# An investigation of actions, change, space within a hole-loop dichotomy

**Paulo E. Santos**
Elect. Eng. Dep.
FEI, São Paulo, Brazil.
E-mail: psantos@fei.edu.br

**Pedro Cabalar**
Department of Computer Science,
University of Corunna, Spain.
E-mail: cabalar@udc.es

**Abstract**

This work investigates the spatial knowledge of a domain composed of non-trivial objects such as *strings* and *holed objects*. To this aim, we consider the formalisation of puzzle-like examples as the starting point for the development of Knowledge Representation systems. The present paper concentrates on the representation of "loops" (or loop-like regions) that can be formed by a flexible string, and that may play an essential part in the solution of physical problems involving strings.

## Introduction

Understanding the reasoning processes involved in spatial knowledge is one of the key issues in the investigation of cognition, as space not only shapes our actions in the commonsense world, but also serves as the scenario in which our everyday experiences take place. Research in*Qualitative Spatial Reasoning* (QSR) (Ligozat 2011) attempts the logical formalisation of spatial knowledge based on primitive relations defined over elementary spatial entities. For instance, QSR theories include a mereotopological theory based on the connectivity between spatial regions (Randell, Cui, and Cohn 1992), the definition of occlusion and parallax (Randell, Witkowski, and Shanahan 2001; Randell and Witkowski 2002), spatial vagueness (Cohn et al. 1997; Guesgen 2002a), the abductive assimilation of sensor data (Santos and Shanahan 2002; 2003; Santos 2007),as well as the definition of qualitative theories about distance(Hernández, Clementini, and di Felice 1995; Guesgen 2002b), boundaries (Meathrel and Galton 2001), shapes (Schlieder 1996; Clementini and Felice 1997) and so forth (Cohn and Hazarika 2001). An interesting and, perhaps, less studied family of QSR problems has to do with domains composed of non-trivial objects such as *strings* and *holed objects*. In a series of papers (Cabalar and Santos 2006; 2011; Santos and Cabalar 2008) we have previously introduced the formalisation of puzzle-like examples involving holes and strings. The use of puzzles as starting point for the development of Knowledge Representation (KR) approaches usually has the advantage of offering a simple scenario with a small number of objects while keeping enough complexity for a KR challenging problem. In our previous work, however, several important limitations were considered to focus on a family of puzzles with common features. Among them, we disregarded the representation of knots, the formation of string loops, and we omitted the intermediate steps where a holed object was crossing another hole.

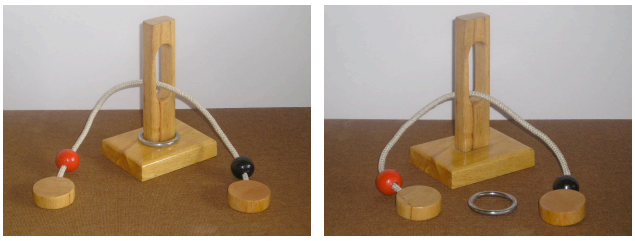
This paper is a first step towards removing the last two restrictions. We outline and identify the main representational problems derived from the introduction of loops in a flexible string and we consider the representation of states where holed objects (or loops) are crossing other holes. We sketch some possible formal solutions whose complete development is still under study.



## Problems with previous formalisations

Previous work (Cabalar and Santos 2006; 2011; Santos and Cabalar 2008) has concentrated on the formalisation and automated solution of a family of puzzles whose goal is to release a ring from an entanglement of objects, including one or several strings, and (possibly holed) rigid objects, maintaining the objects' physical integrity. The main representative of this family of puzzles is probably the so-called *Fisherman's Folly* (shown in Figure 1). The elements of this puzzle are a holed post ($Post$) fixed to a wooden base ($Base$), a string ($Str$), a ring ($Ring$), a pair of spheres ($Sphere1, Sphere2$) and a pair of disks ($Disk1, Disk2$). The spheres can be moved along the string, whereas the disks are fixed at each string endpoint. The string passes through the post's hole in a way that one sphere and one disk remain on each side of the post. 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.

In the initial state (shown in Figure 1(a)) the post is in the middle of the ring, which in its turn is supported on the post's base. The goal of this puzzle is to find a sequence of (non-destructive) transformations that, when applied on the domain objects, frees the ring from the other objects, regardless their final configuration. Figure 1(b) shows one possible goal state. As we would do in natural language, saying that "the string passes through the sphere" (hole) or that "the post passes through the ring" (hole), we will simply identify holes with their host objects (if an object hosted several



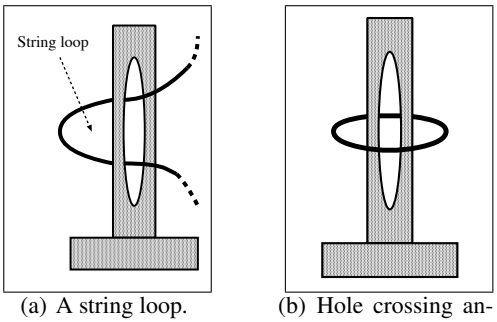
(a) Initial (b) Goal: the ring must be free

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

holes, a more elaborated notation would be required).

A simple planning system capable of finding a solution to the Fisherman's Folly puzzle was presented in (Cabalar and Santos 2006), where the states of the puzzle were represented as lists containing the sections of a long object between hole crossings. Based on the hole ontology from (Casati and Varzi 1999), in (Santos and Cabalar 2008), a mereotopological representation of the domain objects was presented. The work in (Cabalar and Santos 2011) developed a representation of the puzzle actions in a Situation Calculus (McCarthy and Hayes 1969) framework developed in Quantified Equilibrium Logic (Pearce and Valverde 2008), where we were interested in a solution that was tolerant to elaborations.

As said before, in the previous formalisations there were several possible states that were disregarded, since they were irrelevant for the puzzle solution. One of them is the formation of knots and multi-knots, which we keep aside for a future study, but the other two, the formation of string loops and the possibility of holes crossing other holes are in fact relevant for other similar puzzles that could not be captured before. These two features are respectively shown in Figures 2(a) and 2(b). (Cabalar and Santos 2011), for instance, assumed that, when pulling from a string that was crossing a hole, it was always done until loops were undone. Similarly, when a holed-object crossed another hole, it was always done in a complete way, disregarding intermediate states like Figure 2(b).




(a) A string loop. (b) Hole crossing another hole.

Figure 2: Two situations not formalised in (Cabalar and Santos 2011).

These limitations may easily suppose a lack of elaboration tolerance, even without considering new puzzles. For instance, the variation of Fisherman's Folly shown in Figure 3 can be also found in shops and toy stores. It is essentially the *same* puzzle with the difference that the holed post has been replaced by a long metallic arc. However, this is basically the same configuration as before, satisfying the same restrictions (i.e., spheres cannot cross through it, but wooden disks can). In particular, in this variation, the hole formed by the long arc must be crossing the ring hole in the initial state, and there is no way to evade from that.

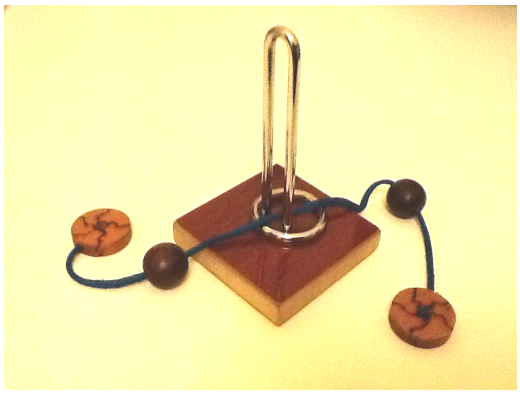


Figure 3: A variation of Fisherman's Folly.

In the Fisherman's Folly, we know at least that an equivalent instance of the puzzle can be solved without dealing with these situations, but in other puzzles, the solution mandatorily requires dealing with these features. One of these puzzles, called the "*easy-does-it*," constitutes our new motivating example and is shown in Figure 4.

Figure 4: Easy does it

## Spatial puzzles as chains

As a starting point, we will consider the simple formalisation presented in (Cabalar and Santos 2006) for representing the Fisherman's Folly and analogous puzzles, extending it afterwards to cope with the new features. The main idea of that formalisation relies on a list structure $chain(S)$ capturing the sequence of hole crossings that each string $S$ traverses from one of its tips to the other. To handle this, objects into three sorts: *holes* (in the example, this includes the post hole, the ring hole and the holes through the spheres), long objects (including the string and the post), and regular objects (including all the remaining objects). For each hole $h$, its faces

are distinguished: $h^-$ and $h^+$; and for each long object $l$ its tips $l^-$ and $l^+$ are defined.

For helping the reader to figure out a puzzle state, we use schematic representations like the one in Figure 5, which shows the initial state of Fisherman's Folly. Arrows correspond to segments of long objects, defined between pairs of hole crossings, or between a hole crossing and a tip. These arrows point in the direction from tip $l^-$ to tip $l^+$ of a same long object $l$. Ellipses represent holes and boxes are linked regular objects. The positive face of a hole is determined by a right thumb rule from the small arrow in the ellipse (similar to a spin direction). Notice how a hole could also be seen as a long object closed to itself joining its two tips.

Although not essential for the $chain$-based representation, it is also helpful to identify the *segments* of a long object. When required, we will represent them using the notation $x:i$ where $x$ is a long object and $i$ a segment label.

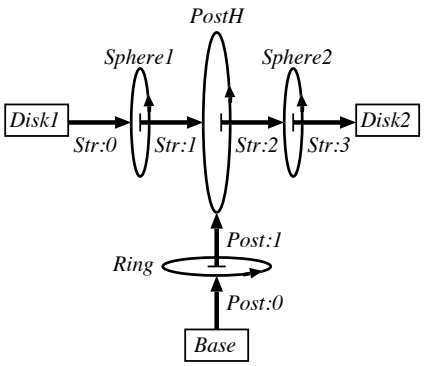



Figure 5: Schematic representation of the initial state.

As said before, given a long object $x$, the structure $chain(x)$ represents the sequence of all hole crossings when traversing $x$ from its negative tip to its positive one. For instance, the state shown in Figure 5 is represented by the following two chains: $chain(P) = [Ring^+]$ (for the post - $P$ object) and $chain(Str) = [Sphere1^+, PostH^+, Sphere2^+]$ (for the string - $Str$ object). The former represents that the long object $Post$ ($P$) crosses the ring hole and the latter states that the string crosses the hole on the sphere 1, the post hole and the hole on the sphere 2, respectively. Note that, for brevity, only the outgoing hole faces are shown, following the direction negative to positive tip. We will sometimes add a second argument to the fluent $chain$ indicating the state related to the chain description; e.g. $chain(x, S_i)$ is a predicate representing the chain of crossings for long object $x$ at state $S_i$.

In (Cabalar and Santos 2006) it sufficed with defining a single action to represent the relevant movements of puzzle objects, This action, denoted $pass(B, h^i)$, represents the operation of passing a bundle of objects $b$ towards the $i$ face of a hole $h$ ($i \in \{+, -\}$). The effects of $pass$ either add or delete hole crossings from the $chain$ on which it is applied. In the case of Fisherman's Folly, for instance, the sequence of actions forming a solution corresponded to:

1. $pass(\{Str^+, Disk2\}, PostH^-)$,
2. $pass(\{Post^+, PostH\}, Ring^-)$,
3. $pass(\{Sphere2\}, Ring^-)$,
4. $pass(\{Ring\}, PostH^+)$ and
5. $pass(\{Sphere2\}, Ring^+)$,

being the resulting sequence of states shown on Figure 6 – see (Cabalar and Santos 2011) for a graphical representation. Note that state 5 has actually reached the goal since, at this point, the ring hole $Ring$ does not occur in any list, i.e., it is not crossed by any long object.

| state | $chain(P)$ | $chain(Str)$ |
|---|---|---|
| $S_0$ | $[Ring^+]$ | $[Sphere1^+, PostH^+, Sphere2^+]$ |
| $s_1$ | $[Ring^+]$ | $[Sphere1^+, PostH^+, Sphere2^+, PostH^-]$ |
| $s_2$ | [ ] | $[Sphere1^+, Ring^-, PostH^+, Ring^+, Ring^-, Sphere2^+, Ring^+, PostH^-]$ |
| $s_3$ | [ ] | $[Sphere1^+, Ring^-, PostH^+, Sphere2^+, PostH^-, Ring^+]$ |
| $s_4$ | [ ] | $[Sphere1^+, PostH^+, Ring^-, Sphere2^+, Ring^+, PostH^-]$ |
| $s_5$ | [ ] | $[Sphere1^+, PostH^+, Sphere2^+, PostH^-]$ |

Figure 6: A formal solution for the Fisherman's puzzle and its graphical representation.

Let us see now which new elaborations are required to formalise the solution steps of the Easy-does-it puzzle at a similar level of abstraction[1].

## Solving the Easy-does-it puzzle

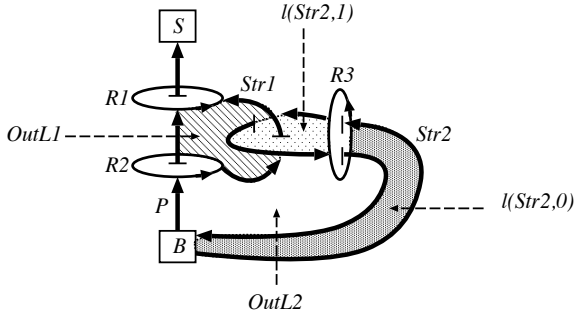



Figure 7: Diagram

Figure 7 shows a possible initial state of the Easy-does-it-puzzle using a diagram as those introduced in the previous section. As a main important novelty, we can see how loops in the string form a new kind of "holes" that are represented with two faces + and -, as done previously for holes in rigid objects (Cabalar and Santos 2006; 2011;

[1]We are also exploring the possibility of directly using Reidemeiter's moves from knot-theory, as they provide a more fundamental and fine-grained description of the movements. However, the result involves many more steps and a considerably higher search space for planning.

Santos and Cabalar 2008). The only distinction here is that loops can appear and disappear and that their sizes may change along time. We will denote loops on a particular string with the expression $l(s, n)$, where $s$ is a string and $n$ represents the ordinal (beginning with 0) of the first segment of $s$ involved in the loop, scanning the string from its negative to its positive tips (note that many loops can be done/undone on a single string). For now we are not taking into account the origin of loops, only that they define an empty space (bounded by a string) through which an object can pass.

The Easy-does-it domain is composed of three rings: two at the tips of a string, and crossed by the post (denoted by $R_1$ and $R_2$) and a third ring ($R_3$), which we call *main* ring (since this is the one that should be released from the system of objects). $R_1$ and $R_2$ are "locked in" the post, since they cannot pass through the sphere ($S$) fixed at the post's tip. The domain also has two strings $Str_1$ and $Str_2$ that are entangled with each other, a post $P$, a base $B$ and six holes: the main ring hole ($R_3$), the holes on the two rings attached to string 1 ($R_1$ and $R_2$), the hole (loop) made by string 1 ($Str_1$). In the initial state (Figure 7) string 2 ($Str_2$) makes two loops, that are separated by $R_3$. The loop at the tip of $Str_2$ (whose area is dot-filled) will be denoted by $l(Str_2, 1)$ and the one connected to the base ($B$) (shadowed in gray) will be denoted by $l(Str_2, 2)$. The reason for this notation shall become clear in the next section. $Str_1$ also forms a loop (filled with diagonal lines) that involves $R1$, $R2$ and the post $P$ too. This loop, called $OutL_1$, is somehow different to loops in $Str_2$ since it is closed by objects that are external to $Str_1$ and its crossings (the post has no direct interaction to $Str_1$). These loops will be called *outer*, as opposed to those using $l(\cdot, \cdot)$ notation that will be called *inner*. In the example, we can find a second outer loop $OutL_2$ (in this case, irrelevant for the solution) formed by the set of objects $Str_1$, $Str_2$, $B$, $R_2$ and $P$. To sum up, the chains of crossings at the initial state $S_0$ depicted in Figure 7 is:

$$chain(P, S_0) = [B, R_2^+, R_1^+, S] \quad (1)$$

$$chain(Str_1, S_0) = [R_2, l(Str_2, 1)^+, R_1] \quad (2)$$

$$chain(Str_2, S_0) = [B, R_3^-, OutL_1^+, R_3^+, B] \quad (3)$$

## Detecting inner loops

Chains of crossings reveal the set of loops formed by a string. Whenever we repeat a link to a same object or a crossing through a same object[2] (regardless the possible crossing direction), we form a loop in a string. For instance, in $chain(Str_2)$ explained above we can detect the two loops:

$$chain(Str_2) = [\underbrace{B, \overbrace{R_3^-, OutL_1^+, R_3^+}^{l(Str_2,1)}, B}_{l(Str_2,0)}]$$

As said before, we denote the loops using the position in $chain(S)$ (the first position is 0) of the origin of the first segment forming the loop. In this case, we have a loop formed by the pair of links to object $B$, that close both tips of the string. This loop is denoted by $l(Str_2, 0)$ because the first $B$ is at position 0. There is no ambiguity, since the closing part of the loop will be the next occurrence of $B$ in the list, from left to right (in this case, position 4, which is the last one). The second loop is formed by the two crossings of ring $R_3$, in this case, in opposite directions. This loop is denoted $l(Str_2, 1)$ signifying that the origin of the loop is at position 1 (crossing $R_3^-$). We can deduce that the loop's end will be the next position to the right in which we find a crossing through object $R_3$ : in the example, position 3.

The same object can form several loops in the string, as in:

$$chain(S) = [\underbrace{R1^+, R2^-, R3^+, \overbrace{R1^-}^{l(S,3)}}_{l(S,0)}, R4^-, R1^+]$$

We can use Allen's interval algebra (Allen 1983) to classify the possible relations involving a pair of loops $x$ and $y$ in a same string. However, not all relations in Allen's algebra are possible here, since two different loops cannot share the same left (resp. right) end. As a result, the relations "starts," "finishes" and their inverses are not allowed. This leaves the remaining 9 possibilities: "before," "meets," "overlaps," and "during", their inverses, plus the relation "equal."

The next section presents a chain description of one possible solution for the Easy-does-it puzzle.

## Solution Steps

This section presents a sequence of changes in the puzzle's states that is one possible solution to the puzzle. We do not claim that this is the optimal solution, or that the formalisation below solves all the representational issues related to the Easy-does-it, but that it is an initial formal description on top of which interesting points can be discussed regarding the representation and reasoning about strings and loops. More importantly, the sequence of states in terms of chains of crossings may help us to explore the general effect of the actions involved in terms of loop creation and removal, something we only outline here, leaving its complete formalisation for future study.

As we saw before, the initial state shown in Figure 7 corresponds to the chains (1)-(3). We analyze next, step by step, each state transition in the puzzle solution.

**First transition** The first step of the solution (from state $S_0$ (Fig. 7) to $S_1$ (Fig. 8(a))) involves passing the segment $Str_2 : 1$ towards $R_1^+$. This results on the state shown in Figure 8(a), with the following chain description.

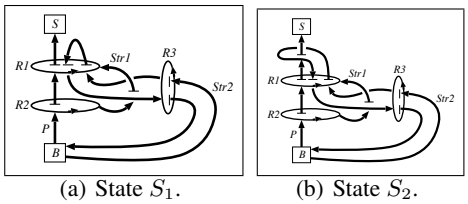
$$chain(P, S_1) = [B, R_2^+, R_1^+, S]$$

$$chain(Str_1, S_1) = [R_2, l(Str_2, 1)^+, R_1]$$

$$chain(Str_2, S_1) = [B, R_3^-, R_1^+, R_1^-, R_3^+, B]$$

In this process we have created a third loop in $Str_2$, $l(Str_2, 2)$, that can be easily seen as the pair $\langle R_1^+, R_1^- \rangle$ in

[2] If we represent the string tips $S^-, S^+$ at both ends of the chain, these are the only exceptions. However, if we make them to coincide in the space, that is, we add the assumption that $S^- = S^+$ represent the same object, then they form again a loop.



(a) State $S_1$. (b) State $S_2$.

Figure 8: First and second transitions.

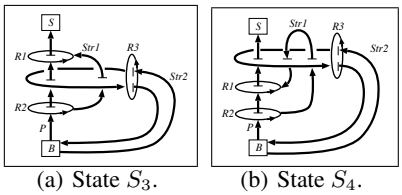
$chain(Str_2)$. On the other hand, $l(Str_2, 1)$ is delimited now by $R_3$, segment $Str_2{:}1$, $R_1$ and segment $Str_2{:}3$.

Note that the crossing through $OutL_1$ has disappeared from $chain(Str_2)$.

**Second transition** The second action is to pass the sphere $S$ towards $l(Str_2, 2)^+$, resulting in state $S_2$ (Fig. 8(b)):

$$\begin{aligned} chain(P, S_2) &= [B, R_2^+, R_1^+, l(Str_2, 2)^+, S] \\ chain(Str_1, S_2) &= [R_2, l(Str_2, 1)^+, R_1] \\ chain(Str_2, S_2) &= [B, R_3^-, R_1^+, R_1^-, R_3^+, B] \end{aligned}$$

**Third transition** Then the segment $Str_2 : 2$ should be pulled towards $R_1^-$ (resulting on the state $S_3$, Fig. 9(a)):



(a) State $S_3$. (b) State $S_4$.

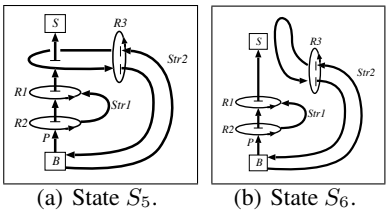
Figure 9: Third and fourth transitions.

$$\begin{aligned} chain(P, S_3) &= [B, R_2^+, l(Str_2, 1)^+, R_1^+, S] \\ chain(Str_1, S_3) &= [R_2, l(Str_2, 1)^+, R_1] \\ chain(Str_2, S_3) &= [B, R_3^-, R_3^+, B] \end{aligned}$$

Notice that this movement has collapsed the segments $Str_2{:}1$, $Str_2{:}2$ and $Str_2{:}3$ into a single $Str_2{:}1$. Similarly, loops $l(Str_2, 1)$ and $l(Str_2, 2)$ have become a single loop $l(Str_2, 1)$ again, as we had in the initial state. It is worth pointing out also that, although the new constraint added to string 2 (that it goes around the post) is not shown in $chain(Str_2, S_3)$, it is clearly represented on $chain(P, S_3)$.

**Fourth transition** We could begin passing $R_1$ down $l(Str_2, 1)^-$. This should leave a loop in $Str_1$ as depicted in Fig. 9(b) (state $S_4$). We would, then, pull from segment $Str_1{:}1$ (the one forming the new loop) down to $l(Str_2, 1)^-$ (state $S_5$, Fig, 10(a)).

However, state $S_5$ could be directly reached in one step by considering a single action $pull(Str_1 : 1, l(Str_2, 1)^-)$, that is, when being at $S_3$, pass the second segment $Str_1 : 1$ of the string $Str_1$ down to $l(Str_2, 1)^-$. As ring $R_1$ is linked



(a) State $S_5$. (b) State $S_6$.

Figure 10: Fifth and sixth transitions.

to this segment, we get that the ring also crosses through $l(Str_2, 1)^-$. This results on the state represented on Fig. 10(a) with the following chain description:

$$\begin{aligned} chain(P, S_5) &= [B, R_2^+, R_1^+, l(Str_2, 1)^+, S] \\ chain(Str_1, S_5) &= [R_1, R_2] \\ chain(Str_2, S_5) &= [B, R_3^-, R_3^+, B] \end{aligned}$$

**Fifth transition** The next action is to move the loop in $Str_2$ upwards above the sphere, obtaining state $S_6$ (Fig. 10(b)). Formally, this means passing the sphere $S$ down to $l(Str_2, 1)^-$, that is $pass(S, l(Str_2, 1)^-)$, as shown below. The chain description of Figure 10(b) is described below.

$$\begin{aligned} chain(P, S_6) &= [B, R_2^+, R_1^+, S] \\ chain(Str_1, S_6) &= [R_1, R_2] \\ chain(Str_2, S_6) &= [B, R_3^-, R_3^+, B] \end{aligned}$$

**Sixth transition** Finally, the loop in $Str_2$, formed by $R_3$, can be removed. Formally, pull segment $Str_2 : 1$ towards $R_3^+$, i.e., $pull(Str_2{:}1, R_3^+)$, resulting in the following chain description (that describe state $S_7$, represented in Figure 11).

$$\begin{aligned} chain(P, S_7) &= [B, R_2^+, R_1^+, S] \\ chain(Str_1, S_7) &= [R_1, R_2] \\ chain(Str_2, S_7) &= [B, B] \end{aligned}$$

Note that the main ring $R$ is not present in any of the last chain descriptions. Therefore, $R$ is free from the set of entangled objects, solving the puzzle.

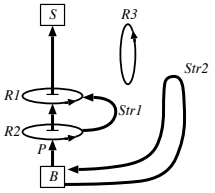



Figure 11: State $S_7$.

## Actions creating and removing loops

In transition 1 and 6 (and also 4, if we consider an intermediate step) we had either creation or unwinding of loops. Note that we can undo all the steps in reverse order from the goal situation to the initial one. Each action has an analogous reverse movement. Transition 1 creates a loop when

done forward, but we can also consider the loop removal when done backwards. Transition 6 removes the loop when done forward and creates it if done backwards.

Let us start by considering the idea of loop creation. Apart from passing an object to a hole side, we need to consider now an action $pick(x : i, p)$ meaning that we pick some arbitrary point in segment $x : i$ towards the hole side $p$. This pick movement does not drag the whole segment completely. In principle, $pick$ always be executed regardless the origin and target of segment $x : i$ and will always create a new loop[3]. Figure 12 illustrates the behavior of $pick$ by showing two consecutive applications of this action. Note how, when we do the second pick on the same string portion, we do not return to the initial situation, but we create a new loop instead.

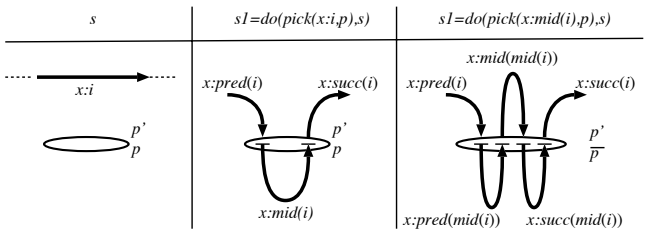



Figure 12: Two consecutive $pick$ actions.

The second movement related to loops we need to consider is the action of *removing a loop*. This movement is done by completely pulling a string segment towards some hole side. Note the difference between $pull(x : i, p)$ in Figure 13 and $pick(x : i, p)$ in Figure 12.

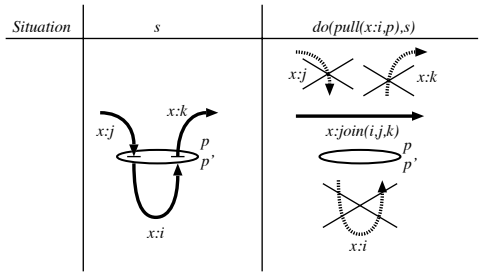



Figure 13: Removing the loop by pulling a string segment.

Using these actions, we can describe the sequence of movements as follows:

1. $pick(Str_2 : 1, R_1^+)$
2. $pass(\{P^+, S\}, l(Str_2, 2)^+)$
3. $pull(Str_2 : 2, R_1^-)$
4. $pass(\{Str_1^+, R_1\}, l(Str_2, 1)^-)$ and then $pull(Str_1 : 1, l(Str_2, 1)^-)$ or perhaps just the latter
5. $pass(\{P^+, S\}, l(Str_2, 1)^-$
6. $pull(Str_2 : 1, R_3^+)$

One could expect that for moving back from the goal to the initial state, we should be able to use the same actions (in reverse order) and simply switch their crossing directions. However, this is not the case, since the reverse movement of a $pick$ action sometimes must be a $pull$ action. Besides, the segments numbering vary depending on the direction in which we apply the movement. If we want to reverse the sequence of movements, the actions should be instead:

1. $pick(Str_2 : 0, R_3^+)$
2. $pass(\{P^+, S\}, l(Str_2, 1)^+)$
3. $pass(\{Str_1^+, R_1\}, l(Str_2, 1)^+)$
4. $pick(Str_2 : 1, R_1^+)$
5. $pass(\{P^+, S\}, l(Str_2, 2)^-)$
6. $pull(Str_2 : 2, R_1^-)$

## Discussion and Conclusions

In this paper we discussed the challenging problem of formally describing a particular characteristic of flexible objects such as strings: their capability of making loops that can be used (and reasoned about) as holes in spatial reasoning processes. We outlined an initial formalisation based on previous investigations on an automated solution for spatial puzzles. There is, however, still a long way to go before deploying this initial formalisation in a real application setting. For instance, we have provided a representation for the states and actions involved in the puzzle solution, but a complete formal description for the derivation of action effects is still part of ongoing work. In particular, we have distinguished between inner and outer string loops. While detection and manipulation of inner loops is clearer now, the treatment of outer loops is more elusive, since they are formed by entanglements of different objects.

Another aspect under study is the "reconstruction" of each state in the puzzle by physically passing one of the strings from one of its tips to the other and traversing all the hole crossings. For instance, in Figure 10(a), suppose we do not have string $Str_2$ and we want to add it to get state $S_5$. To this aim, we would join $Str_2^-$ to the base $B$, then pass tip $Str_2^+$ to $R_3^-$ and $R_3^+$ forming the loop $l(Str_2, 1)$, and finally joining it to $B$ again. This sequence of actions would not reconstruct the puzzle state since, when forming $l(Str_2, 1)$ we need to further specify that we *embrace* the post segment $P : 3$. In this way, a new "embracing" action would be required so that, when closing a loop, we must specify which long object segments may be embraced by that new loop.

Besides, immediately related to this, we leave for a future work the possibility of knots in our domain. We are also studying the physical feasibility of the effects of these actions by simulating their effects on a 3D graphical model of the puzzle.

**Acknowledgements** Paulo E. Santos acknowledges financial support from FAPESP grant 2012/04089-3, and CNPq "bolsa de produtividade em pesquisa" grant 303331/2011-9. Pedro Cabalar was partially supported by Spanish MEC project TIN2009-14562-C05-04.

[3]We are assuming an ideal string that can be arbitrarily stretched. In the real world, the number of loops may be limited by their size and the string length.

## References


Allen, J. F. 1983. Maintaining knowledge about temporal intervals. *Communications of the ACM* 26(11):832–843.

Cabalar, P., and Santos, P. 2006. Strings and holes: an exercise on spatial reasoning. In *Proc. of SBIA-IBERAMIA*, volume 4140 of *LNAI*, 419–429.

Cabalar, P., and Santos, P. E. 2011. Formalising the fisherman's folly puzzle. *Artif. Intell.* 175(1):346–377.

Casati, R., and Varzi, A. C. 1999. *Parts and Places*. MIT press.

Clementini, E., and Felice, P. D. 1997. A global framework for qualitative shape description. *GeoInformatica* 1(1):11–27.

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

Cohn, A. G.; Bennett, B.; Gooday, J.; and Gotts, N. 1997. Representing and reasoning with qualitative spatial relations about regions. In Stock, O., ed., *Spatial and Temporal Reasoning*. Kluwer Academic Publishers. 97 – 134.

Guesgen, H. 2002a. From the egg-yolk to the scrambled-egg theory. In *Proc. FLAIRS*, 476–480.

Guesgen, H. 2002b. Reasoning about distance based on fuzzy sets. *Applied Intelligence (Special Issue on Spatial and Temporal Reasoning)*.

Hernández, D.; Clementini, E.; and di Felice, P. 1995. Qualitative distances. In Kuhn, W., and Frank, A., eds., *LNAI*. Springer-Verlag. 45–58.

Ligozat, G. 2011. *Qualitative Spatial and Temporal Reasoning*. Wiley.

McCarthy, J., and Hayes, P. 1969. Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence Journal* 4:463–512.

Meathrel, R. C., and Galton, A. P. 2001. A hierarchy of boundary-based shape descriptors. In *Proc. of IJCAI*, 1359–1364.

Pearce, D., and Valverde, A. 2008. Quantified equilibrium logic and foundations for answer set programs. In *24th International Conference on Logic Programming*, volume 5366 of *Lecture Notes in Computer Science*. Springer. 546–560.

Randell, D., and Witkowski, M. 2002. Building large composition tables via axiomatic theories. In *Proc. of KR*, 26–35.

Randell, D.; Cui, Z.; and Cohn, A. 1992. A spatial logic based on regions and connection. In *Proc. of KR*, 165–176.

Randell, D.; Witkowski, M.; and Shanahan, M. 2001. From images to bodies: Modeling and exploiting spatial occlusion and motion parallax. In *Proc. of IJCAI*, 57–63.

Santos, P. E., and Cabalar, P. 2008. The space within fisherman's folly: Playing with a puzzle in mereotopology. *Spatial Cognition & Computation* 8(1-2):47–64.

Santos, P., and Shanahan, M. 2002. Hypothesising object relations from image transitions. In van Harmelen, F., ed., *Proc. of ECAI*, 292–296.

Santos, P., and Shanahan, M. 2003. A logic-based algorithm for image sequence interpretation and anchoring. In *Proc. of IJCAI*, 1408–1410.

Santos, P. 2007. Reasoning about depth and motion from an observer's viewpoint. *Spatial Cognition and Computation* 7(2):133–178.

Schlieder, C. 1996. Qualitative shape representation. In Burrough, P. A., and Frank, A. U., eds., *Geographic Objects with Indeterminate Boundaries*. Taylor & Francis Inc. 123–140.