Drawing a Figure in a Two-Dimensional Plane for a Qualitative Representation

Shou Kumokawa and Kazuko Takahashi

School of Science & Technology, Kwansei Gakuin University,
2-1, Gakuen, Sanda, 669-1337, Japan
acy85499@ksc.kwansei.ac.jp, ktaka@kwansei.ac.jp

Abstract. This paper describes an algorithm for generating a figure in a two-dimensional plane from a qualitative spatial representation of PLCA. In general, it is difficult to generate a figure from qualitative spatial representations, since they contain positional relationships but do not hold quantitative information such as position and size. Therefore, an algorithm is required to determine the coordinates of the objects while preserving the positional relationships. Moreover, it is more desirable that the resulting figure meets a user's requirement. PLCA is a simple symbolic representation consisting of points, lines, circuits and areas. We have already proposed one algorithm for drawing, but the resulting figures are far from a "good" one. In that algorithm, we generate the graph corresponding to a given PLCA expression, decompose it into connected subgraphs, determine the coordinates in a unit circle for each subgraph independently, and finally determine the position and size of each subgraph by locating the circles in appropriate positions. This paper aims at generating a "good" figure for a PLCA expression. We use a genetic algorithm to determine the locations and the sizes of circles in the last step of the algorithm. We have succeeded in producing a figure in which objects are drawn as large as possible, with complex parts larger than others. This problem is considered to be a type of "circle packing," and the method proposed here is applicable to the other problems in which locating objects in a non-convex polygon.

1 Introduction

Qualitative Spatial Reasoning (QSR) is a method that treats images or figures qualitatively, by extracting the information necessary for a user's purpose [4,14,15]. It also offers methods to handle and reason about unspecific information. Numerous applications use databases of images or figures including Geographic Information Systems (GIS) and navigation systems. In these applications, responding to queries or frequently updating the data requires a large amount of computation. QSR is a promising method that reduces memory and the workspace required for computations that do not involve strict data. In general, it is easy to transform a figure to a symbolic qualitative representation, but it is difficult to generate a figure from a symbolic qualitative representation. The automatic drawing of figures such as Venn diagram from mereological relationships has been studied in the context of diagrammatic reasoning [1]. However, to the best of our knowledge, no system exists for automatically drawing a figure from mereotopological relationships. Symbolic representation in compact form lends itself to calculation or storage, but drawings are much beter for visualizing the abstract details and context. It is also interesting to view what shape of a figure is generated from a symbolic representation. Drawing a figure from a symbolic representation is applicable to an important service such as schematic maps.

The difficulty in drawing a figure from a symbolic representation arises from two factors. First, one has to judge whether the expression can be embedded in a two-dimensional plane. Second, one must determine the position of each object in the drawing. The latter is necessary since multiple quantitative representations may exist for a single qualitative expression. Therefore, we cannot determine a unique set of appropriate coordinates. Even if a set of coordinates are determined, the resulting complex figure may not support a user to think or to design.

In this paper, we discuss drawing a figure from a PLCA expression, a qualitative representation method. A PLCA expression represents spatial data by focusing on connected patterns of objects, using four simple elements: $point(P), line(L)$ $circuit(C)$ and $area(A)$ [18,19,20]. In PLCA, no pair of areas has a part in common[1]. The entire space is covered with the areas.

A PLCA expression is considered to be a set of mereotopological relationships between objects including areas. Our goal is to draw a figure for such a representation. This is different from other studies on visualization in which the goal is to generate a figure for a set of conceptual relationships to support human cognition and understanding.

In the previous paper, we proved that realizability for a PLCA expression in a two-dimensional plane is reduced to the planarity of a graph and identified the condition for realizability [21]. We also proposed an algorithm for drawing a figure for a PLCA expression that satisfied the realizability condition in a two-dimensional plane. In that algorithm, we decomposed a PLCA expression into several subexpressions each of which corresponds to a connected graph, determined the position of the objects in a unit circle for each graph independently and combined the related parts of them. However, the resulting figure was far from a "good" one. For example, many objects were drawn in a corner, leaving a large vacant space in the center. The main problem was embedding circles in part of another circle in the last step when related parts were combined. In that last step, the challenge was to determine the location and the size of each circle to produce a "good" figure while preserving the relationships of objects described in the PLCA expression.

The problem of embedding is reduced to one of circle packing, putting n circles of different sizes into a non-convex polygon so that they do not intersect. Circle packing is a well-known optimization problem that is NP-complete in general, and many studies have been undertaken [22,16,17]. However, no algorithm has been proposed that covers the conditions in our problem. In this paper, we address the problem using a genetic algorithm (GA) to determine the location and the size of each circle and produce an approximate solution to optimization, that gives a "good" figure. A "good" figure here means one in which the objects are drawn as large as possible, and a complex parts are drawn larger than the other parts.

[1] We use the term $area$ instead of $region$, since $area$ used in this paper is a different entity from the $region$ generally used in qualitative spatial reasoning.

We also discuss another drawing algorithm that does not use a graph-drawing algorithm but generates a figure directly from a PLCA expression. Moreover, we discuss generation of figures from other qualitative representations.

This paper is organized as follows. In section 2, we briefly describe PLCA expressions, and the conditions for their realizability in a two-dimensional plane. In section 3, we introduce the drawing algorithm, and show the results of experiment. In section 4, we present another drawing algorithm for PLCA and discuss drawing for other qualitative representations. And finally, in section 5, we present our conclusions.

2 PLCA Expressions

2.1 Definition of Classes

PLCA has four basic components: $points(P)$, $lines(L)$, $circuits(C)$ and $areas(A)$.

Point is defined as a primitive class.

Line is defined as a class that satisfies the following condition: for an arbitrary instance l of $Line$, $l.points$ is a pair $[p_1, p_2]$ where $p_1, p_2 \in Point$. 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. l^* denotes either l^+ or l^-. Intuitively, a line is the edge connecting two (not always different) points. No two lines are allowed to cross. Note that multiple lines may have the same pair of points. In Fig. 1(a), the arrows denote the orientation of the lines. All of the lines $l_1.points, l_2.points$ and $l_3.points$ are defined to be $[p_1, p_2]$, but they are distinguished by the circuits to which they belong.

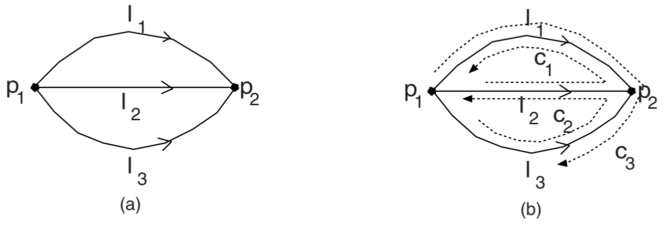


Fig. 1. Multiple lines with the same definition and the associated circuits

Circuit is defined as a class that satisfies the following condition: for an arbitrary instance c of $Circuit$, $c.lines$ is a sequence $[l_1^*, \ldots, l_n^*]$ where $l_1, \ldots, l_n \in Line(n \geq 1)$, $l_i.points = [p_i, p_{i+1}](1 \leq i \leq n)$ and $p_{n+1} = p_1$. $[l_1^*, \ldots, l_n^*]$ and $[l_j^*, \ldots, l_n^*, l_1^*, \ldots, l_{j-1}^*]$ denote the same circuit for any j $(1 \leq j \leq n)$. In Fig. 1(b), we have three circuits: $c_1.lines = \{l_1^-, l_2^+\}$, $c_2.lines = \{l_2^-, l_3^+\}$, $c_3.lines = \{l_3^-, l_1^+\}$.

For $c_1, c_2 \in Circuit$, we introduce two new predicates lc and pc to denote that two circuits share line(s) and point(s), respectively. $lc(c_1, c_2)$ is $true$ iff there exists $l \in Line$ such that $(l^+ \in c_1.lines) \wedge (l^- \in c_2.lines)$. $pc(c_1, c_2)$ is $true$ iff there exists $p \in Point$ such that $(p \in l_1.points) \wedge (p \in l_2.points) \wedge$ $(l_1^* \in c_1.lines) \wedge (l_2^* \in c_2.lines)$. A circuit is the boundary between an area and its adjacent areas viewed from the side of that area.

Area is defined as a class that satisfies the following condition: for an arbitrary instance a of $Area$, $a.circuits$ is a set $\{c_1, \ldots, c_n\}$ where $c_1, \ldots, c_n \in Circuit (n \geq 1)$, and $\forall c_i, c_j \in a.circuits; (i \neq j) \rightarrow (\neg pc(c_i, c_j) \wedge \neg lc(c_i, c_j))$. Intuitively, an area is a connected region which consists of exactly one piece. No two areas are allowed to cross. The final condition means that any pair of circuits that belong to the same area cannot share a point or a line.

The *PLCA expression* e is defined as a five tuple $e = \langle P, L, C, A, outermost\rangle$ where P, L, C and A are a set of points, lines, circuits and areas, respectively, and $outermost \in C$. An element of $P \cup L \cup C \cup A$ is called *a component of* e.

We assume that there exists a circuit in the outermost extremity of the figure called $outermost$. This means that the target figure is drawn in a finite space, and the space can be divided into a number of areas that do not overlap with each other.

In Fig. 2, (a) shows an example of a target figure, and (b) and (c) show the names of the components. Example 1 shows a PLCA expression corresponding to Fig. 2.

Definition 1. *(consistency) A PLCA expression $e = \langle P, L, C, A, outermost\rangle$, is said to be consistent iff the following three constraints are satisfied:*

1. **constraint on P-L:** *For any $p \in Point$ there exists at least one line l such that $p \in l.points$.*
2. **constraint on L-C:** *For any $l \in Line$, there exist exactly two distinct circuits c_1, c_2 such that $l^+ \in c_1.lines, l^- \in c_2.lines$.*
3. **constraint on C-A:** *For any $c \in Circuit$ other than $outermost$, there exists exactly one area a such that $c \in a.circuits$. The $outermost$ is not included in any area.*

Due to these constraints, neither isolated lines nor points are allowed.

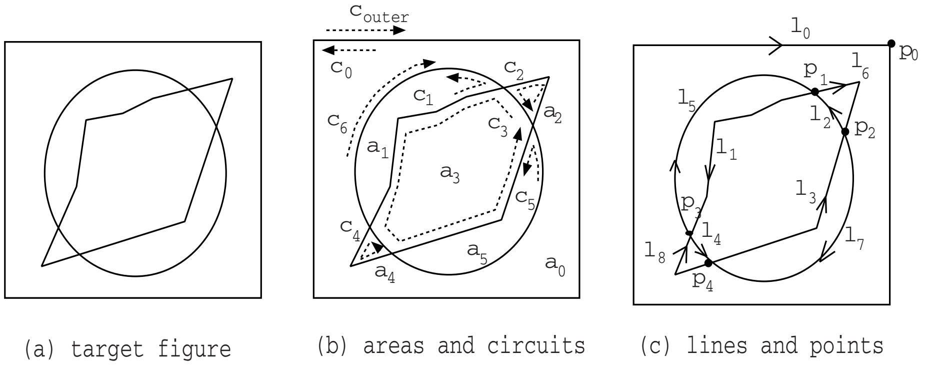


Fig. 2. Example of a target figure

Example 1

$e.points = \{p_0, p_1, p_2, p_3, p_4\}$	$c_{outer}.lines = [l_0^+]$
$e.lines = \{l_0, l_1, l_2, l_3, l_4, l_5, l_6, l_7, l_8\}$	$c_0.lines = [l_0^-]$
$e.circuits = \{c_{outer}, c_0, c_1, c_2, c_3, c_4, c_5, c_6\}$	$c_1.lines = [l_1^-, l_5^-]$
$e.areas = \{a_0, a_1, a_2, a_3, a_4, a_5\}$	$c_2.lines = [l_2^-, l_6^-]$

$e.outermost = c_{outer}$	$c_3.lines = [l_1^+, l_2^+, l_3^+, l_4^+]$
$l_0.points = [p_0, p_0]$	$c_4.lines = [l_4^-, l_8^-]$
$l_1.points = [p_4, p_1]$	$c_5.lines = [l_3^-, l_7^-]$
$l_2.points = [p_1, p_2]$	$c_6.lines = [l_5^+, l_8^+, l_7^+, l_6^+]$
$l_3.points = [p_2, p_3]$	$a_0.circuits = \{c_6, c_0\}$
$l_4.points = [p_3, p_4]$	$a_1.circuits = \{c_1\}$
$l_5.points = [p_1, p_4]$	$a_2.circuits = \{c_2\}$
$l_6.points = [p_2, p_1]$	$a_3.circuits = \{c_3\}$
$l_7.points = [p_3, p_2]$	$a_4.circuits = \{c_4\}$
$l_8.points = [p_4, p_3]$	$a_5.circuits = \{c_5\}$

2.2 Two-Dimensional Realizability

We introduce the concept of connectedness for the components of a PLCA expression.

Definition 2. *(d-pcon) Let* $e = \langle P, L, C, A, outermost\rangle$ *be a PLCA expression. For a pair of components of e, the predicate d-pcon is defined as follows.*

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

Definition 3. *(pcon) Let* α, β, γ *be components of a PLCA expression. pcon is the symmetric and transitive closure of d-pcon.*

1. *If d-pcon*(α, β)*, then pcon*(α, β).
2. *If pcon*(α, β)*, then pcon*(β, α).
3. *If pcon*(α, β) *and pcon*(β, γ)*, then pcon*(α, γ).

Definition 4. *(PLCA-connected) A PLCA expression* e *is said to be PLCA-connected iff pcon*(α, β) *holds for any pair* α *and* β *of components of* e.

Intuitively, PLCA-connectedness guarantees that all the components including the $outermost$ are connected. That is, for any pair of components, there is a trail that can go from one component to the other by tracing components. The PLCA expression in Example 1 is consistent and PLCA-connected. For example, $pcon(c_{outer}, c_6)$ holds since we can move from c_{outer} to c_6 by tracing the components $c_{outer}, l_0, c_0, a_0, c_6$ in that order. On the other hand, the PLCA expression in Example 2 is consistent but not PLCA-connected. It can be divided into two subexpressions: one corresponding to the plane consisting of p_0, l_0, c_{outer}, c_0 and a_0, and the other corresponding to a floating group of other the components. A component of the former is not $pcon$ with that of the latter. For example, $pcon(c_{outer}, c_1)$ does not hold.

Example 2
$e.points = \{p_0, p_1\}$ $c_{outer}.lines = \{l_0^+\}$
$e.lines = \{l_0, l_1, l_2\}$ $c_0.lines = \{l_0^-\}$

$$\begin{array}{ll} e.circuits = \{c_{outer}, c_0, c_1, c_2, c_3\} & c_1.lines = \{l_1^+, l_2^+\} \\ e.areas = \{a_0, a_1, a_2\} & c_2.lines = \{l_2^-\} \\ l_0.points = [p_0, p_0] & c_3.lines = \{l_1^-\} \\ l_1.points = [p_1, p_1] & a_0.circuits = \{c_0\} \\ l_2.points = [p_1, p_1] & a_1.circuits = \{c_1\} \\ & a_2.circuits = \{c_2, c_3\} \end{array}$$

For a consistent connected PLCA expression, the following theorem holds [21].

Theorem 1. *For a consistent connected PLCA expression $e = \langle P, L, C, A, outermost\rangle$, e can be realized in a two-dimensional plane iff $|P| - |L| - |C| + 2|A| = 0$ holds.*

This theorem shows that the two-dimensional realizability for a PLCA expression is judged only by counting the number of the components.

Definition 5. *(planar PLCA expression) A consistent connected PLCA expression that satisfies $|P| - |L| - |C| + 2|A| = 0$ is said to be planar.*

The PLCA expression shown in Example 1 is planar but the expression in Example 2 is not.

2.3 Orientation of a Circuit

As a preparation, we introduce several concepts from graph theory.

A (non-directed) graph is defined to be $G = (V, E)$, where V is a set of vertices and E is a set of edges. An edge of E is defined as a pair of vertices of V. For graphs $G = (V, E)$ and $G' = (V', E')$, if $V' \subset V$ and $E' \subset E$, G' is said to be *a subgraph* of G; if $V \cap V' = \emptyset$ and $E \cap E' = \emptyset$, it is said that G and G' are *disjoint*. Here, when we consider more than one subgraph of G, we assume that they are disjoint. If it is possible to move between any pair of vertices by moving along the edges of the graph, the graph is said to be *connected*; otherwise, it is said to be *disconnected*. A sequence $(v_0, \dots, v_n)$ where (v_i, v_{i+1}) for each i $(0 \leq i \leq n-1)$ is an edge and $v_0 = v_n$, it is said to be *a cycle*. A cycle that is a border of both the graph and the outer infinitely large region is said to be *an outer boundary cycle of g*.

Let $e = \langle P, L, C, A, outermost\rangle$ be a consistent PLCA expression. We can define a non-directed graph $m(e) = (V, E)$ by relating P and L to V and E, respectively. For $p \in P$, $m(p)$ indicates the corresponding vertex, and for $l \in L$, $m(l)$ indicates the corresponding edge. We extend m so that c is mapped to $m(c)$. For each $l_i (i = 0, \dots, n), l_i^* \in c.lines$, if $m(l_i)$ is contained in a graph g, then we say that $m(c)$ *is contained in* g.

Proposition 1. *Let $e = \langle P, L, C, A, outermost\rangle$ be a consistent connected PLCA expression that satisfies $|a.circuits| = 1$ for any area $a \in A$. Then $m(e)$ is a connected graph [21].*

Each circuit of a planar PLCA expression e has an orientation of $inner$ or $outer$. If $m(e)$ is a disconnected graph, then it can be decomposed into connected subgraphs. We determine the orientation of each circuit by considering the relationships among these subgraphs, areas and circuits of e.

[**Algorithm: DCO(determine circuit's orientation)**]

```
1. Make a node $Noutermost$.
2. $setOuterOrientation(outermost, Noutermost)$.
```

Procedure. $setOuterOrientation(c, N_c)$

```
1. Set the orientation of $c$ to be $outer$.
2. For the subgraph $g$ such that $m(c)$ is contained in $g$, make a node $N_g$ and draw an
   edge from $N_c$ to $N_g$.
3. For each $m(c')$ contained in $g$ such that $c' \neq c$, do the following:
   (a) Make a node $N_{c'}$ and draw an edge from $N_g$ to $N_{c'}$ .
   (b) $setInnerOrientation(c', N_{c'})$.
```

Procedure. $setInnerOrientation(c', N_{c'})$

```
1. Set the orientation of $c'$ to be $inner$.
2. For an area $a$ such that $c' \in a.circuits$, make a node $N_a$ and draw an edge from
   $N_{c'}$ to $N_a$.
3. For each $c'' \in a.circuits$ such that $c'' \neq c'$, do the following:
   (a) Make a node $N_{c''}$ and draw an edge from $N_a$ to $N_{c''}$.
   (b) $setOuterOrientation(c'', N_{c''})$.
```

A diagram constructed in this way is called a *DCO diagram*. Each path in the diagram is a sequence of a pattern $N_{c_1} \rightarrow N_g \rightarrow N_{c_2} \rightarrow N_a$ where c_1, c_2 are circuits, a is an area of e, and g is a subgraph of $m(e)$. Fig. 3 is a part of the DCO diagram for Example 1.

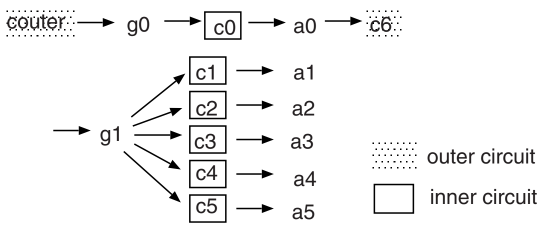


Fig. 3. A part of the DCO diagram for Example 1

Proposition 2. *For a planar PLCA expression* e*, (i) the orientation of each circuit is decidable, (ii) there exists the unique inner circuit in* $a.circuit$ *for each area* a*, and (iii) there exists an outer circuit* c *such that* $m(c)$ *is contained in* g *is an outer boundary cycle of* g *for each subgraph* g *[21].*

3 Drawing a PLCA Expression

3.1 Drawing Algorithm

We describe the outline of an algorithm for drawing a figure from a planar PLCA expression in a two-dimensional plane (Fig. 4).

1. Extract the information of points and lines from the planar PLCA to get the corresponding graph expression.
2. Decompose the graph into disjoint connected subgraphs, and determine the coordinates of nodes and edges in a unit circle for each subgraph independently. We utilize an existing graph-drawing algorithm using straight lines in this step [12].
3. Determine the location and the size of these subgraphs using the information on circuits and areas in the PLCA expression.

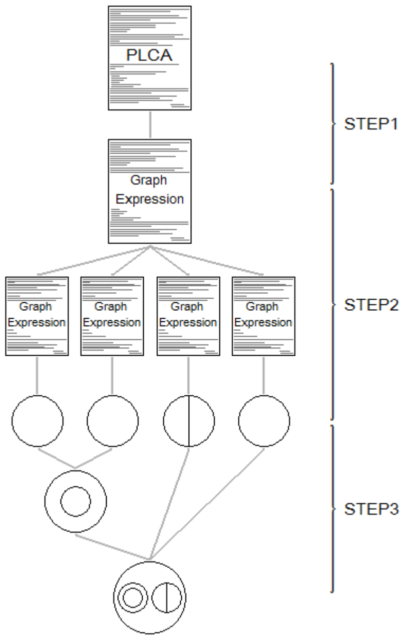


Fig. 4. A drawing process

Definition 6. *(module) Let e be a planar PLCA expression and α be either a point, a line or a circuit of e. If $m(e)$ is decomposed into n disjoint connected subgraphs $g_1, \ldots, g_n$, then we say that e has n modules. If $m(\alpha)$ is contained in g_i, then α is contained in the module corresponding to g_i. For an area a of e, a is contained in the module that contains the inner circuit c in $a.circuits$.*

Note that each component of the PLCA is contained only in one module.

Definition 7. *(e-circle) A unit circle in which each module of the PLCA is embedded in the second step of the algorithm is called an e-circle.*

Definition 8. *(bridge) An area a such that $|a.circuit| \geq 2$ holds is said to be a bridge.*

The third step of the algorithm is the most important since the location and the size of e-circles in a bridge are determined.

For each bridge $a \in A$, do the following: let $a.circuits = \{c_0, c_1, \ldots, c_n\}$, where the orientation of c_0 is $inner$ and those of $c_1, \ldots, c_n$ are $outer$. Let $g_1, \ldots, g_n$ be the subgraphs whose e-circles are $ec_1, \ldots, ec_n$, respectively. For each ec_i $(i = 1, \ldots, n)$, expand or reduce it and draw it in an appropriate location on the inner part of $m(c_0)$.

Fig. 5(a) shows a part of the DCO diagram that includes bridge a. Fig. 5(b) is a realization of this part. A bridge is actually drawn as a polygon.

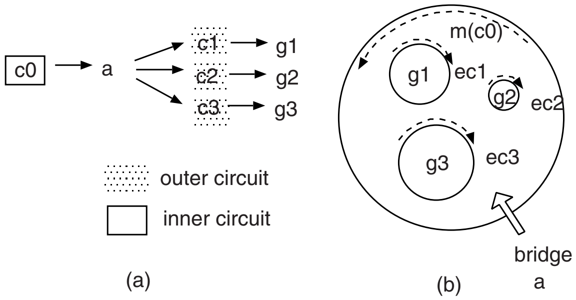


Fig. 5. Realization of a bridge

3.2 Circle Packing

Circle packing is an arrangement of circles inside a given boundary such that no two of them overlap and some, or all, of them are mutually tangent [22,16,17]. This is known as an NP-complete problem in general, but optimal solutions have been found in several cases. The studies on circle packing usually treat simple types: the area to be packed is a simple form such as a circle or a rectangle, and few constraints are imposed on the circles to be packed.

The realization of a bridge is considered to be a type of circle packing problem which is formalized as follows:

> **[Problem $*$]** Pack a non-convex polygon with a specified number n of circles with the constraints: (1) all the circles used for packing are as large as possible, and (2) the corresponding circle increases in size with the increasing number of areas in a module.

This problem is a difficult one and none of the existing algorithms can be applied directly. Therefore, we use a Genetic Algorithm, as a more flexible solution.

3.3 Genetic Algorithm

A Genetic Algorithm (GA) is a search technique to find an optimal solution or an approximation to an optimal solution [8]. It is inspired by evolutionary biology concept such as inheritance, mutation, selection and crossover.

In general, after creating the initial populations of chromosomes, each of which is represented by a bit string, the GA involves repeatedly computing the fitness of each chromosome, taking pairs of chromosomes and creating their offspring until a suitable solution is obtained. In creating offspring, crossover (exchanging selected bits between chromosome) and mutation (flipping chosen bits with a certain possibility) are used. Candidate optimal solutions evolve over time.

3.4 Experiment

Gene Encoding. We implemented the above problem [Problem $*$] as follows. Let (x_i, y_i) denote a coordinate of the center of an i-th circle ($1 \leq i \leq n$). In addition, let $M_1, \ldots, M_n$ be the set of modules obtained by decomposing the graph corresponding to a PLCA expression, and n_i be the number of the areas in M_i ($1 \leq i \leq n$).

Each chromosome corresponds to the locations of n circles and it is denoted by an array of the coordinates of their centers $(x_1, y_1), \ldots, (x_n, y_n)$. Each coordinate (x_i, y_i) is encoded as a sequence $a_{i1}, \ldots, a_{iL}, b_{i1}, \ldots, b_{iL}$, where each a_{ij} ($1 \leq j \leq L$) and each b_{ij} ($1 \leq j \leq L$) is a digit either of $0, \ldots, 9$[2], and L is a sufficiently large number. Let $x_{max}, x_{min}, y_{max}$ and y_{min} be the coordinates defined as follows:

x_{max}: the largest x-coordinate of the drawn bridge
x_{min}: the smallest x-coordinate of the drawn bridge
y_{max}: the largest y-coordinate y where (x_i, y) is in the drawn bridge
y_{min}: the smallest y-coordinate y where (x_i, y) is in the drawn bridge

Then, x_i and y_i are calculated as follows so that there is no lethal gene[3](Fig. 6):

$$\begin{cases} x_i = (x_{max} - x_{min}) \displaystyle\sum_{j=1}^{n_i} \left(\frac{1}{10}\right)^j a_{ij} + x_{min} \\ y_i = (y_{max} - y_{min}) \displaystyle\sum_{j=1}^{n_i} \left(\frac{1}{10}\right)^j b_{ij} + y_{min} \end{cases}$$

The radius of each circle is determined incrementally using distances between the centers of the circles and the boundary of the drawn bridge. We show the pseudo code in the Appendix.

In addition, we use a local search method to compute the fitness for obtaining a better solution.

Parameter Setting. Let S_{ij} ($j = 1, \ldots, n_i$) be the size of an area contained by M_i.

Let S_{total}, N_{total} and S_{av} denote the total size of the areas, the total number of all the areas, and the average size of an area contained in a PLCA expression, respectively. They are defined as follows:

$$S_{total} = \sum_{i=1}^{n} \sum_{j=1}^{n_i} S_{ij}, \quad N_{total} = \sum_{i=1}^{n} n_i, \quad S_{av} = \frac{S_{total}}{N_{total}}$$

Fitness is evaluated so that the total size of the circles used for packing an area is as large as possible, and the size of a circle is proportional to the total number of the

[2] We used digits here for encoding whereas bits are used in general.
[3] This is the basic definition. The actual calculation of y_i is more complicated.

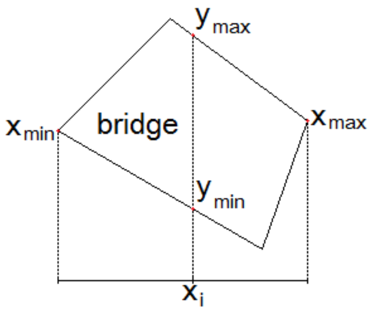


Fig. 6. Calculation of the coordinate

areas that are recursively contained in the corresponding module. Therefore, it can be calculated as:

$$fitness = S_{total} - \sum_{i=1}^{n} \frac{1}{N_i} \cdot \sum_{j=1}^{n_i} |S_{av} - S_{ij}|$$

where N_i is the total number of the areas that are recursively contained in module M_i.

Crossover takes place at n randomly chosen points, and mutation at randomly chosen positions which in our case occurs when the digit is changed to any other digit.

We performed the simulation several times to find appropriate values for the crossover and mutation rates. As a result, the mutation rate is set at the fixed value 1.0, and the crossover rate is set at 0.4 for PLCA expressions with multiple bridges one inside another, and at 0.9 for PLCA expressions with a bridge in which multiple modules are embedded.

Experiment and Evaluation. Experiments are performed using three PLCA expressions corresponding to the figures shown in Fig. 7.

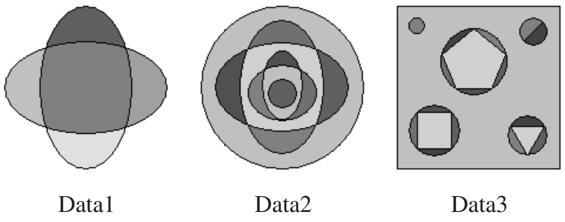
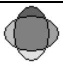
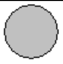
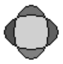
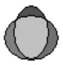
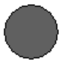
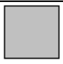
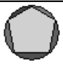
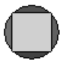
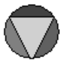
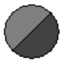
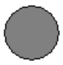


Fig. 7. Figures corresponding to given PLCA expression

1. Data1:This is a simple PLCA expression with one module and no bridge.
2. Data2: This has multiple bridges, one inside another, and is used to check that circles are drawn as large as possible.
3. Data3: This has a bridge in which multiple modules with varying numbers of areas are embedded. This is used to check that the size of a circle is proportional to the total number of areas recursively contained in the corresponding module.

mod	rad	opt rad
	80.0	80.0

mod	rad	opt rad
	80.00	80.00
	34.98	40.00
	22.87	25.51
	10.30	11.44

mod	size	no. of areas
	20066.00	18
	1219.97	6
	1068.04	5
	852.78	4
	425.42	2
	220.72	1

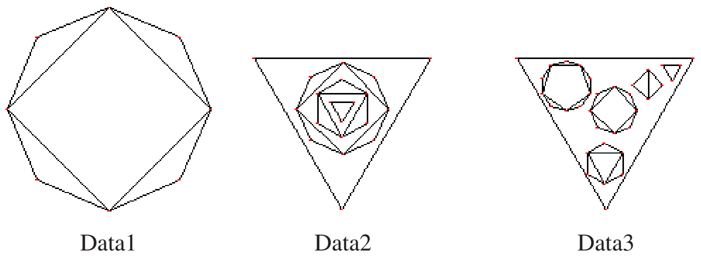


Fig. 8. Results of drawing

We use a $160*160$ rectangle as a drawing field. Fig. 8 shows the result of the drawing. It shows that all the relationships of the PLCA components are preserved. Above all, each module is embedded in the correct bridge.

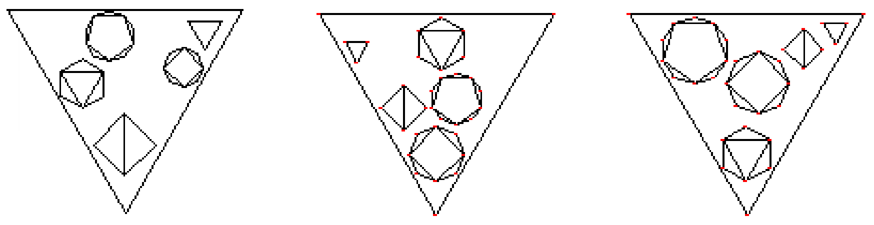
In the tables for Data1 and Data2, mod is the module, rad is the radius of an e-circle and $opt\ rad$ is the radius of an inscribed circle of a bridge, that is the biggest size of a module to be located there. In the table for Data3, mod is the module, $size$ is the total size of all the areas contained in that module, and $no.\ of\ areas$ shows the total number of areas recursively contained in the corresponding module.

In evaluating the results, we ignore the shape of an $outermost$, which is always an inscribed polygon of an e-circle, and discuss the locations and the sizes of the e-circles.

For Data1, the radius is the biggest radius. It follows that a module is drawn as large as possible.

For Data2, the radii of e-circles are slightly smaller than the optimal ones, since the circle should not be tangent to the boundary of the bridge. It follows that the modules are drawn as large as possible.

For Data3, the size of an e-circle is proportional to the number of areas contained in the corresponding module. It is considered that a module containing the larger number of areas is more complex. Therefore, it follows that complex objects are drawn larger than non-complex objects.



(a) naiive location (b) first generation of GA (c) finally obtained

Fig. 9. Cognition of “goodness”

We showed the three figures in Fig. 9, equivalent according to PLCA, to twenty test subjects, and asked them to select the “best” figure. Seventeen of them chose (c), the figure produced by our algorithm.

From these results, we conclude that we have obtained the ”good figure”.

4 Discussion

4.1 Direct Drawing from PLCA

The basic concept of the algorithm used in our experiment was to transform a PLCA expression into a graph, and then draw that graph.

We have proposed another method for drawing a figure directly from a PLCA expression [18]. The basic idea of that algorithm was to draw circuits. Starting from the outermost, draw circuits in the inner area enclosed by the outermost. Draw all the circuits by repeating this procedure recursively.

For that algorithm, we can prove that a figure that preserves the relationship described in the PLCA expression can be drawn, but the method for determining the specific coordinates is not given. This means that we can draw a figure only if we specify proper coordinates. If the circuits share lines or points, a subtle problem exists in determining the size of lines, the location of points, and the shape of circuits. Unfortunately, the actual figure cannot be drawn by automatically using this algorithm.

For example, consider a PLCA expression corresponding to Fig. 10(a). Fig. 10(b) shows three processes for drawing a figure according to this algorithm. If we draw circuits using the process in the right-hand column, then the result is successful. But if we use in the other processes, the drawing fails.

Information from the other circuits to be drawn is not available at the time of drawing the first circuit because this algorithm uses recursive processing. Therefore, appropriate coordinates cannot be determined algorithmically. The algorithm using GA is more flexible in producing a solution.

4.2 Drawing from Other Qualitative Representations

Region Connection Calculus (RCC) [13] and the 9-intersection model [6] are representative frameworks for qualitative spatial reasoning. We examine an algorithm for

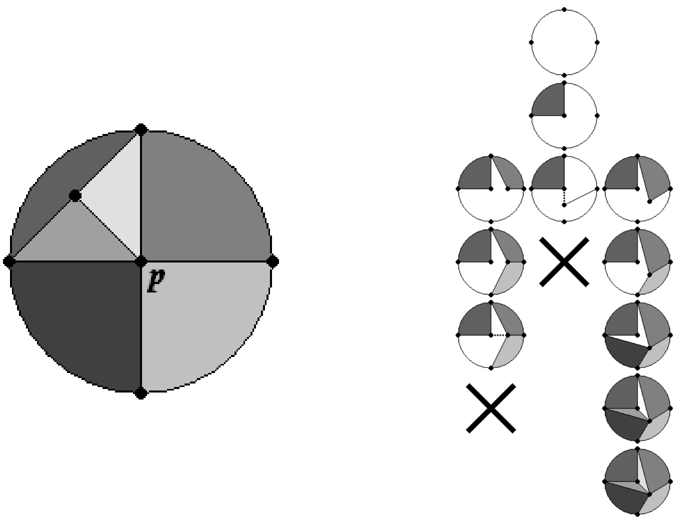


Fig. 10. Drawing from a PLCA expression directly

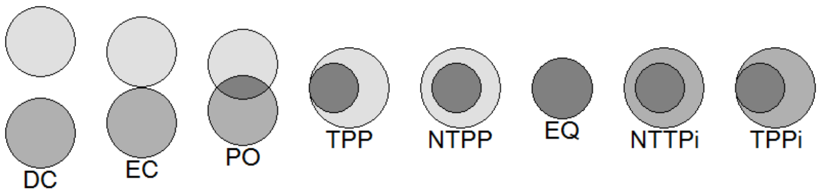


Fig. 11. Fundamental relationships of RCC

drawing a figure in a two-dimensional plane based on these representations. Although the existence of a topological space and planarity for a set of RCC relationships have been discussed in [9,14,23], an algorithm for drawing has not been reported so far.

Fig. 11 shows the eight fundamental relationships of RCC. Consider a set of RCC relationships $S = \{R_1, \ldots, R_n\}$ which is realizable in a two-dimensional plane. If each $R_i (i = 1, \ldots, n)$ is either NTPP, NTPPi, DC or EQ, then S is transformed uniquely into PLCA. In this case, the figure corresponding to S can be drawn using the PLCA drawing algorithm. Otherwise, there exists multiple PLCA expressions. While we can draw one of them, it is an open question as to which one. The 9-intersection model, in which positional relationships between regions are represented in the form of a $3{*}3$ matrix, results in similar uncertainty.

A major difference between PLCA and the other QSR systems is in the way in which relationships among objects are represented. In the other QSR systems, the entire figure is represented in the form of a set of binary relations, while we do not use binary relations. Moreover, objects in the other QSR systems may share parts with each other which is prohibited in PLCA. PLCA uses a more refined classification for equivalent figures than the other QSR systems. It may be possible to determine a one-to-one mapping to PLCA from the extension of RCC, containing information on the connection patterns of regions [5], or the extension of the 9-intersection model [7,10].

5 Conclusion

In this paper, we have proposed an algorithm for drawing a figure in a two-dimensional plane corresponding to a PLCA expression. In general, it is difficult to draw a figure for a qualitative representation since coordinates are not determined uniquely. We thus have proposed an algorithm using GA, and succeeded in producing drawings. The resulting figures not only preserve the relationships in PLCA expressions, but are also "good" figures in the sense that their objects are drawn as large as possible, and complex parts are drawn larger size than those less complex.

We used the size and the number of areas as parameters in the GA. If a specific requirement such as emphasis on a certain object is integrated, we can automatically draw different figures for the same PLCA expression depending on the application, to meet specific requirement details. In future, we are considering the reduction of time in finding a solution in the GA. We will conduct further study on algorithms to draw directly from PLCA expressions and from other qualitative representations. Furthermore, the solution to the problem of packing multiple circles in a non-convex polygon given in this paper could be useful in other applications of packing.

Acknowledgments. This research was supported by KAKENHI19500134.

References

1. Anderson, M., Meyer, N., Olivier, P. (eds.): Diagrammatic Representation and Reasoning. Springer, Berlin (2002)
2. Chartland, G., Lesniak, L.: Graphs & Digraphs, 3rd edn. Wadsworth & Brooks/Cole (1996)
3. Cohn, A.G.: A hierarchical representation of qualitative shape based on connection and convexity. In: Kuhn, W., Frank, A.U. (eds.) COSIT 1995. LNCS, vol. 988, pp. 311–326. Springer, Heidelberg (1995)
4. Cohn, A.G., Hazarika, S.M.: Qualitative spatial representation and reasoning: an overview. Fundamental Informaticae 46(1), 1–29 (2001)
5. Cohn, A.G., Varzi, A.: Mereotopological connection. Journal of Philosophical Logic 32, 357–390 (2003)
6. Egenhofer, M., Herring, J.: Categorizing binary topological relations between regions, lines and points in geographic databases. Technical Report. Department of Surveying Engineering, University of Maine (1990)
7. Egenhofer, M., Franzosa, R.: On the equivalence of topological relations. International Journal of Geographical Information Systems 9(2), 133–152 (1995)
8. Goldberg, D.E.: Genetic algorithms in search, - Optimization and machine learning. Kluwer Academic Publishers, Dordrecht (1989)
9. Grigni, M., Papadias, D., Papadimitriou, C.: Topological Inference. In: International Joint Conference on Artificial Intelligence, pp. 901–907 (1995)
10. Nedas, K., Egenhofer, M., Wilmsen, D.: Metric Details of Topological Line-Line Relations. International Journal of Geographical Information Science 21(1), 21–48 (2007)

11. Overmars, M., de Berg, M., van Kreveld, M., Schwarzkopf, O.: Computational Geometry. Springer, Berlin (1997)
12. Ochiai, N.: Introduction to Graph Theory: Application for Plane Graph. Nihon-Hyouron-sha (In Japanese) (2004)
13. Randell, D., Cui, Z., Cohn, A.G.: A spatial logic based on regions and connection. In: Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning (KR92), pp. 165–176. Morgan Kaufmann, San Francisco (1992)
14. Renz, J.: Qualitative Spatial Reasoning with Topological Information. In: Renz, J. (ed.) Qualitative Spatial Reasoning with Topological Information. LNCS (LNAI), vol. 2293, Springer, Heidelberg (2002)
15. Stock, O. (ed.): Spatial and Temporal Reasoning. Kluwer Academic Publishers, Dordrecht (1997)
16. Stephenson, K.: Circle packings in the approximation of conformal mappings. Bulletin of American Mathematics Society 23, 407–416 (1990)
17. Stephenson, K.: Introduction to circle packing - The theory of discrete analytic functions. Cambridge University Press, Cambridge (2005)
18. Sumitomo, T., Takahashi, K.: DLCS: Qualitative representation for spatial data. In: Twenty-first Annual Meeting of Japan Society for Software Science and Technology (In Japanese) (2004)
19. Sumitomo, T., Takahashi, K.: A qualitative treatment of spatial data. In: The 17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI05), pp. 539–548. IEEE Computer Society Press, Los Alamitos (2005)
20. Takahashi, K., Sumitomo, T.: A framework for qualitative spatial reasoning based on the connection patterns of regions. In: IJCAI-05 Workshop on Spatial and Temporal Reasoning, pp. 57–62 (2005)
21. Takahashi, K., Sumitomo, T., Takeuti, I.: On embedding a qualitative representation in a two-dimensional plane. In: IJCAI-07 Workshop on Spatial and Temporal Reasoning, pp. 101–109 (2007)
22. Williams, R.: Circle packings, plane tessellations, and networks. In: The Geometrical Foundation of Natural Structure: A Source Book of Design. Dover, New York, pp. 34–47 (1979)
23. Wolter, F., Zakharyaschev, M.: Spatio-temporal representation and reasoning based on RCC-8. In: Proc. International Conference on Principles of Knowledge Representation and Reasoning (KR2000), pp. 3–14. Morgan Kaufmann, San Francisco (2000)

Appendix Determining the Radii of Circles

```
$C = \{c_1, c_2, \ldots, c_n\}$   where $c_i$ $(1 \leq i \leq n)$ is the coordinate of the center.
$Decided = \{\}$.        % a set of cirlces whose radii are decided
$Undecided = \{\}$.      % a set of cirlces whose radii are undecided

WHILE ($|C| > 0$)
  Take an arbitrary $c_i$ of $C$.
  Set $d_{i1}$ to the shortest distance between $c_i$ and the boundary of $Area$.
  Set $d_{i2}$ to the half of the shortest distance between $c_i$ and $c_j$ for each $j$ $(i \neq j)$.
  Let $r_i$ be the radius of the circle whose center is $c_i$.
  IF ($d_{i1} < d_{i2}$)
    $r_i = d_{i1}$.
    $Decided = Decided \cup \{c_i\}$.
    IF ($(d_{i2} < d_{i1}) \wedge (d_{j2} < d_{j1})$)
        $r_i = d_{i2}$.
        $Decided = Decided \cup \{c_i\}$.
    ELSE
        $Undecided = Undecided \cup \{c_i\}$.
ENDWHILE
WHILE ($|Undecided| > 0$)
    Take an arbitrary $c_i$ of $Undecided$.
    Set $d_{i2}$ to the half the shortest distance between $c_i$ and $c_j \in Undecided\ (i \neq j)$.
    Set $d_{i3}$ to the shortest distance between $c_i$ and $c_k \in Decided$.
    IF ($d_{i1} = \min(d_{i1}, d_{i2}, d_{i3})$)
        $r_i = d_{i1}$.
        $Decided = Decided \cup \{c_i\}$
    IF ($d_{i3} = \min(d_{i1}, d_{i2}, d_{i3})$)
        $r_i = d_{i3}$.
        $Decided = Decided \cup \{c_i\}$.
    ELSE
        $Undecided = Undecided \cup \{c_i\}$.
ENDWHILE
```