Passing through holes and getting entangled by strings: an automated solution for a spatial puzzle

Paulo Santos[1] and **Pedro Cabalar**[2]

Abstract. This paper investigates the challenging problem of encoding the knowledge and reasoning processes involved in the common sense manipulation of physical objects. In particular we provide a formalisation of a domain involving rigid objects, holes and a string within a reasoning about actions and change framework. Therefore, this work investigates the formalisation and reasoning about flexible objects and void space (holes) in a single domain. Preliminary results of automated reasoning within this domain are also presented.

1 Introduction

The field of qualitative spatial reasoning (QSR) [13] attempts the formalisation of spatial knowledge based on primitive relations defined over elementary spatial entities. One of the best known QSR theories, for instance, is the Region Connection Calculus [8], which is a first order axiomatisation of space based on regions and the connectivity relation. Other representations of spatial knowledge include theories about shape [6], distance [5], position [2] amongst others as surveyed in [4]. However, the use of qualitative spatial knowledge within a planning system remains largely neglected.

One possible reason for the lack of problem solving methods handling qualitative spatial knowledge may be connected to the fact that research on QSR has being conducted independently from research on reasoning about actions and change (RAC) and AI planning (apart from exceptions such as [12] and [11]). One of the motivations for the present work is to approximate RAC to reasoning about spatial knowledge by investigating the formalisation and automatic solution of a challenging spatial puzzle.

This paper assumes the puzzle called *The Fisherman's Folly* (Figure 1) that involves spatial entities such as strings, posts, rings, spheres and holes (through the last ones some (but not all) domain objects can pass). The Fisherman's Folly puzzle consists in going from the configuration shown in Figure 1(a) to the configuration in Figure 1(b) by moving the objects positions respecting some domain restrictions. In this sense, this puzzle is similar to the classic 8-puzzle; however, in the present work the domain objects have non-trivial spatial characteristics, such as flexibility and permeability through holes.

The elements of the puzzle are a holed post fixed to a wooden base, a string, a ring, a pair of spheres and a pair of disks. The disks and spheres are attached to the string, along which the latter can move but not the former, which are fixed to the string endpoints.

In the initial state (shown in Figure 1(a)) the post is in the middle of the ring, which is supported on the post's base. On the other hand, the string passes through the post's hole in a way that one sphere and one disk remain on each side of the post. It is worth pointing out that the spheres are larger than the post's hole, therefore the string cannot be separated from the post without cutting either the post, or the string, or destroying one of the spheres. The disks and the ring, in contrast, can pass through the post's hole. The goal of this puzzle (depicted in Figure 1(b)) is to find a sequence of transformations of the spatial configuration of the puzzle's objects such that the ring is freed from the system *post-base-string*, maintaining the physical integrity of the domain objects. In fact, the goal state is not fixed to the one shown in Figure 1(b). In order to be considered a solution, it is sufficient to move the ring completely out of the rest of the system, regardless the final configuration of the remaining domain objects.

(a) The initial state.

(b) The goal state.

Figure 1: A spatial puzzle: the Fisherman's Folly.

The complexity imposed by the distinct states of the string allied to the existence of holes in the domain objects makes the formalisation and reasoning about this domain a challenging problem. In order to provide a formal account of the spatial relations involved in the Fisherman's Folly we need to consider in our formalisation (and reasoning processes) the holes in objects, such as the post's hole and the space limited by the ring. This calls for assuming holes as real objects, therefore having the same ontological status as spheres and disks. Reasoning about holes and holed objects has been discussed in detail in [14] from the standpoint of topology. However, to the best of our knowledge, this paper presents the first approach that investigates the problem of how these entities could be engaged in actions.

The string brings a further source of complexity which comes from the related infinity of distinct configurations due to its flexibility. The problem of incorporating knowledge about strings and string manipulation has been tackled in [7] where a robotic system capable of learning to tie a knot from visual observation is proposed, this system is called *the Knot Planning from Observation* (KPO) paradigm. In KPO each state of a string is represented by a matrix encoding the string segments, which are defined by the portion of the string that lies in between its endpoints and points where it crosses over itself.

[1] Centro Universitário da FEI, São Paulo, email: psantos@fei.edu.br
[2] Coruña University, Spain, email: cabalar@dc.fi.udc.es

Actions on flexible objects in this context are defined as an extension of the Reidemeister moves in knot theory [10]. This representation is suitable for the identification of string states from a computer vision system; however, it falls short in the context of problem solving, which is the main purpose of the present paper. In this work, we propose a representation for string states that takes into account other objects (including holes) that may be related to the string in the domain. In contrast to the work proposed in [7], we do not take into account knots. Incorporating some of the ideas of the KPO paradigm in our work shall be investigated in the future.

In summary, the purpose of this paper is to investigate the formalisation and autonomous solution of a spatial domain involving holes and a solid objects(including a string), contributing with a novel benchmark problem for common sense knowledge representation. In order to report this work, the present paper is organised as follows: Section 2 presents a basic ontology about holes, where we revisit some concepts from [1]; Section 3 presents a pair of base shape primitives used to define a sufficient condition for object penetrability through holes; Section 4 introduces a formalisation of the puzzle that is used for finding an automated solution for the Fisherman's Folly, as presented in Section 5.

2 A theory about holes

There are at least three distinct types of holes [1]: *cavities*, i.e. holes that are entirely hidden inside their hosts; *hollows*, which are superficial depressions on the host; and, perforating holes (or tunnels), which are holes that have at least two distinct entrance boundaries. In this paper we shall deal only with perforating holes, since only these are relevant to the puzzle's solutions[3].

In the formalisation described below, holes are assumed as open regions which boundaries belong to their host objects. The relationship between holes and their hosts are formalised using the elementary relation: $H(x, y)$, meaning "x is a hole in the object y" (conversely, "y is the host of x") [1].

In this work the domain objects are identified with their occupancy regions, whereas holes are parts of an object's complement that are inside the object's occupancy region and are not parts of any other object. Therefore, as the domain is only populated by spatial regions, it is convenient to include in the basic theory about holes a set of mereological relations accounting for the degree of connectiveness between spatial regions. In this work we assume RCC-8 ([8, 9, 3]) which is a many-sorted first-order axiomatisation of spatial relations based on a dyadic primitive relation of *connectivity* ($C/2$) between two regions. Informally, assuming two regions x and y, the relation $C(x, y)$, read as *"x is connected with y"*, is true if and only if the *closures* of x and y have a point in common. Assuming the $C/2$ relation as primitive, and that x, y and z are variables for spatial regions, the following mereological relations can be defined $DC(x, y)$, which stands for "x is disconnected from y"; $EQ(x, y)$, for "x is equal to y"; $PO(x, y)$, for "x partially overlaps y"; $EC(x, y)$, for " the closure of x and y are externally connected"; $TPP(x, y)$, for "x is a tangential proper part of y"; $NTPP(x, y)$, for "x is a non-tangential proper part of y"; and, $TPPi/2$ and $NTPPi/2$ are the inverse relations of $TPP/2$ and $NTPP/2$ respectively. The conceptual neighbourhood diagram of RCC8 is shown in Figure 2.

Assuming RCC, the relation $H(x, y)$ is constrained by the axioms (1) and (2) below. Axiom (1) guarantees that the host of a hole is not

[3] Therefore, in the remainder of this paper we will use the words: tunnels, perforating holes and holes interchangeably.

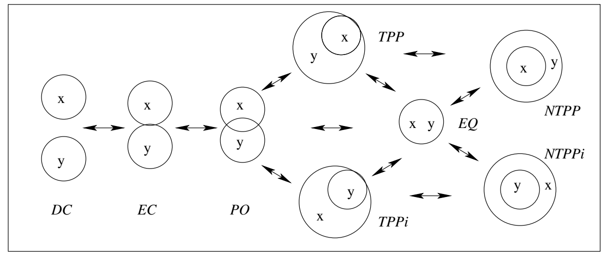


Figure 2: The RCC8 relations and their conceptual neighbourhood diagram.

itself a hole; whereas Axioms (2) states that the hole and it's host object are externally connected.

$$Hxy \rightarrow \neg H(y, z) \quad (1)$$
$$Hxy \rightarrow EC(x, y) \quad (2)$$

Moreover, Axiom 1 implies that the relation H is irreflexive (meaning that no hole hosts itself) and anti-symmetric (i.e., the host cannot be a hole of its resident hole).

An essential characteristic of holes is that they can be interpenetrated by other objects. Therefore, the hole ontology has to include relations about the relative location of a hole wrt the penetrating object. In a world uniquely populated by spatial regions, relative location can be expressed by mereological relations. In order to define relative location wrt a hole, we need the concept of a hole entry boundary (EB) that is defined in [1] by the relation $EB(z, x, y)$, read as "z is the maximally connected part of the hole x (fiat) boundary that is nowhere a boundary of the host y".

We can now express the following relations:

- an object x is *wholly outside* a hole h ($WO(x, h)$) iff $DC(x, h)$;
- an object x is *just outside* a hole h wrt the hole entry boundary h_i ($JO(x, h, h_i)$) iff
$$EB(h_i, h, y) \wedge EC(x, h_i) \wedge \neg TPP(x, h);$$
- x is *partially outside* h wrt the entry boundary h_i ($PO(x, h, h_i)$) iff
$$EB(h_i, h, y) \wedge PO(x, h) \wedge C(x, h_i);$$
- x is *just inside* h wrt the hole entry boundary h_i ($JI(x, h, h_i)$) iff
$$EB(h_i, h, y) \wedge EC(x, h_i) \wedge TPP(x, h).$$
- x is *wholly inside* (WI) h iff $TPP(x, h) \wedge NTPP(x, h)$;

The relations WO, JO, PO, WI and JI are schematised in Figure 3, where the host object is the cuboid, the hole is the cylindrical figure inside the cuboid and the penetrating object is represented by the **v**-shaped figure.

It is worth pointing out that, in contrast to [1], encoding the relative location of an object wrt a hole using RCC relations allowed us to include both JO and JI into the same formalism since RCC is defined over the closure of regions. Therefore, the concepts of just inside and just outside can coexist with the initial assumption of holes as open regions. Another difference between the formalism presented above wrt that proposed in [1] is the inclusion of the hole entry boundary in the definitions of JO, PO and JI, in order to account for the action of an object passing through a particular hole entry boundary.

Figure 3 can be understood as a sequence of continuous transitions

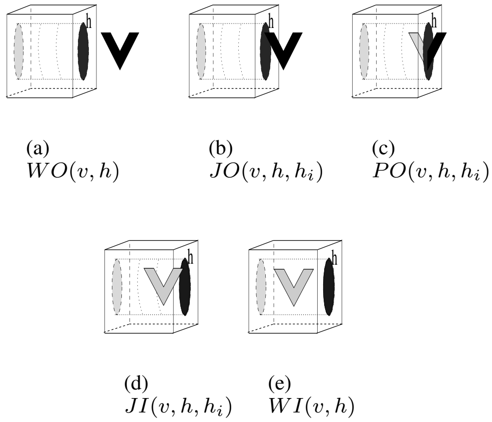


Figure 3: Relative location of an object v wrt a hole h.

from the relation *wholly outside* to *wholly inside*. In order to provide a formal solution to the fisherman's folly, however, we need to be able to locate an object in space that is WO (with respect to every hole) but that is near a particular entry boundary of a tunnel. In effect, tunnels are important qualitative landmarks that could be used as local reference frames. This idea is developed in the next section.

2.1 Hole subspaces

It is not unusual in the common language to characterise sections of a road by the sections *before* and *after* a tunnel. In a domestic domain, we decide where to locate (non-wireless) electronic objects according to the nearby plugs (which are, in fact, tunnel entry boundaries). The issue of reasoning about tunnels becomes quite critical when the problem is to locate buried infrastructure so that repairs can be conducted on a particular network of pipes and cables underground[4]. In spite of these facts, to the best of our knowledge, there are no references that account to the potential use of holes entry boundaries as local reference frames. This section describes an initial attempt to cope with this issue.

In this work, we assume that the entry boundaries are uniquely identified by a symbol referring to their host hole plus a subscript number, that differentiates each of the EBs within a single hole (as shown in Figure 4). If a global reference frame is assumed in the domain, the entry boundaries can be identified by the 3D Cartesian coordinates of their respective centre points; thus, the local reference provided by the entry boundaries could be associated to a global reference frame. In this work, however, objects are located with respect to the near neighbourhoods of the hole entry boundaries.

More formally, an object v is in the near neighbourhood (NN) of a hole EB h_i iff it is *just outside* it or it is connected to another object that is either *just outside* the hole or *partially overlapping* it. In other words:

- x is in the near neighbourhood of a hole entry boundary h_i ($NN(x, h_i)$) iff

$$JO(x, h, h_i) \vee \exists y (C(x, y) \wedge (JO(y, h, h_i) \vee PO(y, h, h_i)))$$

[4] cf. "http://www.mappingtheunderworld.ac.uk/" last access on 6/4/2006.

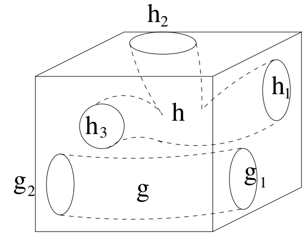


Figure 4: Two holes h an g, and their respective entry boundaries, within a single object.

In the Fisherman's Folly puzzle, however, we only need to consider holes that have only two entry boundaries and null depth (Figure 5). The entry boundaries of this kind on hole are called the *poles* of the hole. These poles subdivide the space local to the hole into two parts, named the *hole subspaces*.

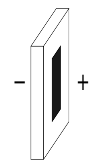


Figure 5: Poles and subspaces of a hole.

Figure 5 represent the hole poles and their relative subspaces, whereby the hole is the shaded region, the poles and the subspaces are represented by a '+' and a '-' sign. Thus, we can represent, for each hole h in the puzzle, its corresponding poles as h^- and h^+. Furthermore, if a is a hole pole, then $-a$ represents the opposite one, so that $-h^- = h^+$ and $-h^+ = h^-$.

We are now capable of expressing formally that an object is near a tunnel (e.g., a car is parked outside the Eurotunnel entrance) or that objects are related to a network of tunnels (which is the case of the puzzle in question). However, in order to account for the main issues involved in the Fisherman's Folly, the theory has to include some basic ideas about object's shape so that it is capable of expressing object's penetrability through holes. The next section discusses some insights on this issue.

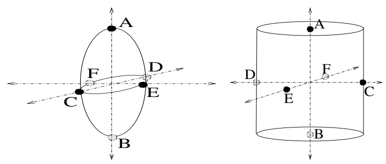
3 The shapes of objects

The shape of objects is, at the same time, the most elusive and the most important issue in reasoning about the common sense space [4]. In this paper we cannot scape from taking into account object's shape (at least in very basic terms), since the solution of the puzzle involves passing an object of a particular shape and size, through a hole entry boundary (also of a particular shape and size). In fact, for the purposes of this work, shapes are only needed to facilitate the proposal of a sufficient condition for an object to pass through an entry boundary. To this end, we propose two shape primitives with which we identify the shapes of the puzzle.

3.1 Shape Primitives

This work assumes ellipsoids and cylinders (Figure 6) as the basic primitives to describe the puzzle's object shapes.

An ellipsoid (Figure 6(a)) is a 3 dimensional geometric figure which every planar cross section is an ellipsis. This figure has three



(a) Ellipsoid (b) Elliptic cylinder

Figure 6: Base shape primitive.

symmetry axes: $\overline{AB}$, $\overline{CD}$ and $\overline{EF}$ (as shown in Figure 6(a)) that are called, respectively, *major*, *mean* and *minor* axes. Thus, the spheres in our puzzle have ellipsoid shapes whose three symmetry axes are of the same lengths. Similarly, the post has the shape of an ellipsoid with major axis much greater than both mean and minor axes.

In an analogous way, we use an elliptic cylinder (Figure 6(b)) as the primitive that accounts for the shapes of the objects not represented by the ellipsoid. An elliptic cylinder is a cylinder whose base is an ellipsis. Therefore, this figure also has a *major*, *mean* and *minor* axes (respectivelly axes $\overline{AB}$, $\overline{CD}$ and $\overline{EF}$ in Figure 6(b)). Therefore, the shape of the puzzle's disks and ring are cylinders whose axis $\overline{AB}$ is much smaller than $\overline{CD}$ and $\overline{EF}$, and the last two have the equal lengths. The shape of the post base is approximated to a cylinder analogous to the previous ones, however with greater proportions. Similarly, the string has the shape of a cylinder where $\overline{AB}$ is much greater than $\overline{CD}$ and $\overline{EF}$.

The string's shape has an extra complication which is its intrinsic flexibility. Thus we should also consider as the string shape every different shapes that it can assume from any sequence of non-destructive transformations. This issue turned out not to be essential in our formal solution for the fisherman's puzzle and is left for a future investigation.

In the next section we present a sufficient condition for an object to pass through a entry boundary.

3.2 A sufficient condition for passing an object through a hole

We first assume a simple common sense conjecture that every object is conducted through a hole in the direction of the largest semi-line connecting any two points of its boundary, this semi-line we call *conducting line*. Thus, the post shall be conducted through the ring hole in the direction of its major axis; similarly, the disks are conducted through the post hole via its diameter (i.e., via its mean and minor axes, not its major).

Let's define the region defined by the orthogonal projection of an object o (taken through the object's conducting line) as $pl(p)$ and the region defined by the orthogonal projection of a hole entry boundary h_i as $p(h_i)$. Now we say that an object can pass through a hole if it is possible to superimpose $pl(p)$ and $p(h_i)$ so that

$$TPP(pl(p), p(h_i)) \vee NTPP(pl(p), p(h_i)) \vee EQ(pl(p), p(h_i)).$$

Thus, we know that the disk passes through the post hole because there is a projection of it that is non-tangential proper part of the hole's entry boundary.

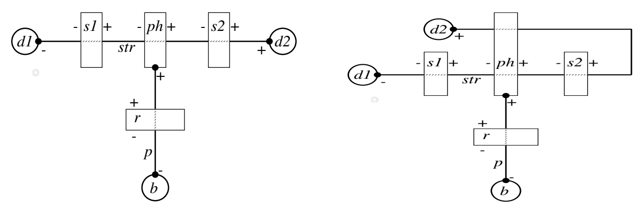
Now that we have a way of checking whether a particular object can pass through a determined hole. In the next section we abstract shapes away and deal only with the object's names allied with a relation $pass\_o$, defining the action of passing an object through a hole (as discussed in Section 5).

4 A formalisation of the Fisherman's Folly

The formalisation of the puzzle assumes a sort for holes, and a second sort called *long object* that includes the string (str) and the post (p). If we momentarily forget the hole in the post, both objects are "long" in the sense that, in principle, they could be simultaneously crossing several holes. Another common feature is that we can recognise two tips in each of these objects, which are the endpoints of their major axes. Thus for each long object l, we represent its two tips by l^- and l^+. Each tip of a long object can be *linked* to something else. For instance, the string tipss are connected to the disks, whereas the bottom of the post is linked to the post base. Although encoded in the same sort, the string and the post have an obvious difference: the flexibility inherent in the former, which is not a characteristic of the latter. As we shall see, in this work the string's flexibility is reflected in the constraints imposed on the movements of the domain objects connected to it.

The rest of objects in the puzzle, that is the disks (d_1, d_2) and the post base (b), will just be classified as *regular* objects, without showing any particular feature, excepting perhaps that by their shapes, they can or cannot pass through each given hole.

We illustrate the formalisation of the puzzle domain using diagrams. In these diagrams a box represents a *hole*, a circle a *regular object*, a thick line stands for a *long object* and a small black circle represents a link or connection. An example of this graphical representation is shown in Figure 7. Note how the post has been divided into a post hole (ph) linked to the top part of the post body (p). It may be reasonably objected that this division is not so natural, but it is also true that it would be possible to build an "equivalent" puzzle where, for instance, ph was a ring and p a second string.



(a) S_0 (initial state) (b) S_1 (d_2 passes left)

Figure 7: A pair of puzzle states.

Since each long object X can be crossing several holes, we will represent this using a list of crossings, called $chain(X)$. This list should collect the sequence of all hole crossings made by object X starting, for instance, from its negative tip and moving towards its positive one, whereas the same hole may occur several times in the list. Furthermore, the *direction* in which the string crosses the hole is also relevant. To see why, assume we represent the situation for Figure 8(a) simply as $chain(str) = [ph, s_1, ph, s_2, ph, ph]$. Then, we could not distinguish Figure 8(a) from Figure 8(b):

Figure 8(b) clearly represents a substantially different situation wrt Figure 8(a): the disk d_2 is now to the *right* (or positive side) of the

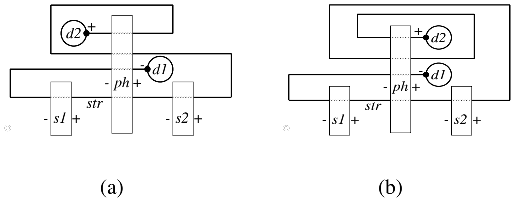


Figure 8: Two different states that could not be distinguished without crossing directions.

post hole ph. Instead, a more suitable representation of Figure 8(a) would be: $chain(str) = [ph^-, s_1^+, ph^+, s_2^+, ph^-, ph^-]$. The list $chain$ is, therefore, a more direct representation of the ideas discussed in Section 2.

Using the formalisation of the puzzle in terms of the list *chain*, presented in this section, we are able to define the basic actions on domain objects, as introduced below.

5 Planning with the puzzle

In this section we define the two actions that implement the basic movements on the puzzle's objects: the action $pass\_o$ (passing an object through a hole) and the action $pass\_h$ (passing a hole through another hole).

5.1 Moving object endings: action $pass\_o$

The action $pass\_o(A, B)$ represents passing a long object tip A through some hole towards its tip B. For example, the execution $pass\_o(str^+, ph^-)$ in the initial state S_0 leads to S_1 (both depicted in Figure 7) and corresponds to moving the positive ending of str (currently linked to disk d_2) to the left of the post hole. It is clear that the execution of $pass(X^+, B)$ (resp. $pass(X^-, B)$) may equally mean that we are adding or removing the hole from $chain(X)$ depending on the context. For instance, the movement described above, $pass\_o(str^+, ph^-)$ should add ph^- to $chain(str)$ leading to the list $[s_1^+, ph^+, s_2^+, ph^-]$ in S_1, whereas performing $pass\_o(str^+, ph^+)$ in that state should return us to the initial situation S_0, removing ph^- from the list.

The possible effects of $pass\_o$ are depicted in Figure 9.

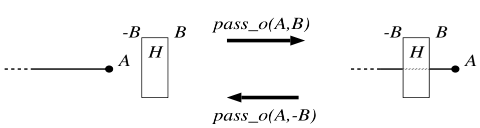


Figure 9: Possible effects of $pass\_o$.

Looking from the right to the left execution of $pass\_o$, we can conclude that, when we are performing $pass\_o(X^+, B)$ on the $chain(X) = [\ldots, -B]$, we must remove $-B$ from the tail of this chain. The analogous case would be $pass\_o(X^-, B)$ with $chain(X) = [-B, \ldots]$ where we would remove $-B$ from the head of $chain(X)$. If none of the two previous cases occur, then $pass\_o(A, B)$ is actually *inserting* a new crossing in $chain(X)$. Thus, $pass\_o(X^+, B)$ adds crossing B in the tail of $chain(X)$ whereas $pass\_o(X^-, B)$ adds crossing $-B$ to the chain head.

5.2 Passing holes through holes: action $pass\_h$

The previous action is not enough for describing the solution of the problem, since it does not take into account the movement of passing an (object containing a single) hole through another hole. To understand why, let us assume that, given the initial situation depicted in Figure 7 we tried to move the ring up. This is equivalent to move the post hole ph down the ring, that is, to pass ph through r^-. So we would need an action such as $pass\_h(A, B)$ where A is now a hole and B a hole pole. Back to the example, we would execute the action $pass\_h(ph, r^-)$ on the initial situation leading to the resulting state depicted in Figure 10.

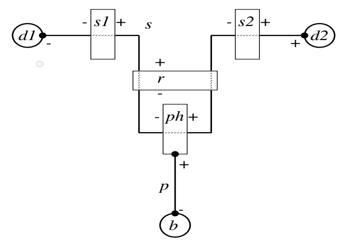
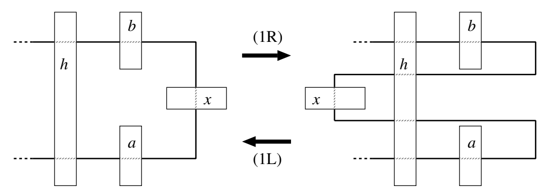


Figure 10: Possible effect of $pass\_h$.

The most relevant effect of this action is that the string chain, which was previously unrelated to the ring hole, has gained two new crossings as an effect of $pass\_h$. In other words, the list: $chain(X) = [s_1^+, \underline{ph^+}, s_2^+]$ has to be updated to: $chain(X) = [s_1^+, \underline{r^-, ph^+, r^+}, s_2^+]$

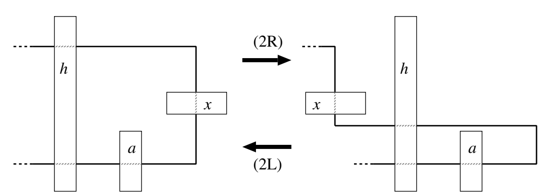
5.3 Possible movements

This section presents some possible movements that can be operated applying the two rules defined above. In the diagrams, we assume that upmost and rightmost endings of long objects are positive.



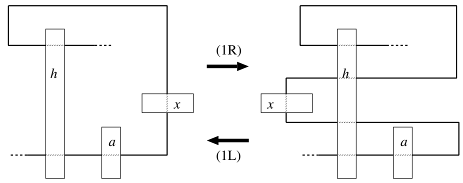
$[\ldots, h^+, a^+, \underline{x^+}, b^-, \ldots]$ $[\ldots, h^+, a^+, \underline{h^-, x^+, h^+}, b^-, \ldots]$

In Move (1R) we have that x is not contiguous to h in the chain. Therefore, by executing $pass\_h(x, h^-)$ we replace x^+ by the triple h^-, x^+, h^+. Movement (1L) starts in a state where x is preceded and succeeded by h in the chain but with alternate signs. A second possible movement would be:



$[\ldots, h^+, a^+, \underline{x^+, h^-}, \ldots]$ $[\ldots, h^+, a^+, \underline{h^-, x^+}, \ldots]$

The problem of Move 2R is that it cannot be applied when x is followed by h^+ instead of h^-, as shown in the instance of movement 1, as follows:



$[\ldots, h^+, a^+, \underline{x^+}, h^+, \ldots]$ $\quad [\ldots, h^+, a^+, \underline{h^-, x^+, h^+}, h^+, \ldots]$

In general, assuming we want to execute $pass\_h(x, e)$ and x is crossed by some string, then for any string Y crossing x, and any occurrence of x in $chain(Y)$ we have the following list of possible movements:

(1R) $chain(Y) = [\ldots, a, \underline{x^z}, b, \ldots] \Longrightarrow [\ldots, a, \underline{e, x^z, -e}, b, \ldots]$ with $a, b \notin \{e, -e\}$ or $a = e, b = -e$.

(1L) $chain(Y) = [\ldots, \underline{-e, x^z, e}, \ldots] \Longrightarrow [\ldots, \underline{x^z}, \ldots]$

(2R) $chain(Y) = [\ldots, a, \underline{x^z, e}, \ldots] \Longrightarrow [\ldots, a, \underline{e, x^z}, \ldots]$ with $a \neq -e$

(2L) $chain(Y) = [\ldots, \underline{-e, x^z}, a, \ldots] \Longrightarrow [\ldots, \underline{x^z, -e}, a, \ldots]$ with $a \neq e\}$

Note that the above movements are complete, in the sense that if x occurs in $chain(Y)$ as follows $[\ldots, a, x^z, b, \ldots]$ both a and b could be equal to e, equal to $-e$ or none of the two. As a result, we would have $3 \times 3 = 9$ possibilities which, for brevity we do not depict here, but can be seen to be all covered by the movements above. The cases in which x is at head or tail end of the chain are also covered by assuming that the ends themselves are elements different from all the rest in the list.

Another important observation is that, while all the represented elements in each movement would be involved in the distinction of the movement type, only the underlined parts constitute the movement effect. This means, for instance, that in movement (2R), a is only used in the predecessor state, to establish that we have a (2R) movement and not a (1L), whereas in the successor state, it could be the case that a results moved to the left or even removed by the effect of another movement (remember that x may occur several times in the chain). An example of this *accumulated* movement would be, for instance, the execution of $pass\_h(x, h^+)$ on the list $[h^-, x^+, h^+, x^-, h^+]$ where we would perform (1L) on the first x and (2R) in the second leading to $[x^+, h^+, x^-]$.

With the representation developed above we can now formally express one solution to the Fisherman's puzzle and the sequence of states involved in its execution. Figure 11 shows this solution step by step and depicts the corresponding spatial configurations of states S_2 through S_5 (S_2 and S_3 were already shown in Figure 7).

Clearly, S_5 is a solution, since the ring r is not passed through any long object. Note that the action performed in state S_1 is actually a combined one. This is because moving the ending p^+ to r^- implies passing also the post hole, as p^+ and ph are linked.

The next section discusses an implementation of the puzzle into an action language.

5.4 A simple Prolog implementation

As an actions domain, our abstraction of the Fisherman's folly is quite simple in the sense that most complex features of actions reasoning are not required for the problem. We deal with two actions,

state	$chain(p)$	$chain(str)$	next action(s)	movements
S_0	$[r^+]$	$[s_1^+, ph^+, s_2^+]$	$pass\_o(str^+, ph^-)$	
S_1	$[r^+]$	$[s_1^+, ph^+, s_2^+, ph^-]$	$pass\_o(p^+, r^-)$ & $pass\_h(ph, r^-)$	(1R) × 2
S_2	$[]$	$[s_1^+, r^-, ph^+, r^+, s_2^+, r^-, ph^-, r^+]$	$pass\_h(s_2, r^-)$	(1L)
S_3	$[]$	$[s_1^+, r^-, ph^+, s_2^+, ph^-, r^+]$	$pass\_h(r, ph^+)$	(2R)+(2L)
S_4	$[]$	$[s_1^+, ph^+, r^-, s_2^+, r^+, ph^-]$	$pass\_h(s_2, r^+)$	(1L)
S_5	$[]$	$[s_1^+, ph^+, s_2^+, ph^-]$		

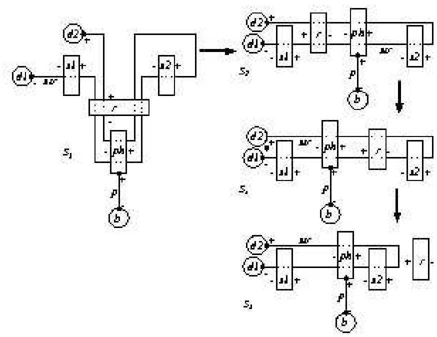


Figure 11: A formal solution for the Fisherman's puzzle.

$pass\_o$ and $pass\_h$ whose execution causes a direct effect on the fluents $chain(X)$, for each long object X. Rather than providing a precondition per each action, we have found more convenient to specify general constraints in which the actions are not executable. We have used a Prolog predicate *nonexecutable(S,A)* to represent when an action *A* cannot be performed in a state *S*, including the rules:

$$nonexecutable(\_, pass\_o([X, \_], [H, \_])) :- \\ cannot\_pass(X, H), !.$$
$$nonexecutable(S, pass\_o(P, [H, \_])) :- \\ member(linked\_to(P) = X, S), cannot\_pass(X, H), !.$$
$$nonexecutable(\_, pass\_h(X, [H, \_])) :- cannot\_pass(X, H), !.$$
$$nonexecutable(S, pass\_h(X, \_)) :- \\ member(linked\_to(\_) = X, S), !.$$

The pairs *[X,Y]* are used to represent tips of long objects, so that for instance, *[p,+]* would stand for p^+. The fourth non-executability condition is used to force that, when an object tip is connected to a hole, the planner tries first to pass the object tip and later the hole in a same transition. In this way we avoid irrelevant solutions were we can try to do it in the opposite ordering, with exactly the same effects.

Of course, the main difficulty of this scenario from the standpoint of planning representation languages (STRIPS, ADL, PDDL) or even formalisms for action reasoning is the need for dealing with lists and pattern matching. In fact, this has motivated the choice of Prolog in order to build this preliminary prototype. Our implementation includes a Prolog predicate *process_chain(X,HP,L1,L2)* to describe the effect of performing $pass\_h(X, HP)$ on chain list *L1* leading to list *L2*. An example showing the implementation of movement (1R) is shown below.

$$process\_chain(X, HP, [A, [X, S], B|Cs], Ds) :- \\ opposite(HP, HP1), A \backslash = HP1, B \backslash = HP, !, \\ process\_chain(X, HP, [B|Cs], Ds0), \\ Ds = [A, HP, [X, S], HP1|Ds0].$$

Note how the right neighbour of *[X,S]*, the crossed tip *B*, is used to keep processing the rest of the chain in the recursive call, and how the result of this recursive call *Ds0* may not contain *B* any more – it could be deleted by an accumulated movement (1L).

From the planning algorithm point of view, we have just implemented a blind search, relying on depth-first forward chaining with an iterative deepening strategy. Since the plan is really short, the Prolog program[5] just takes 10.30 seconds to find a first solution, despite of the inefficient planning strategy.

It is interesting to observe that the program actually finds several solutions in five steps. For instance, apart from the obvious symmetric solution where we begin working with d_1 instead of d_2 making $pass(s^-, ph^-)$, we also get a variant of the depicted solution in Figure 11 where to reach state S_3 we execute instead the sequence $pass\_o(str^+, ph^-)$, $pass\_h(s2, r^-)$ and $pass\_o(p^+, r^-)$ & $pass\_h(ph, r^-)$. This solution is not valid for the original puzzle since, although both the sphere and the post can pass through the ring, they cannot do so *simultaneously*. For immediate future work, we plan extending our representation so that the predicate *cannot_pass* describes when a group of objects cannot be altogether simultaneously passing through a given hole.

We have also made some small variations of the original puzzle by changing some of the premises. For instance, by allowing spheres to pass through the post hole we directly get a shorter solution: $pass\_o(str^+, ph^-)$, then $pass\_h(s2, ph^-)$, that gets the string-disks-spheres tandem out of the post and, finally, $pass\_o(p^+, r^-)$ & $pass\_h(ph, r^-)$ to get the ring free.

6 Concluding remarks

In this work we presented some results of ongoing research on a challenging problem for spatial reasoning and common sense knowledge formalisation, namely, the problem of reasoning about spatial domains that contain non-trivial objects such as holes and strings. We proposed a representation whereby holes identify sub-spaces in which objects could be located. The string in this paper is formalised as a long object that restricts the movement of the objects linked to it. In fact, the flexibility of this object is not fully explored in this work, as we abstracted away the possibility of tying knots. This issue shall be investigated in future work.

The formalisation of spatial knowledge is not the only challenge in the domain assumed in this work. Solving the puzzle also involves interesting issues that are beyond search (or planning) through a state space. For instance, when changing the spatial configuration of the puzzle, a person has a *selective observation* of domain objects, whereby only a portion of the space is observed. Actions are, thus, only applied within this limited view of the scene. A second issue is the minimisation of the *spatial configuration complexity*; the string allows for the application of a sequence of actions rolling the string around the post, or through it, many times. Minimisation of the puzzle's configuration complexity could be used as an heuristic in an automated problem solver. However, how this complexity should be measured is still an open problem. Finally, when trying to solve the puzzle for the first time, a human agent may not know every constraint or every possible movement of the domain objects. Searching for an automated way in which such spatial knowledge can be assimilated is also a very challenging issue for further investigations.

[5] We have used SWI-Prolog 5.2.11 interpreter running on Linux Mandrake 10 on a Pentium IV 1.5 GHz with a RAM of 512 MB.

REFERENCES

[1] R. Casati and A. C. Varzi, *Parts and Places: the structures of spatial representation*, MIT Press, 1999.

[2] E. Clementini, P. di Felice, and D. Hernández, 'Qualitative representation of positional information', *Artificial Intelligence*, (1997).

[3] A. G. Cohn, B. Bennet, J. Gooday, and N.M. Gotts, 'Representing and reasoning with qualitative spatial relations about regions', in *Spatial and Temporal Reasoning*, ed., Oliviero Stock, 97 – 134, Kluwer Academic Publishers, (1997).

[4] A. G. Cohn and S. M. Hazarika, 'Qualitative spatial representation and reasoning: An overview', *Fundamenta Informaticae*, **46**(1-2), 1–29, (2001).

[5] D. Hernández, E. Clementini, and P. di Felice, 'Qualitative distances', in *LNAI*, eds., W. Kuhn and A. Frank, Springer-Verlag, (1995).

[6] R. C. Meathrel and A. P. Galton, 'A hierarchy of boundary-based shape descriptors', in *Proc. of IJCAI*, pp. 1359–1364, (2001).

[7] T. Morita, J. Takamatsu, K. Ogawara, H. Kimura, and K. Ikeuchi, 'Knot planning from observation', in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp. 3887–3892, (2003).

[8] D. Randell, Z. Cui, and A. Cohn, 'A spatial logic based on regions and connection', in *Proc. of the KR*, pp. 165–176, Cambridge, U.S., (1992).

[9] D. A. Randell, A. G. Cohn, and Z. Cui, 'Computing transitivity tables: A challenge for automated theorem provers', in *Proc. of CADE*, LCNS, Saratoga Springs, U.S., (1992). Springer Verlag.

[10] K. Reidemeister, *Knot Theory*, BCS Associates, 1983.

[11] P. Santos and M. Shanahan, 'Hypothesising object relations from image transitions', in *Proc. of the* 15^{th} *European Conference on Artificial Intelligence (ECAI-02)*, ed., Frank van Harmelen, pp. 292–296, Lyon, France, (2002).

[12] M. Shanahan, 'Default reasoning about spatial occupancy', *Artificial Intelligence*, **74**(1), 147–163, (1995).

[13] *Spatial and Temporal Reasoning*, ed., Oliviero Stock, Kluwer Academic Publishers, 1997.

[14] A. C. Varzi, 'Reasoning about space: The hole story', *Logic and Logical Philosophy*, **4**, 3–39, (1996).