An Implmentation of the Boolean Algebra For Yin Sets

Zhixuan Li

October 21, 2021

1 Algorithm

Definition 1. *The subspace $\mathbb{J}^{pl}$ of the Jordan space $\mathbb{J}$ contains $\{\hat{0}\}$, $\{\hat{1}\}$ and all the spadjor forests whose realizable spadjor is a set of oriented polygons.*

Definition 2. *If P is a polygon, denote by $|P|$ the number of vertices of P. If $\mathcal{F} \in \mathbb{J}^{pl}$, $|\mathcal{F}| := \sum_{P \in \mathcal{J}(\mathcal{F})} |P|$. In particular, $|\{\hat{0}\}| = |\{\hat{1}\}| = 0$.*

1.1 The Locater

Given a point and a Yin set in the plane, there are three possibilities for the relative location of this point to the Yin set, i.e. interior, exterior and on boundary. Based on the idea of the line sweep alogorithm [?], Algorithm 1 is designed as follows.

Proposition 1. *Algorithm 1 correctly computes the relative locations.*

Proof. The proof consists of two parts. First we prove that $\mathcal{T}$ records the segments in the order that they intersect with the sweep line from left to right. Next we show that for a particular query point, all possibilities are exhausted in the algorithm. WLOG we may assume S does not contain horizontal segments.

1. The segments in S do not overlap or intersect properly since they are collapsed from a realizable spadjor. It follows that during the presence of a segment in $\mathcal{T}$, it does not exchange position with other segments. Finally the appearance and disappearance of a segment in the line sweep process happens only at endpoints, and Line 9-10 handle that properly.

2. If the current event point $p \in \mathcal{P}(\mathcal{F})$, then p either coincides with an endpoint or lies in the interior of a segment. They are resolved in Line 11-13 and Line 18-19 respectively. Otherwise, by the definition of interior, the relative location of p can be determined from any of the two neighbouring directed segments (Line 21). In the trivial case of p not having a neighrbouring segment, its relative location depends on whether $\rho(\mathcal{F})$ is bounded (Line 17).

□

Proposition 2. *Algorithm 1 takes time $\mathcal{O}((n + q) \log n)$. Here $n = |\mathcal{F}|$ and q is the number of queries.*

Algorithm 1 locate($\mathcal{F}, P$) : Compute the relative locations of a set of points

```
Input :   A non-trivial $\mathcal{F} \in \mathbb{J}^{pl}$. A set of query points $P = \{p_1, \cdots, p_q\}$.
Output :  The relative locations $\mathrm{loc}(p), \forall p \in P$.
 1: Collapse $\mathcal{F}$ into a set of directed segments $S = \{s_1, \cdots, s_n\}$.
 2: Initiailize the ordered set $Q$ to contain the endpoints of all the segments in $S$, as well as the
    query points.
 3: Denote the set of segments whose upper endpoint is $p$ as $U(p)$, and $L(p)$ likewise.
 4: Initialize the status structure $\mathcal{T}$ as empty.
 5:
 6: while $Q$ is not empty do
 7:   Pop the front of $Q$, denoted as $p$.
 8:   if $L(p) \cup U(p) \neq \emptyset$ then
 9:     Remove the segments in $L(p)$ from $\mathcal{T}$.
10:     Add the segments in $U(p)$ to $\mathcal{T}$.
11:     if $p \in P$ then
12:       $\mathrm{loc}(p) \leftarrow$ OnBoundary.
13:     end if
14:   else
15:     Denote the left and right neighbour of $p$ in $\mathcal{T}$ by $s_l, s_r$ respectively.
16:     if neither $s_l$ nor $s_r$ exists then
17:       $\mathrm{loc}(p)$ is determined from whether $\rho(\mathcal{F})$ is bounded.
18:     else if $p \in s_l$ or $p \in s_r$ then
19:       $\mathrm{loc}(p) \leftarrow$ OnBoundary.
20:     else
21:       $\mathrm{loc}(p)$ is determined from either Left$(s_l, p)$ or Left$(s_r, p)$.
22:       /* Left$(\{p_1 \to p_2\}, q) = (p_2 - p_1) \times (q - p_1) > 0$ */
23:     end if
24:   end if
25: end while
```

1.2 The pasting map

The implementation of the pasting map is listed in Algorithm 2. If $\mathcal{J}$ is the output of the algorithm, and $\rho(\mathcal{J})$ is topological simple, we may assume that the processing of each curve segment in E, the output of (MRS-d), has constant complexity. Under such assumption, the running time of Algorithm 2 is roughly linear to $|E|$.

Algorithm 2 $\mathcal{J}$ = PastingMap(G)

Input : The directed multigraph $G = (V, E)$ output by the operation (MRS-d).
Output : A set of Jordan curves $\mathcal{J}$.
Postcondition : $\mathcal{J}$ is a realizable spadjor.

```
1: Initialize $\mathcal{J}$ as empty.
2: Initialize $R$, a linear container of curve segments, as empty.
3: while true do
4:   if $R = \emptyset$ then
5:     Exit if $E = \emptyset$.
6:     Reset $\mathcal{P} \leftarrow \emptyset$.
7:     Select a non-isolated vertex $v_s \in V$. Select an arbitrary edge $s$ that starts from $v_s$.
8:   else
9:     Denote the last curve segment in $R$ as $s'$.
10:    Let $v_s$ be the end point of $s'$.
11:    Among the out-edges incident to $v_s$, select the out-edge $s$ to be the clockwise neighbour
       of $s'$.
12:  end if
13:  Push back $s$ into $R$.
14:  $\mathcal{P} \leftarrow \mathcal{P} \cup \{v_s\}$.
15:  Let $v_e$ be the end point of $s$.
16:  if $v_e \in \mathcal{P}$ then
17:    Let $R^\circ \subset R$ be the subset of curves segments that forms a loop.
18:    $R \leftarrow R \setminus R^\circ$.
19:    For each $q \in \mathcal{P}$, remove $q$ from $\mathcal{P}$ if $q$ is on $R^\circ$ but not the start point of $R^\circ$.
20:    Construct a Jordan curve $\gamma$ by joining the curve segments in $R^\circ$.
21:    $\mathcal{J} \leftarrow \mathcal{J} \cup \{\gamma\}$.
22:  end if
23:  $E \leftarrow E \setminus \{s\}$.
24: end while
```

1.3 The meet operation

Suppose Let $\mathcal{J}_1, \mathcal{J}_2 \in \mathbb{J}^{pl}$ and we are to calculate $\mathcal{J} = \mathcal{J}_1 \wedge \mathcal{J}_2$. We shall need the symbols in Table 1 in the complexity analysis.

1. (MRS-a) calculates the intersections of the Jordan curves in $\mathcal{J}_1$ and $\mathcal{J}_2$. It has complexity $\mathcal{O}((n + I)\log n)$.

2. The segmentation map S_V is performed in (MRS-b). It is of $\mathcal{O}(I)$.

3. (MRS-c) and (MRS-d) filter the output of S_V and construct the set E of curve segments. The function **locate()** is invoked here and it takes $\mathcal{O}((n + k_0)\log n)$.

n	$\lvert\mathcal{J}_1\rvert + \lvert\mathcal{J}_2\rvert$.
V	the set of isolated points characterizing $\mathcal{P}(\mathcal{J}_1) \cap \mathcal{P}(\mathcal{J}_2)$.
I	$\lvert V\rvert$.
k_0	the total number of β_i after the segmentation map S_V is appled. It is bounded below by I.
E	the set of curve segments output by (MRS-d). We have $\lvert E\rvert \le k_0$, but in general $\lvert E\rvert$ and I are not bounded by each other.

Table 1: Notations.

4. The pasting map takes $\mathcal{O}(|E|)$ in normal cases.

To summarize, the meet operation has the complexity $\mathcal{O}((n + k_0)\log n)$ in the worst case. In the normal cases when $\mathcal{J}_1, \mathcal{J}_2$ and $\mathcal{J}_1 \wedge \mathcal{J}_2$ are all topologically simple, we are safe to assume $k_0 = \mathcal{O}(I)$. Thus the complexity of the meet operation is bounded by $\mathcal{O}((n + I)\log n)$, where the majority of the computation time is spent on calculating the intersections of line segments.

1.4 The complementation operation

Suppose $\mathcal{J} \in \mathbb{J}^{pl}$ and we are to calculate $\mathcal{J}'$. Let $n = |\mathcal{J}|$. Reversing the orientations of each Jordan curve requires $\mathcal{O}(n)$ operations. Note that the segmentation map and the pasting map operate on no more than n curve segments, hence the complexities of which are bounded by $\mathcal{O}(n)$. Thus the complementation operation has the complexity $\mathcal{O}(n)$.

1.5 Constructing the Hasse Diagram

Definition 3. *Let $\mathcal{J} = \{\gamma_1, \cdots, \gamma_m\}$ be a non-empty realizable spadjor. The directed graph induced by the covering relation on $\mathcal{J}$ is $G_{\mathcal{J}} = (V, E)$ where*

$$\begin{aligned} V &= \{1, \cdots, m\}, \\ E &= \{(k, l) : \gamma_k \succ \gamma_l, 1 \le k, l \le m\}. \end{aligned} \tag{1}$$

Recovering the Hasse diagram of a $\mathcal{J} \in \mathbb{J}^{pl}$ requires a topological sort on the directed graph induced by $G_{\mathcal{J}}$. Details are listed in Algorithm 3.

Algorithm 3 buildHasse($\mathcal{J}$) : construct the Hasse diagram for $\mathcal{J}$

```
Input :   A non-empty realizable spadjor $\mathcal{J} = \{\gamma_1, \cdots, \gamma_m\}$.
Output :  The Hasse diagram of $\mathcal{J}$.
1: For $i = 1, \cdots, m$, let $p_i$ be an interior point in the bounded component of $\mathbb{R}^2 \setminus \gamma_i$.
2: Initialize $G_{\mathcal{J}} = (V, E)$ where $V = \{1, \ldots, m\}$ and $E = \emptyset$.
3: Let $Q_i = \{1 \le j \le m : j \ne i, \mathrm{BB}(\gamma_i) \supset \mathrm{BB}(\gamma_j)\}, 1 \le i \le m$. BB$(\cdot)$ is the bounding box
   function.
4:
5: for $i = 1, \cdots, m$ do
6:     Call locate($\{\gamma_i\}, \{p_j : j \in Q_i\}$).
7:     For each $j \in Q_i$, set $E \leftarrow E \cup \{(i, j)\}$ if $p_j$ is in the bounded component of $\mathbb{R}^2 \setminus \gamma_i$.
8: end for
9: Return TopologicalSort($G_{\mathcal{J}}$).
```

Proposition 3. *Algorithm 3 takes time $\mathcal{O}(m(m+l)\log l)$ in the worst case, where m is the number of Jordan curves in a realizable spadjor and $l = \max_{\gamma \in \mathcal{J}} |\gamma|$.*