

Reasoning about Space in SNePS

Pedro Moniz
mpc@ursula.inesc.pt

Miguel Correia
mpc@ursula.inesc.pt

Nuno J. Mamede
njm@inesc.pt

Instituto Superior Técnico / INESC
Technical University of Lisbon
Apartado 13069
1000 Lisboa, Portugal

Abstract

The development of non-trivial domain knowledge representation and reasoning, as naive physics, is becoming an important task of AI. One of the challenges that emerge in this domain is the simulation of physical systems, which requires a lot of theorem proving, planning, constraint satisfaction and consistency checking. We have used a general purpose semantic network system to implement a spatial logic based on connectivity and outlined a basic support for qualitative simulation.

1 Introduction

It has been pointed out by several authors [Randell and Cohn 89] that much of the discourse about the every day world exploits topological and metrical rather than geometrical information. Expressions like "side by side", "just after" or "between" are more common than "the car with its right side parallel to the mail box" or "the round box". The topological information is important for the description of the surrounding world in natural language. So it is natural to think that a formalism that can capture this kind of information and support deduction should be very important to AI. Randell, Cohn and Cui developed a logic formalism, based on the dyadic relation "connectivity", which can be used to reason about space [Randell, Cui and Cohn 92].

Our work is an implementation of this spatial logic [Randell and Cohn 89] [Randell, Cui and Cohn 92] [Cui, Cohn and Randell 92] in SNePS (Semantic Network Processing System) - allowing both representation and reasoning about space. It is also a framework for the implementation of a qualitative simulation system that could infer the transactions of a physical system between an initial and a final state.

SNePS [Shapiro 79] [Shapiro and Rapaport 87] [Shapiro 91] [Shapiro and Rapaport 91] is a knowledge representation and reasoning system based on the semantic net paradigm. Its emphasis is on chaining with rules in "common-sense" reasoning situations as well as a theorem prover. The SNePS utilization in a practical case as the present one is extremely important to reveal problems and suggest new directions of development.

The structure of this paper is as follows. Section 2 describes the space ontology: the set of logical relations that allow description and reasoning about space entities; the relational lattice that defines their subsumption hierarchy; an underlying theory for qualitative simulation; and a transitive table between logical relations. Section 3 describes how the space ontology previously described was implemented in SNePS. First we present one possible representation of the space ontology in SNePSLOG and point out some restrictions imposed by SNIP (the SNePS Inference Package) and ways to overcome them. Then we show our final representation, conditioned by the present technological constraints, which can be used

to perform topological deductions, to prove topological theorems or to make spatial simulations. Section 4 contains some examples.

2 The formalism of the Space Ontology

2.1 The Spatial Logic

A spatial logic is a set of logical relations that allow description and reasoning about space entities. The space ontology implemented by us in SNePS is a small subset of the theory described by Randell, Cui and Cohn in [Randell, Cui and Cohn 92]. This subset is described by Randell and Cohn in [Randell and Cohn 89] and it is based on former work by Clarke.

The space ontology is based on a logical description of the space using regions and their connections. This is a simplification of the theory presented by Randell, Cui and Cohn that, beside regions, also includes physical objects and other types of entities. Until now, only the logic of regions has been sufficiently described [Randell and Cohn 89] [Randell, Cui and Cohn 92]. Although regions can have both a spatial and a temporal interpretation, in our work they are always interpreted as spatial regions.

When two regions share a common point, it is said they are connected, represented by the primitive relation - $C(x, y)$ - read as "x connects with y", which is reflexive and symmetric.

From the basic connection relation, a set of dyadic relations is defined to describe different kinds and specialization of connections. These relations are: $DC(x, y)$ (read as "x is disconnected from y"); $P(x, y)$ ("x is part of y"); $PP(x, y)$ ("x is proper part of y"); $EQUAL(x, y)$ ("x is identical with y"); $PO(x, y)$ ("x partially overlaps y"); $EC(x, y)$ ("x is externally connected with y"); $TPP(x, y)$ ("x is tangential proper part of y") and $NTPP(x, y)$ ("x is non tangential proper part of y"). These relations can be defined from the basic connection relation by the following set of axioms:

$$\begin{aligned}
 DC(x, y) &\equiv \neg C(x, y) \\
 P(x, y) &\equiv \forall z [C(z, x) \rightarrow C(z, y)] \\
 PP(x, y) &\equiv P(x, y) \wedge \neg P(y, x) \\
 EQUAL(x, y) &\equiv P(x, y) \wedge P(y, x) \\
 O(x, y) &\equiv \exists z [P(z, x) \wedge P(z, y)] \\
 PO(x, y) &\equiv O(x, y) \wedge \neg P(x, y) \wedge \neg P(y, x) \\
 DR(x, y) &\equiv \neg O(x, y) \\
 TPP(x, y) &\equiv PP(x, y) \wedge \exists z [EC(z, x) \wedge EC(z, y)] \\
 NTPP(x, y) &\equiv PP(x, y) \wedge \neg \exists z [EC(z, x) \wedge EC(z, y)] \\
 EC(x, y) &\equiv C(x, y) \wedge \neg O(x, y)
 \end{aligned}$$

2.2 The Hierarchy

The relations introduced in the former section, can be embedded in a relational lattice that define their subsumption hierarchy (Figure 1). For example, if we assume that the region a is a proper part of region b ($PP(a, b)$) then a is either a tangential or a non tangential proper part of b ($TPP(a, b)$ or $NTPP(a, b)$). Likewise, if $TPP(a, b)$ then $PP(a, b)$, and if $EQUAL(a, b)$ then both relations $P(a, b)$ and $P(b, a)$ hold.

2.3 Simulations

The Spatial Logic has been used as the underlying theory in qualitative simulation [Cui, Cohn and Randell 92] where a state is defined by a set of atomic formulas. The simulation takes an initial state that evolves according to envisioning axioms describing direct topological transitions that can be made between pairs of regions. Such evolving is done until a final state is reached. A representation of the topological transitions can be found in Figure 2.

In this context, the basic theory also includes the topological transitions of the base relations,

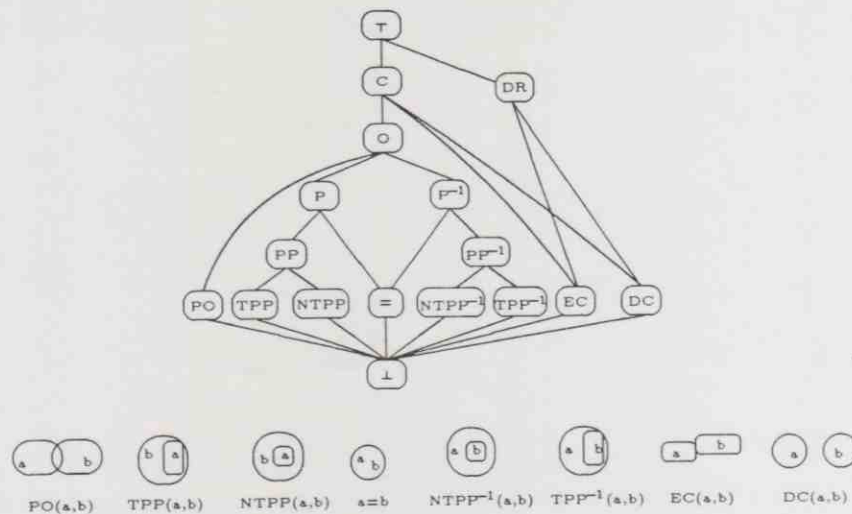


Figure 1: A lattice defining a subsumption hierarchy of the dyadic relations defined solely in terms of the primitive relation $C(x, y)$.

constraints that apply within a state and between adjacent states, add-rules that sanction the introduction of new objects into the domain of the next state, and delete-rules to ratify the elimination of objects in the next state.

To make a simulation possible it is important that the consistency of the initial state description be checked along the envisioning process. This is done using a transitivity table. In the next section transitivity tables will be discussed. According with [Randell and Cohn 89] the base relations and their topological transitions are depicted in Figure 2.

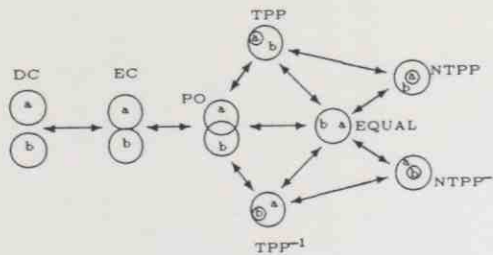


Figure 2: Base relations and their topological transitions.

2.4 Transitivity Tables

One of the problems that will be referred to later is the need of consistency checking of a set of formulas. One way to do this, is to prove the transitivity between pairs of formulas. Suppose that we have a set of relations $P = \{P_{y1}(x_1, x_2), \dots, P_{yn}(x_i, y_j)\}$ between regions ($P(a, b)$ reads "region a has a P connection with region b ") describing the state of a system modeled by regions and their connective relations. For each pair of relations that hold in P , for example $P_1(a, b)$ and $P_2(b, c)$, we can ask which relations $P_j(a, c)$ or $P_i(c, a)$ we can have in P and still have a logically consistent set P . This is a typical problem in theorem proving.

There are several strategies to construct such tables, but they are all complex and tedious and in some cases even too difficult to secure. These difficulties, along with the necessity of using transitivity tables, give a major incentive to the development of a theorem prover that can generate the entries for

$R_2(b,c)$	DC	EC	PO	TPP	$NTPP$
$R_1(a,b)$					
DC	any	DR, PO, PP	DR, PO, PP	DR, PO, PP	DR, PO, PP
EC	DR, PO, PP^{-1}	DR, PO, TPP, TPP^{-1}	DR, PO, PP	EC, PO, PP	PO, PP
PO	DR, PO, PP^{-1}	DR, PO, P^{-1}	any	PO, PP	PO, PP
TPP	DC	DR	DR, PO, PP	$NTPP$	PP
$NTPP$	DC	DC	DR, PO, PP	$NTPP$	$NTPP$

Table 1: Transitivity Table (from [Randell, Cohn and Cui 92]).

such tables. Such a theorem prover would be a significant progress in automated theorem proving. Our work in SNePS is an attempt to solve this problem using the capability of representation and reasoning of this system. Some examples shown in Section 4 are entries of this table.

3 SNePS Implementation

3.1 SNePS Restrictions and Implementations Issues

Using the expressiveness of SNePSLOG [Matos and Martins 89] [Shapiro *et al.* 93] it is possible to translate the space formalism expressions almost directly.

```

all(x)(region(x) => C(x,x))
all(x,y)({region(x),region(y)} => {C(x,y) <=> C(y,x)})
all(x,y)({region(x),region(y)} => {DC(x,y) <=> DC(y,x)})
all(x,y)({region(x),region(y)} => {PP(x,y) <=> ~PP(y,x)})
all(x,y)({region(x),region(y)} => {O(x,y) <=> O(y,x)})
all(x,y)({region(x),region(y)} => {DR(x,y) <=> DR(y,x)})
all(x,y)({region(x),region(y)} => {PO(x,y) <=> PO(y,x)})
all(x,y)({region(x),region(y)} => {EC(x,y) <=> EC(y,x)})
all(x,y)({region(x),region(y)} => {TPP(x,y) <=> ~TPP(y,x)})
all(x,y)({region(x),region(y)} => {NTPP(x,y) <=> ~NTPP(y,x)})
all(x,y)({region(x),region(y)} => {DC(x,y) <=> (~C(x,y))})
all(x,y)({region(x),region(y)} => {P(x,y) <=> (all(z)({region(z),C(z,x)} => {C(z,y)}))})
all(x,y)({region(x),region(y)} => {PP(x,y) <=> (P(x,y) and ~P(y,x))})
all(x,y)({region(x),region(y)} => {EQUAL(x,y) <=> (P(x,y) and P(y,x))})
all(x,y)({region(x),region(y)} => {O(x,y) <=> exists(z)(region(z) and P(z,x) and P(z,y))})
all(x,y)({region(x),region(y)} => {PO(x,y) <=> (O(x,y) and ~P(x,y) and ~P(y,x))})
all(x,y)({region(x),region(y)} => {DR(x,y) <=> ~O(x,y)})
all(x,y)({region(x),region(y)} => {TPP(x,y) <=> exists(z)(region(z) and PP(x,y) and EC(z,x) and EC(z,y))})
all(x,y)({region(x),region(y)} => {NTPP(x,y) <=> exists(z)(region(z) and PP(x,y) and (~EC(z,x) or ~EC(z,y))})})
all(x,y)({region(x),region(y)} => {EC(x,y) <=> (C(x,y) and ~O(x,y))})

```

These expressions although correct according to SNePSLOG syntax, do not allow the SNIP (the SNePS Inference Package) to do many of the expected deductions. We are going to analyze why this happens and how to avoid this problem.

Negation

The use of negations can cause problems. For example, if SNePS knows $C(a,b)$, it still cannot answer the query $\sim C(a,b)?$. To solve this problem we can add the second-order expression:

```
all(r,x,y) (r(x,y) <=> (~(~r(x,y))))
```

or we can substitute a simple expression like:

```
all(x,y)({region(x),region(y)} => {C(x,y) <=> ~DC(x,y)})
```

with: $all(x,y)({region(x),region(y)} => {C(x,y) <=> ~DC(x,y)})$

```
all(x,y)({region(x),region(y)} => {DC(x,y) <=> ~C(x,y)})
```

This solution, although less elegant is more efficient than the previous one which causes much additional inference.

Existential Quantifiers

Since the existential quantifiers are not implemented in SNePS (yet), they have to be substituted by Skolem functions following the usual way. For example, the formula:

```
all(x,y)({region(x),region(y)} &=> {TPP(x,y) <=> exists(z)(region(z) and PP(x,y) and EC(z,x) and EC(z,y))})
```

is replaced by:

1. all(x,y) ({region(x),region(y)} &=> {TPP(x,y) => {PP(x,y) and region(SKF_TPP(x,y)) and EC(SKF_TPP(x,y),x) and EC(SKF_TPP(x,y),y)}}})
2. all(x,y,z) ({region(x),region(y),region(z)} &=> {{PP(x,y),EC(z,x),EC(z,y)} &=> {TPP(x,y)}}})

During the our experiments in theorem proven (see example 3 in section 4) with the spatial logic, in order to perform some deduction we needed a mechanism that could recognize an instance of a Skolem function as a variable (region) already in use. For example, if we know that a region A is a tangential proper part of region B therefore exists an region X that is external connected with B and with A. If we knew that an region C is external connected with region A, it's only natural to think that this region could be the region X, so instead of instantiated the region X as Skolem(A,B) we identified X with the region C. This is a very sensible domain because logic consistency can be violated, so it must be used with care. In our case this kind of reasoning was very useful.

To make this kind of reasoning possible we used the following set of formulas:

```
; unification of the Skolem function with a region...
all(r,f,x,y,z)({skf(r),f(r(x,y),x),f(z,y),region(x),region(y),region(z),region(r(x,y)))}&=>{unif(r(x,y),z)})
all(r,f,x,y,z)({skf(r),f(r(x,y),y),f(z,y),region(x),region(y),region(z),region(r(x,y)))}&=>{unif(r(x,y),z)})
all(r,f,x,y,z)({skf(r),f(x,r(x,y)),f(z,y),region(x),region(y),region(z),region(r(x,y)))}&=>{unif(r(x,y),z)})
all(r,f,x,y,z)({skf(r),f(y,r(x,y)),f(z,y),region(x),region(y),region(z),region(r(x,y)))}&=>{unif(r(x,y),z)})

;...the replication of nodes using the unification done previously ...
all(g,x,y,z) ({unif(x,y), g(x,z)} &=> {g(y,z)})
all(g,x,y,z) ({unif(x,y), g(z,x)} &=> {g(z,y)})

;.. and the following predicates must exist "a priori".
skf(skolem_function_name(a,b))
region(skolem_function_name(a,b))
```

3.2 The Hierarchy of Relations in SNePSLOG

The formulas that can be written by the simple inspection of the hierarchy of relations, depicted in Figure 1, gives birth to a second set of formulas, redundant to the first set. This set of formulas is more efficient in the inference of topological relations although is not as powerful as the first one. The hierarchy relations in SNePS are:¹

```
;First level of the hierarchy
all(x,y)({region(x),region(y)} &=> {C(x,y) => andor(1,1){O(x,y), EC(x,y)}})
all(x,y)({region(x),region(y)} &=> {O(x,y) => C(x,y)})
all(x,y)({region(x),region(y)} &=> {EC(x,y) => C(x,y)})
all(x,y)({region(x),region(y)} &=> {DR(x,y) => andor(1,1){EC(x,y), DC(x,y)}})
all(x,y)({region(x),region(y)} &=> {EC(x,y) => DR(x,y)})
all(x,y)({region(x),region(y)} &=> {DC(x,y) => DR(x,y)})

;Second level of the hierarchy
all(x,y)({region(x),region(y)} &=> {O(x,y) => andor(1,1){P(x,y),P(y,x),PO(x,y)}})
all(x,y)({region(x),region(y)} &=> {P(x,y) => O(x,y)})
all(x,y)({region(x),region(y)} &=> {P(y,x) => O(x,y)})
all(x,y)({region(x),region(y)} &=> {PO(x,y) => O(x,y)})

;Third level of hierarchy
all(x,y)({region(x),region(y)} &=> {P(x,y) => andor(1,1){PP(x,y),EQUAL(x,y)}})
all(x,y)({region(x),region(y)} &=> {PP(x,y) => P(x,y)})
```

¹The andor(1,1) connective implements the "exclusive or" in SNePSLOG.

```

all(x,y)({region(x),region(y)} &=> {{EQUAL(x,y)} => andor(1,1){P(x,y),P(y,x)}})

;Fourth level of the hierarchy
all(x,y)({region(x),region(y)} &=> {PP(x,y) => andor(1,1){TPP(x,y), NTTP(x,y)}})
all(x,y)({region(x),region(y)} &=> {TPP(x,y) => PP(x,y)})
all(x,y)({region(x),region(y)} &=> {NTTP(x,y) => PP(x,y)})

```

3.3 Implementation of the Spatial Logic According to SNePS Technological Constraints

This section lists the SNePSLOG formulas of the spatial logic used in topological deductions and theorem proving. Our goal was to prove spatial topology theorems based on these formulas. This goal was partially achieved, SNePS proved the simpler theorems.

The formula list is divided into two groups. The first implements the axioms of the relations. The second contains the rules that allow the deduction of a relation from other relations. These formulas are obtained from two sources: The formal definition of the relation and their subsumption hierarchy.

```

;Connectivity axioms - C
all(x)({region(x)} &=> {C(x,x)})
all(x,y)({region(x),region(y)} &=> {C(x,y) => C(y,x)})

;axioms of derived relations
all(x,y)({region(x),region(y)} &=> {DC(x,y) => DC(y,x)})
all(x,y)({region(x),region(y)} &=> {PP(x,y) => ~PP(y,x)})
all(x,y)({region(x),region(y)} &=> {O(x,y) => O(y,x)})
all(x,y)({region(x),region(y)} &=> {DR(x,y) => DR(y,x)})
all(x,y)({region(x),region(y)} &=> {PO(x,y) => PO(y,x)})
all(x,y)({region(x),region(y)} &=> {EC(x,y) => EC(y,x)})
all(x,y)({region(x),region(y)} &=> {TPP(x,y) => ~TPP(y,x)})
all(x,y)({region(x),region(y)} &=> {NTTP(x,y) => ~NTTP(y,x)})

; Relation definitions
;connected - C (the primitive relation)
all(x,y)({region(x),region(y)} &=> {C(x,y) => andor(1,1){O(x,y), EC(x,y)}})

;disconnected - DC
all(x,y)({region(x),region(y)} &=> {DC(x,y) => (~C(x,y))})
all(x,y)({region(x),region(y)} &=> {C(x,y) => (~DC(x,y))})
all(x,y)({region(x),region(y)} &=> {(~DC(x,y)) => C(x,y)})
all(x,y)({region(x),region(y)} &=> {(~C(x,y)) => DC(x,y)})
all(x,y)({region(x),region(y)} &=> {DC(x,y) => DR(x,y)})

;part - P
all(x,y,z)({region(x),region(y),region(z)} &=> {P(x,y) => (~C(z,x) or C(z,y))})
all(x,y,z)({region(x),region(y),region(z)} &=> {{P(x,y), C(z,x)} &=> {C(z,y)}})
all(x,y,z)({region(x),region(y),region(z)} &=> {{P(x,y), ~C(z,y)} &=> {~C(z,x)}})
all(x,y)({region(x),region(y)} &=> {P(x,y) => O(x,y)})
all(x,y)({region(x),region(y)} &=> {P(x,y) => andor(1,1){PP(x,y), EQUAL(x,y)}})

;equal - EQUAL
all(x,y)({region(x),region(y)} &=> {{EQUAL(x,y)} &=> {P(x,y),P(y,x)}})
all(x,y)({region(x),region(y)} &=> {{P(x,y), P(y,x)} &=> {EQUAL(x,y)}})

;proper part - PP
all(x,y)({region(x),region(y)} &=> {{PP(x,y)} &=> {P(x,y), ~P(y,x)}})
all(x,y)({region(x),region(y)} &=> {{PP(x,y) => andor(1,1){TPP(x,y), NTTP(x,y)}})
all(x,y)({region(x),region(y)} &=> {{P(x,y), ~P(y,x)} &=> {PP(x,y)}})

;overlaps - O
all(x,y)({region(x),region(y)} &=> {{O(x,y)} &=> {region(SKF_O(x,y)),P(SKF_O(x,y),x),P(SKF_O(x,y),y)}})
all(x,y,z)({region(x),region(y),region(z)} &=> {{P(z,x), P(z,y)} &=> {O(x,y)}})
all(x,y)({region(x),region(y)} &=> {O(x,y) => C(x,y)})
all(x,y)({region(x),region(y)} &=> {O(x,y) => andor(1,1){PO(x,y),P(x,y)or P(y,x)}})

```

```

;discrete from - DR
all(x,y)({region(x),region(y)} &=> {0(x,y) => ~DR(x,y)})
all(x,y)({region(x),region(y)} &=> {~0(x,y) => DR(x,y)})
all(x,y)({region(x),region(y)} &=> {DR(x,y) => ~0(x,y)})
all(x,y)({region(x),region(y)} &=> {~DR(x,y) => 0(x,y)})
all(x,y)({region(x),region(y)} &=> {DR(x,y) => andor(1,1){EC(x,y),DC(x,y)}})

;partially overlaps - PO
all(x,y)({region(x),region(y)} &=> {{PO(x,y)} &=> {0(x,y), ~P(y,x), ~P(x,y)}})
all(x,y)({region(x),region(y)} &=> {{0(x,y), ~P(y,x), ~P(x,y)} &=> {PO(x,y)}})

;externally connected - EC
all(x,y)({region(x),region(y)} &=> {{EC(x,y)} &=> {C(x,y),DR(x,y)}})
all(x,y)({region(x),region(y)} &=> {{C(x,y),DR(x,y)} &=> {EC(x,y)}})

;tangential proper part - TPP
all(x,y) ({region(x),region(y)} &=>
  {TPP(x,y) => {PP(x,y) and region(SKF_TPP(x,y)) and EC(SKF_TPP(x,y),x) and EC(SKF_TPP(x,y),y)}})
all(x,y) ({region(x),region(y),region(z)} &=> {{PP(x,y),EC(z,x),EC(z,y)} &=> {TPP(x,y)}})
all(x,y)({region(x),region(y)} &=> {TPP(x,y) => PP(x,y)})

;non-tangential proper part - NTPP
all(x,y) ({region(x), region(y)} &=>
  {NTPP(x,y) => {PP(x,y) and region(SKF_NTPP(x,y)) and (~EC(SKF_NTPP(x,y),x)) or ~EC(SKF_NTPP(x,y),y)}})
all(x,y) ({region(x), region(y), region(z)} &=> {{PP(x,y), (~EC(z,x) or ~ EC(z,y))} &=> {NTPP(x,y)}})
all(x,y)({region(x),region(y)} &=> {NTPP(x,y) => PP(x,y)})

```

3.4 Relation Transitions in SNePS

Our present implementation include a simple simulation. Our simulation uses a SNePS's module [Pinto-Ferreira 91] that given an initial state and a final state infers the actions that should be taken in order to go from one state to another. This module allows the description of actions and the respective modification induced to the system. The actions are defined by their name, a delete list (the negation of certain relations as the action becomes true) and an add list (introduction of new relations).

An example of an action definition is:

```

all(x)(U({ALIVE(x)} {DEAD(x)} DECLARED-DEAD(x)))
all(x)(U({<delete-list>} {add-list} <name>))

```

The topological transitions, depicted in Figure 2, in SNePS are represented by the following actions:

```

all(x,y) (U({DC(x,y)} {EC(x,y)} DC-EC(x,y)))
all(x,y) (U({EC(x,y)} {PO(x,y)} EC-PO(x,y)))
all(x,y) (U({PO(x,y)} {TPP(x,y)} PO-TPP(x,y)))
all(x,y) (U({PO(x,y)} {EQUAL(x,y)} PO-EQUAL(x,y)))
all(x,y) (U({PO(x,y)} {TPP(y,x)} PO-TPP-1(x,y)))
all(x,y) (U({TPP(x,y)} {EQUAL(x,y)} TPP-EQUAL(x,y)))
all(x,y) (U({EQUAL(x,y)} {TPP(y,x)} EQUAL-TPP-1(x,y)))
all(x,y) (U({TPP(x,y)} {NTPP(x,y)} TPP-NTPP(x,y)))
all(x,y) (U({EQUAL(x,y)} {NTPP(x,y)} EQUAL-NTPP(x,y)))
all(x,y) (U({EQUAL(x,y)} {NTPP(y,x)} EQUAL-NTPP-1(x,y)))
all(x,y) (U({TPP(y,x)} {NTPP(y,x)} TPP-1-NTPP-1(x,y)))

```

And an inverse transition for each action previously represented. For example:

```

all(x,y)(U({EC(x,y)} {DC(x,y)} EC-DC(x,y)))

```

4 Examples

In all examples the queries are made with one option saying to find only one answer in spite of searching for all. The times are in seconds and were measured on a SUN Sparc Station 2.

Example 1

This example consists of proving that if one region is inside another and the latter is disconnected from a third one then the first one and the third are disconnected:

```
;State description:
region(A)
region(B)
region(C)
DC(A,B)
PP(C,B)
```



```
;Relations used:
```

```
all(x,y) ( {region(x), region(y)} &=> {DC(x,y) => (~C(x,y))} )
all(x,y) ( {region(x), region(y)} &=> { (~C(x,y)) => DC(x,y)} )
all(x,y,z) ( {region(x), region(y), region(z)} &=> {P(x,y) => (~C(z,x) or C(z,y))} )
all(x,y) ( {region(x), region(y)} &=> {PP(x,y)} &=> {P(x,y), ~P(y,x)} )
```

```
; Query
DC(a,c)?
```

```
.....
Since {PP(C,B)} &=> {P(C,B), ~P(B,C)}
and PP(C,B)
I infer P(C,B)
```

```
Since P(X,Y) => (~C(Z,X) or C(Z,Y))
and P(C,B)
I infer ~C(A,C) or C(A,B)
```

```
.....
Since all(X,Y) ( {REGION(X), REGION(Y)} &=> {DC(X,Y) => (~C(X,Y))} )
and REGION(A)
and REGION(B)
I infer DC(A,B) => (~C(A,B))
```

```
.....
Since ~C(A,C) or C(A,B)
and it is not the case that C(A,B)
I infer ~C(A,C)
```

```
.....
Since (~C(X,Y)) => DC(X,Y)
and ~C(A,B)
I infer DC(A,B)
```

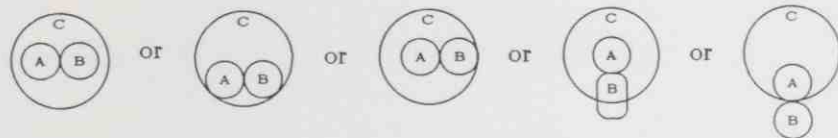
```
DC(A,C)
```

```
CPU time: 26.85
```

Example 2

In this example it is proved that if a region A is a proper part of another region C and is also connected with a third region B than region B and C must be connected.

```
;State description:
region(A)
region(B)
region(C)
PP(A,C)
EC(A,B)
```



```
;Relations used:
```

```
all(x,y) ( {region(x), region(y)} &=> {C(x,y) => C(y,x)} )
all(x,y) ( {region(x), region(y)} &=> {EC(x,y)} &=> {C(x,y), DR(x,y)} )
all(x,y) ( {region(x), region(y)} &=> {PP(x,y)} &=> {P(x,y), ~P(y,x)} )
all(x,y,z) ( {region(x), region(y), region(z)} &=> {P(x,y), C(z,x)} &=> {C(z,y)} )
```



```

; Query:
C(b,c)?
.....
Since {PP(A,C)} &=> {P(A,C),~P(C,A)}
and PP(A,C)
I infer P(A,C)
.....
Since {PP(A,C)} &=> {P(A,C),~P(C,A)}
and PP(A,C)
I infer ~P(C,A)
.....
Since all(X,Y)({REGION(X),REGION(Y)} &=> {C(X,Y) => C(Y,X)})
and REGION(B)
and REGION(A)
I infer C(B,A) => C(A,B)
.....
Since C(X,Y) => C(Y,X)
and C(A,B)
I infer C(B,A)
.....
Since C(X,Y) => C(Y,X)
and C(C,B)
I infer C(B,C)
.....
I know C(C,B)
I know C(A,B)

C(B,C)

CPU time: 70.62

```

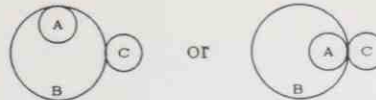
Example 3

In this example it is shown that if a region A is a tangential proper part of a region B and that region B is externally connected with a third region C , then C is disjoint from region A . In this deduction we need to use a non-topological set of formulas in order to unify the Skolem functions with variables.

```

;State description:
region(A)
region(B)
region(C)
TPP(A,B)
EC(C,B)

```



```

;Relations used:
all(x,y) ({region(x),region(y)} &=>
  {{TPP(x,y)} &=> {PP(x,y),region(SKF_TPP(x,y)),EC(SKF_TPP(x,y),x),EC(SKF_TPP(x,y),y)}})
all(x,y) ({region(x),region(y)} &=> {{EC(x,y)} &=> {C(x,y),DR(x,y)}})
all(x,y) ({region(x),region(y)} &=> {DR(x,y) => DR(y,x)})
all(r,f,x,y,z) ({skf(r),f(r(x,y),y),f(z,y)} &=> {unif(r(x,y),z)})
all(g,x,y,z) ({unif(x,y),g(x,z)} &=> {g(y,z)})
skf(SKF_TPP)

```

```

;Query:
DR(a,c)?

```

```

Since {TPP(A,B)} &=> {PP(A,B),EC(SKF_TPP(A,B),A),EC(SKF_TPP(A,B),B)}
and TPP(A,B)
I infer EC(SKF_TPP(A,B),B)
.....
Since {TPP(A,B)} &=> {PP(A,B),EC(SKF_TPP(A,B),A),EC(SKF_TPP(A,B),B)}
and TPP(A,B)
I infer EC(SKF_TPP(A,B),A)

Since all(Y,Z,R,F,X)({SKF(R),F(R(X,Y),Y),F(Z,Y)} &=> {UNIF(R(X,Y),Z)})
and SKF(SKF_TPP)

```

```

and EC(SKF_TPP(A,B),B)
and EC(C,B)
I infer UNIF(SKF_TPP(A,B),C)
.....
Since all(G,X,Y,Z)({UNIF(X,Y),G(X,Z)} => {G(Y,Z)})
and UNIF(SKF_TPP(A,B),C)
and EC(SKF_TPP(A,B),A)
I infer EC(C,A)
.....
Since {EC(C,A)} => {DR(C,A),C(C,A)}
and EC(C,A)
I infer DR(C,A)

Since DR(X,Y) => DR(Y,X)
and DR(C,A)
I infer DR(A,C)
.....

DR(A,C)

CPU time : 338.07

```

These examples lead to several conclusions. First, SNePS really permits to represent spatial relations in a simple way and to make inference with them. A second conclusion is that although the first conclusion is true, the system managed to do inference only to answer to queries with high hierarchical level relations. A third conclusion would be that the time of inference grows with the number of formulas, as expected.

Example 4

In this example we show the capability of SNePS to plan the transitions between a initial state and a final state using the topological transitions. This transcript has been edited to save space.

```

; Initial state
:DC(a,b)

; Final state
: NTPP(b,a) plan

I try to prove the goal conjunction
NTPP(B,A)
in the description:
primitive wffs:
all(X,Y) (U{DC(X,Y)} {EC(X,Y)} DC-EC(X,Y))
all(X,Y) (U{EQUAL(X,Y)} {NTPP(X,Y)} EQUAL-NTPP(X,Y))
all(X,Y) (U{EC(X,Y)} {DC(X,Y)} EC-DC(X,Y))
all(X,Y) (U{PO(X,Y)} {EC(X,Y)} PO-EC(X,Y))
all(X,Y) (U{TPP(X,Y)} {PO(X,Y)} TPP-PO(X,Y))
all(X,Y) (U{EQUAL(X,Y)} {PO(X,Y)} EQUAL-PO(X,Y))
all(X,Y) (U{EQUAL(X,Y)} {TPP(X,Y)} EQUAL-TPP(X,Y))
all(X,Y) (U{TPP(Y,X)} {EQUAL(X,Y)} TPP-1-EQUAL(X,Y))
all(X,Y) (U{NTPP(X,Y)} {TPP(X,Y)} NTPP-TPP(X,Y))
all(X,Y) (U{EC(X,Y)} {PO(X,Y)} EC-PO(X,Y))
all(X,Y) (U{TPP(Y,X)} {NTPP(Y,X)} TPP-1-NTPP-1(X,Y))
all(X,Y) (U{NTPP(X,Y)} {EQUAL(X,Y)} NTPP-EQUAL(X,Y))
all(X,Y) (U{NTPP(Y,X)} {EQUAL(x,y)} NTPP-1-EQUAL(X,Y))
DC(A,B)
all(X,Y) (U{PO(X,Y)} {TPP(X,Y)} PO-TPP(X,Y))
all(X,Y) (U{PO(X,Y)} {TPP(X,Y)} PO-EQUAL(X,Y))
all(X,Y) (U{PO(X,Y)} {TPP(Y,X)} PO-TPP-1(X,Y))
all(X,Y) (U{TPP(X,Y)} {EQUAL(X,Y)} TPP-EQUAL(X,Y))
all(X,Y) (U{EQUAL(X,Y)} {TPP(Y,X)} EQUAL-TPP-1(X,Y))
all(X,Y) (U{TPP(X,Y)} {NTPP(Y,X)} TPP-NTPP(X,Y))
all(X,Y) (U{TPP(Y,X)} {TPP(Y,X)} TPP-1-TPP-1(X,Y))
actions: none - initial description

```

```

I wonder if NTPP(B,A)
holds within the BS defined by context DEFAULT-DEFAULTCT
.....
I know NTPP(B,A)
A plan to achieve
  NTPP(B,A)
is to:
DC-EC(A,B)
EC-PO(A,B)
PO-TPP-1(A,B)
TPP-1-NTPP-1(A,B)

CPU time: 994.48

```

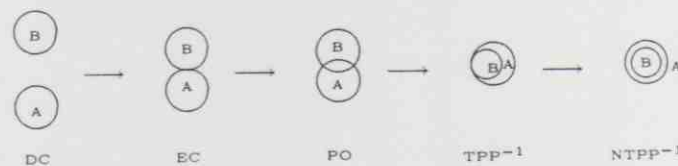


Figure 3: Solution of Example 4.

5 Conclusions

We demonstrated how SNePS allowed us to test a spatial logic without having to develop a dedicated implementation in a programming language. This demonstrates both the knowledge representation capability of the spatial logic presented in [Randell and Cohn 89] [Randell, Cui and Cohn 92] and the versatility of SNePS as a knowledge representation system.

Our work is mainly concerned with implementation issues in SNePS, so there were no considerations about the adequacy of the spatial logic used. Our conclusions are drawn in order to establish limitations and future developments that SNePS can offer to the implementation of “reasoning about space” systems.

Our implementation was mainly focused on the possibility of theorem proving in spatial logic using SNePS. This is an important issue because it is the backbone of qualitative simulation, it can be used in the creation of transitivity tables used in the consistency checking of the different states in logic descriptions of a physical system in a qualitative simulation. A fully automated construction of such tables would be a breakthrough in qualitative simulation [Randell, Cohn and Cui 92]. With a limited set of formulas of the original set presented in [Randell and Cohn 89] [Randell, Cui and Cohn 92], we were able to prove some of the more general entries of a transitivity table. This was a very encouraging step although it raised several problems due to the current development stage of SNePS.

Perhaps the main problem that we were confronted with was the discrepancy between the expressive power of SNePSLOG and the effective power of the inference engine. This made necessary a careful choice of the formulas presented to SNIP. Issues such as the implementation of the existential quantifier and the unification mechanism have been examined in detail in sections 3.1 and 3.3. Another problem that we were confronted with was the long reasoning times that the system requires to make inferences. This delayed the testing and development cycle of an effective set of formulas.

The most promising point in this work is the use of the planning capability of SNePS. It allows, in a simple way, the implementation of state description and transitions of a physical system. The enhancement of the topological transitions and the restrictions of those transitions due to the nature of the objects involved is work that is currently being done by the authors. Work that we still have to do is the extension of the set of formulas in order to include the IN and OUT concepts proposed in [Randell

and Cohn 89] [Randell, Cui and Cohn 92], and the concept of "convex hull". We believe that this would allow a larger number of theorems to be proved. Finally, the formulas used to implement the existential quantification and the unification of Skolem functions may be improved.

The work described in this paper raises more challenges than it solves problems, but it gives us some clues about the way to go when it is necessary to test an ontological description. We hope that this work becomes a starting point for a broader use of SNePS as a general purpose logical and reasoning system.

References

- [Cui, Cohn and Randell 92] Z. Cui, A. Cohn and D. Randell, Qualitative simulation based on a logical formalism of space and time, in: *Proceedings of the Tenth National Conference on Artificial Intelligence*, San Jose, CA (The MIT Press, Menlo Park, CA, 1992) 679-684.
- [Matos and Martins 89] P. Matos and J. Martins, SNePSLOG - a logic interface to SNePS, Technical Report GIA 89/3, Grupo de Inteligência Artificial, Instituto Superior Técnico, Lisbon, Portugal (1989).
- [Pinto-Ferreira 91] C. Pinto-Ferreira, A lógica de mutações e o problema da quiescência, Ph.D. Thesis, Instituto Superior Técnico, Technical University of Lisbon, Lisbon, Portugal (1991).
- [Randell and Cohn 89] D. Randell and A. Cohn, Modelling topological and metrical properties in physical processes, in: *Proceedings First International Conference on Principles of Knowledge Representation and Reasoning*, Toronto, Ont (Morgan Kaufmann Publishers, San Mateo, CA, 1989) 357-368.
- [Randell, Cohn and Cui 92] D. Randell, A. Cohn and Z. Cui, Computing transitivity tables: A challenge for automated theorem provers, in: *Proceedings CADE 11*, Berlin, Germany (SpringerVerlag, 1992).
- [Randell, Cui and Cohn 92] D. Randell, Z. Cui and A. Cohn, A spatial logic based on regions and connection, in: *Proceedings Third International Conference on Principles of Knowledge Representation and Reasoning*, Cambridge, MA (Morgan Kaufmann Publishers, San Mateo, CA, 1992) 165-176.
- [Shapiro 79] S. Shapiro, The SNePS semantic network processing system, in: N. Findler, ed., *Associative Networks: Representation and Use of Knowledge by Computers* (Academic Press, New York, NY, 1979) 179-203.
- [Shapiro 91] S. Shapiro, Case studies of SNePS, *ACM SIGART Bulletin* 2 (3) (1991) 14-18.
- [Shapiro and Rapaport 87] S. Shapiro and W. Rapaport, SNePS considered as a fully intentional propositional semantic network, in: N. Cercone and G. McCalla, eds., *The Knowledge Frontier* (Springer-Verlag, New York, NY, 1987) 262-315.
- [Shapiro and Rapaport 91] S. Shapiro and W. Rapaport, Models and minds: Knowledge representation for natural-language competence, in: R. Cummins and J. Pollock, eds., *Philosophical AI: Computational Approaches to Reasoning* (SpringerVerlag, New York, NY, 1991).
- [Shapiro et al. 93] S. Shapiro et al., SNePS-2.1 user's manual, SNeRG Technical Note, Department of Computer Science, State University of New York at Buffalo, Buffalo, NY (1993).