A Qualitative Representation of a Figure and Construction of Its Planar Class

Kazuko Takahashi, Mizuki Goto and Hiroyoshi Miwa
School of Science&Technology, Kwansei Gakuin University, 2-1, Gakuen, Sanda, 669-1337, Japan
{*ktaka, bub85144, miwa*}*@kwansei.ac.jp*

Keywords: Qualitative Spatial Reasoning, Formalization, PLCA, Planarity.

Abstract: *PLCA* is a framework for qualitative spatial reasoning. It provides a symbolic expression of spatial entities and allows reasoning on this expression. A figure is represented using the objects used to construct it, that is, points, lines, circuits and areas, as well as the relationships between them without numerical data. The figure is identified by the patterns of connection between the objects. For a given PLCA expression, the conditions for planarity, that is, an existence of the corresponding figure on a two-dimensional plane, have been shown; however, the construction of such a PLCA expression has not been discussed. In this paper, we describe a method of constructing such expressions inductively, and prove that the resulting class coincides with that of the planar PCLA. The part of the proof is implemented using a proof assistant Coq.

1 INTRODUCTION

There are many areas in which spatial data are encountered, including image data and video data. As part of image recognition or analysis, we typically extract pertinent aspects of image data, such as dimensions, position, and direction, depending on our purpose. For example, we may focus on the relative positional relationships of the objects, that is, whether the objects are connected and the relative spatial location, as well as changes in these relationships for moving objects. Using this information, we can create an abstract description of moving objects or find a path that describes the trajectory. Qualitative spatial reasoning is a method of representing spatial data by extracting the topological, mereological, or geometric properties without requiring numerical data, which may depend on the application (Stock, 1997; Cohn and Renz, 2007; Ligozat, 2011; Hazarika, 2012). This reflects human cognition and reasoning of common-sense knowledge. Logical expressions are typically adopted in such a representation. Logical expressions for a figure enable us to perform mechanical reasoning on symbols, which reduces the computational complexity. There are various promising practical applications of qualitative spatial reasoning: simulation on Geographic Information System, query-answering system on spatial database, navigation on mobile robots, and so on.

To certify a system on qualitative spatial reasoning, we must prove that an expression correctly represents the properties of the image data and that there is a corresponding image for a given expression. Although there have been lots of works of qualitative spatial reasoning in the field of artificial intelligence (Randell et al, 1992; Egenhofer, 1995; Freksa, 1991; Borgo, 2013), little work has been carried out from the viewpoint of the computational model.

On representation, most research claim expressive power for spatial knowledge but do not refer to the class the expression stands for. We do not know whether a proposed expression is valid or reliable. It is necessary to clarify to what extent the expression is effective, if we implement a system based on the expression. On reasoning, most research focus on consistency check, that is, whether there exists a space that can satisfy all the given relationships among spatial objects, and efficient algorithms for solving this problem (Renz, 2002). However, they do not discuss how to construct such a consistent set.

In this paper, we describe a computational model for a qualitative representation.

Takahashi et al. have proposed a framework for qualitative spatial reasoning, *PLCA*[1] (Takahashi and Sumitomo, 2007; Takahashi, 2012), which focuses on the patterns of connections between regions. This method distinguishes patterns in which regions are

[1]The name of PLCA is originated from an acronym for Point(P), Line(L), Circuit(C) and Area(A).

connected in different ways, for example, by a single point, by two points, by a line and so on. For example, in Figure 1, (a),(b) and (c) are regarded as the same, while (d),(e) and these figures are regarded to be different.

PLCA expressions represent the properties of spatial data by describing the constituent objects, and the relationships between them, without considering attributes such as the size, direction, or shape.

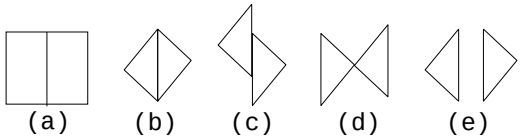


Figure 1: Classification of figures in PLCA. (a)-(c) Regions connected by a line, (d) regions connected by a point, and (e) regions that are not connected.

Takahashi et al. have described the conditions for planarity of a given PLCA expression (Takahashi et al, 2008), that is, an existence of the corresponding figure on a two-dimensional plane, and given a proof for this; however, they have not discussed the construction of such a planar PLCA expression. In this paper, we describe the construction of a planar PLCA expression inductively, and prove that the resulting class coincides with that of the planar PLCA. The part of this proof is implemented using a proof assistant Coq (Bertot and Castŕan, 1998).

The remainder of this paper is organized as follows. In Section 2, we describe a PLCA expression. In Section 3, we describe the inductive construction of a PLCA expression. In Section 4, we prove that the constructed class coincides with that of planar PLCA. In Section 5 we compare our work with the related work, and Section 6 concludes the paper.

2 PLCA

2.1 Target Figure

The target figure of PLCA is considered as a region segmentation of a finite space. In addition, PLCA admits regions with holes, and regards a hole itself to be a region. It does not admit isolated lines or points, because a region cannot be properly defined. Here, we describe a target figure using a simple closed curve (Kosniowski, 1980).

Definition 2.1. *(simple closed curve) A non-self-intersecting continuous loop in a plane is called a* simple closed curve *or a* Jordan curve.

The following is the well-known theorem on Jordan curve.

Theorem 2.1. *Every Jordan curve divides the plane into an interior region bounded by the curve and an exterior region containing all of the nearby and far away exterior points.*

Formally, our target figure is a finite region on a two-dimensional plane, divided into a finite set of subregions of which each boundary is a simple closed curve. In Figure 2, (a) and (b) are target figures, whereas (c) and (d) are not.

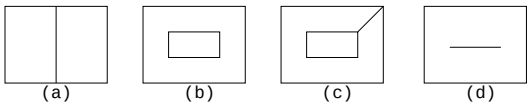


Figure 2: Examples of (a) and (b) target figures, and (c) and (d) non-target figures.

2.2 PLCA Expression

A PLCA expression is defined as a five-tuple, $\langle P, L, C, A, outermost \rangle$, where P is a set of points, $L \subseteq P^2$, $C \subseteq L^n$ $(n \geq 3)$, $A \subseteq C^m$ $(m \geq 1)$, $outermost \in C$.

In PLCA, there are four basic types of object: points P, lines L, circuits C and areas A. An element $l \in L$ is defined as a pair of points p_1 and p_2, and denoted by $l.points = [p_1, p_2]$, where p_1 and p_2 are distinct. Intuitively, a line is an edge between points. No two lines are allowed to cross. A line has an inherent orientation. When $l.points = [p_1, p_2]$, l^+ and l^- mean $[p_1, p_2]$ and $[p_2, p_1]$, respectively. They are called *directed lines*. l^* denotes either l^+ or l^- and l^{*re} denotes the line with the inverse orientation of l^*.

An element $c \in C$ is defined as a list of directed lines and denoted by $c.lines = [l_1^*, \ldots, l_n^*]$, where $l_i^* \neq l_j^*$ if $i \neq j$ $(0 \leq i, j \leq n)$, $l_i^* = [p_{i-1}, p_i] (1 \leq i \leq n)$ and $p_n = p_0$. If $p \in l.points \wedge l^* \in c.lines$, it is said that p *is on* c. A circuit has a cyclic structure, that is, $[l_0^*, \ldots, l_n^*]$ and $[l_j^*, \ldots, l_n^*, l_0^*, \ldots, l_{j-1}^*]$ represent the same circuit for any j $(1 \leq j \leq n)$. Intuitively, a circuit is the boundary between an area and its adjacent areas.

An element $a \in A$ is defined as a set of circuits and denoted by $a.circuits = \{c_0, \ldots, c_n\}$, where any pair of circuits c_i and c_j $(0 \leq i \neq j \leq n)$ cannot share a point. Intuitively, an area is a connected region which consists of exactly one piece.

In addition, *outermost* is a specific circuit in the outermost side of the figure.

Example 2.1. *(PLCA Expression)*

A PLCA expression $\langle P, L, C, A, outermost \rangle$ corresponding to the example target figure shown in Figure 3 is given below.

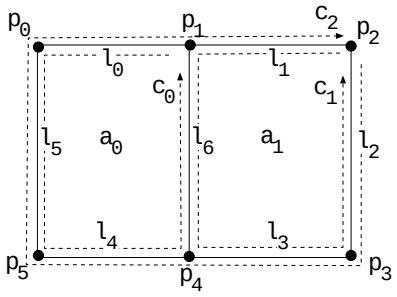


Figure 3: An example of a target figure.

$P = \{p_0, p_1, p_2, p_3, p_4, p_5, p_5\}$
$L = \{l_0, l_1, l_2, l_3, l_4, l_5, l_6\}$
$C = \{c_0, c_1, c_2\}$
$A = \{a_0, a_1\}$
$outermost = c_2$
$l_0.points = [p_0, p_1]$
$l_1.points = [p_1, p_2]$
$l_2.points = [p_2, p_3]$
$l_3.points = [p_3, p_4]$
$l_4.points = [p_4, p_5]$
$l_5.points = [p_5, p_0]$
$l_6.points = [p_1, p_4]$
$c_0.lines = [l_0^-, l_5^-, l_4^-, l_6^-]$
$c_1.lines = [l_1^-, l_6^+, l_3^-, l_2^-]$
$c_2.lines = [l_0^+, l_1^+, l_2^+, l_3^+, l_4^+, l_5^+]$
$a_0.circuits = \{c_0\}$
$a_1.circuits = \{c_1\}$

2.3 Basic Concepts of PLCA Expressions

For $c_1, c_2 \in C$, we introduce two new predicates *lc* and *pc* to indicate that two circuits share line(s) and point(s), respectively.

$$lc(c_1, c_2) \stackrel{\text{def}}{=} \exists l \in L; (l^* \in c_1.lines) \wedge (l^{*re} \in c_2.lines)$$

$$pc(c_1, c_2) \stackrel{\text{def}}{=} \exists p \in P; (p \in l_1.points) \wedge (p \in l_2.points) \wedge (l_1^+ \in c_1.lines) \wedge (l_2^- \in c_2.lines).$$

If $lc(c_1, c_2)$, then either $pc(c_1, c_2)$ or $pc(c_2, c_1)$ holds. For any pair of circuits $c_1, c_2 \in C$, if $c_1, c_2 \in a.circuits$, then $\neg pc(c_1, c_2)$ holds from the definition of Area.

For a circuit c, we define a corresponding circuit-segment.

Definition 2.2. *(circuit-segment) Let* $c.lines = [l_0^*, \ldots, l_n^*]$. *A sequence* $cs = [m_0^*, \ldots, m_k^*]$ $(0 \leq k \leq n)$, *where* $m_i^* = l_{(i+j) \bmod n}^*$ $(0 \leq j \leq n-1)$ *is said to be* a circuit-segment of c, *and denoted by* $cs \sqsubseteq c$.

For a circuit-segment $cs = [m_0^*, \ldots, m_k^*]$, we define its inverse as $inv(cs) = [m_k^{*re}, \ldots, m_0^{*re}]$.

Example 2.2. *(circuit-segments) In Example 2.1,* $[l_0^-, l_5^-]$, $[l_4^-, l_6^-, l_0^-]$, $[l_0^-, l_5^-, l_4^-, l_6^-]$ *are some circuit-segments of* c_0. *Furthermore,* $inv([l_0^-, l_5^-])$ *is* $[l_5^+, l_0^+]$.

For a pair of circuits c_1 and c_2, $S_{scs}(c_1, c_2)$ represents a set of their shared circuit-segments, that is, $S_{scs}(c_1, c_2) = \{cs \mid cs \sqsubseteq c_1, inv(cs) \sqsubseteq c_2\}$. For any $cs \in S_{scs}(c_1, c_2)$, $inv(cs) \in S_{scs}(c_2, c_1)$ holds.

Definition 2.3. *(MSCS) An element* $cs \in S_{scs}(c_1, c_2)$ *is said to be a* maximal shared circuit-segment of c_1 and c_2 *if there does not exist* $cs' \in S_{scs}(c_1, c_2)$ *such that* cs *is a subsequence of* cs'. *A set of maximal shared circuit-segments of* c_1 *and* c_2 *is denoted by* $S_{MSCS}(c_1, c_2)$.

When $c.lines$ is contained in $S_{MSCS}(c_1, c_2)$, c_1 and c_2 are the inner and the outer circuits of a simple closed curve, respectively.

Example 2.3. *(shared circuit-segments) In Figure 4,* $S_{scs}(c_0, c_1) = \{[], [l_0^+], [l_1^+], [l_2^+], [l_3^+], [l_0^+, l_1^+], [l_2^+, l_3^+]\}$. *Furthermore,* $S_{MSCS}(c_0, c_1) = \{[l_0^+, l_1^+], [l_2^+, l_3^+]\}$ *and* $S_{MSCS}(c_1, c_0) = \{[l_1^-, l_0^-], [l_3^-, l_2^-]\}$.

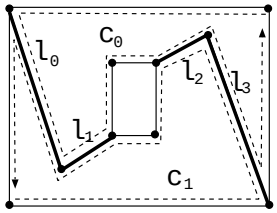


Figure 4: Shared circuit-segments of c_0 and c_1.

Here, we introduce a new type *Path*. An instance *path* of type *Path* is defined as a list of directed lines and used to construct a new circuit. For *path*, $start(path)$, $end(path)$ and $inner\_lines(path)$ show the starting point, ending point and list of directed lines, respectively. The length of $inner\_lines(path)$, which may be 0, is said to be the *length of the path*. Clearly, any circuit-segment is a *Path*.

2.4 Consistency

A consistent PLCA expression does not allow an isolated point or an isolated line, and all of the objects should be correctly defined by the incidence relations. For any point, there exists at least one line that contains it. For any line, there exist exactly two distinct circuits that contain it and its inverse direction, respectively. For any circuit, there exists exactly one area that contains it. The *outermost* is not included in any area. The consistency is formally defined as follows.

Definition 2.4. *(PLCA consistency)*

- **[Consistency of Point-Line]**
  $\forall p \in P(\exists l \in L; p \in l.points)$
  $\forall l \in L(\forall p \in l.points; p \in P)$
- **[Consistency of Line-Circuit]**
  $\forall l \in L(\exists c, c' \in C; l^+ \in c.lines \wedge l^- \in c'.lines)$
  $\forall c \in C(\forall l^* \in c.lines; l \in L)$
  $\forall l \in L(l^* \in c_1.lines, l^* \in c_2.lines \rightarrow c_1 = c_2)$
- **[Consistency of Circuit-Area]**
  $\forall c \in C(\exists a \in A; c \in a.circuits)$
  $\forall a \in A(\forall c \in a.areas; c \in C)$
  $\forall c \in C(c \in a_1.circuits, c \in a_2.circuits \rightarrow a_1 = a_2)$
- **[Independence of outermost]**
  $\neg\exists a \in A; outermost \in a.cuicuit$.

2.5 PLCA-connectedness

Intuitively, PLCA-connectedness guarantees that no objects are separated, including the *outermost*. In other words, for any pair of objects, there exists a trail from one object to the other via further objects.

Definition 2.5. *(d-pcon) Let* $e = \langle P, L, C, A, outermost \rangle$ *be a PLCA expression. For a pair of objects of e, the binary relation d-pcon on* $P \cup L \cup C \cup A$ *is defined as follows.*

1. $d\text{-}pcon(p, l)$ *iff* $p \in l.points$.
2. $d\text{-}pcon(l, c)$ *iff* $l \in c.lines$.
3. $d\text{-}pcon(c, a)$ *iff* $c \in a.circuits$.

Definition 2.6. *(pcon) Let* α, β *and* γ *be objects of a PLCA expression.*

1. *If* $d\text{-}pcon(\alpha, \beta)$, *then* $pcon(\alpha, \beta)$.
2. *If* $pcon(\alpha, \beta)$, *then* $pcon(\beta, \alpha)$.
3. *If* $pcon(\alpha, \beta)$ *and* $pcon(\beta, \gamma)$, *then* $pcon(\alpha, \gamma)$.

Definition 2.7. *(PLCA-connected) A PLCA expression e is said to be PLCA-connected iff* $pcon(\alpha, \beta)$ *holds for any pair of objects* α *and* β *of e.*

2.6 Planar PLCA Expression

Intuitively, PLCA-Euler guarantees that a PLCA expression can be embedded in a two-dimensional plane so that the orientation of each circuit can be correctly defined.

Definition 2.8. *(PLCA-Euler) For a PLCA expression* $\langle P, L, C, A, outermost \rangle$, *if* $|P| - |L| - |C| + 2|A| = 0$, *then it is said to be* PLCA-Euler.

Takahashi et al. have given a proof of the following theorem on the planarity of a PLCA expression (Takahashi et al, 2008).

Theorem 2.2. *For a consistent, connected PLCA expression, it is PLCA-Euler iff there exists a corresponding target figure on a two-dimensional plane.*

Planar PLCA is defined as follows.

Definition 2.9. *(planar PLCA) For a PLCA expression, if it is consistent, PLCA-connected and PLCA-Euler, then it is said to be* planar PLCA[2].

For example, the PLCA expression in Example 2.1 is planar.

The following lemmas hold for a planer PLCA expression, and are used in the subsequent proof for the realizability of an inductively constructed PLCA.

Lemma 2.1. *For a planar PLCA expression, there exists an area that has a single circuit.*

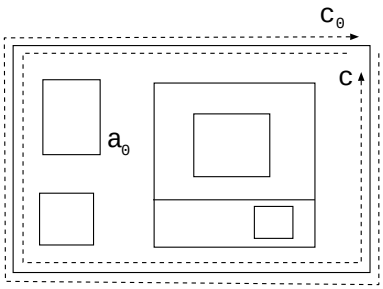
Proof. Let $\langle P, L, C, A, outermost \rangle$ be a planar PLCA expression. Assume that for any area $a \in A$, $|a.circuits| \geq 2$ holds. Set $k = 0$ and c_0 be *outermost*. Take c such that $lc(c_k, c)$ holds. Take an area a_k such that $c \in a_k.circuits$ holds. Let $a_k.circuits$ be $\{c, c_{k_1}, \ldots, c_{k_n}\}$. Note that $\neg pc(c, c_{k_i})$ holds for all i from the definition of Area. Take an arbitrary c_{k_i} ($c_{k_i} \neq c$) and let c_{k+1} be c_{k_i} Increment k and repeat this procedure, then we can take an infinite sequence of circuits $SeqC = c_0, c_1, \ldots$.

Figure 5 illustrates each step of this procedure. Take c_0 as an outermost and c such that $lc(c_0, c)$ holds. Take an area a_0 such that $c \in a_0.circuits$ holds (Figure 5(a)). There are three circuits in $a_0.circuits$ other than c. Take an arbitrary circuit among them and set it as c_1; take c such that $lc(c_1, c)$ holds. Take an area a_1 such that $c \in a_1.circuits$ holds. (Figure 5(b)). There is one circuit in $a_1.circuits$ other than c. Take this circuit and set it as c_2; take c such that $lc(c_2, c)$ holds. Take an area a_2 such that $c \in a_2.circuits$ holds. (Figure 5(c)). We continue this procedure.

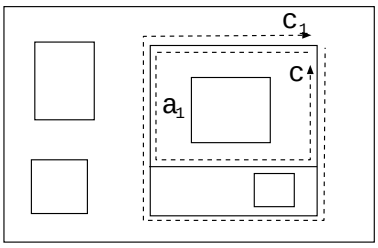
Each circuit is a simple closed curve. $\neg pc(c_i, c_{i+2})$ holds for each i, from Theorem 2.1, since c_i and c_{i+2} are circuits in the exterior region and interior region of c_i, respectively. On the other hand, the number of circuits is finite. Therefore, we cannot take an infinite sequence of circuits *SeqC*. Hence, there exists an area $a \in A$ such that $|a.circuits| = 1$. □

Lemma 2.2. *For any circuit c of a planar PLCA expression, there exists a circuit that has only one maximal shared circuit-segment with c.*

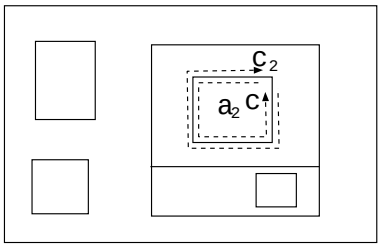
[2] Strictly, the original PLCA admits a curved line, and multiple lines between the same pair of points. If we admit only straight lines, we convert a PLCA expression in the original definition by adding the same number of points and lines, and this conversion does not affect the condition for planarity or the proof thereof.



(a) step1: take a_0



(b) step2: take a_1



(c) step3: take a_2

Figure 5: Existence of an area with a single circuit.

Proof. Let $\langle P, L, C, A, outermost \rangle$ be a planar PLCA, and $c \in C$ be an arbitrary circuit. Assume that for all circuits $c' \in C$, $|S_{MSCS}(c,c')| \neq 1$ holds. For a circuit c' such that $\neg lc(c,c')$ holds, $|S_{MSCS}(c,c')| = 0$ holds. Therefore, we take a circuit c' such that $lc(c,c')$ holds. Let $S_{MSCS}(c,c') = \{cs_1, cs_2\}$. Circuit-segments cs_1 and cs_2 do not share a point. Since cs_1 and cs_2 are considered to be paths, we can take their starting points and ending points: $start(cs_1) = p, end(cs_1) = q$, $start(cs_2) = r, end(cs_2) = s$. Then there exists $cs \sqsubseteq c$ such that $start(cs) = q, end(cs) = r$, and each line in cs is not included in $c'.lines$. Since c' is a circuit, there exists cs'; $cs' \sqsubseteq c'$, $start(cs') = r, end(cs') = q$. On the other hand, from the consistency of Line-Circuit, there exists c_0; $inv(cs) \sqsubseteq c_0$, $start(inv(cs)) = r, end(inv(cs)) = q$. Then, circuit c_0 is defined by appending two circuit-segments $inv(cs')$ and $inv(cs)$. Therefore, $S_{scs}(c,c_0) = \{cs\}$. It follows that $|S_{MSCS}(c,c_0)| = 1$, which is a contradiction (Figure 6).

□

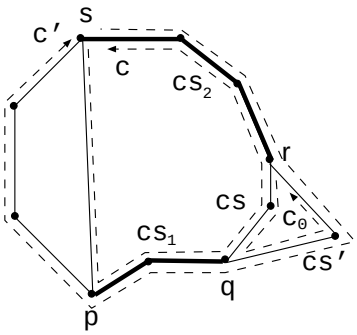


Figure 6: Existence of an area with a single maximal shared circuit-segments. (Relationships of circuit-segments: $cs_1, cs_2, cs \sqsubseteq c$, $inv(cs_1), inv(cs_2), cs' \sqsubseteq c'$ $inv(cs), inv(cs') \sqsubseteq c_0$, $start(cs_1) = p, end(cs_1) = q$, $start(cs_2) = r, end(cs_2) = s$. $start(cs) = q, end(cs) = r$, $start(cs') = r, end(cs') = q$.)

3 CONSTRUCTION OF PLCA

Theorem 2.2 gives the conditions for planarity of a given PLCA expression. The next issue to address is how to construct such an expression.

We can construct a PLCA expression of elements P, L, C and A in this order, for example. In this approach, we must check all of the constraints on the objects carefully during each stage. For example, we must make a circuit so that there exist exactly two distinct circuits: one that contains a line, and one the line in the inverse direction. If this is not satisfied, we must backtrack to construct these lines. This not only requires time, but it is also very difficult to prove that the resulting structure is a planar PLCA expression.

Therefore, we take a different approach, in which we begin with *outermost* and construct a PLCA expression inductively.

We define a class for PLCA expressions using the following three constructors: *single_loop*, *add_loop* and *add_path*. A constructor *single_loop* corresponds to the base case, and the other two correspond to operations that construct a new PLCA expression by dividing an existing area in a current PLCA expression using a path. An arbitrary path, the length of which is more than one is introduced, makes a new circuit using it. Points and lines contained in the path are added simultaneously, and the area is divided into two areas.

We must add objects of four different types simultaneously during an induction step because the objects of a PLCA expression are mutually related. We take the number of areas as a measure of induction, and the number of other objects increases following the application of each constructor. We cannot take the number of points or lines as such a measure, because the expression that is obtained as a result of adding

a single point or a single line to a PLCA expression may not be a PLCA expression.

An alternative method of generating a new area is to add a path to the outer part of the *outermost*. That is, we take two points on the current *outermost* and combine these with a path in the exterior region of *outermost*. In this case, *outermost* changes during each step where a constructor is applied. Because the construction of a new *outermost* is the base case in an inductive definition, we cannot succeed in a proof if we change the definition of *outermost* during each step. Therefore, we do not adopt this method.

We now describe the construction. The idea of construction is based on drawing a figure. Although we demonstrate the construction process on a figure to provide an intuitive discussion, the construction itself is performed symbolically.

A constructor *single_loop* is for a base case, and corresponds to the simplest target figure with one area. There are only two circuits: the outermost circuit and the inner side thereof. Consider an arbitrary path *path*, such that $start(path) = x$, $end(path) = y$, and $inner\_lines(path) = [l_0^+, \ldots, l_n^+]$. Then we create new circuits *outermost* such that $outermost.lines = [l^+, l_0^+, \ldots, l_n^+]$ and c such that $c.lines = [l_n^-, \ldots, l_0^-, l^-]$, where $l.points = [y,x]$ (Figure 7).

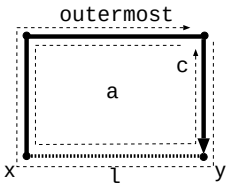


Figure 7: The constructor *single_loop*.

We now define *add_loop*. Consider an arbitrary area a (Figure 8(a)). Take an arbitrary path *path*, such that $start(path) = x$, $end(path) = y$ and $inner\_lines(path) = [l_0^+, \ldots, l_n^+]$. Make a line l such that $l.points = [y,x]$ (Figure 8(b)). Then make new circuits c_1 and c_2 such that $c_1.lines = [l^+, l_0^+, \ldots, l_n^+]$, and $c_2.lines = [l_n^-, \ldots, l_0^-, l^-]$. Add c_1 to $a_1.circuits$ and c_2 to $a_2.circuits$ (Figure 8(c)). As a result, a is divided into two areas, a_1 and a_2 (the hatched part). The points and lines contained in *path* are added accordingly. If a contains more than one circuit, all of them remain in a_1, and a_2 contains none.

Now we define *add_path*. Consider a circuit c such that $c \in a.circuits$, and two points y,z on c. Here y and z may be identical. Because a circuit-segment is a path, consider a circuit-segment $cs \sqsubseteq c$ such that $start(cs) = y$, $end(cs) = z$. Then c is divided into two circuit-segments: cs and cs'. Let $c.lines = [ll_0^+ \ldots, ll_m^+]$, $cs = [ll_0^+ \ldots, ll_k^+]$ $(0 \leq k \leq m)$ and $cs' = [ll_{k+1}^+ \ldots, ll_m^+]$ (Figure 9(a)). Take an arbitrary path *path*, such that $start(path) = s$, $end(path) = e$ and $inner\_lines(path) = [l_0^+, \ldots, l_n^+]$. Make lines l_s and l_e such that $l_s.points = [s,y]$ and $l_e.points = [z,e]$, respectively (Figure 9(b)). Then make new circuits c_1 and c_2 such that $c_1.lines = [l_s^-, l_0^+, \ldots, l_n^+, l_e^-, ll_{k+1}^+ \ldots, ll_m^+]$ and $c_2.lines = [l_e^+, l_n^-, \ldots, l_0^-, l_s^+, ll_0^+ \ldots, ll_k^+]$. Add c_1 to $a_1.circuits$ and add c_2 to $a_2.circuits$ (Figure 8(c)). As a result, a is divided into two areas, a_1 and a_2 (the hatched part), c is eliminated, and two new circuits are created. The points and lines contained in *path* are added and the objects are changed. If a contains circuits other than c, all of them remain in a_1, and a_2 contains none.

Note that *add_loop* is applied to a specific area, whereas *add_path* is applied to a specific circuit and two points on it.

Definition 3.1. *(IPLCA) PLCA expressions constructed by the above three constructors are said to be* Inductive PLCA (IPLCA).

4 PROOF OF FORMALIZATION

Here we prove that IPLCA coincides with planar PLCA.

4.1 Proof of Planarity

We first prove that IPLCA is planar. From Theorem 2.2, we prove the following theorem.

Theorem 4.1. *(planarity for IPLCA) If e is IPLCA, e is (i) consistent, (ii) PLCA-connected, and (iii) PLCA-Euler.*

We implement IPLCA and prove these three properties using the proof assistant Coq (Bertot and Castŕan, 1998). Coq is based on typed logic adopted for higher-order functions. The data types and functions are defined in recursive form, and the proof proceeds by connecting suitable tactics. The definition of IPLCA and the proof of Theorem 4.1 required approximately 5500 lines of code in total The advantage of using Coq is to certify the correctness of the formalization. We do not show the detail of the proof here, since it is out of the focus of this paper. The entire code is shown in (Goto, 2014).

4.2 Proof of Realizability

We prove that a planar PLCA is IPLCA. This means that any target figure can be drawn by applying the

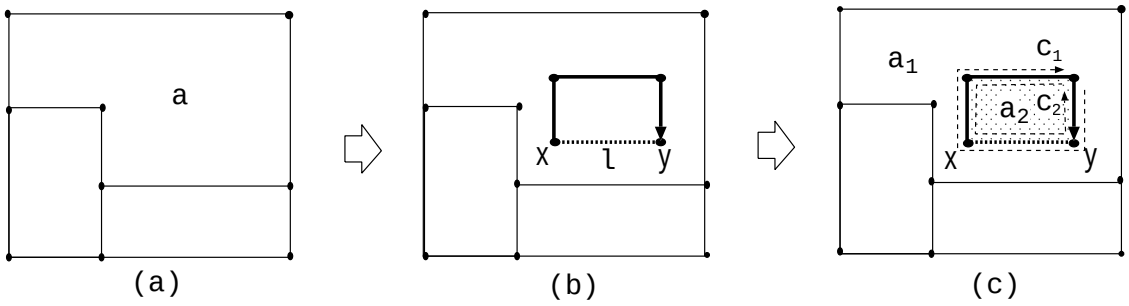


Figure 8: The constructor *add_loop*.

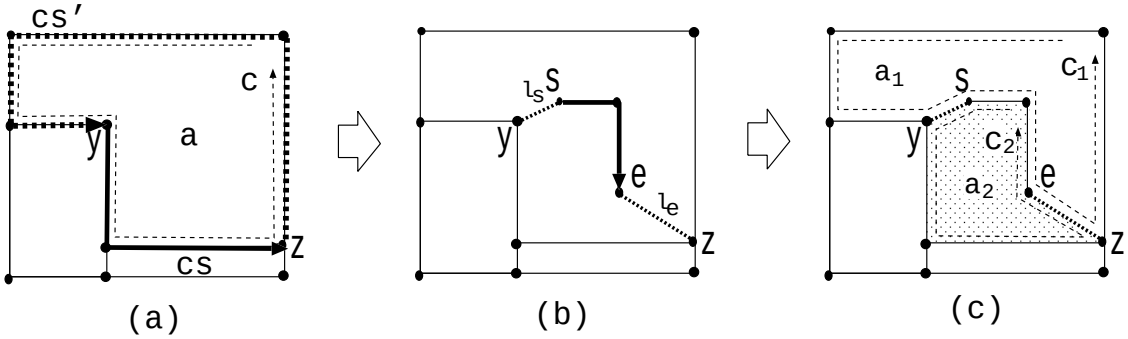


Figure 9: The constructor *add_path*.

constructors of IPLCA in a suitable order. For example, consider Figure 10. If we apply *add_loop* first, we cannot successively apply constructors, because any intermediate figure is not the target figure (Figure 10(a)). However, if we apply *add_path* first, we can successively add areas by applying *add_path* again (Figure 10(b)).

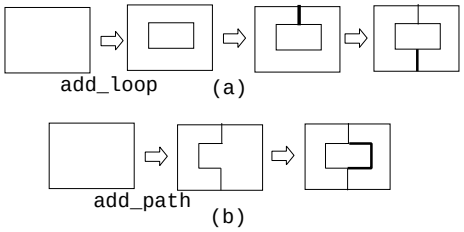


Figure 10: Constructing figures (a) by first applying *add_loop*, and (b) by first applying *add_path*.

Theorem 4.2. *(Realizability for IPLCA) A planar PLCA is IPLCA.*

Proof. Let F be a target figure. We prove the theorem using induction on the number of areas of F.

(Base case) The number of areas is 1.

F consists of only a simple closed curve. This is clearly a base case of IPLCA, and is constructed by applying *single_loop*.

(Induction step) The number of areas is $n+1$.

The principle of our proof via induction is as follows. For a planar PLCA e, of which the number of areas is $n+1$, we remove a suitable area a such that we can form a planar PLCA e', where the number of areas is n. Because e' is IPLCA from the induction hypothesis, we can apply *add_loop* or *add_path* to obtain e. We proceed the proof based on this principle.

We can take an area a with a single circuit c from Lemma 2.1. Then, there exists c' such that $|S_{MSCS}(c,c')| = 1$, from Lemma 2.2. Assume that $c' = outermost$. Since the number of areas is more than one, a contains more than one circuit, which is a contradiction. Therefore, $c' \neq outermost$.

Case 1. $S_{MSCS}(c,c') = \{c.lines\}$.

In this case, we remove a, c, c', and all objects on c and c' to obtain a planar PLCA e' such that $|e'.areas| = n$. Note that since $c' \neq outermost$, e' has an *outermost*. Here e' is IPLCA from the induction hypothesis. Then we can construct e by applying the constructor *add_loop* on a' (Figure 11).

Case 2. $S_{MSCS}(c,c') \neq \{c.lines\}$.

Let $S_{MSCS}(c,c') = \{cs\}$. In this case, c is divided into two circuit-segments cs and cs_1, and c' is divided into two circuit-segments $inv(cs)$ and cs_2 (Figure 12). We remove a, c, c', and all objects on c and c', and add a circuit *newC* by appending cs_1 and cs_2. We obtain a planar PLCA expression e' such that $|e'.areas| = n$.

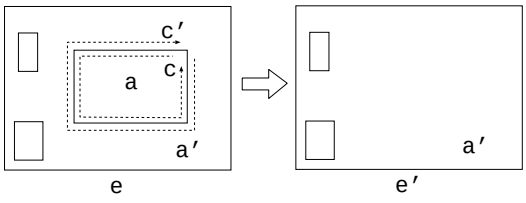


Figure 11: Removing an area with case 1.

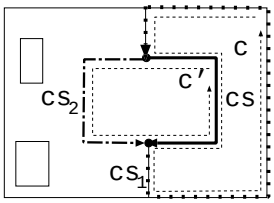


Figure 12: Circuit-segments in case 2. Circuit c is divided into cs and cs_1, and circuit c' is divided into $inv(cs)$ and cs_2.

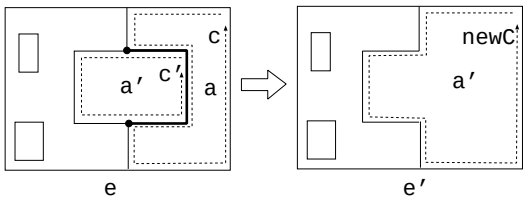


Figure 13: Removing an area with case 2.

e' is IPLCA from the induction hypothesis. Then we can construct e by applying the constructor *add_path* on *newC*, $start(cs_1)$ and $end(cs_1)$ (Figure 13).

□

5 RELATED WORK

There exist several symbolic expressions other than qualitative spatial representations for a figure on a two-dimensional plane, including computational geometry (de Berg et al, 1997) and graph theory (Harary, 1969). Different from qualitative spatial representations, the main objective of computational geometry is to analyze the complexity of algorithms for problems expressed in terms of geometry and to develop efficient ones, rather than to recognize or to analyze the characteristics of a figure. Graph theory can be used to provide symbolic expressions of spatial data. The topological structure of spatial data can be represented as a graph by treating spatial objects, such as points and lines, as nodes and the relationships between them as edges. There exists a condition to determine the planarity of a given graph; however, in general, a graph does not contain any information on an area, and therefore we only know that we can embed a graph by locating areas properly. In contrast, a PLCA expression places constraints on the locations of areas. In this respect, a PLCA expression is more specific than a graph.

One of the challenges for symbolic expressions of a figure on a two-dimensional plane is the concept of a *hypermap*. A hypermap is an algebraic structure that represents objects and relationships between them, and can be used to distinguish the topological and geometric aspects. There are several works that use a hypermap and give a formalization and a proof of the properties of these aspects using proof assistants. Gonthier et al. formalized and proved the four-color theorem and showed a proof (Gonthier, 2008). In this work, planar subdivisions are described by a hypermap. Dufourd applied a hypermap to formalize and to prove a Jordan curve theorem (Dufourd, 2009). He also showed a treatment of surface subdivision and planarity based on a hypermap (Dufourd, 2010). Brun et al. showed a derivation of a program to compute a convex-hull for a given set of points from their specification using a hypermap (Brun et al, 2012). They specified the algorithm and proved its correctness using a structural induction. Hypermap is a strong method for providing a mechanical proof of the topological or geometric properties in a symbolic form; however, the representation is too complicated to understand intuitively.

6 CONCLUSION

We have described a method of constructing a PLCA expression inductively, and have proved that the defined class coincides with that of planar PCLA. Formalization and part of the proof was implemented using the proof assistant Coq. Our main contribution is giving a computational model to a qualitative spatial representation, which is the first attempt in the research field on qualitative spatial reasoning.

Mechanical proof using a proof assistant provides a rigorous proof of correctness of the formalization. In future, we will complete the mechanical proof of the part currently done manually.

ACKNOWLEDGEMENTS

This work is supported by JSPS KAKENHI Grant Number 25330274.

REFERENCES

de Berg, M., M. van Kreveld, M. Overmars and O. Schwarzkopf (1997). *Computational Geometry*. Springer-Verlag.

Bertot, Y. and P. Castŕan (1998). *Interactive Theorem Proving and Program Development - Coq'Art: The Calculus of Inductive Constructions*. Springer Verlag.

Borgo, S. (2013). RCC and the theory of simple regions R^2. *Conference On Spatial Information Theory (COSIT13)*, pp.457-474.

Brun, C., J. -F. Dufourd and N. Magaud (2012). Designing and proving correct a convex hull algorithm with hypermaps in Coq. *Computational Geometry : Theory and Applications*, 45(8):436-457.

Cohn, A. G. and J. Renz (2007). Qualitative spatial reasoning. in *Handbook of Knowledge Representation*. F. Harmelen, V. Lifschitz and B. Porter(eds.), Chapt 13, pp.551-596, Elsevier.

Dufourd, J. -F. (2009). An intuitionistic proof of a discrete form of the Jordan curve theorem formalized in Coq with combinatorial hypermaps. *Journal of Automated Reasoning*, 43(1):19-51.

Dufourd, J. -F. and Y. Bertot (2010). Formal study of plane Delaunay triangulation. *Interactive Theorem Proving*, pp.211-226, LNCS 6172, Springer-Verlag.

Egenhofer, M. and J. Herring (1995). Categorizing binary topological relations between regions, lines, and points in geographic databases. *Department of Surveying Engineering*, University of Maine.

Freksa, C. (1991). Conceptual neighborhood and its role in temporal and spatial reasoning. *Proceedings of the IMACS Workshop on Decision Support Systems and Qualitative Reasoning*, pp.181-187.

Gonthier, G. (2008). Formal proof - The four color theorem. *Notices of the AMS*, 55(11):1382-1393.

Goto, M. and K. Takahashi (2014). http://ist.ksc.kwansei.ac.jp/∼ktaka/IPLCA/

Harary, F. (1969). *Graph Theory*. Reading, MA, Addison-Wesley.

Hazarika, S. (2012). *Qualitative Spatio-Temporal Representation and Reasoning: Trends and Future Directions*. IGI Publishers.

Kosniowski, C. (1980). *A First Course in Algebraic Topology*. Cambridge University Press.

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

Randell, D. A., Z. Cui and A. G. Cohn (1992). A spatial logic based on regions and connection. *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning (KR92)*, pp.165-176.

Renz, J. (2002). *Qualitative Spatial Reasoning with Topological Information*. LNAI 2293, Springer-Verlag.

Stock, O. (Ed.) (1997). *Spatial and Temporal Reasoning*, Kluwer Academic Publishers.

Takahashi, K. (2012). PLCA: A framework for qualitative spatial reasoning based on connection patterns of regions. in (Hazarika, 2012), Chapt 2, pp.63-96.

Takahashi, K. and T. Sumitomo (2007). The qualitative treatment of spatial data. *International Journal on Artificial Intelligent Tools*,16(4):661-682.

Takahashi, K., T. Sumitomo and I. Takeuti (2008). On embedding a qualitative representation in a two-dimensional plane. *Spatial Cognition and Computation*, 8(1-2):4-26.