Comprehensive representation of feasible combinations of alternatives for dynamic production planning using Zero-Suppressed Binary Decision Diagram

Keita TAKAHASHI*, Masahiko ONOSATO* and Fumiki TANAKA*
*Graduate School of Information Science and Technology, Hokkaido University
N14W9, Sapporo, Hokkaido 060-0814, Japan
E-mail: ktakahashi@dse.ssi.ist.hokudai.ac.jp

Received 28 February 2014

Abstract
The goal of the production-planning problem is to find the optimal solution from combinations that satisfy constraints. Traditionally, production-planning problems have been treated separately, due to the problem size and complex constraints, whereas the constraints in production planning spread to the entirety of production planning. For example, when the dynamic changes as machine breakdowns and new machine addition occur, a production plan should be modified with new constraints caused by these dynamic changes. Therefore, it is required to represent a set of solution candidates that satisfy constraints in production planning. Zero-Suppressed Binary Decision Diagram (ZDD) is a directed graph representation of Boolean function and can efficiently represent a set of combinations. This paper describes the validity of application of the ZDD to production planning to represent a set of solution candidates. Experimental results applied to a sample-planning problem demonstrate that the ZDD is efficiently used for production planning and for dealing with dynamic changes.

Key words : Dynamic production planning, Production-planning domains, Production-planning attributes, ZDD

1. Introduction

In manufacturing sites, various changes – such as machine breakdowns, unplanned orders and cancellations – happen frequently and unexpectedly (Phanden et al., 2011). These dynamic changes often cause the manufacturing execution plans prepared in advance to be inaccurate or infeasible. To deal with this problem, production managers should modify certain parts of production planning. In this research, changeable parts in production planning are defined as production-planning attributes. Each production-planning attribute has a relationship with other attributes. A change of one attribute may influence other attributes and cause the propagation of attribute changes. Since these influences could spread to the whole of production planning, constraints among production-planning attributes should be carefully considered in production planning for dynamic changes. Traditional approaches that treat production planning separately, due to the problem size, can result in quasi-optimal solutions such as the integration of process planning and scheduling problems (Rajabnasab and Mansour, 2001) (Wong et al., 2006) (Shao et al., 2009) and the layout problem in cellular manufacturing (Kia et al., 2013). However, the works of research may be not able to address these dynamic changes due to the spread. For this reason, in this research, production planning is defined as comprehensive.

The comprehensive solution space for production planning is quite large and combinatorial explosion can often occur. However, the solution space includes useless combinations that do not satisfy constraints. These combinations should be removed before searching for the solutions.

In this paper, we describe the representation of solution candidates that satisfy constraints for production planning. For the realization thereof, we use the Zero-Suppressed Binary Decision Diagram (ZDD) (Minato, 1993, 2001). The ZDD is a special type of Binary Decision Diagram (BDD) (Bryant, 1986) used to represent a binary decision tree in
graph form, and is suitable for representing and processing combinatorial set data. ZDDs can be effectively used for analyzing a space of solutions of the problem with incrementally added constraints.

This paper firstly introduces the definitions of production-planning domains and attributes. Next we describe a combinatorial problem in production planning, followed by the introduction of ZDD. Moreover, we apply the ZDD to production planning. Finally, we make a conclusion with a brief summary and a mention of future works.

Nomenclature

In this paper, notations are defined as follows.

\( i \): index of machine type \((i, k = 1, \ldots, I)\)

\( j \): index of machine type instance \((j = 1, \ldots, J)\)

\( k \): index of feasible combination of operation type and machine type \((k = 1, \ldots, K)\)

\( r \): index of process sequence \((r = 1, \ldots, R)\)

\( s \): index of part type \((s = 1, \ldots, S)\)

\( w, \alpha, \beta, \gamma \): index of operation \((w, \alpha, \beta, \gamma = 1, \ldots, W)\)

\( F \): the maximum number of installable machine instances in a factory

\( M \): a set of all combinations that consist of variables of all machine type instances

\( M_i \): a set of combinations of instances of machine type \( i \) represented by ZDDs \( (M_i \in M) \)

\( M_{ij} \): variable of instance \( j \) for machine type \( i \) \((M_{ij} \in M_i)\)

\( M_{iwr} \): variable of machine type \( i \) for operation \( w \) performed on the \( r \) th sequence for part \( s \)

\( O_{wr} \): variable of operation \( w \) performed on the \( r \) th sequence for part \( s \)

\( X \): a set of combinations of production plans

\( X^s \): a set of combinations of process sequences for part \( s \) represented by ZDDs

\( |X^s| \): the number of combinations contained in \( X^s \)

\( X^s_w \): a combination of process sequence for part \( s \) represented by ZDDs \( (X^s_w \in X^s) \)

\( Z^{swt} \): a set of combinations of feasible machine types to process \( O_{wr} \) and \( O_{wr}^s \) represented by ZDDs

\( |Z^{swt}| \): the number of combinations contained in \( Z^{swt} \) \((|Z^{swt}| = K)\)

\( Z^k_{swt} \): a combination of a feasible machine type to process \( O_{wr}^s \) and \( O_{wr}^s \) represented by ZDDs \( (Z^k_{swt} \in Z^{swt}) \)

2. Production-planning domains and attributes

In this research, we categorize production planning into four domains.

1. Process planning; creation of product information including operation processes.
2. Resource planning; arrangement of equipment and workers for manufacturing execution.
3. Execution planning; assignment of jobs to equipment and creation of production schedule.
4. Order management; decision of boundary conditions in production processes.

Each domain has production-planning attributes. Table 1 shows the production-planning attributes. As shown in Table 1, 23 production-planning attributes are defined.

In process planning, we defined eight attributes. The aim of product redesign (1.1) is to modify product design. Grouping of the process plan (1.2) is used to merge identical processes in process plans for different products. The merged processes are treated as one process. Production process (1.4) means the product can be achieved through different combinations of operations. Process sequence (1.5) allows changing to different sequences of operations. Different machine type (1.6), tool type (1.7) and tool approach direction (TAD) (1.8) may exist for each operation. Attributes of (1.4), (1.5), (1.6), (1.7) and (1.8) are called as process planning flexibilities (K. Lian et al., 2012).

In resource planning, four attributes are considered. The domains are categorized into capacity and configuration. In regards to capacity, there is quantity of resource (2.1) and available time (2.2). The quantity of resource allows modification of the number of resources, such as tools and machines, in order to allocate the resources on a production site. The available time allows extending or shortening of the resource-available time. Then, resource allocation (2.3) and control rule (2.4) are considered as the configuration. Resource allocation means setting up resources as machines, tools and so on. The control rule is to modify operational methods for AGV and automated warehouses.

Attributes in execution planning are principally related to the assignment of jobs to equipment and are classified into three categories. First, attributes related to jobs are considered, which are input order to each facility (3.1), an
Assignment to other facilities (3.2) and lots splitting (3.3). Second, there are three kinds of attributes in regard to scheduling. They are modification of objective function (3.4), change of dispatching rule (3.5) and alleviation of constraints (3.6). Finally, two attributes of inventory are considered, which are inventory placement (3.7) and quantity of inventory (3.8).

Attributes in this domain are related to production load, which are an extension of the due date (4.1), order cancellation (4.2) and outsourcing (4.3) to reduce load in a production site. Production-planning attributes as defined above have relations to other attributes. In the following sections, interactions between the production-planning attributes are described in detail.

### 3. Combinatorial problem in production planning

#### 3.1 Interactions between attributes

The goal of the production-planning problem is to find the optimal solution from a set of combinations that satisfy constraints. A combination in production planning consists of selected alternatives from each production-planning attribute. Figure 1 shows a combinatorial problem of alternatives and an influence of a dynamic change, which represents a solution space of production planning as an example. In Fig.1, \( S_i, S_j \) and \( S_k \) signify production-planning attributes. Any alternative in \( S_i \) is defined as \( S_i(p) = S_i(p \in S_i) \) and a set of combinations \( S_j(p \subseteq S_j) \) in \( S_i \) in \( S_j \). The relationship between \( S_i(p) \) and alternatives in \( S_j \) can be represented as follows.

\[
f^{i\rightarrow j}(S_i(p)) = S_j
\]  

Furthermore, the relation between \( S_i \) and \( S_j \) can be defined as Eq. (2).

\[
f^{i\rightarrow j}(S_i(p \subseteq S_i)) = \cup f^{i\rightarrow j}(S_i(p))
\]
In Eq. (2), the right hand side expresses a union of alternatives in $S_j$ that have a relation with $S_{i,p}$. Each alternative has such relationships. A change of alternative may affect other alternatives in other attributes.

Moreover, a dynamic change makes some current feasible combinations of production plans infeasible. If a dynamic change affects an alternative in $S_j$ in Fig. 1 and then the alternative cannot be selected anymore due to the dynamic change, combinations of production plans including the alternative can be infeasible. If the production plan carried out in a factory includes such alternatives, the production plan should be modified. To address such dynamic changes, another alternative in the attributes or in other attributes may be selected. Then, the modification will influence other alternatives in other attributes as well. Such influence may spread to the entirety of production planning. Therefore, production planning is treated comprehensively in this research.

### 3.2 Reduced set of combinations

As shown in Fig. 1, there are some alternative combinations that are not related to $S_{i,p}$. In practical production planning problems, these combinations exist. Such combinations do not need to be considered. Therefore, these combinations should be removed from the solution space. In other words, solution candidates should represent only...
feasible combinations.

For the realization of the solution candidates, we use the Zero-Suppressed Binary Decision Diagram (ZDD) proposed by Minato (Minato, 1993), which is typically known as a particularly efficient representation of sparse combination sets. In the following chapters, we give an overview of ZDDs and describe the standard operations supported by ZDDs, before discussing our ZDD encoding of the production planning in more detail.

4. Zero-Suppressed Binary Decision Diagram

A Binary Decision Diagram (BDD) is a directed graph representation of a Boolean function. BDDs have the two terminal nodes, called 0-terminal node and 1-terminal node, and many decision nodes with the two edges, called 0-edge and 1-edge. In order to represent a Boolean function efficiently, the following reduction rules are usually applied (Minato, 1993).

1. Share all the equivalent sub-graphs (Fig.2 (a)).
2. Eliminate all the redundant nodes whose two edges point to the same node (Fig.2 (b)).

A ZDD is a variant of BDD that can efficiently represent a set of combinations. In a ZDD, a path from the root node to the 0-terminal node represents that such a combination does not exist in a set of combinations. The reduction rules of ZDDs, which are slightly different from those of BDDs, are as follows (Minato, 1993).

1. Share equivalent nodes as well as ordinary BDDs.
2. Delete all nodes of which 1-edge directly points to the 0-terminal node, and jump through to the 0-edge’s destination (Fig.2 (c)).

Consider Boolean function \((a\bar{b}c) \lor (a\bar{b}\bar{c})\), which can be equivalently represented by using a set of combinations({ac}, {b}). Figure 3 shows a BDD and ZDD representation for this function and set of combinations. In a tree, a node represents a variable. A ZDD is more concise than a BDD. If a variable never appears within any elements in a set of combinations, a node representing the variable is removed from the ZDD.

There are some basic arithmetic operations for ZDD: addition (+), subtraction (−), multiplication (×), division (/) and modulo (%).

- **Addition**: \(F + G\) means the union of \(F\) and \(G\).
  Ex) when \(F = \{a, b\}\) and \(G = \{c\}\), \(F + G = \{a, b, c\}\)

- **Subtraction**: \(F - G\) means difference between \(F\) and \(G\).
  Ex) when \(F = \{a, b, c\}\) and \(G = \{a, c\}\), \(F - G = \{b\}\)

- **Multiplication**: \(F \times G\) generates all possible concatenations of two items in respective \(F\) and \(G\).
  Ex) when \(F = \{ab, b, c\}\) and \(G = \{ab, 1\}\), \(F \times G = (ab \times ab) + (ab \times 1) + (b \times ab) + (b \times 1) + (c \times ab) + (c \times 1)\)
  \[= ab + ab + ab + b + abc + c\]
  \[= \{ab, abc, b, c\}\]

  “1” in \(G\) includes only a “null” item.

- **Division**: \(F/v\) (quotient) is to extract items that include variable \(v\) and remove \(v\) from the extracted items.
  Ex) when \(F = \{ab, b, c\}\) and \(v = \{b\}\), \(F/v = \{a, 1\}\)

- **Modulo**: \(F\%v\) (remainder) is to extract items that do not include variable \(v\).
  Ex) when \(F = \{ab, b, c\}\) and \(v = \{b\}\), \(F\%v = \{c\}\)

Therefore, \(F = v \times (F/v) + (F\%v)\)

Moreover, we use the **Restrict** and **Permitsym** operation. This operation is described as follows.

- **Restrict** operation: \(F.\text{Restrict}(G)\) extracts the product terms from \(F\) such that the item combination is a superset of at least one item combination in \(G\).
  Ex) when \(F = \{b, ac, cd, abc, bcd\}\) and \(G = \{a, bc\}\), \(F.\text{Restrict}(G) = \{ac, abc, bcd\}\)

- **Permitsym** operation: \(F.\text{Permitsym}(n)\) filters the product terms in \(F\), each of which consists of less than or equal to \(n\) items.
  Ex) when \(F = \{b, ac, cd, abc, bcd\}\) and \(n = 1\), \(F.\text{Restrict}(n) = \{b\}\)

Formulations for production planning in ZDDs are defined by the above operations. Other operations refer to (S. Minato, 1993, 2001) for more details.
5. Application of ZDD to production planning
5.1 Range of use in production planning

In this section, the combinatorial problem of production planning is formulated by the ZDD to represent only feasible combinations that satisfy constraints. The problem is formulated under the following assumptions:

1. Four production-planning attributes in two production-planning domains are considered as the range of use for the ZDD: manufacturing method, process sequence, machine type and quantity of resource.
2. Each machine type has its instances with the same performance.
3. In a factory, the maximum number of installable machine instances is defined with respect to each machine type.
4. The sum of installable machine instances is less than or equal to \( F \).
5. Locations of machine instances are not considered.
6. Dynamic changes occur before carrying out the production.

The formulation of production planning on the ZDD is described in the following section.

5.2 Solution method

In creating the representation of a set of combinations by the ZDD, we take an approach that extends a set of combinations and adds constraints to it. The first step is to create a set of combinations that represent process sequences satisfying preference relations for a part type. Figure 4 shows feasible process sequences for part types. The number of part types \((P_i)\) is 8 and the number of operation types \((O_w)\) is 15. In Fig. 4, there are three types of nodes, namely, starting node, intermediate node, and ending node. Both the starting node and the ending node are dummy ones that indicate the beginning and the end of the manufacturing process of a part, respectively. An intermediate node represents an operation. The arrow that connects two nodes indicates the precedence relations between the nodes. Moreover, there are two types of brackets. The vertical bars (\(||\)) enclosing nodes mean alternative and the square brackets (\([|]\)) enclosing nodes express arbitrary sequence order. The alternative means \((O_1 \rightarrow O_2 \rightarrow O_3), (O_3 \rightarrow O_2 \rightarrow O_1), (O_3 \rightarrow O_1 \rightarrow O_2)\) is selectable in \(P_1\). The arbitrary sequence order means \(O_4\) in \(P_3\) can be the first process or the second process as well as \(O_7\). Equation (3) represents a feasible process sequence \(X^s_u\) for part type \(s\).

\[
X^s_u = O^s_{a_1} \times O^s_{b_2} \times \ldots \times O^s_{r_r}
\]  

(3)

For example, the first feasible process sequence \((u = 1)\) for \(P_1\) can be defined in the ZDD as \(X^s_1 = \{O^1_{11}O^1_{22}O^1_{33}\}\). When a set of \(X^s_u\) is \(X^s\) and \(|X^s| = U\), \(X^s\) is defined as follows.

\[
X^s = X^s_1 + X^s_2 + \ldots + X^s_U
\]  

(4)

\(X^s\) represents a set of combinations satisfying constraints about process sequences for part type \(s\). A set of feasible process sequences for \(P_1\) can be represented in the ZDD as \(X^1 = \{O^1_{11}O^1_{22}O^1_{33}, O^1_{31}O^1_{12}O^1_{43}, O^1_{31}O^1_{32}O^1_{33}\}\). The next step is to represent a set of combinations of feasible machine types to process each operation. Table 2 represents information of each operation and machine type. As shown in Table 2, the number of machine types is 8. When operation \(w\) performed on the \(r\) th sequence for part \(s\) can be processed on machine type \(i\), a ZDD representing the combination expresses itself as follows.

\[
Z^{s wr} = O^{s wr}_{w r} \times M^s_{i wr}
\]  

(5)

When a set of \(Z^{s wr}\) is \(Z^{swr}\) and \(|Z^{swr}| = K\), \(Z^{swr}\) is defined as follows.
In this case, a set of combinations of feasible machine types to process $O_1$ and $O_1$ can be represented as $Z^{111} = \{O_1^1 M_1^{111}, O_1^1 M_1^{111}, O_1^1 M_1^{111}\}$. $X^s$ in Eq. (4) is a ZDD that is a set of combinations of only feasible process sequences for part $s$. The next step is to create a ZDD that satisfies constraints of process sequences and feasible machine types by extending combinations in $X^s$ with $Z^{swr}$. By using Eqs. (5) and (6), the algorithm for adding feasible machine types for each operation to $X^s$ is as follows.

1. $w, r \leftarrow 1$;
2. while $w \leq W$ do
3. while $r \leq R$ do
4. $X^s \leftarrow (X^s / O_{wr}^s) \times Z^{swr} + (X^s \% O_{wr}^s)$;
5. $r \leftarrow r + 1$
6. end while
7. $r \leftarrow 1$, $w \leftarrow w + 1$;
8. end while

$X^s / O_{wr}^s$. In the fourth line of this algorithm extracts a set of combinations including $O_{wr}^s$ from $X^s$ and removes $O_{wr}^s$ from the extracted set. By multiplying the extracted ZDD by $Z^{swr}$, the ZDD changes to a set of combinations satisfying the constraint of machine types and operation types. Consider simplification equation on the fourth line of this algorithm, $X \leftarrow (X / a_1) \times Y + (X \% a_1)$. Figure 5 shows the procedures of this equation. When $X$ is $\{a_1, a_2, a_3\}$ and $Y$ is $\{a_1 b_1, a_1 b_2, a_2 b_1, a_2 b_2\}$, $(X / a_1)$ extracts combinations that include $a_1$ and remove $a_1$ from the extracted combinations, which is $\{a_2, a_2 b_1, a_2 b_2\}$. $(X / a_1) \times Y$ generates $\{a_1 a_2 b_1, a_1 a_2 b_2, a_2 a_1 b_1, a_2 a_1 b_2\}$. As the results, $X$ becomes $\{a_1 a_2 b_1, a_1 a_2 b_2, a_2 a_1 b_1, a_2 a_1 b_2, a_3\}$. For example, when combinations is $\{O_1^1 O_2^1 O_3^1 O_4^1, O_1^1 O_2^1 O_3^1 O_4^1, O_1^1 O_2^1 O_3^1 O_4^1\}$ in $X^1$ as feasible process sequences for $p^1$, $w = 1$ and $r = 1$, $X^1$ is changed to $\{O_1^1 O_2^1 O_3^1 O_4^1, O_1^1 O_2^1 O_3^1 O_4^1, O_1^1 O_2^1 O_3^1 O_4^1, O_1^1 O_2^1 O_3^1 O_4^1\}$ through the equation. The number of combinations in this example is changed to 5. When the number of parts is $S$, a set of all combinations satisfying constraints in process planning is defined as follows.
The ZDD $X$ in Eq. (7) represents a set of combinations of feasible process plans in process planning.

Each machine type has instances. The next step is to consider how many instances are installed in this factory. The number of these instances for each machine type is shown on the left in Table 2, as the column of “Instance”. The number of instances that can be installed in a factory is defined as a finite number. When the number of available instances of machine type $i$ is $T_i (1 \leq T_i \leq f)$, a set of all combinations $M_i$ that consist of variables of instances of machine type $i$ except “null” is defined as follows.

$$
M_i = (M_{i1} + 1) \times (M_{i2} + 1) \times \ldots \times (M_{iT_i} + 1) - 1
$$

(8)

If an operation in a process plan requires machine type $i$, the instance of machine type $i$ must be installed in the factory. If an operation in a process plan does not require machine type $i$, the instance must not be installed. This is a constraint between process planning and resource planning. The procedure to reflect this constraint to the ZDD $X$ is as follows.

1. $s,w,r,i \leftarrow 1$, ZDD $Y = \emptyset$;
2. while $s \leq S$ do
3. while $w \leq W$ do
4. while $r \leq R$ do
5. while $i \leq I$ do
6. $Y \leftarrow X.Restrict(M_i^{\text{swt}})$;
7. $X \leftarrow (Y - Y.Restrict(M_i)) \times M_i + X\%M_i^{\text{swt}} + Y.Restrict(M_i)$;
8. $i \leftarrow i + 1$;
9. end while
10. $i \leftarrow 1, r \leftarrow r + 1$;
11. end while
12. $r \leftarrow 1, w \leftarrow w + 1$;
13. end while
14. $w \leftarrow 1, s \leftarrow s + 1$;
15. end while

Fig. 5 Example in the fourth line of the algorithm for adding feasible machine types for each operation

$$
X = X^1 \times X^2 \times \ldots \times X^S
$$

(7)
The sixth line of this algorithm $Y \leftarrow X.\text{Restrict}(M^{SW}_{i})$ means that combinations of process plans that require $M^{SW}_{i}$ are extracted from $X$ and then stored to $Y$. If the number of the extracted combinations is 0, $Y$ is empty. In the seventh line of the algorithm, $Y \leftarrow Y.\text{Restrict}(M_{i})$ means that combinations of process plans including at least one combination in $M_{i}$ are removed from $Y$. The multiplication operation is applied to the left combinations to generate all possible concatenations of the left combinations and $M_{i}$. Finally, the union of left combinations, combinations of process plans that do not require $M^{SW}_{i}$ and removed combinations that have already included variables of instances of machine type $i$ are generated and the union is stored in $X$. For a simple example, consider that a combination of process plan for $P_{1}$ is $\{O_{11}^{1}, O_{22}^{1}, O_{33}^{1}, M_{11}^{1}, M_{21}^{1}, M_{31}^{1}\}$. When $s, w, r = 1$ and $i = 2$ in the algorithm, $Y$ in the sixth line is $\{O_{11}^{1}, O_{22}^{1}, O_{33}^{1}, M_{11}^{1}, M_{22}^{1}, M_{31}^{1}\}$. Since $Y.\text{Restrict}(M_{2})$ and $X \% M_{11}^{1}$ are empty in the seventh line, $Y = Y.\text{Restrict}(M_{2})\times M_{2}$ is $\{O_{11}^{1}, O_{22}^{1}, O_{33}^{1}, M_{11}^{1}, M_{22}^{1}, M_{31}^{1}\}$. The multiplication operation is applied to the left combinations to generate all possible concatenations of the left combinations and $M_{2}$. When $s = 1$, $w, r = 3$ and $i = 3$ in the algorithm, the combination changes to $\{O_{11}^{1}, O_{22}^{1}, O_{33}^{1}, M_{11}^{1}, M_{22}^{1}, M_{33}^{1}\}$. The output of this algorithm shows a set of combinations representing process plans and resource plans without the constraint of the maximum number of installable machine instances.

The maximum number of installable machine instances is defined as a finite number in a factory. If combinations of process plans and resource plans do not satisfy this constraint, the combinations are removed from the ZDD. The first step is to create a ZDD $M$ representing a set of all combinations that consist of variables of all machine type instances except “null”. The ZDD $M$ can be represented as Eq. (9).

$$M = (M_{1} + 1) \times (M_{2} + 1) \times \ldots \times (M_{i} + 1) - 1$$ (9)

When the maximum number of installable machine instances in a factory is $F$, a set of combinations that contain more than $F$ variables is defined as follows.

$$M_{NG} = M - M.\text{Permitsym}(F)$$ (10)

$M_{NG}$ is a set of combinations in which the number of variables representing instances is more than $F$ instances. For example, when $M = \{M_{11}M_{12}M_{13}, M_{11}M_{12}M_{13}, M_{12}M_{13}, M_{1}, M_{2}, M_{3}\}$ and $F = 1$ , the output of $M.\text{Permitsym}(F)$ is $\{M_{1}, M_{2}, M_{3}\}$. In this case, $M_{NG}$ is $\{M_{11}M_{12}M_{13}, M_{11}M_{12}M_{13}, M_{12}M_{13}, M_{1}, M_{2}, M_{3}\}$. Combinations including at least one combination in $M_{NG}$ should be removed from $X$. This manipulation is defined as Eq. (11).

$$X = X - X.\text{Restrict}(M_{NG})$$ (11)

In this equation, $X.\text{Restrict}(M_{NG})$ extracts combinations including at least one combination $M_{NG}$ and then the extracted combinations are subtracted from the ZDD $X$.

Finally, redundant combinations are considered, which is related to the same performance of instances of machine types. For example, when machine type $i$ has three instances and two instances out of three can be installed, three combinations consisting of two variables of instances of machine types can be defined as $\{M_{11}M_{12}, M_{11}M_{13}, M_{12}M_{13}\}$. In terms of machine performance, each combination is redundant. In such a case, a combination consisting of two variables that are earlier declared is kept and others are removed. In the case of the above example, $\{M_{11}M_{12}\}$ is kept and others are removed. These redundant combinations can be removed from the ZDD $X$ by the following algorithm.

1. ZDD $A, B, C = \emptyset, i = 1$;
2. while $i \leq I$ do
3.    count $\leftarrow T_{i}$;
4.    if $\text{count} > 1$ then
5. while count ≥ 1 do
6. \( A \leftarrow M_i.\text{Permitsym}(\text{count}) - M_i.\text{Permitsym}(\text{count} - 1); \)
7. if the number of combinations in \( A \) is more than 1, then
8. \( B \leftarrow X.\text{Restrict}(A); \)
9. \( X \leftarrow X - B; \)
10. \( C \leftarrow \) a combination consisting of earlier declared variables in \( A; \)
11. \( B \leftarrow B.\text{Restrict}(C); \)
12. end if
13. \( X \leftarrow X + B; \)
14. count \leftarrow count - 1;
15. end while
16. end if
17. \( i \leftarrow i + 1; \)
18. end while

In the sixth line of this algorithm, combinations of instances of machine types that consist of \textit{count} variables are extracted from \( M_i \) and stored in \( A \). If the number of combinations of instances in \( A \) is more than 1, the procedure in the eighth and ninth line extracts combinations in \( X \) that include at least one combination in \( A \) and then removes the extracted combinations from \( X \). Moreover, a combination of instances of machine types including earlier declared variables is extracted through the procedure of tenth line, for which the combination is stored in \( C \) in the algorithm. In the eleventh line, combinations of process plans and resource plans including a combination in \( C \) are extracted and stored in \( B \). Finally, the union of a set of reduced redundant combinations of process plans and resource plans \( B \) and a set of combinations of process plans and resource plans is generated. For example, consider redundant combinations of instances of machine type \( m_4 \). The number of instances of machine type \( m_4 \) is 2 as shown in Table 3, thus \( M_4 = \{M_{41}, M_{42}\} \) and \textit{count} is 2 in this algorithm. Firstly, \( \{M_{41}, M_{42}\} \) is extracted in the sixth line. Since the number of combinations in \( A \) is 1, ZDD \( X \) is not changed and \textit{count} is changed to 1 from 2. Next, \( \{M_{41}, M_{42}\} \) is extracted as well. The number of combinations in \( A \) is 2. This means redundant. If any combinations of process plans and resource plans require at least one instance of machine type \( m_4 \), these combinations are extracted from \( X \) and stored in \( B \). \( M_{41} \) is earlier declared as variable. Therefore, \( C = \{M_{41}\} \). Only combinations in \( B \) that include \( M_{41} \) in \( C \) are extracted from \( B \). Finally, the union of \( X \) and \( B \) is generated and stored in \( X \) in the 13th line. The output of this algorithm shows a set of combinations consisting of no redundant combinations in terms of machine performances and the ZDD \( X \) shows solution candidates that satisfy constraints in process planning and resource planning.

### 5.3 Influence due to dynamic change

Generally, two types of disturbances occur on the shop floor: internal and external. Internal disturbance involves machine breakdown, the arrival of new machines, tool failure, etc., whereas external disturbance involves order cancellation, rush order arrival and so on. These changes make the current production schedule infeasible.

We consider two types of dynamic changes in internal disturbance: new machine instance addition and machine breakdown. These changes are positive and negative in terms of the ZDD. In other words, they have a different impact on the ZDD, which is the addition of new variables respectively representing the instance of machine types and the removal of current feasible combinations including the variables that signify instances of machine breakdowns.

New machine instance addition means that the number of instances of any machine type is increased. In term of ZDDs, for new machine instance addition, it is necessary to generate a set of combinations of process plans and resource plans including the new variable of the instance. When the variable of new instance of machine type \( i \) is \( M_{iA} \), the algorithm to generate a ZDD that represents such combinations is as follows.

1. \( j \leftarrow 1 \)
2. while \( j \leq T_i \) do
3. \( X \leftarrow (X/M_{ij}) + (X\%M_{ij}); \)
4. \( j \leftarrow j + 1; \)
5. end while
6. \( M_i \leftarrow (M_i + 1) \times (M_{ia} + 1) - 1; \)
7. \( s, w, r \leftarrow 1; \)
8. ZDD \( Y = \emptyset \)
9. while \( s \leq S \) do
10. while \( w \leq W \) do
11. while \( r \leq R \) do
12. \( i \leftarrow i + 1; \)
13. \( r \leftarrow r + 1 \)
14. end while
15. \( r \leftarrow 1, w \leftarrow w + 1; \)
16. end while
17. \( w \leftarrow 1, s \leftarrow s + 1; \)
18. end while
19. Do the algorithms for the constraint of the maximum number of installable machine instances and redundant combinations;

The third line in this algorithm removes combinations of process plans and resource plans including \( M_{ij} \) from \( X \). The procedure of the fourth line means to generate a set of all combinations \( M_i \), except “null”, which consist of variables of instances of machine type \( i \) including new instances. The 16th and 17th line are equivalent as the procedure of algorithm between Eqs. (8) and (9). Finally, the algorithms for the maximum number of installable machine instances and redundant combinations are performed. For example, a new instance of machine type \( m_2 \) is added and \( M_2 \) is currently \( \{M_{21}\} \). First, the combinations of process plans and resource plans including \( M_{21} \) are removed from \( X \) by the third line. And then, \( M_2 \) changes to \( \{M_{21}, M_{22}, M_{21}, M_{22}\} \) through the fourth line. Procedures after this are performed as well as the algorithms between Eqs. (8) and (9).

Machine breakdown means that the number of instances of any machine type is decreased. For machine breakdown, combinations in \( X \) that include the variable of instances of the broken-down machine type are removed from \( X \). The procedure to deal with this dynamic change is easier than new machine instance addition. When an instance of machine type \( i \) is broken-down, the removal procedure is as follows.

\[
X = X - X.\text{Restrict}(M_{iT_i})
\] (12)

The manipulation involves the subtraction of combinations including the broken-down machine instance from \( X \). As a result of this, \( X \) represents only feasible combinations.

6. Sample problem for comprehensive representation

6.1 Experiment conditions and types of dynamic changes

In this experiment, the number of part types is 8. Process sequences for each part type are shown in Fig. 4. The number of operation type is 15. Feasible machine types to process each operation are defined as shown in Table 2. In terms of resource planning, eight machine type instances out of twelve can be installed in a factory. In this experiment, \( S, W, R \) and \( J \) are 8, 15, 4 and 3, respectively.

Moreover, two dynamic changes are considered, which are the arrival of a new instance of machine type \( m_2 \) and a breakdown of an instance of \( m_3 \).

All the algorithms implemented in this paper were developed in VSOP coded on Ruby (Minato, 2006) and run on a workstation with a Xeon E5-2643 3.30GHz and 256GB memory. The VSOP program is for calculating combinatorial item set data specified by symbolic expressions based on ZDD techniques.
6.2 Result and Discussion

Table 3 shows the result of a set of combinations of process plans in process planning. The column of the number of nodes means how many nodes are included in a ZDD. Number of combinations is the number of combinations of process plans in a ZDD. Computation time is the time to create a ZDD for a part and comprehensive process plans. The computation time of each ZDD is quite short. The number of combinations in the ZDD of comprehensive process plans is about $3.3 \times 10^{13}$ and indicates that these combinations in the ZDD are feasible in process planning. ZDDs can also enumerate these combinations. Table 4 shows the result of the ZDD considering instances of machine type without the constraint of the maximum number of installable machine instances and with that constraint in resource planning. As shown in Table 4, the ZDD is extended, when compared to process planning and the number of combinations of production planning in the ZDD is about $1.9 \times 10^{15}$. This manipulation takes the longest computation time, when compared to other manipulations. Figure 6 shows the change of the number of combinations in each phase. The size of each box is proportionate to the number of combinations in each phase. By adding the constraint of the maximum number of installable machine instances, the number of combinations in the ZDD is reduced to about 3.5%. The number of combinations in the ZDD satisfying all constraints is approximately $6.6 \times 10^{13}$. Thus, this represents solution candidates. In this experiment, the number of used variables is 4344. Thus, the number of all combinations is $2^{4344}$, when considering whole solution space. By using the ZDD, the number of combinations is reduced by 99.9%. The computation time to create the set satisfying constraints is approximately 5.79 sec including input files loading.

Table 5 shows the result of experiments after each dynamic change. Each computation time is the time to obtain a set of combinations satisfying the constraints of each dynamic change from the ZDD before each dynamic change. The input data for each dynamic change is the ZDD after applying the constraint of the maximum number of installable machine instances, as shown in Table 4. The new machine instance is for adding new variables. Therefore, the number of nodes and combinations is increased when compared to the results before dynamic changes. However, the number of nodes and combinations for the machine breakdown is greatly decreased. This is because $m_3$ has only one instance and some process sequences become infeasible, due to $m_3$ being necessary for operation type $O_8$. ZDDs can confirm how many combinations of process plans are feasible after the machine breakdown. The number of combinations of feasible process plans after machine breakdown was about $5.7 \times 10^{13}(572,778,086,400)$. Therefore, the number of

---

### Table 3  Results of a set of combinations of process plans

| Process plans of $P_i$ | Number of nodes | Number of combinations | Computation time |
|------------------------|-----------------|-----------------------|------------------|
| $P_1$                  | 27              | 36                    | 0.004[s]         |
| $P_2$                  | 54              | 75                    | 0.006[s]         |
| $P_3$                  | 31              | 36                    | 0.005[s]         |
| $P_4$                  | 37              | 90                    | 0.005[s]         |
| $P_5$                  | 45              | 75                    | 0.006[s]         |
| $P_6$                  | 14              | 12                    | 0.005[s]         |
| $P_7$                  | 16              | 16                    | 0.005[s]         |
| $P_8$                  | 56              | 261                   | 0.006[s]         |
| Comprehensive process plans | 280            | 32,878,483,200,000    | 0.005[s]         |

### Table 4  Results of a set of combinations in process and resource planning

| Process and Resource planning without constraints | Number of nodes | Number of combinations | Computation time |
|---------------------------------------------------|-----------------|-----------------------|------------------|
| 19,510                                            | 1.864,400,617,433,088 | 5.130[s]             |
| With the constraint of the maximum number of installable machine instances | 19,443           | 65,501,043,610,240    | 0.215[s]         |

### Table 5  Results of a set of combinations after dynamic change

| Dynamic change | Number of nodes | Number of combinations | Computation time |
|----------------|-----------------|-----------------------|------------------|
| New machine instance | 19560         | 78,227,619,692,032    | 2.483[s]         |
| Machine breakdown   | 4991           | 2,597,473,408,512     | 0.012[s]         |
feasible process plans is reduced to approximately 98.3% by the machine breakdown, when compared to the number of feasible process plans before the machine breakdown.

7. Conclusion

In this paper, we applied the ZDD to the production-planning problem for the representation of solution candidates that satisfy constraints. By using ZDD operations, constraints in production planning could be formulated. The experimental result shows that the number of combinations in the ZDD was reduced to about 3.9% by adding the constraint of the maximum number of installable machine instances and the computation time was 0.215 sec. The number of combinations was reduced by 99.9%, when compared to the number of all combinations. Two different dynamic changes were considered in terms of resource addition and reduction. For the resource addition as new machine instance addition, firstly, the ZDD was expanded to include the variable of new resources by using ZDD operations such as the \( \text{Restrict} \) operation and basic arithmetic operations. Finally, the combinations in the ZDD were reduced by applying the constraint of the maximum number of installable machine instances. For the resource reduction as a machine breakdown, the \( \text{Restrict} \) operation and subtraction were used to remove solution candidates that include the variable of broken-down machine type.

Since the ZDD could represent a solution candidate for process planning and resource planning, further applications for the ZDD in production planning are expected. For example, when a production manager wants a candidate, for which a specific operation type for \( r \)th sequence for a part type \( s \) is fixed, such candidates can be obtained by the ZDD operations. When the operation type is \( v \) (the variable is \( O^s_{vr} \)) and a set of combinations of process plans and resource plans is \( X \), a set of combinations of process plans and resource plans to which \( O^s_{vr} \) is fixed can be obtained by \( X' = X.\text{Restrict}(O^s_{vr}) \). If \( X' \) is empty as the result, this means that \( O^s_{vr} \) is a redundant variable and independent of solution candidates. Moreover, by representing solution candidates in ZDDs, we can know the number of solution candidates and the depth of ZDDs. As a solution method for production planning, meta-heuristics like genetic algorithms are often used. Such meta-heuristics need to set some parameters such as population size, crossover rate and mutation rate in the genetic algorithm, for example. ZDDs have the possibility to help adjust these parameters by using the number of solution candidates and the depth of ZDDs.

Future works will focus on the formulation of other constraints such as job assignment and job order, including more production-planning attributes. Moreover, an algorithm to find the quasi-optimal solution from the ZDD will be developed.

Acknowledgements

We are grateful to Prof. Shin-ichi Minato at the Graduate School of Information Science and Technology, Hokkaido University, for providing advice as well as for his permission to use VSOP coded on Ruby for this research.

References
Bryant, R. E., Graph-Based Algorithms for Boolean Function Manipulation, IEEE Transactions on Computers, Vol.C-35, No.8 (1986), pp.677-691.

Kia, R., Baboli, A., Javadian, N., Tavakkoli-Moghaddam, R., Kazemi, M. and Khorrami, J., Solving a group layout design model of a dynamic cellular manufacturing system with alternative process routings, lot splitting and flexible reconfiguration by simulated annealing, Computers & Operations Research, Vol.39, No.11 (2012) pp.2642-2685.

Lian, K., Zhang, C. and Shao, X., Optimization of process planning with various flexibilities using an imperialist competitive algorithm, The International Journal of Advanced Manufacturing Technology, Vol.59, No.5-8 (2012), pp.815-828.

Minato, S., Zero-Suppressed BDDs for Set Manipulation in Combinatorial Problems, 30th ACM/IEEE Design Automation Conference, 1993, pp.272-277.

Minato, S., Zero-Suppressed BDDs and Their applications, International Journal on Software Tools for Technology Transfer, Vol.3, No.2 (2001), pp.156-170.

Minato, S., VSOP (Valued-Sum-of-Products) Calculator for Knowledge Processing Based on Zero-Suppressed BDDs, Federation over the Web, Lecture Notes in Computer Science Volume 3847, 2006, pp.40-58.

Phanden, R. K., Jain, A. and Verma, R., Integration of process planning and scheduling: a state-of-the-art review, International Journal of Computer Integrated Manufacturing, Vol.24, No.6 (2011), pp.517-534.

Rajabasab, A. and Mansour, S., Dynamic flexible job shop scheduling with alternative process plans: an agent-based approach, The International Journal of Advanced Manufacturing Technology, Vol.54, No.9-12 (2011), pp.1091-1107.

Shao, X., Li, X., Gao, L. and Zhang, C., Integration of process planning and scheduling-A modified genetic algorithm-based approach, Computers & Operations Research, Vol.36, No.6 (2009), pp.2082-2096.

Wong, T. N., Leung, C. W., Mak, K. L. and Fung, R. Y. K., Integrated process planning and scheduling/rescheduling-an agent-based approach, International Journal of Production Research, Vol.44, No.18-19 (2006), pp.3627-3655.