal recall* TR could be defined as a value pair $(R(G), r(G))$, or preferably as a combination:

$$TR(G) = R_t(G) + \frac{1}{v(K_{min})} r_t(G) \tag{6}$$

  With this formula, we ensure that one single major relation is better than all minor ones and that the recall can not exceed 1 (see next Section).

  In our example: $TR(G1) = (0.5, 0.25)$ or or $TR(G1) = 0.5 + \frac{0.25}{8} = 0.53$

- *Temporal precision* $TP(G)$ is simply the ratio between the *correct value* of G_{min}, $v_c(G_{min})$, and its full *value* $v(G_{min})$.

$$TP(G) = \frac{v_c(G_{min})}{v(G_{min})} \tag{7}$$

  In our example: $TP(G1) = \frac{5}{9} = 0.56$

4.2.3 Boundaries

Temporal precision. Temporal precision is a value between 0 and 1 since $v_r(G) \leq v(G)$ (errors and merges are zero or positive).

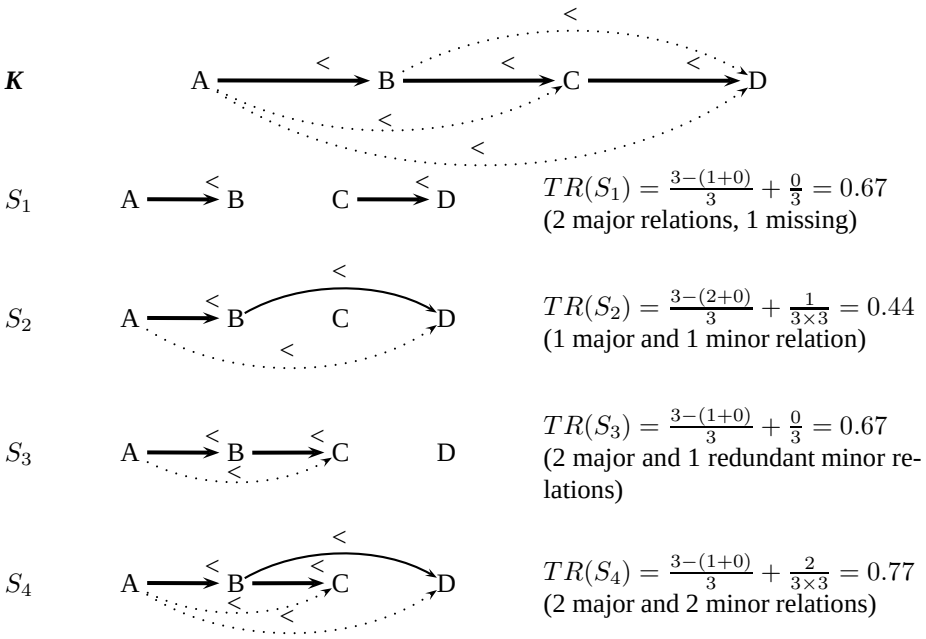


Figure 14: Temporal recall of simple graphs, compared to key K.

Temporal Recall. Major recall is between 0 and 1.

- If 1, then minor recall $r_t(G) = 0$, because all relations in G_{min} are already in K_{min} (and then cannot be in $K^* - K_{min}$). In this case temporal recall cannot exceed 1.
- If not 1,
  $R_t(G) \leq (v(K_{min}) - 1)/v(K_{min})$. Yet, $\frac{1}{v(K_{min})} r_t(G) < \frac{1}{v(K_{min})}$, and the full temporal recall stays below 1.

Then $0 \leq TR(G) \leq 1$.

4.2.4 Simple examples

Temporal recall and precision as described above lead to the expected values for the sample graphs pictured in Figure 5 and analysed in Section 2.4.

Figure 14 recalls these graphs and details the temporal recall values for each of them (given that $v(K_{min}) = 3$). As for precision, for each graph S_i, $TP(S_i) = 1$.

5 Disjunctive convex relations

We now apply the metric to sets of convex relations. Convex relations are a set of relations that are *conceptual neighbors*, that is they encode relations that may be vague but in which intervals endpoints can only be in convex subsets of the timeline (see Section 2.2).

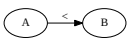
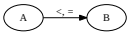
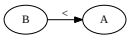
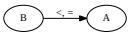
Rel.	*Repres.*	=	<	≤	>	≥
=		1	0	0.5 *(“$\frac{1}{2}$-split”)*	0	0.5 *(“$\frac{1}{2}$-split”)*
<		0	1	0.5 *(disjunction)*	0	0
≤		0.5 *(“$\frac{1}{2}$-merge”)*	0.5 *(disjunction)*	1	0	0.5 *(disjunction)*
>		0	0	0	1	0.5 *(disjunction)*
≥		0.5 *(“$\frac{1}{2}$-merge”)*	0	0.5 *(disjunction)*	0.5 *(disjunction)*	1

Table 3: Example weights for relaxed measures.

Building the minimal graph follows the same procedure as explained above (transitive reduction). The measures do not differ at all if we choose a *strict* scoring scheme (see Section 2.3). But in a *relaxed* scheme, it becomes necessary to apply a weighting scheme; a response is no longer assigned a binary value (0 or 1), but for example one of the values in Table 3.

As shown in that table, the relaxed measure has an effect on *misses* values and on v_c (a relation can get a half-point), but also on merge and split values (where half-points are also possible).

Consider a new key $K2$ with convex relations (see also Figure 15):

	A	B	C	D	E	F
A		s	$> \vee mi \vee oi \vee fi \vee = \vee f \vee di \vee si$	$<$	s	$<$
B			$di \vee si \vee oi \vee mi \vee >$	m	s	m
C				$<$	$d \vee s \vee o \vee m \vee <$	$<$
D					f	$s \vee = \vee si$
E						$di \vee fi \vee o$

The characteristics of K2 are:

- 7 nodes for 6 events (5 equality relations)
- 2 relations: $[C2 \leq A2], [A2 < \{B2, D1, F1\}]$
- $K2^* = [C2 \leq A2]$, $[A2 < \{B2, D1, F1\}]$, $\mathbf{C1 < A2}$, $\mathbf{C1 < \{B2, D1, F1\}}$, $\mathbf{C1 < \{E2, D2\}}$, $\mathbf{C1 < F2}$, $\mathbf{C2 < \{B2, D1, F1\}}$, $\mathbf{C2 < \{E2, D2\}}$, $\mathbf{C2 < F2}$,

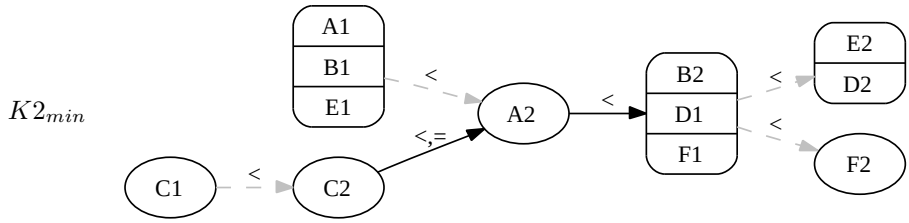


Figure 15: Key with convex relations, $K2_{min}$.

$\{\mathbf{A1}, \mathbf{B1}, \mathbf{E1}\} < \mathbf{F2}$, $\mathbf{A2} < \{\mathbf{E2}, \mathbf{D2}\}$, $\mathbf{A2} < \mathbf{F2}$

- $v(K2_{min}) = (12 - 7) + 2 = 7$.

Let us consider an evaluated graph $G2$ (Figure 16) which has:

- 10 nodes for 6 events (2 equality relations)
- 5 relations: $[C1 \leq D1], [D2 < C2], [C2 \leq A2], [\{A1, B1\} \leq E1], [A2 < \{B2, F1\}]$
- $G2^* = [C1 \leq D1]$, $[D2 < C2]$, $[C2 \leq A2]$, $[\{A1, B1\} \leq E1]$, $[A2 < \{B2, F1\}]$, $[\mathbf{C1} < \mathbf{D2}]$, $[\mathbf{C1} < \mathbf{A2}]$, $[\mathbf{C1} < \{\mathbf{B2}, \mathbf{F1}\}]$, $[\mathbf{C1} < \mathbf{F2}]$, $[\mathbf{D1} < \mathbf{C2}]$, $[\mathbf{D1} < \mathbf{A2}]$, $[\mathbf{D1} < \{\mathbf{B2}, \mathbf{F1}\}]$, $[\mathbf{D1} < \mathbf{F2}]$, $[\mathbf{D2} < \mathbf{A2}]$, $[\mathbf{D2} < \{\mathbf{B2}, \mathbf{F1}\}]$, $[\mathbf{D2} < \mathbf{F2}]$, $[\mathbf{C2} < \{\mathbf{B2}, \mathbf{F1}\}]$, $[\mathbf{C2} < \mathbf{F2}]$, $[\mathbf{A2} < \mathbf{F2}]$, $[\{\mathbf{A1}, \mathbf{B1}\} < \mathbf{F2}]$, $[\{\mathbf{A1}, \mathbf{B1}\} < \mathbf{E2}]$
- $v(G2_{min}) = (12 - 10) + 5 = 7$.

There is no merge and the number of splits is 2.5. The "half-split" comes from the fact that E1 may be equal to $\{A1, B1\}$, because of the $<,=$ edge.

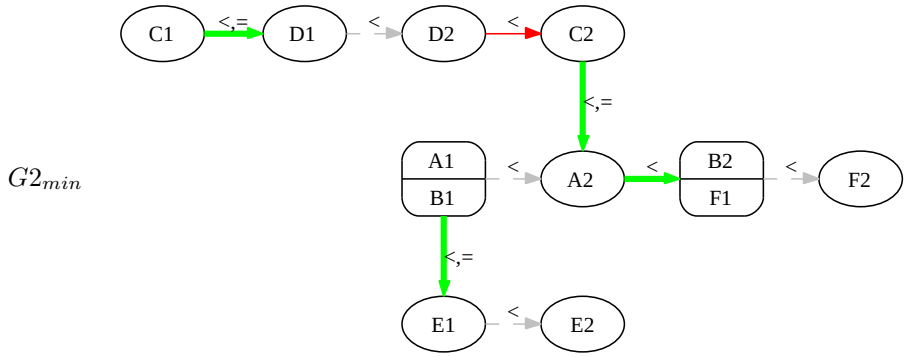


Figure 16: Evaluated graph with convex relations, $G2_{min}$.

The application of the same measures leads to the following values:

$$R_t(G2) = \frac{7 - (0 + 2.5)}{7} = 0.64$$

$$r_t(G2) = 0$$

$$TR(G2) = 0.64$$

$$v_c(G2) = (12 + 2 * 1/2) + 2 = 15$$

The two half-points in $v_c(G2)$ hold for relations $C1 \leq D1$ (instead of $C1 < D1$ in the key) and $B1 \leq E1$ (instead of $B1 = E1$ in the key).

As an example, and following Table 3, we can see that $E1$ leaving the group $\{A1, B1, E1\}$ costs a half-point to the recall value, while relation $B1 \leq E1$, correct but imprecise, gets a half-point of precision. Both results are logical behaviors of the measure in our opinion.

6 Experiments

As stated in [21], a recall measure is expected to decrease in a linear way when the amount of information decreases. This behavior can be evaluated by comparing a given annotation with the same annotation where temporal information is taken out one piece at a time.

6.1 Artificial graphs

First we perform an experiment on artifically built graphs. This experiment is performed in the following way:

- For a given number E of events, a temporal graph is built randomly. To be comparable to a graph built from a text, events are considered as intervals with some uncertainty about their endpoints, giving rise to possibly disjunctive relations between the generated events. This graph can contain any kind of convex (possibly disjunctive) Allen relations. The graph is closed, leading to $N = (E \times (E-1))/2$ relations.

- From this graph, R relations are randomly removed. Recall is expected to decrease in proportion to the number of removed relations. This operation is achieved several times for R between 1 and N [5].

- Figure 17 shows the different values of strict recall, generalized (relaxed) recall and our temporal recall (called *point recall*) according to the proportion of relations kept in the graph.

It is interesting to note that the curve based on the minimal graphs is almost linear, while other measures increase faster, in a more parabolic way. Full graphs contain redundant information, and recall increases then artificially. This is not the case when considering minimal graphs, where redundancy has been removed. This is all the more important as a corpus includes texts of various lengths and with a varying number of

[5] In Figure 17, E = 10, and 5 different graphs are built for each value of R; this leads to a total number of 216 incomplete graphs. This is done for 10 full graphs; The figure is then a smooth curve made of averaged points coming from 2160 different graphs.

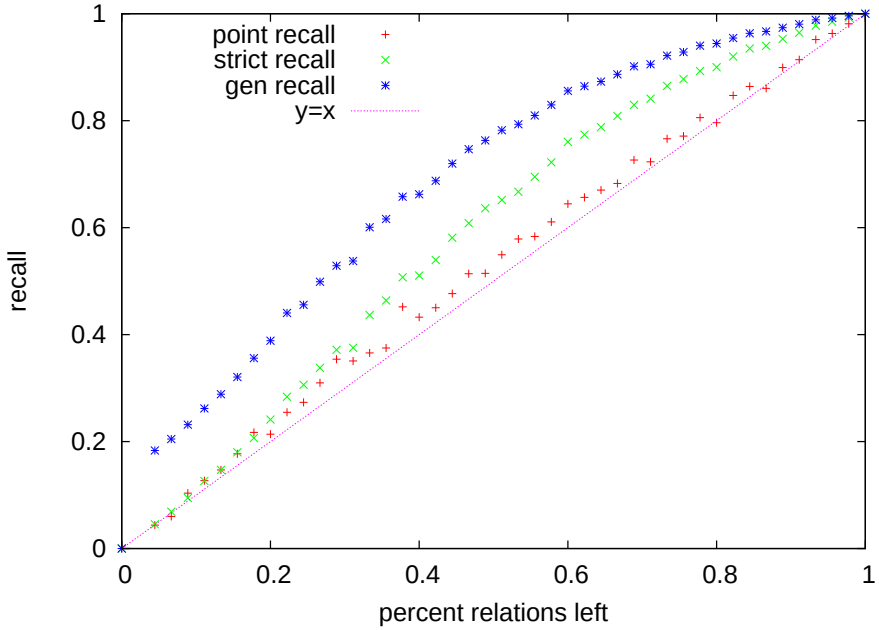


Figure 17: Behavior of recall measures according to the amount of temporal information

events, in order to avoid to give too much weight to the longer texts, either in the evaluation of the task or in training classifiers. We have for instance also compared the number of relations present in a minimal graph obtained by transitive reduction with that of the temporal closure of the interval-based graph, with respect to the number of events present in the text, on the whole TimeBank Corpus 1.1. It appears that minimal graph grows roughly linearly as expected (the variance being due to variable multiple branching when there is a lot of uncertainty), while the temporal closure is larger, much more irregular, with greater variance when the number of events grows.

6.2 TimeBank corpus

We also did the same experiment on a sample of texts from TimeBank, with much more irregular results (Figure 18) but the same phenomenon can be observed (a linear decrease of point recall), with a larger unstability when almost all relations are removed.

TimeBank is a real ressource, but it is less controlled and homogenous than a set of artificial graph, because human annotators make mistakes, forget relations, or introduce inconsistencies. That is at least in part why we observe a lot more noise in the experimental results.

Finally, to estimate the behavior of precision measures, we slightly change the above experiment by switching more and more relations to different ones, thus 'disturbing' the initial graph, while trying to keep it consistent. Again we did this a number of

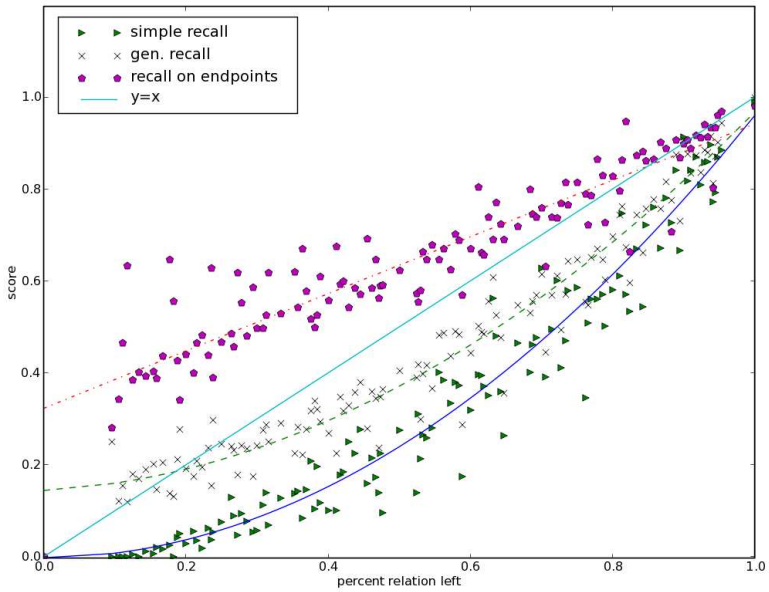


Figure 18: Behavior of recall measures according to the amount of temporal information (TimeBank)

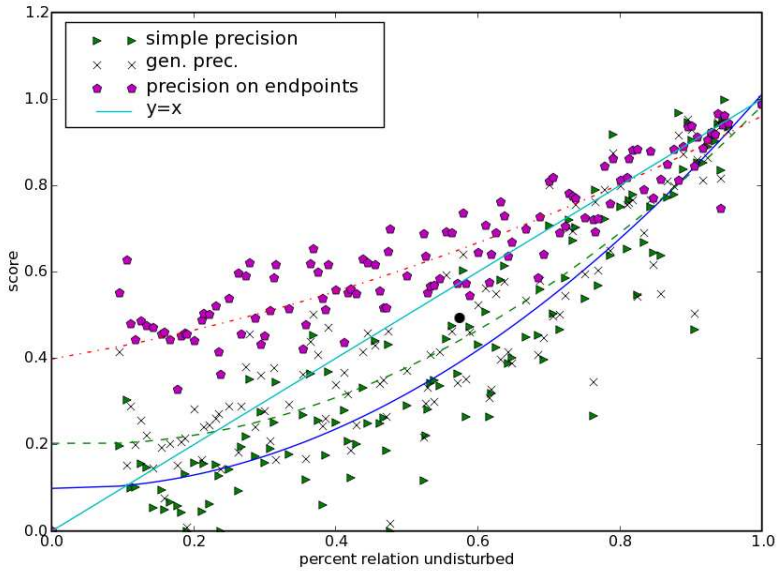


Figure 19: Behavior of recall measures according to the amount of temporal information (TimeBank)

times, and averaged the results on points with similar rate of undisturbed relations. The result, shown Figure 19, confirms that the point-based measure follows more closely the ideal 'y=x' function, with again some unstability when the graph is very disturbed.

7 Conclusion

Comparing temporal constraints graphs is crucial in the task of extracting temporal information from texts, both from an evaluation point of view and in the perspective of incorporating global constraints in statistical learning procedures. We argue here for comparison measures devoid of some of the biases inherent in the commonly used comparisons of closures of Allen-based temporal graphs. The measure is defined on the transitive reductions of the graph of (partially) ordered interval endpoints. Transitive reduction is conceptually intuitive, easy to compute and is unique in the cases considered. We have shown empirically that the behavior of this kind of measures is appropriate with the goals we had in mind.

We do not claim that ordering interval endpoints should be considered as annotation provided by humans, only that the translation is possible and useful. It remains unclear if this could also be an acceptable way of presenting temporal information to humans, or how the resulting minimal graph could be meaningfully re-translated into interval-based relations. We also plan to check our assumption that this procedure could be

useful in the task of learning temporal constraints by integration of global constraints (for instance as a good indication of how close are two temporal situations).

References

[1] A. Aho, M. Garey, and J. Ullman. The Transitive Reduction of a Directed Graph. *SIAM Journal on Computing*, 1(2):131–137, 1972.

[2] James Allen. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, pages 832–843, 1983.

[3] Philip Bramsen, Pawan Deshpande, Yoong Keok Lee, and Regina Barzilay. Inducing temporal graphs. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 189–198, Sydney, Australia, jul 2006.

[4] Nathanael Chambers and Dan Jurafsky. Unsupervised Learning of Narrative Event Chains. In *Proceedings of ACL-08: HLT*, pages 789–797, Columbus, Ohio, June 2008. Association for Computational Linguistics, Morristown, NJ, USA.

[5] Michel F. Dubois and Sylviane R. Schwer. Classification topologique des ensembles convexes de Allen. In *Proceedings of Reconnaissance des Formes et Intelligence Artificielle (RFIA)*, volume III, pages 59–68, 2000.

[6] Christian Freksa. Temporal reasoning based on semi-intervals. *Artificial Intelligence*, 54(1-2):199–227, March 1992.

[7] J.A. La Poutré and J. van Leeuwen. Maintenance of transitive closures and transitive reductions of graphs. In *Proceedings of the International Workshop WG '87 on Graph-theoretic concepts in computer science*, pages 106–120, 1988.

[8] Mirella Lapata and Alex Lascarides. Learning Sentence-internal Temporal Relations. *Journal of Artificial Intelligence Research*, 27:85–117, 2006.

[9] Inderjeet Mani, James Pustejovsky, and Robert Gaizauskas, editors. *The Language of Time: A Reader*. Oxford University Press, 2005.

[10] Inderjeet Mani and Barry Schiffman. Temporally Anchoring and Ordering Events in News. In James Pustejovsky and Robert Gaizauskas, editors, *Time end Event Recognition in Natural Language*. John Benjamin, 2005.

[11] Inderjeet Mani, Marc Verhagen, Ben Wellner, Chong Min Lee, and James Pustejovsky. Machine learning of temporal relations. pages 753–760.

[12] Inderjeet Mani, Ben Wellner, Marc Verhagen, and James Pustejovsky. Three Approaches to Learning TLINKs in TimeML. Technical report, Computer Science Department, Brandeis University. Waltham, USA., 2007.

[13] Philippe Muller and Xavier Tannier. Annotating and measuring temporal relations in texts. pages 50–56.

[14] James Pustejovsky, Patrick Hanks, Roser Saurí, Andrew See, Robert Gaizauskas, Andrea Setzer, Dragomir Radev, Beth Sundheim, David Day, Lisa Ferro, and Marcia Lazo. The TIMEBANK Corpus. In *Proceedings of Corpus Linguistics*, pages 647–656, Lancaster University, UK, March 2003.

[15] H. Reichenbach. *Elements of Symbolic Logic*. McMillan, New York, 1947.

[16] Andrea Rodríguez, Nico Van de Weghe, and Philippe De Maeyer. Simplifying Sets of Events by Selecting Temporal Relations. In *Geographic Information Science, Third International Conference, GIScience 2004*, volume 3234/2004 of *Lecture Notes in Computer Science*, pages 269–284, Adelphi, MD, USA, October 2004. Springer Berlin / Heidelberg.

[17] Roser Saurí, Jessica Littman, Robert Knippen, Robert Gaizauskas, Andrea Setzer, and James Pustejovsky. TimeML Annotation Guidelines, Version 1.2.1, 2006. http://timeml.org/site/.

[18] Frank Schilder. A Hierarchy for Convex Relations. *TIME*, pages 86–93, 1997.

[19] Andrea Setzer, Robert Gaizauskas, and Mark Hepple. The Role of Inference in the Temporal Annotation and Analysis of Text. *Language Resources and Evaluation*, 39:243–265, 2006.

[20] Sébastien Sorlin. *Mesurer la similarité de graphes*. PhD thesis, Université Claude Bernard, Lyon I, November 2006.

[21] Xavier Tannier and Philippe Muller. Evaluation Metrics for Automatic Temporal Annotation of Texts. In European Language Resources Association (ELRA), editor, *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, May 2008.

[22] Marta Tatu and Munirathnam Srikanth. Experiments with Reasoning for Temporal Relations between Events. In *Proceedings og the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 857–864, Manchester, UK, August 2008.

[23] Marc Verhagen, Robert Gaizauskas, Franck Schilder, Mark Hepple, Graham Katz, and James Pustejovsky. SemEval-2007 - 15: TempEval Temporal Relation Identification.

[24] Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. A model-theoretic coreference scoring scheme. In *MUC6 '95: Proceedings of the 6th conference on Message understanding*, pages 45–52, Columbia, Maryland, USA, 1995. Association for Computational Linguistics, Morristown, NJ, USA.

[25] Marc Vilain, Henry Kautz, and Peter van Beek. Constraint propagation algorithms for temporal reasoning: a revised report. In *Readings in qualitative reasoning about physical systems*, pages 373–381. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.