BOOLEAN EMBEDDING
— From a Partially Ordered Structure to a Boolean Algebra —

JINSEI YAMAGUCHI

Department of Information and Computer Sciences, Kanagawa University
2946 Tsuchiya, Hiratsuka, Kanagawa 259-1293, Japan

We prove the following theorem.
Theorem Let $(P,\leqq)$ be an arbitrary partially ordered structure. Then, there is a canonical embedding (1:1, order-preserving map)

$$e : (P,\leqq) \to \mathbb{B}$$

such that

(i). $\mathbb{B}$ is a complete Boolean algebra.

(ii). $(\forall p,q\in P)(p$ and q are compatible $\to e(p)\wedge e(q)\neq \mathbb{0})$

(iii). $\{e(p) \mid p\in P\}$ satisfies " a small property " in the sense that;

(a) If P is finite, then $\mathbb{B}$ is finite.

(b) If P is infinite, then for any infinite branch *l* in $\mathbb{B}-\{\mathbb{0}\}$,

$(\forall \mathbb{p}\in l)(\exists q\in P)(e(q)\leqq \mathbb{p})$.

(c) for any minimal element a in P, $e(a)$ becomes a minimal element in $\mathbb{B}-\{\mathbb{0}\}$. □

Especially, when $(P,\leqq)$ is finite, we give an algorithm called " the Normal Separativization " which realize the embedding e. As the byproduct, we obtain an interesting insight that the set of all finite partially ordered structures can be classified to categories called " Boolean complexity of the type (n,m) ", where n and m are natural numbers.

Keywords: Partially ordered structure; Boolean algebra; embedding; algorithm

C.R.Categories: F.2, G.2

§ 1. INTRODUCTION

Consider some practical phase. Suppose we are given the task of making a model for a system S and assume that we abstract a partially ordered structure $(P,\leqq)$ from the system as an appropriate approximation. Now, let p, q be two elements in P. We are sometimes faced to the case that it is interesting (or necessary) to consider the least upper bound $p\vee q$ or the greatest lower bound $p\wedge q$ of p, q. However, there does not always exist $p\vee q$ or $p\wedge q$ for arbitrary elements p, q in P, because P is not a lattice in general.

In this case, we usually reform $(P,\leqq)$ to a new structure $(P',\leqq')$ in order to obtain $p_i\vee q_i$ or $p_i\wedge q_i$ for some particular pairs $\{(p_i,q_i) \mid 1\leqq i\leqq n\}$ in P, so that the resulting new structure $(P',\leqq')$ better fits to the original system S. Here, there occurs one problem, if the reform is not a minor change. That is, if P is big enough and the number of pairs (p_i,q_i) is large, we can not employ the one-by-one repairing technique. In this case, we need some

" systematic and genetic approach " to enrich $(P,\leqq)$ so as to get a lattice structure $(P'',\leqq'')$ such that $(P',\leqq') \subset (P'',\leqq'')$. (Then, ignore useless elements in $P''-P'$.) Once we employ this kind of general methodology, then there is a possibility that we can obtain a new insight into the original system S via the enrichment. (**Discovery**)

By the way, from a viewpoint of application, the most popular and the most useful lattice structure is a Boolean algebra. In this sense, it is convenient to consider the algorithm such that, given an arbitrary partially ordered structure $(P,\leqq)$, $(P,\leqq)$ is automatically enriched to the corresponding " canonical " Boolean algebra $\mathbb{B}$. Here, the " canonicity " is given by a **"** small property **"** of the resulting $\mathbb{B}$ compared with $(P,\leqq)$. If $\mathbb{B}$ is too big compared with $(P,\leqq)$, $\mathbb{B}$ is not useful as a practical application, because $\mathbb{B}$ may contain additional noisy elements as a model of the original system S. Moreover, theoretically speaking, any $(P,\leqq)$ is always embeddable to a large enough $\mathbb{B}$ trivially. So, we recognize that the importance at this stage is the notion of the small property. In this context, we obtain the next theorem.

Theorem 1-1. Let $(P,\leqq)$ be an arbitrary partially ordered structure. Then, there is a canonical embedding (1:1, order-preserving map)

$$e : (P,\leqq) \to \mathbb{B}$$

such that

(i). $\mathbb{B}$ is a complete Boolean algebra.

(ii). $(\forall p,q \in P)(p$ and q are compatible $\to e(p) \wedge e(q) \neq \mathbb{0})$

(iii). $\{e(p) \mid p \in P\}$ satisfies " a small property " in the sense that;

(a) If P is finite, then $\mathbb{B}$ is finite.

(b) If P is infinite, then for any infinite branch *l* in $\mathbb{B}-\{\mathbb{0}\}$,

$(\forall \mathbb{p} \in l)(\exists q \in P)(e(q) \leqq \mathbb{p})$.

(c) for any minimal element a in P, $e(a)$ becomes a minimal element in $\mathbb{B}-\{\mathbb{0}\}$. □

Thus, we notice that the resulting $\mathbb{B}$ is small enough, compared with P. In the above theorem, we use the usual definition of the " compatibility " in the sense that;

Definition 1-2. Let $(P,\leqq)$ be an arbitrary partially ordered structure. Then,

1. $p,q \in P$ are " compatible " iff $(\exists r \in P)(\ r \leqq p$ and $r \leqq q\ )$
2. $p,q \in P$ are " incompatible " iff p,q are not compatible. ⊣

The main purpose of this paper is prove this theorem by proposing an algorithm which realizes the above stated embedding e. As the byproduct, we reach the theoretically interesting recognition that all finite partially ordered structures are classified to categories called " Boolean complexity of the type (n,m) ", where n and m are natural numbers.

In the following, we use the terminology " p.o.s. " for the abbreviation of " partially ordered structure ", for the sake of convenience.

§ 2. REGULAR OPEN COMPLETE BOOLEAN ALGEBRA

In this section, we present a preliminary lemma which is necessary to prove Theorem 1-1. To state the result, we need a few definitions.

Definition 2-1. Let $(P, \leqq)$ be an arbitrary p.o.s.. Then,

1. For any p, q in P, q is an " immediate predecessor " of p (p is an " immediate successor " of q) iff

   $p < q$ and there is no element between p and q.

2. $(P, \leqq)$ is " separative " iff

   $(\forall p,q \in P)(\neg(p \leqq q) \rightarrow (\exists r \leqq p)(\ r$ and q are incompatible$))$

3. $X \subset P$ is " dense " in P iff $(\forall p \in P)(\exists q \in X)(q \leqq p)$. ┤

Here, in 1, we use " the reverse definition with respect to $\leqq$ " compared with the usual definition of the" immediate predecessorness(successorness) " . Note that this is only a matter of convention. We can use the usual terminology concerning the predecessorness and/or the successorness. In the following, however, we use the above definition, because it well fits to the figures drawn in this paper.

Now, in order to catch the notion of the separative intuitively, we exhibit a few examples.

Example 2-2.

1. A tree which has at least two immediate successor nodes at each node (except the final nodes (leaves)) is separative.

2. A simple separative p.o.s. which is not a tree is, for example, illustrated by the figure 1.

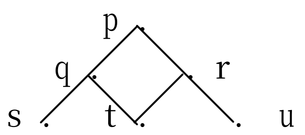

Figure 1

(In the following, an upper point is larger than a lower point in the sense of $\leqq$.)

3. A p.o.s. illustrated by the figure 2 is not separative.

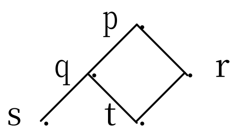

Figure 2 ┤

Concerning the notion of the " dense ", we can introduce a topology on an arbitrary p.o.s. in the following manner.

Definition 2-3. Let $(P,\leqq)$ be an arbitrary p.o.s.. We can introduce a topology T on P by using the notion of a " cut "

$[p] =\{q\in P \mid q\leqq p\}$ for each $p\in P$

as a basic open set. Based on this topology T, we can obtain the set

r.o.$(P)=\{T\in\mathsf{T} \mid T$ is a regular open set $\}$ ⊣

Here, it is easy to check that, for any p.o.s. $(P,\leqq)$, (r.o.$(P),\subseteq)$ becomes a complete Boolean algebra. In addition, when $(P,\leqq)$ is separative, it is not so difficult to see that, for each $p\in P$, $[p]\in$r.o.(P).

Example 2-4. Let $(P,\leqq)$ be a p.o.s. illustrated by the figure 3.

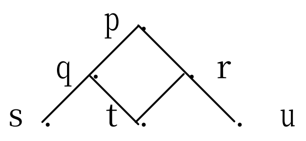

Figure 3

Then, r.o.(P) becomes a complete Boolean algebra illustrated by the figure 4.

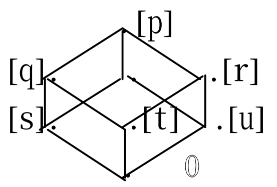

Figure 4 ⊣

So, we can define " the canonical embedding (1:1, order-preserving map) "

$e : (P,\leqq) \rightarrow$ (r.o.$(P),\subseteq)$

for each separative p.o.s. Using this embedding, we obtain the desired result.

Lemma 2-5. Let $(P,\leqq)$ be a separative p.o.s. Then, the canonical embedding

$e : (P,\leqq) \rightarrow$ (r.o.$(P),\subseteq)$

satisfies the following properties.

(1). r.o.(P) is unique up to isomorphism.
(2). $p,q \in P$ are compatible $\rightarrow$ $e(p) \wedge e(q) \neq \mathbb{0}$
(3). $\{e(p) \mid p \in P\}$ is dense in r.o.(P)$-\{\mathbb{0}\}$,

where $\mathbb{0}$ is the least element in r.o.(P).

Proof: See, for example, [1]. $\square$

However, if $(P, \leqq)$ is not separative, the above defined

$$e : (P, \leqq) \rightarrow (\text{r.o.}(P), \subseteq)$$

does not become 1:1, though r.o.(P) is unique up to isomorphism. The reason is that, for any non-separative $(P, \leqq)$, $(\exists p \in P)([p] \notin \text{r.o.}(P))$. In other words,

$(P, \leqq)$ is separative iff $[p] \in$ r.o.(P) for any $p \in P$.

Example 2-6. Let $(S, \leqq)$ be a p.o.s. illustrated by the figure 5.

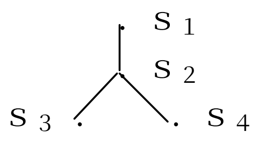

Figure 5

Then, this tree is not separative and r.o.(S) becomes a Boolean algebra illustrated by the figure 6.

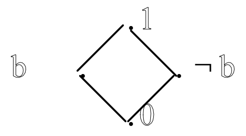

Figure 6

Here, $[s_1]=\{s_1, s_2, s_3, s_4\}$, $[s_3]=\{s_3\}$, $[s_4]=\{s_4\}$ become regular open. However, $[s_2]=\{s_2, s_3, s_4\}$ is not regular open. So, the correspondence is illustrated by the figure 7.

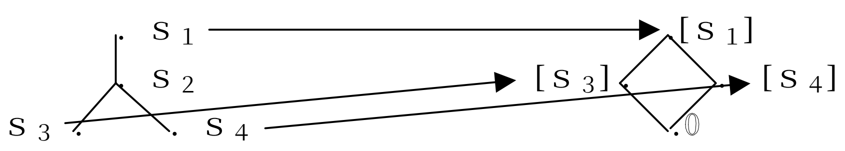

Figure 7 ⊣

With these facts in mind, in order to prove our main theorem, we employ the following methodology.

Fundamental Methodology

「Given an arbitrary p.o.s. （P , ≦） .

Ⅰ. Firstly, we enrich （P , ≦） to a suitable "separative" p.o.s. （P' , ≦'） so that

$$g : (P, \leqq) \rightarrow (P', \leqq')$$

is a 1:1, order preserving map.

Ⅱ. Secondly, we use the above results

$$e : (P', \leqq') \rightarrow (\text{r.o.}(P'), \subseteq)$$

to obtain a complete Boolean algebra r.o.(P').

Ⅲ. The enrichment at the stage Ⅰ must satisfy the conditions stated in Theorem 1-1 via the stage Ⅱ, so that $e \cdot g : (P, \leqq) \rightarrow \text{r.o.}(P')$ becomes the required result. 」 ⊣

In the following, we propose several algorithms which realize the stage Ⅰ and Ⅲ at the same time.

§ 3. NATURAL SEPARATIVIZATION

In general, given an arbitrary p.o.s. （P , ≦） , there are many techniques which enrich （P , ≦） to a separative p.o.s. （P' , ≦'） . Among them, there are some grand tactics which make the resulting （P' , ≦'） be not so large compared with the original （P , ≦） . We employ a few of them, which are based on the same strategy. To state the strategy simply, we need a terminology.

Definition 3-1. Let （P , ≦） be a p.o.s..

(1) Let $\{p_1, p_2, \cdots, p_n, \cdots\}$ be a subset of P. By taking completely new symbols $\{a_1, a_2, \cdots, a_n, \cdots\}$ which are not in P, we can enrich $(P, \leqq)$ to a new p.o.s. $(P, \leqq) \cup \{(a_1, p_1), (a_2, p_2), \cdots, (a_n, p_n), \cdots\}$.
We say that “$(P, \leqq)$ grow beards” through the process.

(2) Let $(P, \leqq)$ be a p.o.s. and let $(P, \leqq)$ grow beards to $(P', \leqq')$. If $(P', \leqq')$ becomes separative, then we say that $(P', \leqq')$ is “a natural separativization” of $(P, \leqq)$. ┤

Now, we are ready to state our basic strategy.

Basic Strategy:
「We employ a natural separativization to realize Ⅰ and Ⅲ .」 ┤

In the following, based on this strategy, we propose several algorithms which embody our fundamental methodology.

≪Tactics 1≫ **Trivial Separativization on a Tree**

The first case is to consider an arbitrary tree structure instead of a general p.o.s.. From a viewpoint of applications, this case is of worth being considered independently, because a variety kinds of knowledge are represented by tree structures.

Let $(P, \leqq)$ be an arbitrary tree structure.

Algorithm 1

```
「
Step 1. Try to find a node $s_n$ in $P_n$ whose immediate successor node consists of only one
        element, where $P_0 = P$.
Step 2. If there is no node whose immediate successor node consists of one element,
        Then, stop.
        Else, grow a beard at the node $s_n$ to obtain a new tree $P_{n+1}$ and go to Step 1.
」
```

It is easy to check that, by this algorithm, every tree becomes separative. In the following, for the sake of convenience, let’s call the technique of growing one beard at each node whose immediate successor node consists of just one element “the trivial separativization” in a general p.o.s.. Thus, the above result can be restated as

“every tree becomes separative by the trivial separativization.”

Now, for an arbitrary p.o.s. $(P, \leqq)$, the trivial separativization does not always make $(P, \leqq)$ be separative in general.

Example 3-2. Let $(P, \leqq)$ be a p.o.s. illustrated by the figure 8.

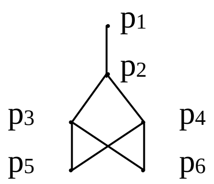

Figure 8

Then, the trivial separativization makes $(P, \leqq)$ be a p.o.s. illustrated by the figure 9.

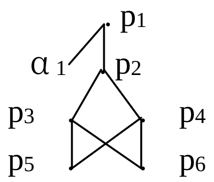

Figure 9

However, the above p.o.s. is not separative. ⊣

So, we need an additional technique to make $(P, \leqq)$ be separative. To state the technique, we need a terminology.

Definition 3-3. Let $(P, \leqq)$ be a p.o.s.. If $(P, \leqq)$ is not separative, then there is at least one pair (p, q) of elements in P such that

$$\neg(p \leqq q) \text{ and } ((\forall r \leqq p)(\exists f(r) \in P)(f(r) \leqq r \text{ and } f(r) \leqq q)) \quad \cdots①$$

This kind of pair (p, q) is called a " witness " of the non-separativity of $(P, \leqq)$. Here, let

$W(P) = \{(p, q) \mid (p, q)$ is a witness of the non-separativity of $(P, \leqq)\}$. ⊣

Using this terminology, we can propose a general separativization technique for an arbitrary p.o.s..

≪Tactics 2≫ **Canonical Separativization on a p.o.s.**

Let $(P, \leqq)$ be an arbitrary p.o.s..

A lgorithm 2

```
「
Step 1. If $(P, \leqq)$ has any node having only one immediate successor,
        Then, do the trivial separativization on $(P, \leqq)$ to obtain $(P', \leqq')$.
        Else, take $(P, \leqq) = (P', \leqq')$.
Step 2. Try to find a witness $(p_n, q_n)$ of non-separativity of $P_n$, where $P_0 = P'$.
Step 3. If there is no witness $(p_n, q_n)$ of non-separativity of $P_n$,
        Then, stop.
        Else, grow a beard at the node $p_n$ to obtain a new tree $P_{n+1}$ and go to Step 2.
」
```

If there is an algorithm to represent $(P, \leqq)$, then this algorithm becomes concrete and it is obvious that the resulting p.o.s. becomes separative. Though this algorithm can be applied to an arbitrary p.o.s., the resulting p.o.s. grows heavy beards in general. If there is a little restriction on the condition of a p.o.s., we can give a lighter separativization algorithm.

§4. REGULAR SEPARATIVIZATION AND NORMAL SEPARATIVIZATION

Definition 4-1. Let $(P, \leqq)$ be an arbitrary p.o.s. Then,
$(P, \leqq)$ is "(downward-)hierarchical" iff $(\forall p \in P)(\{q \in P \mid p \leqq q\}$ is finite $)$. ⊣

Suppose $(P, \leqq)$ is hierarchical. Then, for any $p < q$ in P, the number of paths from p to q are finite and, for any path *l* from p to q, the number of nodes on *l* between p and q are finite, too. As the result, for any $p \in P$, p is not a limit point in P, though P itself may be an infinite set. In addition, P "begins" from "maximal" (or "topmost" or "root") elements in the ordering $\leqq$ of $(P, \leqq)$, in the sense that

$r \in P$ is a maximal element in P iff $\neg(\exists s \in P)(r < s)$.

Remark: If we define the notion of the "hierarchical" by

"$(\forall p \in P)(\{q \in P \mid q \leqq p\}$ is finite$)$" (upward-hierarchical),

then P begins by "minimal" elements. Throughout this paper, we employ the downward definition. The following argument becomes essentially the same by suitably inverting the ordering, if we employ the upward definition.

To tell the truth, we can employ a more general notion of the "bi-hierarchicalness" defined by

$(P, \leqq)$ is "bi-hierarchicalness" iff $(\forall p \leqq \forall q \in P)(\{r \in P \mid p \leqq r \leqq q\}$ is finite $)$,

instead of the "hierarchicalness", and we can continue the following arguments with a slight modification. However, many partially ordered structures used in the field of computer science belong to the above category of hierarchicalness, so the hierarchicalness is general enough, we think. In the following, we use the terminology of "h.p.o.s." for the abbreviation of "hierarchical p.o.s.". ⊣

Perhaps, a famous h.p.o.s. is a (downward) $\leqq\omega$-tree. (A tree T is a "$\leqq\omega$-tree" iff the height $h(T)$ of T is at most ω, where $h(T) = \max\{\text{length}(l) \mid l$ is a branch of $T\}$.) Here, the notion of h.p.o.s. is strictly more general than the notion of $\leqq\omega$-tree, because it has a (downward) merging point.

Now, as far as h.p.o.s. concerns, we can give a lighter separativization algorithm by using the following concept.

Definition 4-2. Let $(P, \leqq)$ be a h.p.o.s. and let $W(P)$ be a witness of the non-separativity of $(P, \leqq)$.

1. Let $(p,q) \in W(P)$. Then, the non-empty set

$T(p,q) = \{f(p) \mid f(p)$ is a merging point satisfying the condition ① $\}$

is called "the set of targets" selected by (p,q).

2. Let $f(p) \in T(p,q)$ be a target selected by (p,q). Then, the non-empty set

$B(f(p)) = \{s \in P \mid s$ is an immediate predecessor of $f(p)$ on a path between p and $f(p)\}$.

is called "the bearding points" of $f(p)$. ⊣

Using these notions, we can define a general separativization technique. This technique is applicable to an arbitrary h.p.o.s..

≪Tactics 3≫ **Standard Separativization on a H.P.O.S.**

Let $(P, \leqq)$ be an arbitrary h.p.o.s..

```
「
Step 1. If $(P, \leqq)$ has any node having only one immediate successor,
          Then, do the trivial separativization on $(P, \leqq)$ to obtain $(P', \leqq')$.
          Else, take $(P, \leqq) = (P', \leqq')$.
Step 2. Check the separativity of $(P', \leqq')$.
          If $(P', \leqq')$ is separative,
          Then, stop.
          Else, choose a witness $(p,q) \in W(P')$.
                For this $(p,q)$, choose a target $f(p) \in T(p,q)$.
                For this $f(p)$, choose a bearding point $s \in B(f(p))$.
Step 3. Grow a beard at $s$ to obtain $(P'', \leqq'')$.
(As the result, $(p,q)$ is no more the witness of the non-separativity of $(P'', \leqq'')$.)
          and go to Step 2.
」
```

Thus, if there is a representing algorithm of a h.p.o.s. $(P, \leqq)$, the above standard separativization becomes concrete. Here, depending on the selecting criterion of $((p,q), f(p), s)$, there are many techniques which realize the standard separativization. One typical criterion use the following concept.

Definition 4-3. Let $(P, \leqq)$ be a h.p.o.s. and let $p \in P$ be an arbitrary element. Then, we can define a rank for p so that

$$\mathrm{rank}(p) = \max\{\mathrm{length}(l) \mid l \text{ is a path from a maximal element to } p\},$$

where $\mathrm{length}(l) =$ " the number of nodes on l " -1. ┤

Thus, any topmost element in P has rank 0 and any element in P corresponds to a natural number as a rank. (If P is bi-hierarchical, then any element in P corresponds to an integer by suitably taking starting points of rank 0 in P.) By utilizing the notion of the rank, we may efficiently embody the standard separativization in the following form.

Algorithm 4. **Regular Separativization**

```
「
At the process of $W(P')$,
    Choose a $(p,q)$ such that
        1)  rank(q) is minimum
        and, concerning p, our idea is that we would like to make rank(p) as large as possible.
```

So, practically, we decide that

2) rank(p) is as large as “ a possible computation ” in the representing algorithm of $(P, \leqq)$

At the process of $T(p,q)$

Choose a $f(p)$ such that

3) rank($f(p)$) is minimum

At the process of $B(f(p))$

Choose a s such that

4) rank(s) is minimum

」

The effectiveness of the regular separativization compared with the standard separativization in general can be checked easily. The lower point (the higher rank) a beard grows, the more witnesses of non-separativity it kills (at the same time).

This criterion can be applied to any h.p.o.s.. However, if a h.p.o.s. is finite, we can employ an ultimately effective tactic. Here, remember the fact that every finite p.o.s. is always hierarchical. For this reason, the next algorithm is of worth being considered independently.

A lgorithm 5. **Normal Separativization on a Finite P.O.S.**

「

At the process of $W(P')$,

Choose a (p,q) such that

1) rank(q) is minimum

and

2) rank(p) is maximum

At the process of $T(p,q)$,

Choose a $f(p)$ such that

3) rank($f(p)$) is minimum

(So, $f(p)$ becomes the immediate successor of p, because rank(p) is maximum.)

At the process of $B(f(p))$,

Choose a s such that

4) $s = p$. (This is a direct consequence of 3).)

」

As the consequence of this algorithm, the set $B(f(p))$ of bearding points of $f(p)$ is uniquely determined as a singleton $\{p\}$. So, this algorithm becomes not only a special case of the Standard Separativization but also a special case of the Canonical Separativization.

§ 5. BOOLEAN COMPLEXITY OF THE TYPE (n,m)

In the following, let's stick our attention only to the case that a p.o.s. P is finite. From Lemma 2-5, it is easy to check that, for any separative p.o.s. P,

e maps the set of all atoms in P 1:1, onto the set of all atoms in r.o.(P), ⋯②

where
$e : (P, \leqq) \to$ (r.o.(P),$\subseteq$)
is the canonical embedding defined in Lemma 2-5. More generally, the next result holds.

Proposition 5-1. Let $(P, \leqq)$ be a finite separative p.o.s. and let $e : P \to$ r.o.$(P) - \{\mathbb{O}\}$ be the canonical embedding defined in Lemma 2-5. Then, for $p \in P$,

$$e(p) = \vee \{e(q) \mid q \leqq p \text{ and } q \text{ is an atom in } P\} .$$

where $\vee$ is the Boolean operation.

Proof: Easy. (See, for example, [2], pp95-97.) □

By combining this result and our result, we reach a new mathematical insight that

「any finite p.o.s. $(P, \leqq)$ can be measured by the " Boolean complexity " $\mathbb{B}$ based on the normal separativization,」 ⋯【1】

because the target $\mathbb{B}$ is uniquely determined by the algorithm.

Here, as a direct consequence of Proposition 5-1, we notice that

Corollary 5-2. Let $(P, \leqq)$ be a finite p.o.s. whose topmost nodes consist of more than one elements $\{p_1, p_2, \cdots\}$ and let $\mathbb{B}$ be the Boolean complexity of P. By putting a " dummy " maximum element t to P, we obtain a new p.o.s. $(P \cup \{t\}, \leqq)$ such that
$p_1 \leqq t$, $p_2 \leqq t$, $\cdots$.
Then, $(P \cup \{t\}, \leqq)$ has the same Boolean complexity $\mathbb{B}$ as $(P, \leqq)$. Moreover, t corresponds

to the maximum element $\mathbb{1}$ in $\mathbb{B}$ by the canonical embedding e.

Proof: Check the process of the Normal Separativization. Then, we notice that t does not influence the Normal Separativization algorithm. So, by Proposition 5-1, we get the required result. □

Now, from a viewpoint of Boolean complexity, a more interesting fact is;

Corollary 5-3. Let $(P, \leqq)$ be a finite p.o.s. and let $e : P \to \mathbb{B}$ be the canonical embedding via the normal separativization. Then, an atom $\mathbb{a}$ in $\mathbb{B}$ corresponds either an atom in P or a beard grown by the normal separativization.

Proof: Easy. □

So, we can refine the above result【1】by distinguishing the atoms generated by growing beards from the atoms corresponding to the original p.o.s. $(P, \leqq)$ in the following form.

「Any finite p.o.s. $(P, \leqq)$ has a unique Boolean complexity of the type (n,m), where n is the number of atoms corresponding to the original P and m is the number of beards generated by the algorithm, where n+m is the number of atoms in the target Boolean algebra $\mathbb{B}$.」

…【2】

Example 5-4.

(1) a finite p.o.s. illustrated on the left side of the figure 10 becomes a p.o.s. illustrated on the right side of the figure 10 via the normal separativization.

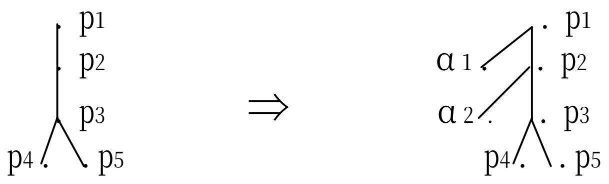

Figure 10

So, the Boolean complexity is (2,2).

(2) a finite p.o.s. illustrated on the left side of the figure 11 becomes a p.o.s. illustrated on the right side of the figure 11 via the normal separativization.

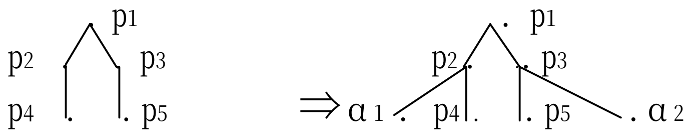

Figure 11

So, the Boolean complexity is (2,2). ⊣

Above examples typically show that two different finite p.o.s. may have the same Boolean complexity (n,m). Thus, we have the conclusion that:

Theorem 5-5. The set of all finite p.o.s.s can be classified to the class of Boolean type (n,m) for each natural numbers n and m, via the normal separativization.

Proof: Easy. □

§ 6. CONCLUSION

By proposing several natural separativization algorithms in § 3 and § 4, we finish to prove Theorem 1-1 via Lemma 2-5. Among several algorithms, the most interesting is the Normal Separativization algorithm, which can be applied to any finite p.o.s. By using this algorithm, we can classify the set of all finite p.o.s.s to mathematically new concepts called “Boolean Complexity of the type (n,m)” where n and m are natural numbers.

References

[1] Jech,T.J. , *Set Theory*, Academic Press, 1978.

[2] Takeuti,G. *Introduction to Modern Set Theory*,(in Japanese), Nihon Hyouron-sha, 1971, revised edition 1989.

[3] Yamaguchi,J., LIFE-Ⅲ[5] From a Viewpoint of Situation Theory (2) (in Japanese) *7th Conference Proceedings Japan Soc.Software Sci.& Tech.* (1990), 369-372.

[4] Yamaguchi,J., LIFE-Ⅲ[5] From a Viewpoint of Situation Theory (3) (in Japanese) *JSAI, SIG-FAI-9101*(1991), 21-29.

[5] Yamaguchi,J., LIFE-Ⅲ[5] From a Viewpoint of Situation Theory (4) (in Japanese) *8th Conference Proceedings Japan Soc.Software Sci.& Tech.* (1991), 453-456.