A Uniform Representation for Time and Space and Their Mutual Constraints

Roger T. Hartley
Computing Research Laboratory
New Mexico State University
Las Cruces, NM 88003

March 16, 1999

Abstract

Much recent work in reasoning systems has concentrated on the role of time in planning, action modeling, and tasks in domains where time is important. On the other hand, there are systems that concentrate on spatial reasoning, especially where manipulation or managing of the environment is important, as in robot route planning. The integration of the two themes is a goal which, if possible, would allow the interaction of space and time to be explored. A problem-solving system would then be able to reason about the times at which actions might occur, in the light of spatial constraints, or vice versa, to reason about the places in which actions take place, and the temporal constraints involved. This paper shows a way to integrate the representation of both time and space in a framework that allows uniform reasoning across both dimensions. An ontology for objects, events, states, processes is provided using conceptual graphs for representation, and a syntactic extension to Sowa's conceptual graph formalism (see Sowa's article, this volume) is presented to support the effort.

1 Introduction

This paper has two aims. The first aim is to improve the support that Sowa's conceptual graph theory has for automated reasoning, especially in the spatio-temporal domain. In order to achieve this, syntactic and semantic extensions have been made to the basic theory of conceptual graphs. The second aim is to develop the semantic extensions through an ontology of time and space that allows their essential duality to emerge. The syntactic extensions are the addition of an *overlay* graph that contains actor nodes to support reasoning, rather than relying on purely logical inference techniques, as Sowa does ([Sowa, 84]). The explicit representation of reasoning elements is preferable in practical knowledge representation systems, as has been shown repeatedly in the knowledge engineering literature. Conceptual Programming (CP) is a working implementation of the ideas presented in this paper, and has been used successfully to support the Model Generative Reasoning systems developed at the Computing Research Laboratory (see [Coombs and Hartley, 87], [Coombs and Hartley, 88]).

2 Conceptual graphs with actors

A conceptual graph is a labeled bi-partite directed graph. The two kinds of nodes are *concept* nodes, which are rectangular boxes labeled with the name of a type taken from a lattice of such types, and *relation* nodes, which are oval boxes labeled with the name of a relation taken from a set of such relations. Additionally, the concept node can contain a *referent* field (separated from the type label by a colon) that names an individual object that conforms to the type whose label is in the node. In expressiveness these graphs are equivalent to FOPC with sorts. The relation nodes correspond exactly to n-place predicates, and the concept nodes to either an existentially quantified, but sorted variable, or to a constant whose sort is known. Universal quantification may be achieved through special individuals (essentially an "individual" that represents the set of all known instances of the type mentioned) or through negated *contexts*. Sowa's context corresponds to a scoped expression in FOPC.

In his book ([Sowa, 84]), Sowa also shows how unknown objects (nodes with no individual field) can be computed by an *actor* node that corresponds to a function in standard logics. Thus the inclusion of functions in FOPC can be handled by conceptual graphs in a highly visual way, that has many benefits when it comes to providing practical knowledge systems to be used by people not well-versed in formal methods. Actor nodes of this kind are diamond-shaped boxes connected to concept nodes with dashed lines. In our extensions to conceptual graph theory, we give these actors the capability of computing *quantitative* constraints in a Prolog fashion, i.e. of doing constraint propagation through a system of values and variables. This is the constraint level of CP.

The focus of this paper is our extension of conceptual graph to a spatio-temporal level that uses *qualitative actors* that propagate constraints among moments in time when acts occur and locations of objects in space. This level requires a syntactic extension to conceptual graphs in order that the diagrams not become too confused and thus lose their force. In this paper we shall concentrate on the spatio-temporal level only. Interested readers can see a report on the constraint level in [Eshner and Hartley, 88]. In the rest of the paper an 'actor' refers to this spatio-temporal actor, not to the quantitative constraint variety.

2.1 The semantics of actors

An actor can best be thought of as an implicit relation between semantic objects represented in graphical form. The relation can be made explicit by interpreting the constraints expressed by the actor and its connections (inputs and outputs, roughly speaking) just as a rule in a rule-based system can be thought of as an implicit relation between its left and right-hand sides. 'Firing' the rule computes the relation. The CP actors however, can be run forwards or backwards, or operate as constraint checkers, just as a Prolog rule can. In this manner an actor can compute a missing relation. Temporal actors compute missing temporal relations, and spatial actors compute missing spatial relations.

2.2 The syntax of actors

The inputs and outputs of a spatio-temporal actor are things like acts, events, objects, regions, etc. i.e. the language of the ontology of space and time. This ontology is presented briefly in the next section. In conceptual graphs many of these ontological entities are represented by two or more objects related through a single relation node. Some way of connecting an actor to one of these partial graphs is thus required. We have chosen to extend the syntax of conceptual graphs for these actor connections by allowing arcs to come out of a relation node, and be connected to an actor through a special spatio-temporal relation node. In the basic theory relation nodes can only connect to concept nodes. Our extension presents no ambiguity, however, since a relation node connected to another relation node must be an actor input or output.

Since the actor and its relations are additions to an already existing graph, we can think of the addition as an *overlay* graph. The analogy here is of overhead slides being laid on top of one another to produce a complete diagram. Figure 1 shows a graph, a simple one-actor overlay and the graph that results from laying the overlay on the first graph.

3 The Duality of Space and Time.

In this section we summarize an ontology for space/time that will prove suitable for representation in the extended conceptual graph theory outlined above. It is mainly aimed at determining what things can be

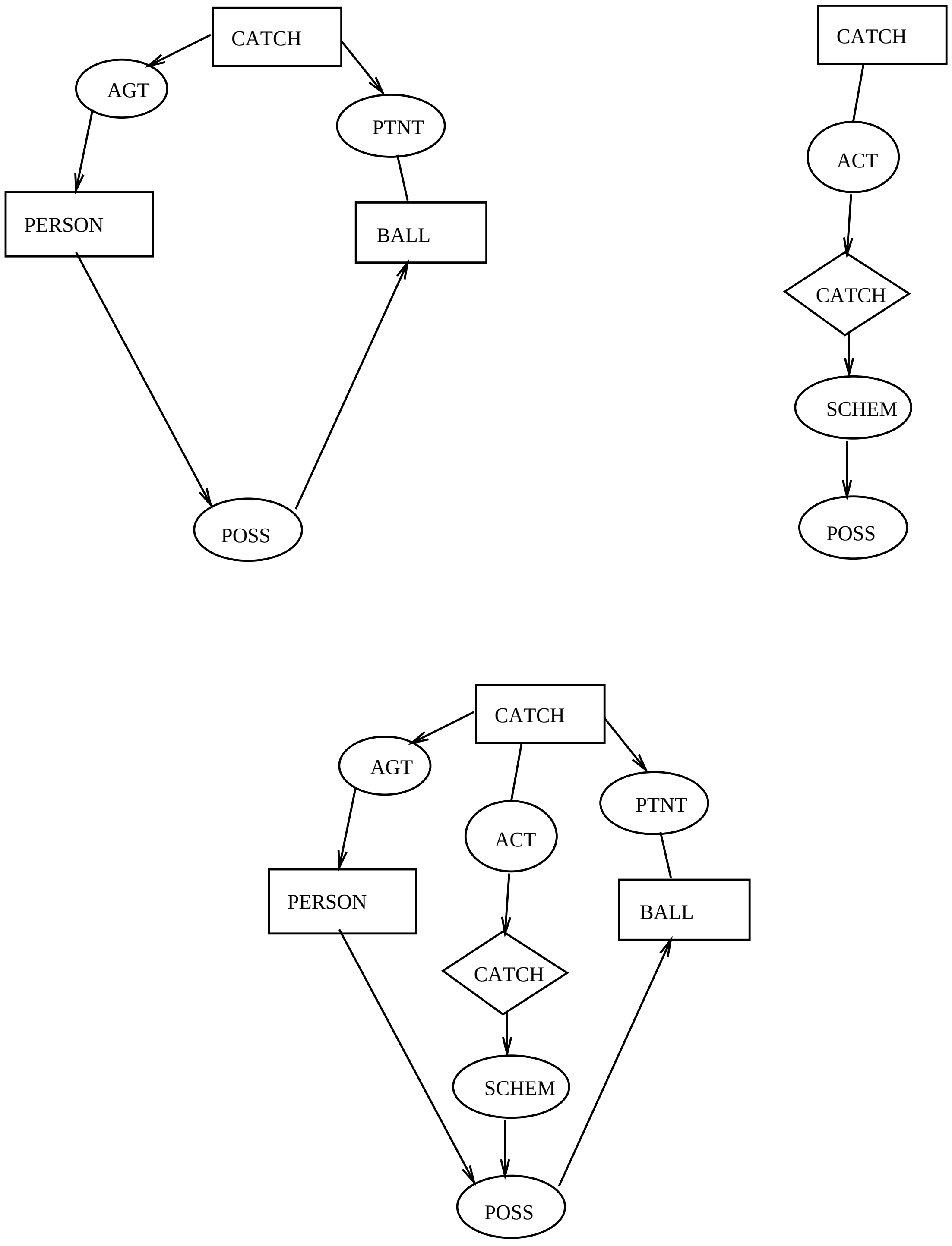


Figure 1: A conceptual graph, an actor overlay and the overlaid graph.

inputs or outputs to spatio-temporal actors. The discussion will not be deep, in a philosophical sense; we only aim to explore the worth of adding spatio-temporal actors to conceptual graph theory in order to provide a useful tool. Most of the discussion is therefore at a commonsense level; the level at which practical knowledge representation is typically done. All of Sowa's base theory is incorporated, including the primitive relations, and the simple breakdown of concept types into objects, acts and properties. We only consider ordinary objects and acts at a human scale. There is no attempt to incorporate the micro-scale of quarks, waves et al. or the astronomical scale of galaxies and gravity. Each ontological entity will be introduced and defined as the interpretation of a particular canonical conceptual graph. These graphs are summarized after the discussion in Figure 3.

The appropriate concepts in the domain of time are the *moment* (sometimes called an *instant*) and the time *interval.* The corresponding duals in the spatial domain are the *location*, and the *region*, which is intended to capture extent in three dimensions, just as an interval captures extent in the single dimension of time. We will also need to talk of *objects* (usually physical objects) and *acts.* Both entities have aspects of both space and time, of course, and the interesting things start to occur when the two are related.

4 Adding structure: States and Processes.

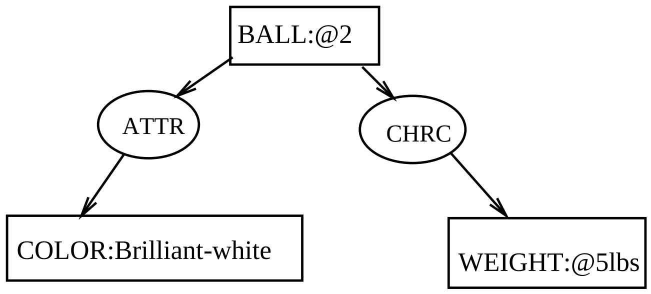
The simple existence of objects and acts (and perhaps relations between them) is not sufficient alone to form a model of space/time. In particular we need to talk about *properties* of objects and acts. This not only serves to differentiate different types and instances of these types, but also will serve as the basis of the integration of space and time. Objects can have two sorts of property. One sort is an intrinsic property, which we shall call (after [Sowa, 84]) a *characteristic.* This is a member of that set of features without which the object would not be an object, such as its size, shape and mass. They are basically spatial in nature, with a less important temporal component. For instance, an object has a shape and size and mass, the persistence of which ensures the object's continued existence. On the other hand, objects can have accidental features, that are more temporal (i.e. changeable) in nature. These include, color, temperature, speed, etc. These are *attributes.* The object with its collection of properties makes a *state.* One relation and its associated object and property (or properties in the case of multi-way relations) is a *partial state.*

Acts also have attributes and characteristics. These properties are either temporal or spatial in nature. Characteristics include rate, acceleration (or qualitative counterparts like quickness) and start and end times (both moments). Attributes, which are more spatial in nature, include direction, range, and orientation. The act and its properties makes a *process.* One relation, with its act and property is a *partial process.*

4.1 Representation of objects and acts.

At this point we should introduce part of the representation to be used later for integrating space and time. Here we follow Sowa (*op cit*). Both objects and acts are assumed to be typed, both can be instantiated with an individual. A type is represented by an upper-case label and any individual of this type is represented by the label in square brackets. Thus $[BALL]$ represents a ball, and $[CATCH]$ represents an act of catching. Properties are also typed, and represented in the same way. So $[COLOR]$ represents a color, and $[SPEED]$ a speed. Individuals of any type are placed after the type label, separated by a colon. Thus $[BALL : @2]$ denotes two balls, and $[COLOR : Brilliant - white]$ might be their color. $[CATCH : \#234]$ is the act of catching with the unique identifier 234 which distinguishes this act from all others, including those of different types.

The relationships of properties to the objects and acts they modify is done with a relation label in parentheses. The direction of the relation is indicated by arrows. Figure 2 shows the conceptual graph representing the color and weight of two balls, together with its linear form. In the paper we will use either form as appropriate, although CP uses the pictorial form exclusively.



$$[BALL : @2] \\ \rightarrow (ATTR) \rightarrow [COLOR : Brilliant - white] \\ \rightarrow (CHRC) \rightarrow [WEIGHT : @5lbs]$$

Figure 2: The ball's properties

Acts can be similarly represented. e.g.

$$[THROW : \#234] \rightarrow (MANR) \rightarrow [SOFT]$$

represents a gentle throw, where THROW:#234 is an act of throwing, MANR is a relation between the act and a property indicating the manner in which the act is carried out. In this case the manner is SOFT, for softly.

4.2 Relationships between entities.

Now that we have a notation for describing objects and acts, we can consider relationships between objects, between acts, and between objects and acts. These relationships will give us a basic set of representations to support reasoning.

Like an object's properties, relationships between objects are largely spatial in nature. These include out-and-out spatial relations like left-of, above etc., but also relationships of containment, support, and inclusion (e.g. part/whole). A single object and its properties and their values (if known) give the familiar notion of a *state*. The set of all objects and their spatial relations we will call a *schematic*. If we gather all the objects in our universe together with their properties and their spatial relationships at one moment we have the common notion of a *snapshot*. We must include in the ideas of state, schematic, and snapshot that it exists at a particular moment, and that any part of it can, and will, change through time. One spatial relation and its associated object(s) is a *partial schematic*.

The relationships between acts are largely temporal in nature. Again, there are obvious ones like after, before, overlap *a la* [Allen, 85]. There can also be abstract relationships like causes, enables, or triggers. Their common link is their pervasiveness throughout space, whether it is the action at a distance involved in causality, or their applicability irrespective of location. They will tend to pervade space unless delimited by an object, just as state relationships persist unless changed by an act. A number of acts and their temporal relationships forms a *chronicle*. The collection of acts, their properties and temporal relationships we will call a *history*. This is a "temporal snapshot" (there is no term in English for this, apart from history) fixed in space. Any aspect of a chronicle i.e. a single relationship and its associated acts is a *partial chronicle*. It can, and does, change with location. The THROW example is a partial process, whereas

$$[THROW : \#234] \rightarrow (BFOR) \rightarrow [CATCH : \#345]$$

is a partial chronicle, since there are two acts, and one completes before the second starts.

Objects participate with a single act to form an *event*. The relationships here are the standard case relations well-known in knowledge representation. Thus, agent, patient, experiencer, instrument etc. relate an act to its participants. The spatial cases, such as location, path, direction etc. are considered here as properties of an act, as are the temporal cases such as duration and rate. The case relations cannot form partial processes or states, whereas the spatial and temporal ones can. In fact, it is probably better to think of the *event* as the atomic unit (albeit with structure) corresponding to an act, and the *experience* as atomic in the spatial sense. An experience is a single object and all its associated acts. Events are time-independent since their acts carry with them a time interval, one of the distinguished characteristics mentioned above. Experiences are location-independent, since their objects carry an intrinsic region with them.

Figure 3 shows canonical graphs for each of the ontological entities introduced above. In the diagram, three concept types are used: $PROP$, ACT and OBJ. There are five kinds of relation label, each of which stands for a set of relations disjoint from any other set. They are:

- SR: a Spatial Relation, between two objects,

- TR: a Temporal Relation, between two acts,

- CR: a Case Relation, between an act and an object,

- APR: an Act Property Relation between an act and one of its properties,

- OPR: an Object Property Relation between and object and one of its properties

5 Reasoning with events and experiences

Reasoning has two main requirements. Firstly, a set of linking forms that relate knowledge structures together according to a pre-defined principle, and secondly a way of operationalizing the inference of one structure given another. In much of AI this amounts to using rules expressed as implications and then using modus ponens for inference in some variety of classical logic. Other approaches, such as production systems, semantic networks and frames are reducible in some sense or other to logic. These alternative approaches, however, all have aspects that are not easily reducible to logic. They mostly revolve around the mechanisms used to make inferences. Logic, *per se* does not specify such mechanisms, although proof techniques are essential in all but trivial cases. The question is how can we specify mechanical procedures that allow the correct, desired inferences to be made without also making incorrect, unwanted ones. The current concentration in AI on non-monotonic systems is one such an attempt. Here we employ an approach borne out of a combination of simulation, that we call small-case reasoning, and a combining technique based on Sowa's maximal join. The first requirement for reasoning mentioned above is satisfied by rules, both temporal and spatial at the level of spatio-temporal entity, and by maximal join at the level of whole graphs.

As its major form of knowledge structure, CP employs the *schema*, which is similar in many ways to the cases of case-based reasoning (CBR) (e.g. [Hammond, 86]). However, schemata in CP are "small" cases in the sense that each schema only focuses on one term (a single concept type). They are more likely to be correct in the sense that their place in a larger structure will be secure, without the fix-up rules that CBR employs when cases do not match known facts. We call each schema a *definition* of the term in question, although there can be many such definitions, reflecting the variety of localized situations where the term can be used. Lansky's use of localized representations is similar ([Lansky and Fogelsong, 87]), although our use

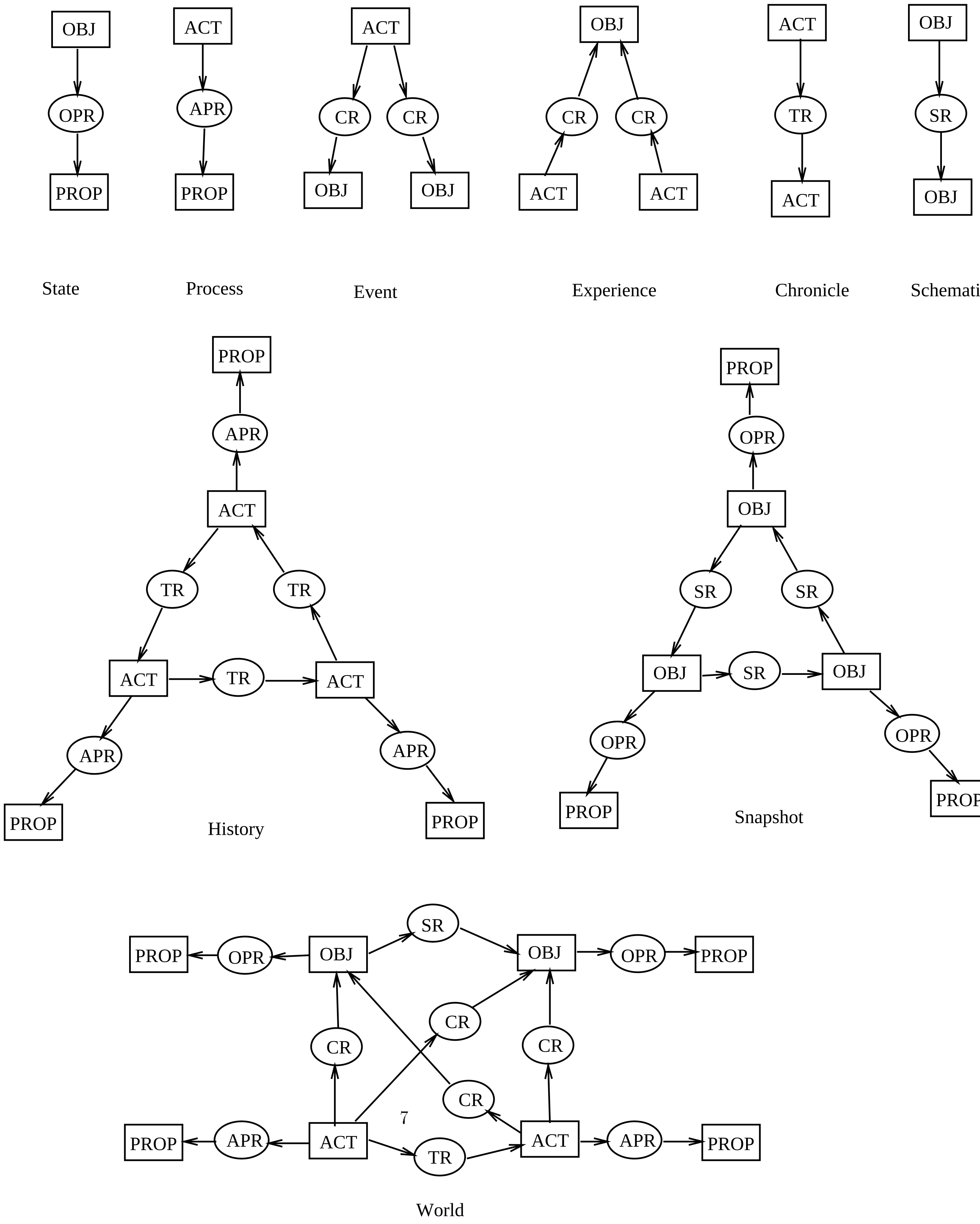
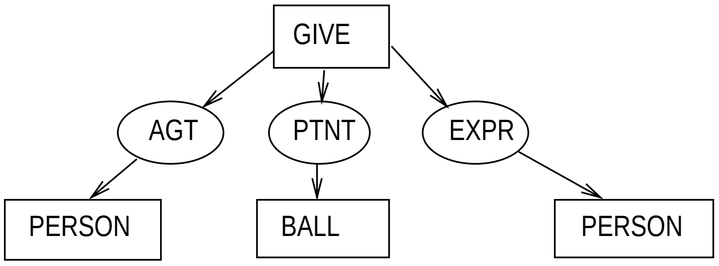
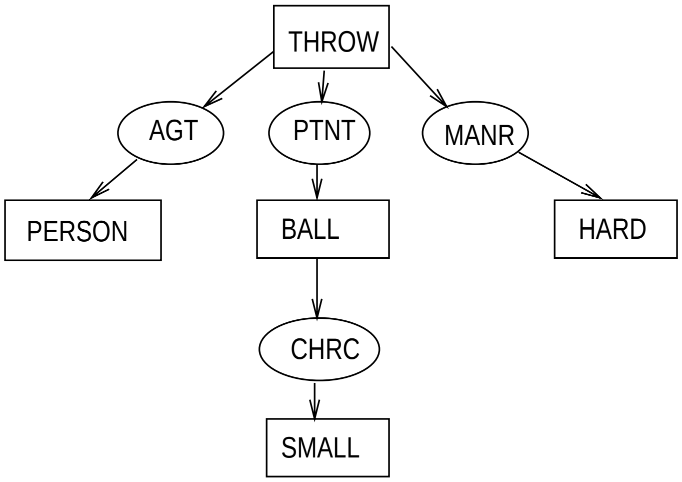


Figure 3: Canonical graphs for each of the spatio-temporal entities



$[GIVE]$ -
$\rightarrow (AGT) \rightarrow [PERSON : *x]$
$\rightarrow (PTNT) \rightarrow [BALL]$
$\rightarrow (EXPR) \rightarrow [PERSON : *y]$

Figure 4: The graph for GIVE



$[THROW]-$
$\rightarrow (AGT) \rightarrow [PERSON]$
$\rightarrow (PTNT) \rightarrow [BALL] \rightarrow (CHRC) \rightarrow [SMALL]$
$\rightarrow (MANR) \rightarrow [HARD]$

Figure 5: The graph for THROW

of the term more accurately coincides with Hayes' notion of conceptual closure ([Hayes, 85a]) in that each definition includes an adequate set of related terms to make a complete, coherent definition. The multiple schemata for a term allow for the fact that such closures are not unique.

A schema contains only events (and experiences), the base components of reasoning. Figure 4 shows a definition of giving. It shows an event involving three objects none of which has a partial state. Throwing might however, contain a partial state for the object, as well as a partial process of the direction in which the object is thrown, as in Figure 5 where the ball's characteristic of smallness forms part of its state.

On top of a schema there can be up to three 'overlays', one to infer temporal relationships, one to infer spatial relationships, and one that computes functional relationships between the individuals involved. The last overlay, the constraint overlay, will not be discussed here, but briefly it can be used to carry out quantitative computing such as can be found in more standard simulation languages ([Eshner and Hartley, 88]).

6 The representation of rules for temporal reasoning.

Rules in CP are represented as *actors*, whose sole job is to act as confluence points for the knowledge structures that have to be related. All of the actors are constraint-like in that they can operate forwards or backwards. However, temporal actors are often regarded as operating forwards, in the direction of time.

Thus, inputs to an actor are pre-conditions for the actor's firing, and outputs are post-conditions. In the temporal domain, inputs are partial states and schematics, since these are exactly what is expected to change in time. For instance, the possession of a ball by a person is a partial schematic; the possession of the same ball by another person is another one. Each temporal actor also has an act (really a process) as input, at least one partial state or schematic as input and one as output. The crucial part of the whole idea of the overlay comes, however, with the temporal relationships that the act bears to the inputs and outputs. We have appealed to simple ideas of causality to analyze the possible relationships. Firstly, let us call partial states and schematics collectively *situations*. Causally speaking, a situation can enable or trigger an act, and the act can terminate that situation (or not). If, in addition, we allow the *absence* of a situation to have the same causal status i.e. enabling or triggering, then we arrive at a total of eight combinations of a single input situation and an act. These are displayed in Figure 6 in time chart form (time increases to the right) with each one having the situation on top and the act below. A vertical line on an end-point indicates a change start to stop (or vice versa), and an arrow head indicates an unknown end-point. These time charts are similar in use and meaning to the time maps of Dean and McDermott ([Dean and McDermott, 87]). Notice that each of the interval pairs (except 6 and 7) have correspondence to Allen's relations ([Allen, 85]).

The same ideas can be applied to output situations, changes in which are caused by an act. An output situation can be started by an act's starting, started by the event's ending or coincide exactly with the event. [1] Again the inverses involving the situation's absence complete the picture. Now we get six possibilities (Figure 7).

When relations like these are used in a temporal overlay, we get a rule-like structure that can be incorporated into a larger structure and used in a simulation. A simple example is shown in Figure 8. The relations on the actor node $< CATCH >$ effectively type the inputs and outputs. The type can either be ACT, indicating connection to an act node, $SCHEM$ for connection to a partial schematic, or $STATE$ for connection to a partial state. Recall that partial schematics or states are *situations* and both have an interval associated with them in the time chart. The direction of the arrows connecting through the type node indicate whether it is an input or output. Each actor has a time chart associated with it that gives the exact relationships of the intervals and moments involved. Note that the act is incompletely specified (there are no case relations), but the time chart, and hence any temporal inferences do not depend on them. This graph could be overlaid on a more complete description (mentioning, for instance the direction of travel and the nature of the ball).

The overlay for the example involving GIVE might be as in Figure 9. This then is the purpose of a temporal overlay, to show how situations change in time when directly affected by acts. Note that unless changed by an act, a situation will persist, as demanded by the basic assumptions. Note also that if the absence of a situation is to participate in causality (usually in enablement) then the relation (property or spatial) must be represented explicitly. For this reason, relations in CP cannot be given interpretations of true or false, but only of possibility.

7 The representation of rules for spatial reasoning.

We can now apply the same notions to create spatial actors corresponding to the temporal actors just discussed. Where the temporal overlay placed partial states or schematics in temporal relationship, the spatial one places partial processes or chronicles in spatial relationships. Clearly we need at least two partial processes or chronicles to do this, together with an object (properly an experience) so we will need at least two acts. Notice that in both the single act cases, the only spatial inference is that all objects occupy the same region as the act. Only when two acts occur can entities be spatially differentiated. Similarly, when only one object participates in several events, no temporal differentiation can be made, since there can be no change of state.

[1] Moreover, the effect can be delayed, as in the delay between a ball being thrown and it breaking a window.

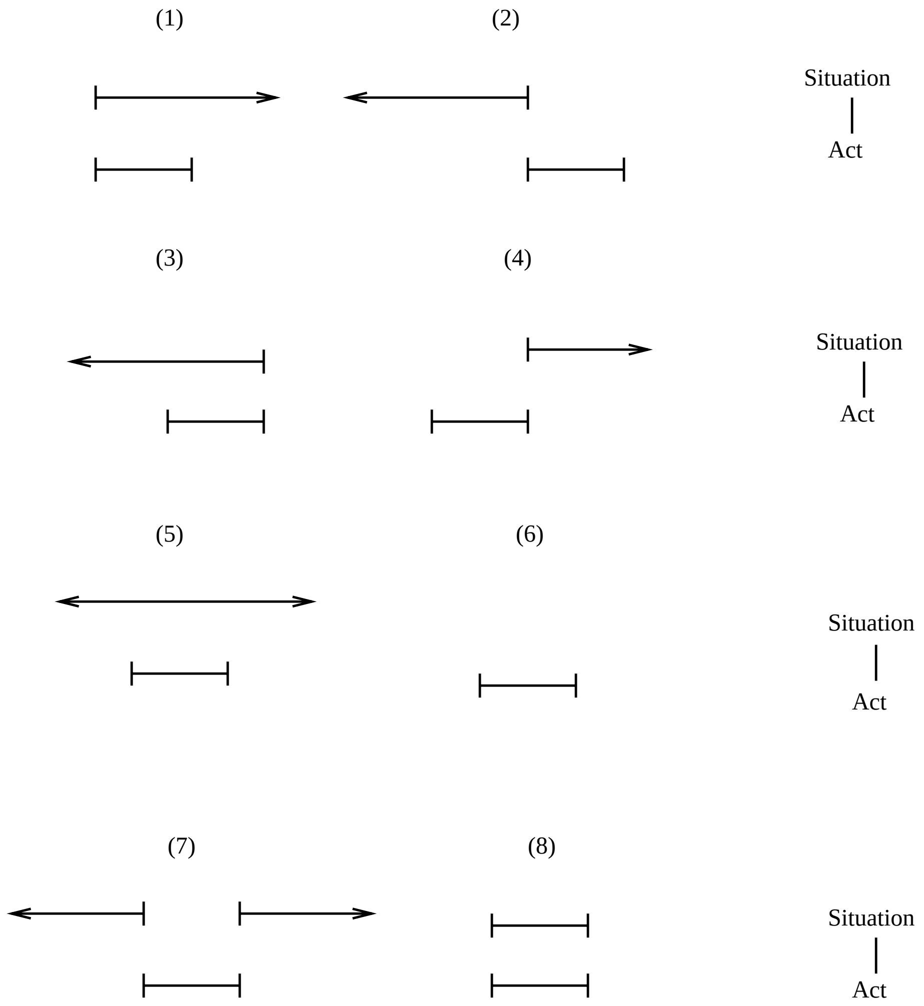


Figure 6: Time charts for the input temporal relations. (1) is a triggering enablement where the situation triggers the act and persists after the act finishes. (2) is the inverse where the absence of the situation is the trigger. (3) is a temporary enabling condition where the situation that enables the act disappears at the end of the act. (4) is again the inverse. (5) and (8) are permanent enabling conditions where the situation (or its absence) persists even though the act terminates. (7) shows an interrupt enablement where the situation is interrupted by the act, but resumes after the act finishes. (8) is the inverse of this, but can also be seen as the situation and act coinciding in their start and end points

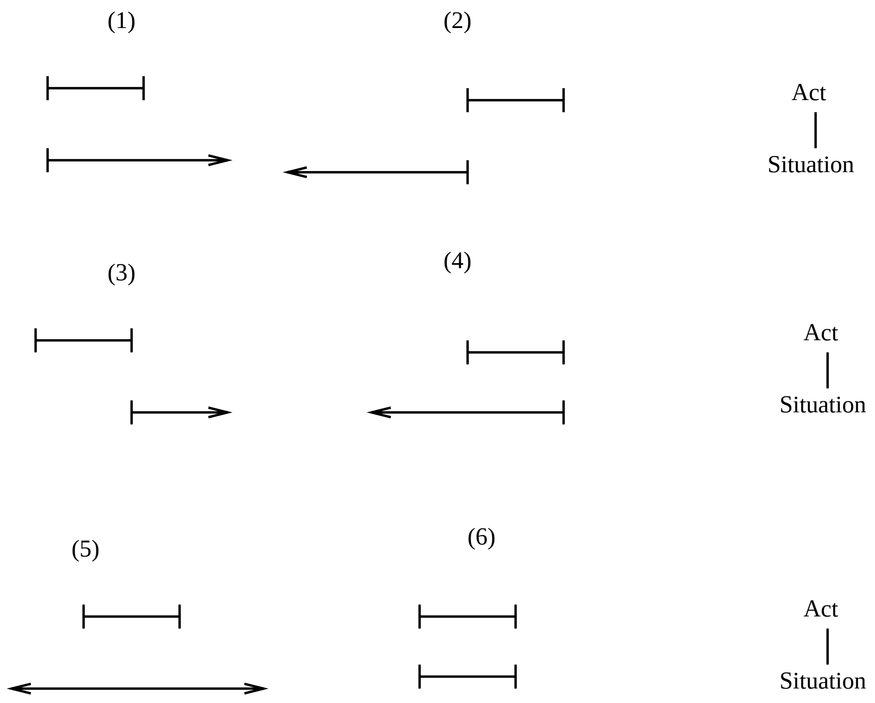
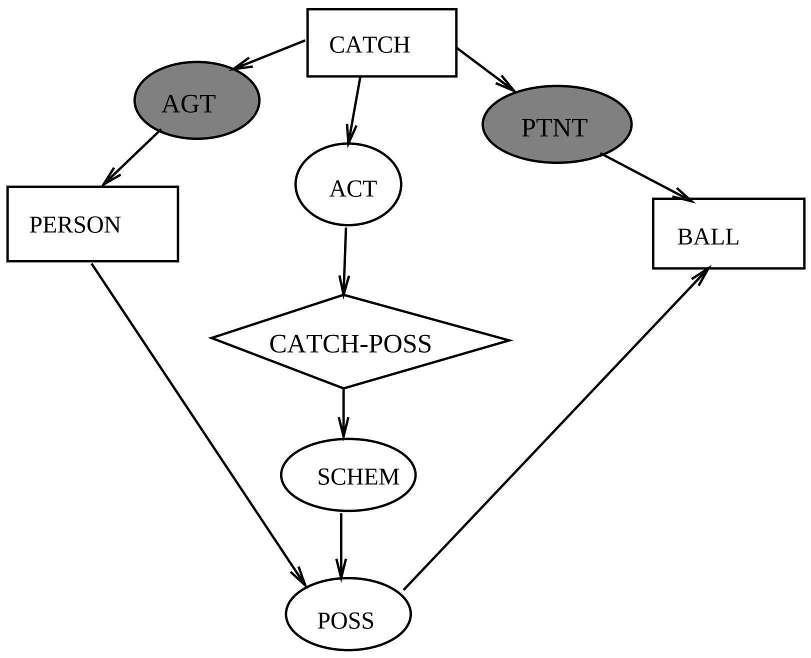
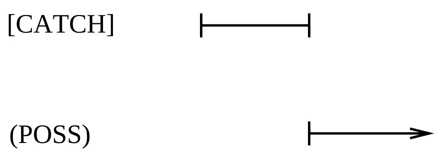


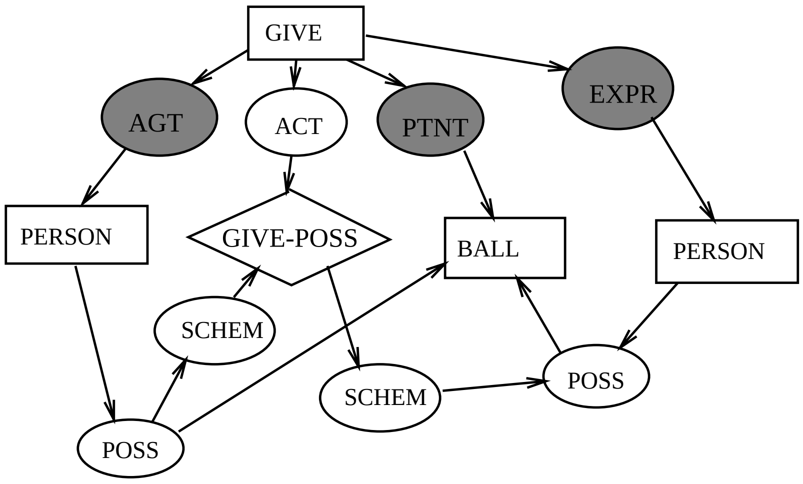
Figure 7: Time charts for the output temporal relations.





$$\begin{array}{l} < CATCH - POSS > - \\ \quad \leftarrow (ACT) \leftarrow [CATCH] \\ \quad \rightarrow (SCHEM) \rightarrow (POSS) - \\ \qquad \rightarrow [BALL] \\ \qquad \leftarrow [PERSON] \end{array}$$

Figure 8: The temporal overlay for CATCH with the actor's time chart. The shaded nodes are not part of the overlay, but are part of the schema for CATCH.



$$
\begin{array}{l}
< GIVE - POSS > - \\
\quad \leftarrow (ACT) \leftarrow [GIVE] \\
\quad \leftarrow (SCHEM) \leftarrow (POSS) - \\
\qquad \rightarrow [BALL : *b] \\
\qquad \leftarrow [PERSON : *x], \\
\quad \rightarrow (SCHEM) \rightarrow (POSS) - \\
\qquad \rightarrow [BALL : *b] \\
\qquad \leftarrow [PERSON : *y]
\end{array}
$$

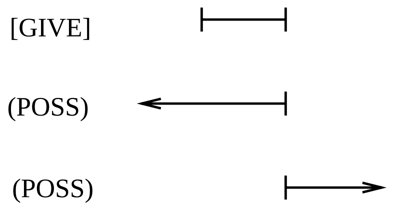


Figure 9: The temporal overlay for GIVE with the actor's time chart. The shaded nodes are part of the schema for GIVE, but are not included in the overlay.

Each spatial rule will be represented by an actor corresponding to an object. Whereas the temporal actors are directional, according to the forward flow of time, there is no such constraint on spatial actors. They execute, therefore, very much like Prolog rules, i.e. they can operate forwards or backwards. Since there is no distinction between inputs and outputs, no relation to a spatial actor has an arrow-head.

Consider a person grasping a ball, throwing it and catching it again. There are two partial chronicles - the succession of temporal relations between the three acts. The $BALL$ spatial actor will then constrain the regions occupied by the partial chronicles to intersect the ball's region, just as the temporal actor constrains the intervals of the partial states to bear a pre-determined relationship to its act's interval. Objects can partially or totally constrain a partial chronicle, or not at all. These factors give rise to seven possible classes of spatial relations (Figure 10). These are similar to the set of mereological relations (i.e. those concerning the connections between regions) discussed by Randell and Cohn in [Randell and Cohn, 89]. We have added a distinction, however, between two regions which share a common space — $INSD$, (inside) and a region which takes space away from another — $INVD$ (invades). The former covers the case of two acts one of whose regions contains the other. The latter covers the case of a solid object invading a region in which an act occurs; the act does not extend to the inside of the object, but only surrounds it. The overlay is shown below followed by a diagram (Figure 11) that approximates in two dimensions the relationships between the regions occupied by the ball and the three events. Really it is only the intersection of the ball's region with the *overlap* between the events' regions (i.e. that occupied by the partial chronicle) that can be inferred. Note, for instance, that there is no information to constrain the $CATCH$ to the same region as the $THROW$. However, they must overlap, and this overlap must (at least) intersect the ball's region. Note that an act's region cannot be determined unless all of its objects are specified, although one object can provide a partial constraint.

8 The integration of temporal and spatial rules.

The symmetries alluded to in the introduction can be seen by considering a canonical graph with two acts and two objects (actually objects or properties) and their possible relations. A fully connected version is shown in Figure 12. The CR labels denote case relations, TR is a temporal relation and SR is a spatial relation, as in the diagram of the canonical forms (Figure 3). Since both TR and SR are present, there is no reason to specify an act actor, whether spatial or temporal, since their job is to compute missing relations. Figure 13 shows the square firstly with a missing temporal relation, and secondly with a missing spatial relation. The missing relations are computed by the actors as shown. When both sorts of overlay are included in the same structure, the result is a constraint network of actors, that is called a *world.* Whether the actor is temporal or spatial it can only fire if all of its inputs and/or outputs are satisfied. Currently CP 'executes' such a network (a CP *program*) with a breadth-first traversal of the network, passing tokens to partial states, schematics, chronicles or processes when actors do fire. When an actor fires, its time chart (or spatial map) is merged with its global form in such a manner that the constraints between moments and intervals or locations and regions are obeyed. It is possible for the merge to prove impossible. In this case, the simulation fails, and no final inference may be drawn. This is very like a Prolog program that says 'no' to a query, or to an over-constrained system of equations that has no solution.

The end result is a global time chart and a spatial map showing the relationships of the various regions and time intervals. The two can be correlated through the objects and acts involved. The execution of the program is best considered as a simulation of the behavior of that part of the world being represented. The success of the simulation (i.e. the successful computation of all temporal and spatial constraints) is evidence for the accuracy of the model. If the program fails to run, i.e. there are actors left unfired, then the model is inaccurate. This technique forms the basis of our problem-solving technique where the schemata are pieced together with the aim of producing a successful simulation ([Coombs and Hartley, 87], [Coombs and Hartley, 88]).

CP, like other constraint satisfaction systems suffers from potential exponential worst case complexity

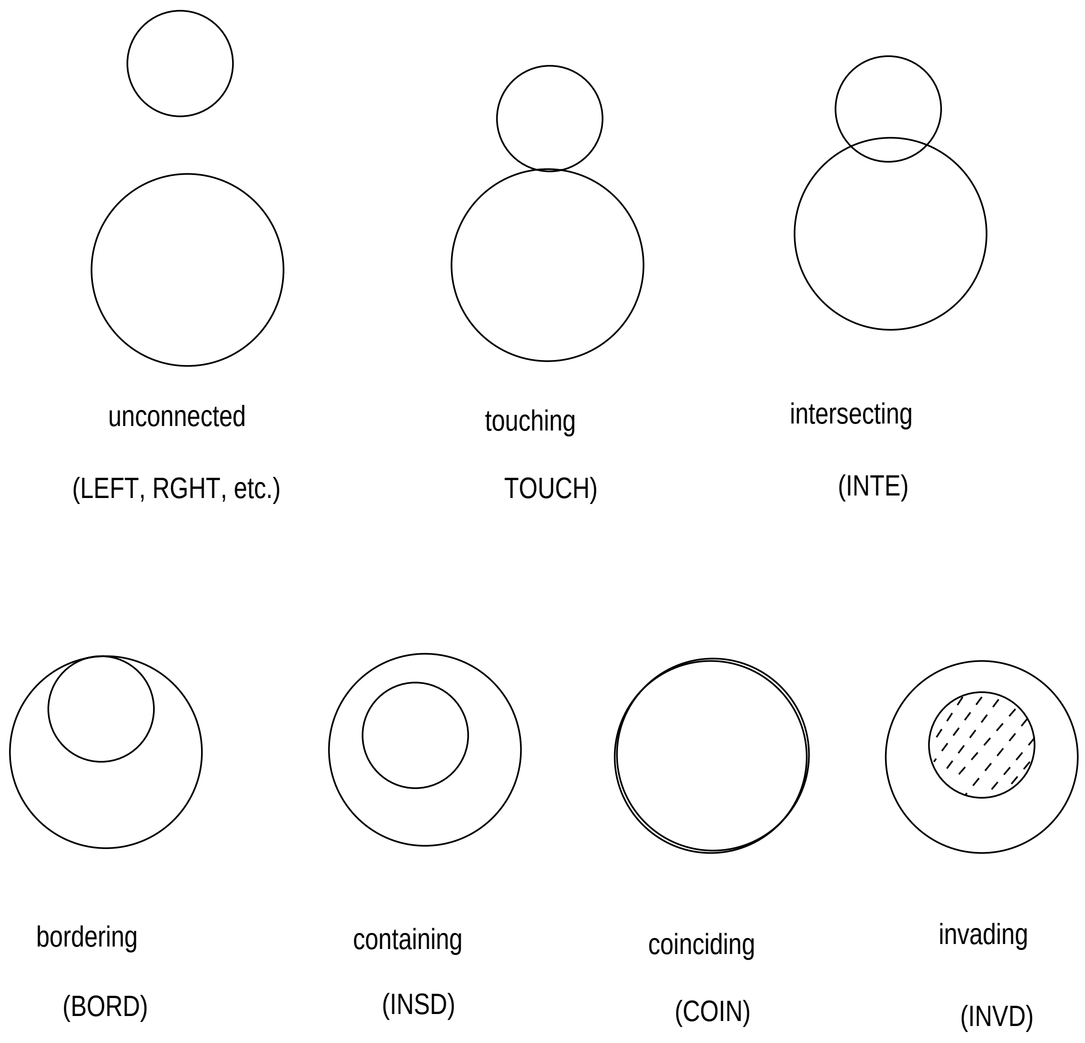
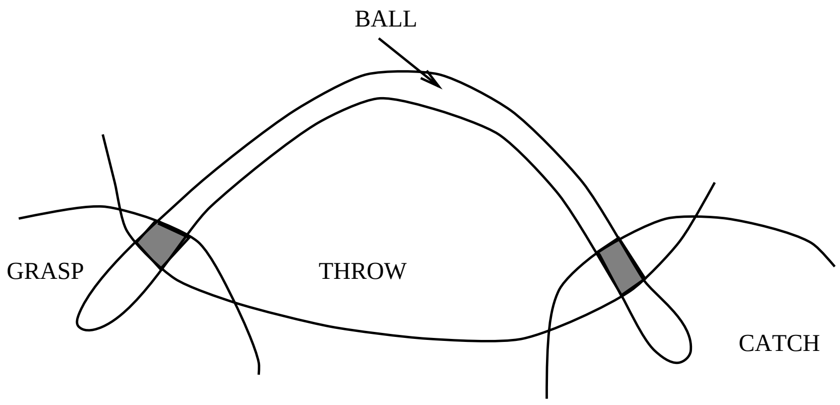


Figure 10: The dyadic spatial relations which can appear in a spatial map.



$$
\begin{array}{l}
< BALL > - \\
\quad -(OBJ) - [BALL] \\
\quad -(CHRON) - (BFOR) - \\
\qquad \rightarrow [THROW : *t] \\
\qquad\quad \leftarrow [GRASP], \\
\quad -(CHRON) - (BFOR) - \\
\qquad \rightarrow [CATCH] \\
\qquad\quad \leftarrow [THROW : *t]
\end{array}
$$

Figure 11: The spatial layout for BALL. The ball's region ties together the regions of grasping, catching and throwing since the shaded regions must be non-empty.

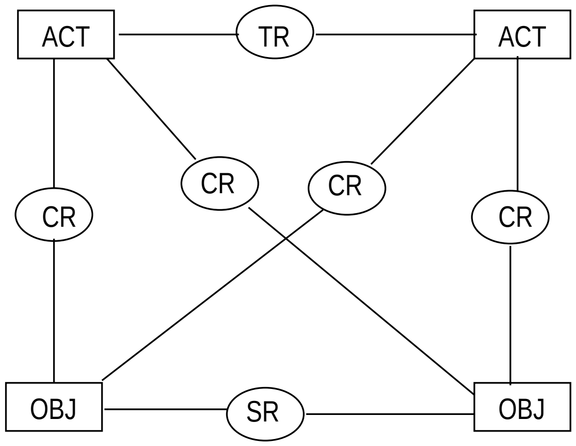


Figure 12: The canonical space/time square, where ACT denotes any act, OBJ any object, and CR, SR and TR denote any case, temporal or spatial relation respectively. No relations are missing, so no actors are needed. The directionality of the relations is left unspecified.

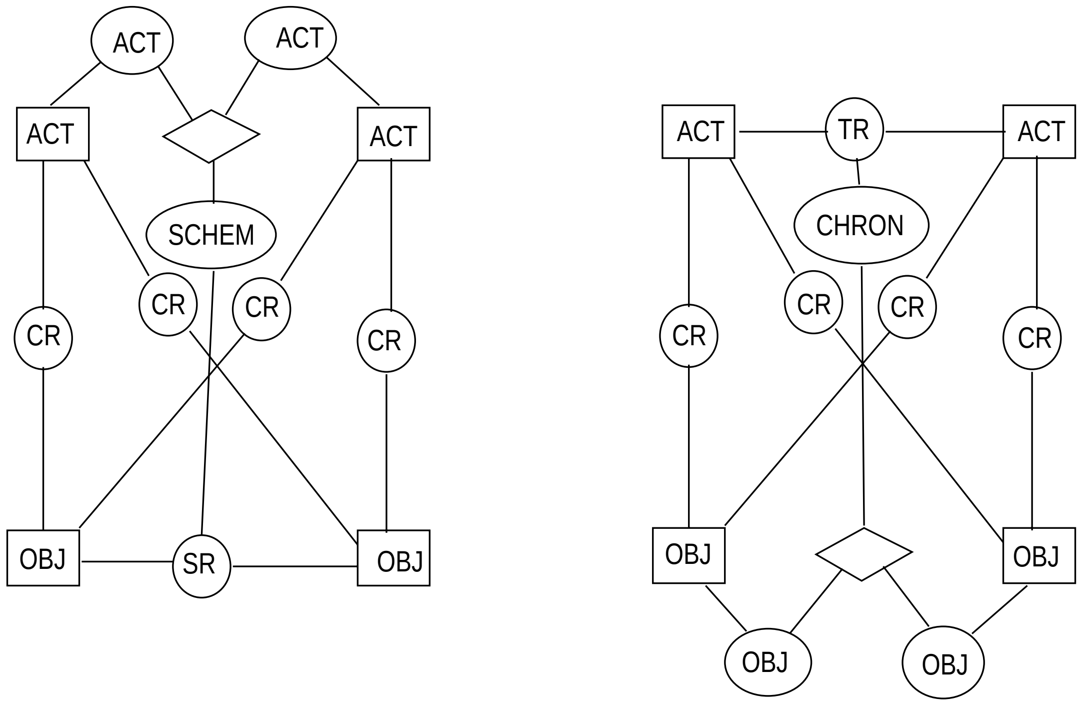


Figure 13: The canonical graph with actors supplying missing relations. On the left a temporal actor defines an implicit relation between the acts. On the right, a spatial actor defines an implicit relation between the objects.

(see, for instance [Dechter, et al, 89], [Tsang, 87]). This will only be a problem in large, pathological cases, however. Our experience is that by concentrating on the 'small' cases mentioned before, the performance is acceptable. In other words, by limiting the program's size to only be concerned with local effects we work around the frame problem, and produce efficient software. Another problem involves simulations where cycles of cause and effect are present. One of our examples is of a bouncing ball that loses energy as it does so. The simulation has to 'wait' until the ball loses all its kinetic energy to discover when the ball will roll instead of bounce, and how many times it will bounce. CP provides no meta-level reasoning to infer these results without running the simulation.

8.1 An extended example — the double play in baseball

We now present an extended example that brings together all of the ideas discussed above. We presume that the reader has some familiarity with the American game of baseball, although a deep understanding of the many complex rules is not necessary. Some of the parts of the example have been shown already, when we referred to the throwing and catching of balls. The example is the "standard" double play in baseball. We assume that the batter (B) has hit the ball toward short stop (SS) with a man on first base (the runner, R). SS has the ball and attempts to initiate the double play by throwing to second base (2), where the fielder 2B is standing. If 2B catches the ball before the runner touches the base, then the runner is out. Fielder 2B, in turn, throws to first base (1) where the fielder 1B is standing. Again if 1B catches the ball before the batter touches base 1, then he is out. If both throws are successful, then a double play has been 'turned'. Either runner may be out, independently, or neither. Figure 14 is a picture showing the initial arrangement of the fielders and runners. Our task is to show how a qualitative framework can be set up in which it is possible to apply numeric reasoning to discover whether the double play will succeed or not. We will not show this numeric working, which would involve the actual times, distances and speeds of runners and the ball as it is thrown, but we will show how the framework can be set up.

Firstly, there are three agents involved, the fielders, whose actions need to be coordinated. SS throws the ball to 2B, who catches it, and throws it to 1B who catches it (we assume for simplicity, that both fielders are touching their respective bases all the time). Then there are two agents, the runners, whose actions are uncoordinated, but it is usual for the runner at first base to start running before the ball is hit. The runners run and tag the base to which they are running. There may be complications such as "getting a lead", "stealing a base" that we shall ignore. However, the qualitative analysis should provide for the runners either succeeding or failing to beat their respective throws. There are three objects apart from the human agents. They are the ball, and the two bases, 1 and 2. Figure 16 shows the spatial configuration of objects, and figure 15 shows the temporal configuration of acts. Note that there is no requirement for either graph to be connected, especially where no relation can be computed.

The next diagram, figure 17 shows, in summary form, the events/experiences. The top line shows the acts, and the bottom line show the objects. No temporal or spatial relations are included here. The case relations are omitted for clarity.

It remains to show how the schematic can be computed from known temporal relations, or the chronicle from known spatial ones. If there are some of each that are known, then the missing ones can be computed using the same actor-based mechanisms described above. In order to reduce the complexity we only show two localized computations, one for a temporal relation and one for a spatial relation.

8.1.1 Computing a double-play temporal relation

Figure 18 shows the graphs for catch and throw with their temporal overlays. Below each graph is the time chart for the actor. This time chart is part of the definition of the graph, i.e. it is part of the knowledge of what it is to throw a ball to a fielder or to catch it. Figure 19 shows the two combinations of the individual catch and throw time charts (the two graphs join in two ways on $FIELDER$ and $BALL$). It shows that if the sequence of acts is $THROW$–$CATCH$ (one fielder to another) then $THROW$ must precede $CATCH$

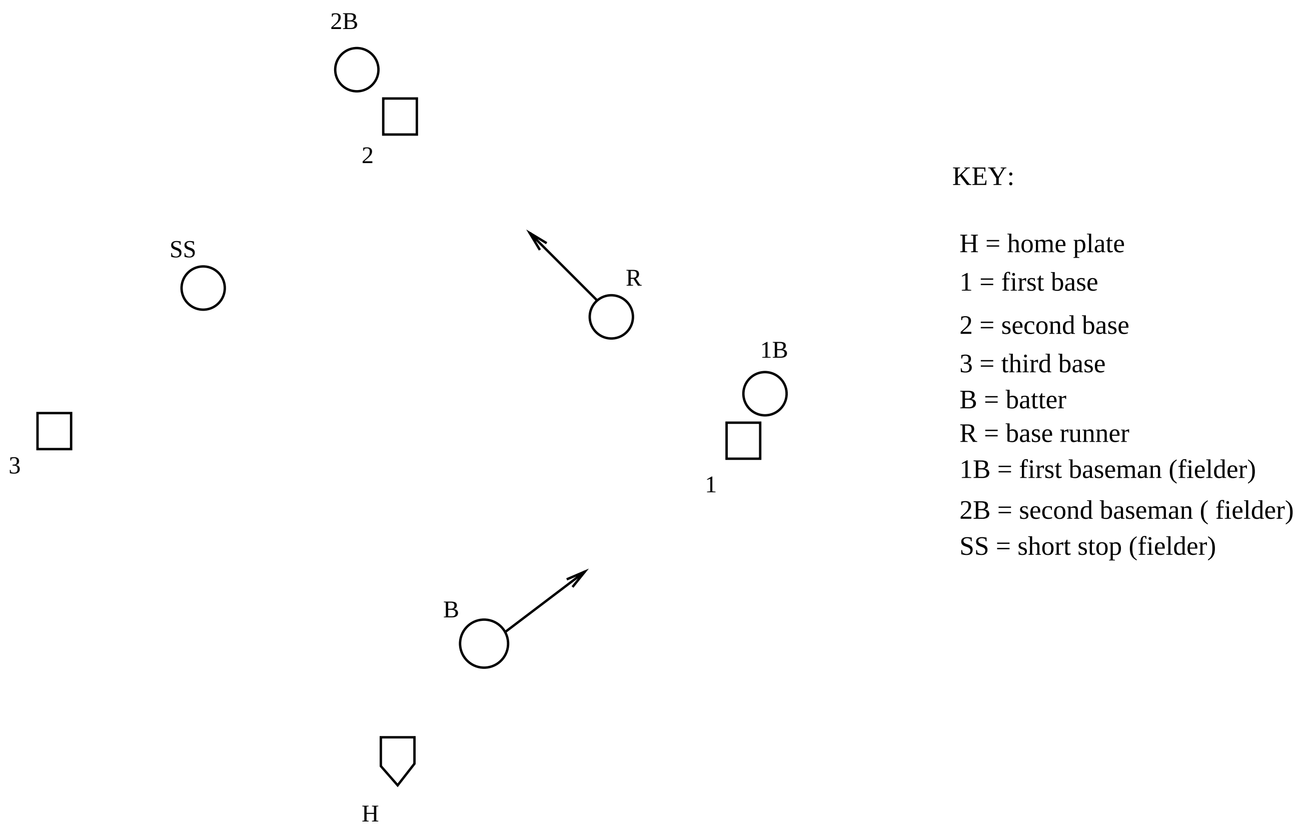


Figure 14: The “standard” double play in baseball

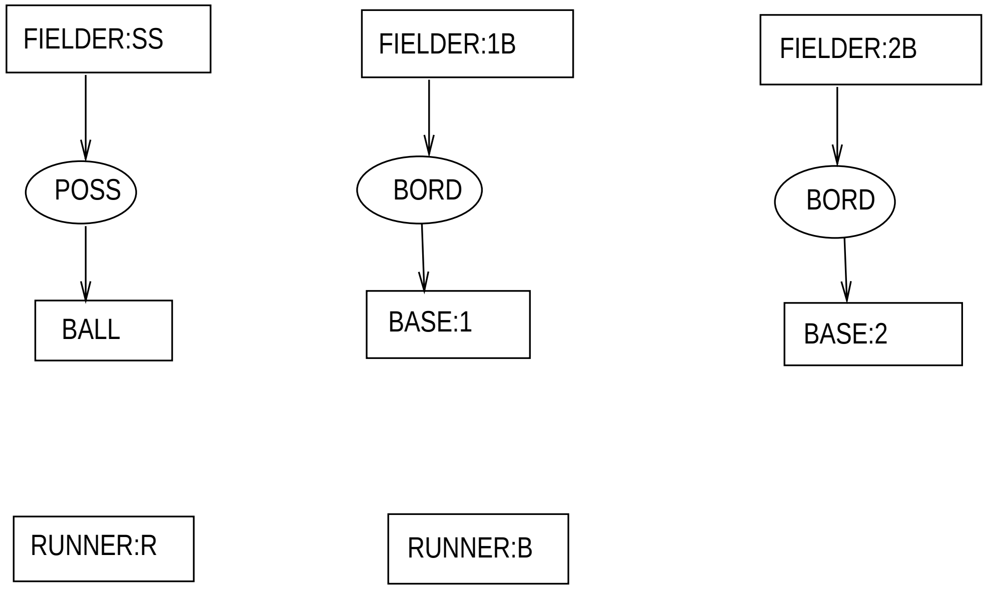


Figure 15: The double play schematic, representing the spatial configuration of the relevant objects at the time the double play starts.

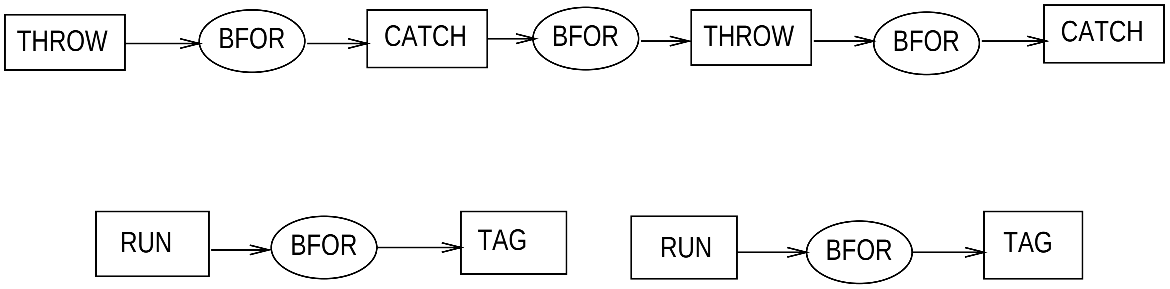


Figure 16: The double play chronicle, representing the temporal configurations of the relevant events during the double play. The top graph is the short stop throwing to second base before the second baseman catches it and throws it to the first baseman, who catches it. The bottom two graphs show the independent acts of the two runners, the batter and the base runner.

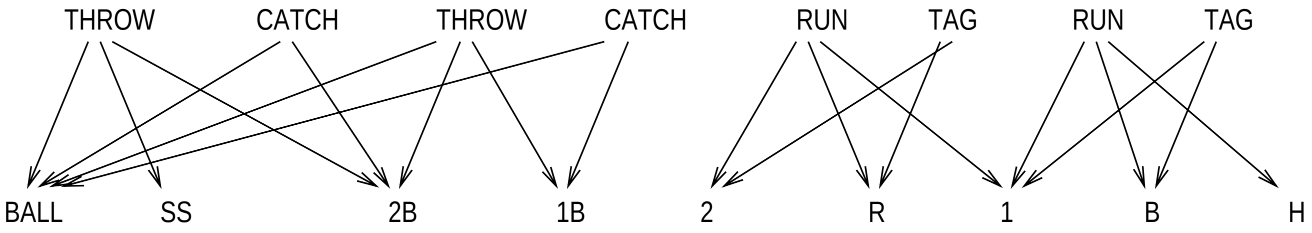


Figure 17: The double play events/experiences. This shows the (generic, unspecified) linkages between the acts on the top row, and the objects on the bottom. Read downwards there are eight events; read upwards there are nine experiences.

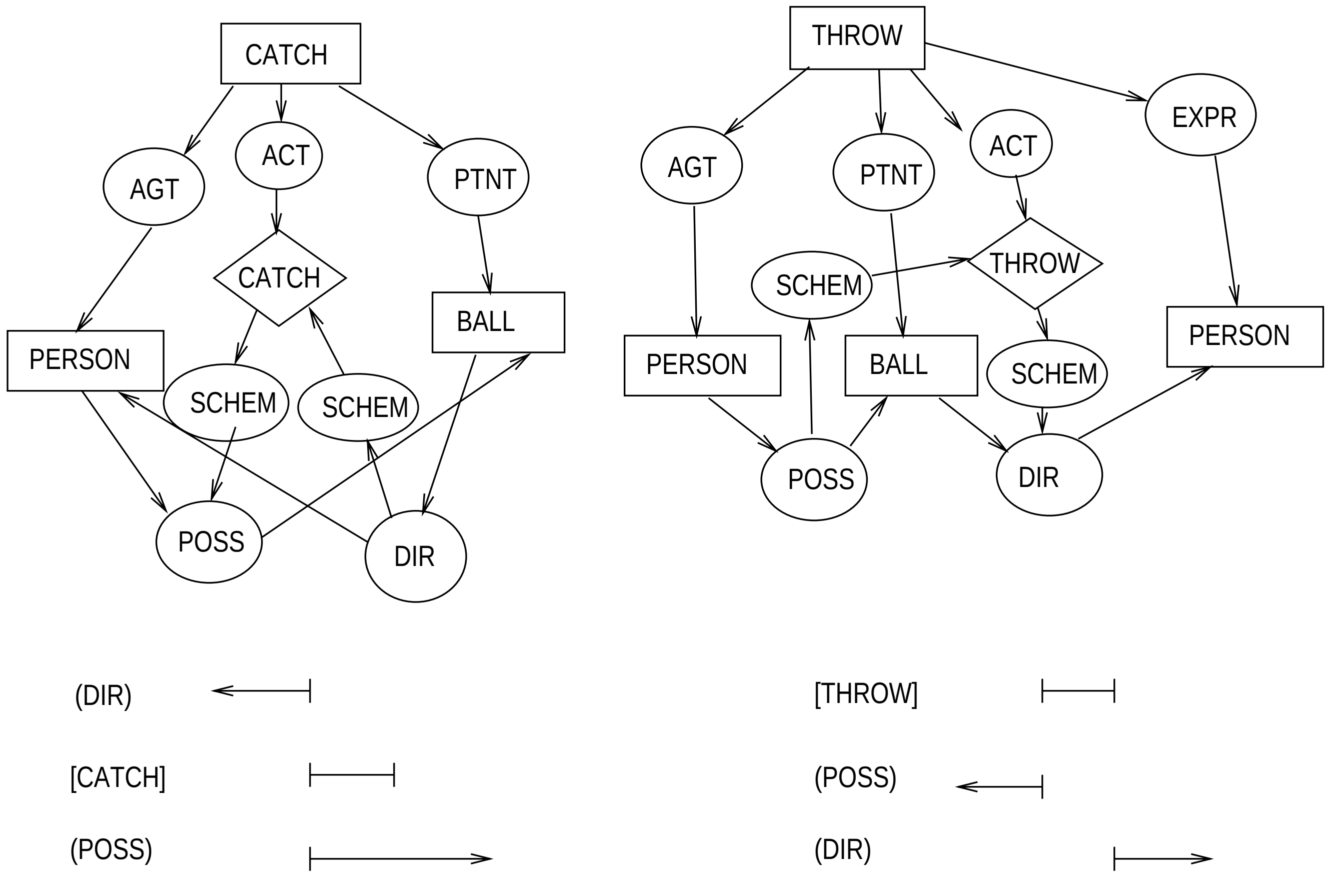


Figure 18: The graphs for CATCH and THROW, with individual time charts.

(hence the temporal relation computed is $BFOR$). However, if the sequence is $CATCH–THROW$ (by the same fielder), then there is ambiguity. Either the throw comes after the catch, when the fielder hangs on to the ball for a period of time, or the throw immediately follows the catch. The ambiguity is show by the dashed line for the interval for $POSS$ after $CATCH$. This indicates a partial ordering among the end-points of these intervals. In the $CATCH–THROW$ case we might talk of catch and throw 'in one motion'. These time charts are produced by constraint propagation through the actor network. In this example there are only two actors, but in general there could be a network of multiply connected actors where there are multiple events happening simultaneously.

8.1.2 Computing a double-play spatial relation

Just as a temporal actor can constrain the end-points of open-ended time intervals, a spatial actor can constrain the boundaries of regions. Figure 20 shows a spatial actor constraining the regions occupied by two bases, the runner running between them and the region over which RUN precedes TAG. In the example, the only uncertain spatial relations concern the runners, hence this actor (one for each runner) is the only one used. The actor constrains the partial chronicle to occupy exactly the same region as the runner's, which in turn must touch (the relation BORD) both bases. The spatial map is produced by merging the dyadic relation diagrams from Figure 10. Here two $BORD$ diagrams have been merged on the region occupied by the runner during the time when run comes before tag.

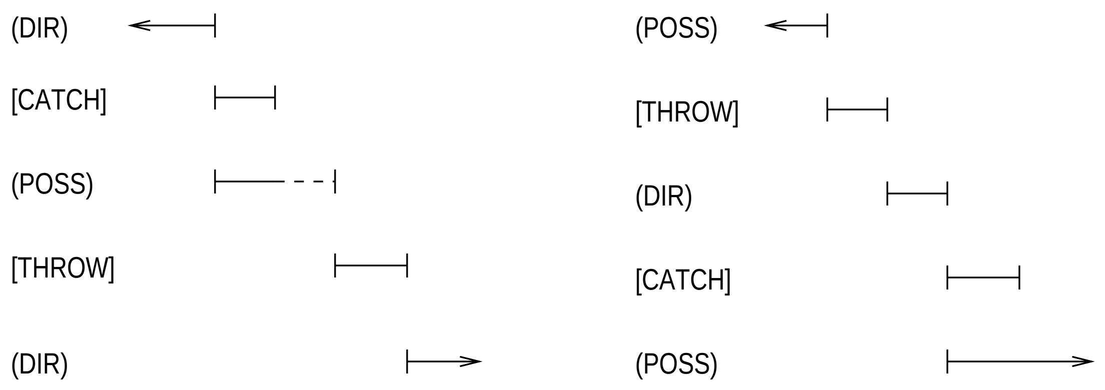
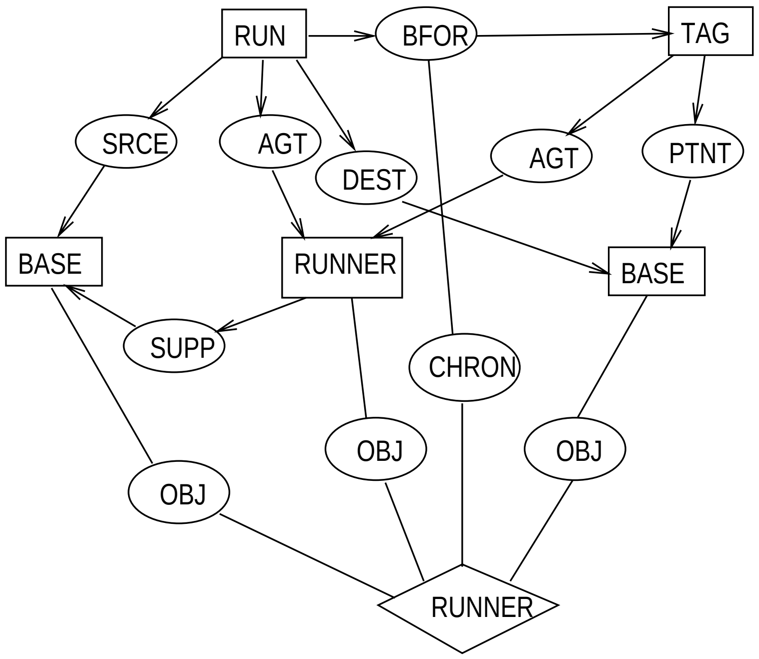


Figure 19: The two possible time charts for CATCH and THROW combined.

9 Related work

Much of the work presented here matches similar efforts in either spatial or temporal reasoning. The ontological discussions have their origins in Hayes' work in naive physics ([Hayes, 85a], [Hayes, 85b]), but more philosophical treatments can be found in, for instance in [Rescher, 68]. Related artificial intelligence treatments can be found in [Davis, 90]. Conceptual graphs are becoming popular in a variety of fields. Applications are now under way in natural language ([Sowa, 90]), and knowledge acquisition ([Velardi, 88]), as well as numerous knowledge engineering efforts. Directly related work can be found in [Esch and Nagle, 90] and [Moulin and Côté, 90] where the base conceptual graph theory is used to represent time and temporal happenings. Randell and Cohn (op cit) come to similar conclusions regarding the spatial relations, citing work by Kautz in [Hobbs et al., 85]. Allen's work in the temporal domain (Allen, op cit) remains seminal, through several attempts to build on it.

The use of explicitly represented inference mechanisms (our spatio-temporal actors) has origins in the whole knowledge engineering enterprise, where explicit meta-rules for control were the name of the game (see, for instance, [Clancey, 83]). Shapiro and his co-workers have advocated the inclusion of explicit inference nodes in the semantic networks ([McKay and Shapiro, 81]), although we believe we are the only attempt to extend conceptual graphs in the same way. Our network of actors follows work in constraint propagation (Dechter et al., op cit.; Tsang, op cit), although, as we have stated before, we believe that the 'small' case nature of our representations can keep the complexity in bounds.



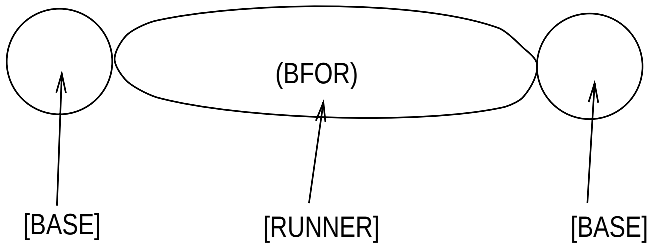


Figure 20: The graph for RUNNER with associated spatial map

10 Conclusions

We have shown that spatial and temporal reasoning can be integrated under a representation formalism founded upon dualities in the concepts appropriate in each area. We have defined and given examples of the spatial concepts of state, schematic and snapshot, and the temporal concepts of process, chronicle and history, as well as the linking concepts of event, experience and world. We have also shown how the elements of reasoning with these concepts can be assembled into knowledge structures that are capable of representing complex world situations and how, through the idea of simulation, such a representation may be validated. A working implementation of these ideas, the Conceptual Programming knowledge representation environment exists and has been used in several reasoning tasks.

References

[Allen, 85] Allen, J. Maintaining Knowledge about Temporal Intervals. *Comm. of the Association of Computing Machinery*, 26(11), 1985, pp. 832-843.

[Clancey, 83] Clancey, W.J. The advantages of abstract control knowledge in expert system design, in Proceedings of 3rd. National Conference on Artificial Intelligence, Washington, D.C., 1983, pp. 74-78.

[Coombs and Hartley, 88] Coombs, M. J. and R. T. Hartley. Explaining novel events in process control through Model Generative Reasoning. *Int. J. Expert Systems.* 1, 1988, pp. 87-109.

[Coombs and Hartley, 87] Coombs, M. J. and R. T. Hartley. The MGR algorithm and its application to the generation of explanations for novel events. *Int. J. Man-Machine Studies.* 27, 1987, pp 679-708.

[Davis, 90] Davis, E. Representations of Commonsense Knowledge. San Mateo: Morgan Kaufmann, 1990.

[Dechter, et al, 89] Dechter, R, Meiri, I, and Pearl, J. Temporal Constraint Networks. Proceedings of KR'89, Principles of Knowledge Representation and Reasoning, San Mateo, CA: Morgan Kaufmann, pp. 83-93.

[Dean and McDermott, 87] Dean, T.L and D.V. McDermott (1987). Temporal data base management. Artificial Intelligence, 32, 1-55.

[Esch and Nagle, 90] Esch, J.W. and Nagle, T.E. Representing temporal intervals using conceptual graphs, in Proceedings of the Fifth Annual Workshop on Conceptual Structures, Boston, 1990.

[Eshner and Hartley, 88] Eshner, D.P. and Hartley, R.T. Conceptual programming with constraints. Proc. Conceptual Graphs Workshop, Minneapolis, 1988.

[Hammond, 86] Hammond, K. J. CHEF: A model of case-based planning. Proc. Natl. Conf. on Artificial Intelligence, 1986. Los Altos, CA: Kaufmann, pp. 267-271.

[Hartley, 88] Hartley, R.T. CP - The Manual. Memorandum MCCS 88-128, Computing Research Laboratory, New Mexico State University, 1988.

[Hayes, 85b] Hayes, P. Ontology for liquids, in Hobbs, J.R. and Moore, R.C. (eds.) Formal Theories of the Commonsense World. Ablex, 1985.

[Hayes, 85a] Hayes, P. The second naive physics manifesto. In J.R. Hobbs and R.C. Moore (eds), Formal Theories of the Commonsense World. Norwood, NJ: Ablex, 1985.

[Hobbs et al., 85] Hobbs, J.R. et al. Commonsense summer: final report. Center for the Study of Language and Information, Report CLSI-85-35, 1985.

[Lansky and Fogelsong, 87] Lansky, A.L. and Fogelsong, D.S., Localized representation and planning methods for parallel domains. Proc. Natl. Conf. on Artificial Intelligence, Seattle, 1987, pp. 240-245.

[McDermott, 82] McDermott, D. A temporal logic for reasoning about processes and plans.

[Moulin and Côté, 90] Moulin, B. and Côté, D. Extending the conceptual graph model for differentiating temporal and non-temporal knowledge, in Proceedings of the Fifth Annual Workshop on Conceptual Structures, Boston, 1990.

[Randell and Cohn, 89] Randell, D.A. and Cohn, A.G. Modeling topological and metrical properties on physical processes. Proceedings of KR'89, Principles of Knowledge Representation and Reasoning, San Mateo, CA: Morgan Kaufmann, 1989, pp. 357-368.

[Rescher, 68] Rescher, N. (ed.) The logic of decision and action. Pittsburg University Press, 1968.

[McKay and Shapiro, 81] McKay, D.P, and Shapiro, S.C. Using active connection graphs for reasoning with recursive rules, in Proceedings of Seventh International Joint Conference on Artificial Intelligence, Vancouver, 1981, pp. 368-374.

[Shoham, 88] Shoham, Y., Chronological Ignorance: Experiments in Nonmonotonic Reasoning. *Artificial Intelligence*, 36, 1988, pp. 279-331.

[Sowa, 90] Sowa, J.F. Lexical structures and conceptual structures, in Pustejovsky, J. Semantics in the lexicon, Kluwer.

[Sowa, 84] Sowa, J. Conceptual Structures. Reading, MA: Addison-Wesley, 1984.

[Tsang, 87] Tsang, E.P.K. The consistent labeling problem in temporal reasoning. Proc. Natl. Conf. on Artificial Intelligence, Seattle, 1987.

[Velardi, 88] Velardi, P, and Pazienza, M.T., Computer-aided acquisition of lexical co-occurrences, in Proceedings of Conference of the Association of Computational Linguistics, Vancouver, 1989.