# Qualitative Simulation Based On A Logical Formalism Of Space And Time *

Z. Cui, A.G. Cohn and D.A. Randell
Division of Artificial Intelligence
School of Computer Studies
University of Leeds, Leeds, LS2 9JT, England
{cui,agc,dr}@dcs.leeds.ac.uk

## Abstract

We describe an envisionment-based qualitative simulation program. The program implements part of an axiomatic, first order theory that has been developed to represent and reason about space and time. Topological information from the modelled domain is expressed as sets of distinct topological relations holding between sets of objects. These form the qualitative states in the underlying theory and simulation. Processes in the theory are represented as paths in the envisionment tree. The algorithm is illustrated with an example of a simulation of phagocytosis and exocytosis - two processes used by unicellular organisms for garnering food and expelling waste material respectively.

## Introduction

Envisionment-based simulation programs used in Qualitative Reasoning (QR) are now well established. The notion of an envisionment originated in de Kleer's NEWTON program, but now appears as a central program design feature in many QR simulation programs - see Weld and de Kleer (1990). An envisionment takes a set of predetermined qualitative states, and expresses them in the form of graph or a tree. This represents a temporal partial ordering of all the qualitative states a modelled physical system can evolve into given some indexed state. The term "envisionment" refers to the generated tree of possible states of a modelled system, the term "envisioning" to the actual process of deriving this tree. Envisionments can be attainable or total. Attainable envisionments generate the tree from some particular initial state of the modelled system; total envisionments are generated from all possible states - see Weld and de Kleer (1990) for examples of both types. Our simulation program currently produces an attainable envisionment.

The simulation program described below shares many of its general design features with Kuipers' (1986) QSIM approach to qualitative simulation. QSIM uses a set of symbols that represent physical parameters of a modelled system, together with a set of constraint equations (which are taken to be qualitative analogues of standard differential equations commonly used in mathematics and physics). The qualitative simulation starts with a structural description of the modelled domain (being the description of the parameters and constraint equations which relate the parameters to each other) and an initial state. The program produces a tree which represents the initial state of the system as the root node, and possible behaviours of the modelled system as paths in the tree from the root node to its leaf nodes.

In our simulation program, QSIM's physical parameters map to a set of mutually exhaustive and pairwise disjoint set of dyadic relations that can hold between pairs of regions. Similarly, QSIM's set of transition rules map to a set of transition rules in our theory (which determine the manner in which pairs of objects can change their degree of connectivity over time), and QSIM's constraint model maps to domain independent and dependent constraints that apply to states, and between adjacent states. While both QSIM and our simulation program take particular physical systems as a model, unlike QSIM, our simulation program first requires the user to abstract out a logical description of the physical model in terms of a set of topological relationships holding between the set of objects in the modelled domain. An analogue of QSIM's consistency filtering also appears in our simulation program.

The structure of the rest of the paper is as follows. First we outline that part of the underlying theory upon which the present simulation program is based. Then we discuss the simulation program. We give an example model and resulting envisionment, and finally we discuss related and future work.

## Overview of the Spatial Theory

The formal theory which underpins the simulation program (see Randell and Cohn 1989, Randell, Cohn and Cui 1992 and Randell 1991) is based upon Clarke's (1981, 1985) calculus of individuals based on "connection" and is expressed in the many sorted logic LLAMA (Cohn 1987). The theory supports regions having either a spatial or temporal interpretation. Informally, these regions may be thought to be infinite in number,

*The support of the SERC under grant no. GR/G36852 is gratefully acknowledged.

and any degree of connection from external contact to identity is allowed in the intended model.

The basic part of the formal theory assumes a primitive dyadic relation: C($x, y$) read as 'x connects with y' which is defined on regions. C($x, y$) is reflexive and symmetric. In terms of points incident in regions, C($x, y$) holds when regions $x$ and $y$ share a common point. Using the relation C($x, y$), a basic set of dyadic relations are defined. These relations are DC (is disconnected from), P (is a part of), PP (is a proper part of), = (is identical with), O (overlaps), DR (is discrete from), PO (partially overlaps), EC (is externally connected with), TP (is a tangential part of), NTP (is a nontangential part of), TPP (is a tangential proper part of), NTPP (is a nontangential proper part of), TPI (is the identity tangential part of), and NTPI (is the identity nontangential part of). The relations P, PP, TP, NTP, TPP and NTPP support inverses. Of the defined relations, the set DC, EC, PO, TPP, NTPP, TPI, NTPI, and the inverses for TPP and NTPP form a mutually exhaustive and pairwise disjoint set. From now on we shall refer to this particular set, as the set of base relations defined solely in terms of the primitive relation C. A pictorial model for this set of base relations (excepting the relation NTPI) is given in Figure 1 [1]. Atomic formulae whose predicate symbol is a base relation will be called basic atoms. Note that all the relations described above can be expressed as disjunctions of sets of base relations.

For the temporal part of the theory assumed by the simulation program, we first introduce temporal regions into our ontology, which we call periods. Periods are subdivided into intervals and moments, where a moment is defined as a period that has no constituent parts such that one part is before another. In addition to periods, a new primitive relation of temporal precedence 'B($x, y$)' read as '$x$ is before $y$' is added to the formalism and axiomatised to be irreflexive and transitive. A set of 13 dyadic temporal relations are then defined - see Randell (1991). These may be viewed as analogues of all the 13 interval relations common to interval logics - see e.g. Allen and Hayes (1983). However, for the purposes of this paper, only the relation Meets($x, y$) which is irreflexive and transitive is needed. Two periods $x$ and $y$ are then said to meet if and only if $x$ is before $y$ and no other period $z$ exists such that $x$ is before $z$, and $z$ is before $y$.

In the general theory, an ontological distinction is made between physical objects (bodies) and the regions of space they occupy. Bodies and regions are represented in the formal theory as disjoint sorts. The mapping between the two is done by introducing a transfer function 'space($x, y$)' read as 'the space occupied by $x$ at $y$', that takes a body at a given moment in time, and maps this to the region of space it occupies. The transfer function is used in the theory to define a set of ternary relations of the form $\Phi(x, y, z)$ which are used in a set of envisioning axioms meaning that body $x$ is in relation $\Phi$ to body $y$ during period $z$. However, in this paper, the temporal parameters in formulae used in the simulation program remain implicit, e.g. the formula NTPP($n, a$) abbreviates the temporally indexed formula NTPP($n, a, t$) - where $t$ denotes a specific period during which the state obtains.

The general theory contains a set of envisioning axioms and encodes a set of theorems (derivable from the part of the theory described above) in the form of a transitivity table - cf. Allen's (1983) transitivity table. The envisioning axioms describe direct topological transitions that can be made between pairs of regions. Thus, for example, given two regions that DC in one state, a direct transition to EC is allowed, and from EC back to DC or to where the regions PO, and so on. These axioms rule out certain transitions - for example no direct transition between DC and PO is allowed. A pictorial representation of of the envisioning axioms is illustrated in Fig. 1.

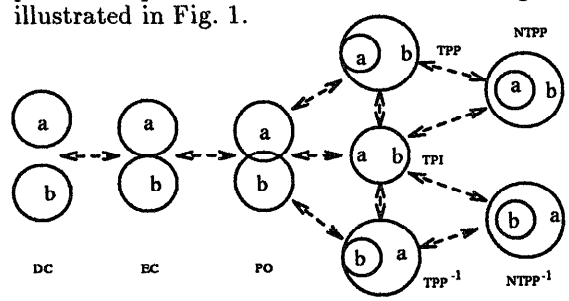


Figure 1: A pictorial representation of the base relations and their direct topological transitions.

The theory also uses a precomputed transitivity table for the set of dyadic base relations described above - for details see Randell, Cohn and Cui (1992). Each R3(a,c) entry in the table represents a disjunction of all the possible dyadic relations holding between regions a and c, for each R1(a,b) and R2(b,c) conjunction - where R1, R2, R3 are elements of the set of base relations in the theory. The transitivity table is used in the simulation program for checking consistency of state descriptions in the envisioning process.

The general theory also includes an additional primitive function 'conv($x$)' read as 'the convex hull of $x$', which is axiomatised and is used to generate a further set of dyadic relations. These additional relations are used to describe regions that are either inside, partially inside or outside other regions - see Randell, Cohn and Cui (1992). As with the set of relations defined solely in terms of C, the extended theory including the new set of inside and outside relations also admits the possibility of constructing several further sets of base relations, depending upon the degree of representational

[1] In this paper we make the assumption that all the regions are topologically closed (i.e. include their boundaries). The relation NTPI is only satisfied if the regions it is predicated on are topologically open. Thus we ignore NTPI here.

detail required by the user. For the basic extension to the theory, the set of base relations extend from 9 to 23. However, here we simply concentrate upon the set of base relations defined solely in terms of C which turns out to be sufficient to demonstrate the general utility of our approach.

## The Simulation Program

State descriptions in the simulation program are represented as conjunctions of ground atomic formulae. The program first of all takes an initial state description, then evolves successive states according to the restrictions imposed by direct topological transitions encoded in the envisioning axioms, by sets of constraints that apply within a state or between states, and by any sets of add or delete rules that sanction the introduction and deletion of named entities in the modelled domain respectively. A consistency check is made for each state, first for the initial state, and then for all potential evolved states generated in the envisioning process. The envisioning process terminates when for each path generated in the envisionment tree, the last state repeats an earlier one. Each path of states $S_1$, $S_2$, ... corresponds to a sequence of periods, $t_1$, $t_2$, ... such that Meets($t_i, t_{i+1}$), and the state description of $S_i$ obtains during $t_i$. Each complete path corresponds to a possible behaviour of the physical model as predicted by the program. However, because the transition rules always allow the possibility that the relationship between two entities continues indefinitely, each initial subpath also corresponds to a predicted behaviour.

The program requires a complete n-clique as the initial state, i.e. $n(n-1)/2$ atomic formulae. This requirement is needed for consistency and constraint checking by the program to function correctly.[2]

### Constraints

The simulation program supports *intrastate* and *interstate* constraints. Intrastate constraints are constraints that apply within a state, and interstate constraints between adjacent states - that is to say, between consecutive states, or states which meet. For example, in the physical system which is used to illustrate this simulation program below - namely modelling phagocytosis of unicellular organisms - an intrastate constraint would be the assertion that the cell's nucleus is always part of the cell, and an interstate constraint would be the fact that once the food is ingested during phagocytosis and becomes a part of the amoeba, it will remain so. Formally, both types of constraints assume the following forms:

Intrastate constraint: $\Phi$, where $\Phi$ is a quantifier free formula, and all terms are variables or constants (in this case all variables are implicitly quantified). Note here that in the current implementation of the theory, $\Phi$ must be composed of basic atoms.

Interstate constraints: $\Phi \rightarrow (R_0 \Longrightarrow (R_1 \vee ... \vee R_n))$ and $\Phi \rightarrow (R_0 \not\Longrightarrow (R_1 \wedge ... \wedge R_n))$ where $\Phi$ is as above, and the $R_i$ are basic atoms predicating the same terms. In the first case, where $\Phi$ holds, if $R_0$ then in any next state the disjunction $R_0 \vee R_1 \vee ... \vee R_n$ holds, while in the second case the disjunction $R_0 \vee R'_1 \vee ... \vee R'_m$ must hold where each $R'_i$ is a base atom predicating the same terms as $R_0$ and $R'_i \neq R_j$ for any $i$, $j$. The presence of an interstate constraint does not force a transition to take place.

### Add and Delete rules

In addition to the set of constraint rules described above, the simulation program also supports add and delete rules. Both sets of rules can be viewed as a special kind of inter-state constraints. Add rules simply sanction the introduction of new objects into the domain at the next state, and delete rules the elimination of particular objects in the next state. In the model used to illustrate our program, an example of an add rule is where, having enveloped the food, a vacuole is formed in the amoeba, while an example of a delete rule is where the vacuole containing waste material passes out of existence as it opens up and discharges its contents into its environment.

Add and delete rules assume the following forms: **add** $O_1$, ..., $O_n$ **with** $\Psi_1$ **when** $\Psi_2$. **Delete** $O_1$, ..., $O_n$ **when** $\Psi_2$ . $\Psi_1$ is a conjunction of basic atoms, and $\Psi_2$ is a quantifier free Boolean composition of atoms. $O_1$, ..., $O_n$ must be ground terms (at least in the current implementation). An add rule is fired when the 'when' condition is true for some instantiation of any free variables in the condition, and will add $O_1$, ..., $O_n$ to all next states with the specified relations. Similarly, delete rules will be fired when the 'when' condition is true and will delete all the specified objects in all next states.

### The Algorithm

The algorithm first of all takes an initial state of the modelled physical system, then proceeds to generate the envisionment. Each state in the envisioning process is checked for intrastate consistency before the next state in the envisionment is generated. The completed tree representing the envisionment has the initial state as the root node, and paths tracing to leaf nodes as distinct sequences of transitions undergone by the set of modelled objects.

The algorithm first puts the initial state $s_0$, in a set S of unexpanded states; and then executes the following steps:

[2]However, in practice, a partial description of the initial is usually supplied by the user and a) the program computes the complete description, and b) only those state descriptions that arise from evolved transitions from pairs of entities described in the initial state are explicitly represented in the envisionment. Actually, our program is slightly different to the procedure specified below but this need not concern us here.

1. If S is empty then stop.
2. Select and remove a state $S_i$ from S.
3. If $S_i$ is inconsistent, then go to 1.
4. Select applicable transition rules by applying interstate constraints.
5. Apply all the selected transition rules to produce a set of possible next states.
6. Apply add and delete rules.
7. Check intrastate constraints.
8. Add remaining states generated to S; go to 1.

We discuss the details of steps 3 to 7 in the subsections below.

**Consistency checking** In step 3 the algorithm uses a simple form of consistency checking step to filter out sets of atomic formulae (being a potential state in the simulation and thus in the physical model) whose conjunction is inconsistent in the underlying theory, and thus supports no model. In this instance, we use the results encoded in the transitivity table. Given n-objects in the modelled domain, there are exactly $n(n-1)/2$ atomic formulae in a state. In particular for each tuple of objects $x, y, z$, there are three atomic formulae of the $R_1(x, y)$, $R_2(y, z)$ and $R_3(x, z)$. Consistency checking simply consists of checking that each $R_3(x, z)$ formula is logically implied by $R_1(x, y)$ and $R_2(y, z)$ for each $y \notin \{x, z\}$. In use this is effectively the same as Allen's (1983) constraint satisfaction algorithm, except that our algorithm can be simplified since we have no "disjunctive labels", i.e. we have restricted state descriptions to predicating a single base relation to any pair of objects.

**Generating next states** In steps 4 through to 7, the algorithm takes a state produced in step 3, and proceeds to generate a new state. The selected state $S_i$ is a set of basic atoms. For each atom there are between 1 and 5 applicable transition rules - see Figure 1. In step 4 possible transitions for each atom which violate an interstate constraint are filtered out. In step 5 the remaining transitions are applied in all possible combinations to yield a set of possible next states. In step 6 the add and delete rules are then applied in that order. Finally, in step 7 any next states which violate an intrastate constraint are deleted.

**Correctness** The program terminates when for each path generated in the envisioning process, the last state repeats an earlier one. It should be evident that the algorithm will terminate if there are no add rules, but the same applies if there are finitely many add rules. This follows from the syntactic restriction on add rules, that the objects must be ground terms, so only finitely many new objects can ever be introduced.

It is important to show that all the behaviours predicted by the simulation correspond to possible behaviours of the physical system being modelled. This issue brings brings to the fore the question whether or not the simulation can be proved to be "sound" and "complete". In our case by "soundness" we need to show that every frontier of the envisionment tree (viewed as a disjunction) generated in the simulation is a provable consequence in the underlying theory, and by "completeness", to show that, given an initial state, every 'minimal' provable disjunction of conjoined basic atoms in the underlying theory will be expressed in the envisionment. Whereas Kuipers has proved the correctness of QSIM relative to ordinary differential equations, our gold standard is the logical formalism presented in Randell (1991). We have shown the system to be be sound and we conjecture the system is complete but have still to complete the proof of this.

**Complexity** The critical point about the algorithm (and its complexity) is that states are complete, i.e., all relations between all objects are explicitly given in terms of base atoms and there is no disjunctive or indefinite information. This means that all constraints and add/delete rules can be considered individually. The complexity of step 3 is $O(n^3)$ because there are $n^3 - 3n^2 + 2$ different triples given $n$ objects in a state. For step 4, suppose there are $c$ interstate constraints and each constraint contains at most $v$ variables and there are $n$ objects, then each constraint can be applied at most $C_n^v$ ways. This is polynomial of degree of $v$. Applying a constraint is linear to the number of connectives in it. For step 5, if there are $n$ objects there are $(n^2 - n)/2$ relations. The maximum branching rate in the graph for direct topological transitions is 5 (from equality) so there are at most $5^n$ successor states (but more likely $2^n$ which is of course still exponential). This compares to the situation in QSIM. In practice, consistency checking will prune the number of next states dramatically. The complexity of steps 6 and 7 is the same as step 5, i.e. $O(n^v)$.

## An Example

By way of a simple example, we shall demonstrate the simulation program by modelling cellular behaviour - in particular, the processes known as phagocytosis and exocytosis. Phagocytosis is the process by which cells surround, engulf and then digest food particles. It is the feeding method used by some unicellular organisms of which the amoeba is an example, and which is adopted here. The same process is used by white blood cells in an attempt to deal with invading microorganisms. Exocytosis is the name given to a similar inverse process where waste material originally contained in a cell is subsequently expelled from the cell.

In the proposed model, an amoeba is depicted in a fluid environment containing other organisms which are its food. Each amoeba is credited with vacuoles (being fluid filled spaces) containing either enzymes or food which the animal has ingested. The enzymes are used by the amoeba to break down the food into nutrient and waste. This is done by routing the enzymes to the food vacuole. Upon contact the enzyme and food

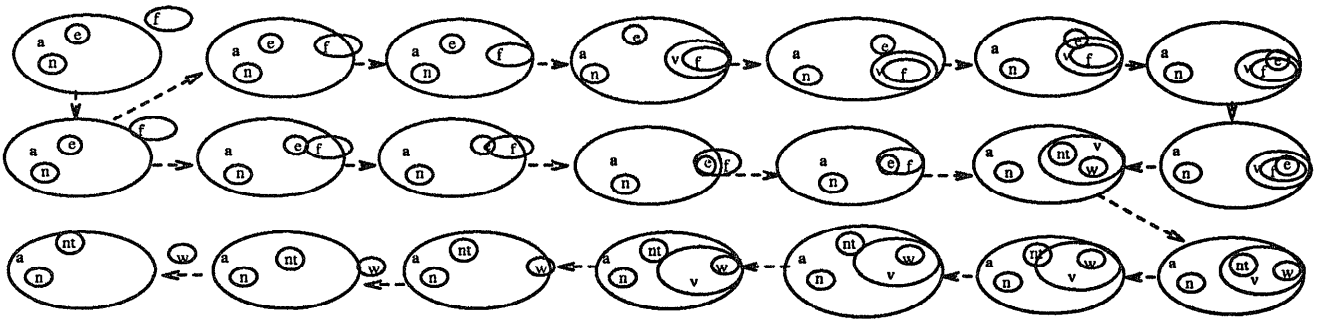

Figure 2: A pictorial representation of two paths generated in the envisionment.

vacuoles fuse together and the enzymes merge into the fluid containing the food. After breaking down the food into nutrient and waste, the nutrient is absorbed into the amoeba's protoplasm, leaving the waste material in the vacuole ready to be expelled. The waste vacuole passes to the exterior of the protozoan's body, which opens up, letting the waste material pass out of the amoeba and into its environment.

The formal description of the physical model is as follows. We assume six physical objects: a, f, n, e, nt, w and v, standing for the amoeba, its food, the amoeba's nucleus, a packet of enzymes, a body of waste material, and a vacuole respectively. In the simulation, the vacuole, the nutrient and the waste are generated dynamically as the process is undergone.

The initial state is represented by the conjunction of the following atomic formulae: DC(a,f), NTPP(n,a), NTPP(e,a), DC(n,e) and DC(e,f). [3]

Next we introduce our set of domain constraints for the physical model. First the interstate constraints:

1)EC$(f,a) \not\Rightarrow$ DC$(f,a)$ 8)NTPP$(nt,v) \not\Rightarrow$ TPI$(nt,v)$
2)PO$(f,a) \not\Rightarrow$ EC$(f,a)$ 9)EC$(nt,v) \not\Rightarrow$ PO$(nt,v)$
3)TPP$(f,a) \not\Rightarrow$ PO$(f,a)$ 10)PO$(nt,v) \Rightarrow$ EC$(nt,v)$
4)TPP$(f,a) \not\Rightarrow$ TPI$(f,a)$ 11)TPP$(nt,v) \not\Rightarrow$ TPI$(nt,v)$
5)DC$(nt,v) \Rightarrow$ DC$(nt,v)$ 12)NTPP$(f,a) \not\Rightarrow$ TPI$(f,a)$
6)EC$(w,a) \not\Rightarrow$ PO$(w,a)$ 13)EC$(e,f) \not\Rightarrow$ DC$(e,f)$
7)PO$(w,a) \Rightarrow$ EC$(w,a)$ 14)PO$(e,f) \Rightarrow$ TPP$(e,f)$

Constraints 1 to 3, 6 and 7, and 13 and 14 respectively impose a unidirectionality of movement between the food and the amoeba, between the waste material and the amoeba and between the enzyme packet and the food. In the first case when the food is in contact with the amoeba it is always ingested to become a proper part of the animal; in the second case once the waste material is in external contact with the animal, it will never be reingested, and in the last case once the enzyme packet contacts the food, it will always pass into it becoming a part. Constraints 4 and 8, and 6 and 7 respectively impose the conditions that once the food is ingested (and is thus a proper part) it will remain a proper part of the animal, and that nutrient once produced (being a proper part of the vacuole) remains a proper part. Without these constraints the transition from being a proper part to being identical sanctioned by the envisioning axioms is not violated; this would simply result in a possible state being generated in the envisionment with the amoeba being part of the food, and the vacuole part of the nutrient!

The interstate constraints are all straightforward to understand and just impose the obvious static topological constraints between the domain entities.

NTPP(e, a), PP(nt, a), PP(v, a), DR(n, v), DR(n, e), NTPP(n, a), PP(w, v), PP(f, v), PP(w, a) $\rightarrow$ PP(v,a)

In the simulation, two add-rules are given. The first rule introduces nutrient and waste into the food vacuole when the enzyme packet is a proper part of the food, while the second rule sees the creation of the vacuole when the food is a proper part of the amoeba. The delete rules govern the deletion of the enzyme and food, and vacuole respectively. Since the first add rule below contains no basic atoms in the 'with relations' component, it is actually schematic for 4 rules in which only basic atoms are used.

**add** $nt, w$ **with** PP$(nt,v) \wedge$ PP$(w,v)$ **when**TPP$(e,f)$
**add** $v$ **with** TPP$(v,a) \wedge$ TPP$(f,v)$ **when** TPP$(f,v)$
**delete** $e, f$ **when** P$(e,f)$
**delete** $v$ **when** TPP$(v,a) \wedge$ PP$(w,v) \wedge$ DR$(nt,v)$

The simulation program produces an envisionment with 76 distinct states. Our constraints are sufficiently strong because each complete path corresponds to the English description of phagocytosis and exocytosis given above. A pictorial representation of two paths generated in the envisionment is given in Fig. 2.

In both paths generated we can see that the food is

[3] In the initial state, since there are 5 objects, there are really 10 relationships to be specified. As mentioned earlier, the program expands a user supplied partial description to a complete description. In fact although the formula DC(e,f) is formally derivable in the general theory from the first four atomic formulae, it is represented explicitly in the input language here, otherwise no relation between e and f will be generated in subsequent states in the envisioning process - see earlier footnote.

ingested by the amoeba, a vacuole is formed which then contains that food, digestion takes place transforming the food into nutrient and waste, and finally the waste is expelled. Note that in one path the enzyme packet begins to be absorbed into the food before the food is completely enveloped by the amoeba, while in another path the vacuole is formed before the enzyme packet is similarly absorbed.

Altogether there are 6 terminal states although there are 264 paths leading from the initial state to these final states representing different orderings of the topological transformations. However all the complete paths predict that phagocytosis and exocytosis will be undergone. Some of the paths exhibit oscillatory behaviour.

## Related and Further Work

For a detailed discussion of the ontology and formalism used in the simulation see Randell (1991) . We have already discussed the relationship between this simulation program and Kuiper's QSIM above. The volume (Weld and de Kleer 1990) contains several papers on qualitative spatial simulation, Forbus (1980) reports on a simulator called FROB and Gardin and Meltzer (1989) describe an analogical spatial simulator. Forbus et al (1991) in the context of the CLOCK project give a general framework for qualitative reasoning concerning mechanisms, but all use very different ontologies to our work.

We mentioned how further dyadic relations describing bodies that are either inside, partially inside or outside each other. This could be exploited in the amoeba example to give a richer and more realistic model where the food can be made to pass from being outside to being inside the animal, and then options would be available once the food has been engulfed to whether the food is modelled as forming a part of the animal, or not. However, in order to do this an extended transitivity table needs to be built, containing at least 529 cells. This is a formidable task which we have not yet completed; we have recently constructed a transitivity table via a program which reasons about bitmap representation of space but the resulting transitivity table has not yet been verified with respect to the modeling theory. It should also be pointed out here that in addition to the set of inside and outside relations mentioned, an set of containment relations can also be defined and exploited, in which one body completely wraps around another - see Randell (1991). At present the modelling primitives simply capture topological information. These could be extended to include metric information, capturing for example notions of relative size and distances between objects. The possibility of introducing a metric extension to the theory is outlined in Randell (1991). Further envisaged extensions to the theory that would include a subtheory of motion to the modelling language, for at present motion is represented implicitly by specified topological transitions between sets of objects. Other useful extensions would include explicit information about causality and processes, the latter including teleological accounts of a physical system's behaviour.

In the present implementation, constraints and objects have to be individually specified. It would be useful to generalise this restriction to allow for generic constraints and typed objects in the programs description language, relating individuals of particular types. The syntax of constraints described in this paper is not necessarily the most liberal that could be efficiently implemented, but corresponds to the current version of the system. We intend to investigate this expressiveness/efficiency trade off.

## References


Allen, J. F. 1983. Maintaining knowledge about temporal intervals. *CACM*, Vol. 26, pp. 832–843.

Clarke, B. L. 1981. A Calculus of Individuals Based on Connection. *Notre Dame J. of Formal Logic,* 2(3) 204–218.

Clarke, B. L.1985. Individuals and Points. *Notre Dame Journal of Formal Logic* 26(1) 61–75.

Cohn, A. G.1987. A More Expressive Formulation of Many Sorted Logic. *J. of Automatic Reasoning.*, 3(2) 113–200.

Forbus, F. 1980. Spatial and Qualitative Aspects of Reasoning about Motion. *in Proc. AAAI-80*, Morgan Kaufmann, San Mateo, Ca.

Forbus K. D., Nielsen P. and Faltings B. 1991. Qualitative spatial reasoning: the CLOCK project, *Art. Int.* 51, pp. 417–471, Elsevier.

Gardin F., and Meltzer B. 1989. Analogical Representations of Naive Physics. *Art. Int.* 38: 139–159.

Kuipers, B. 1986. Qualitative Simulation. *Artificial Intelligence* 29: 298–338.

Randell, D.; Cohn, A. G.; and Cui, Z. 1992. Naive Topology: modeling the force pump. *in Recent Advances in Qualitative Reasoning,* ed B Faltings and P Struss, MIT Press, in press, 1992.

Randell, D. A. 1991. Analysing the Familiar: Reasoning about space and time in the everyday world. Ph.D. Thesis, Dept. of Comp. Sci., Warwick Uni. UK

Randell, D., and Cohn, A. G. 1989. Modeling Topological and Metrical Properties in Physical Processes. *in Proc. of the 1st Int. Conference on Principles of Knowledge Representation and Reasoning,* ed. R.J. Brachman, et al, Morgan Kaufmann, San Mateo, Ca.

Randell, D. A.; Cohn, A. G.; and Cui, Z. 1992. Computing Transitivity Tables: a Challenge for Automated Theorem Provers. to appear in Proceedings of CADE–11, Springer Verlag.

Weld, D., and Kleer, J. de. 1990. *Readings in Qualitative Reasoning about Physical Systems.* Morgan Kaufmann, San Mateo.