SSI-DT79125038

博士論文

ゼロサプレス型 BDD を用いた
動的生産プランニングの統合的計画法に関する研究

高橋　啓太

2015 年 2 月

北海道大学 大学院情報科学研究科

システム情報科学専攻

本論文は北海道大学大学院情報科学研究科に

博士 (情報科学) 授与の要件として提出した博士論文である.

高橋　啓太

審査委員：　主査　小野里雅彦　教授

副査　金井　理　教授

田中　文基　准教授

ゼロサプレス型 BDD を用いた 動的生産プランニングの統合的計画法に関する研究*

高橋　啓太

概要

世界の市場はよりグローバルに，動的に，予測不可能に，そして顧客第一という状況へと変化している．そのような状況下で製造企業は企業間競争において，第一線を維持するために敏捷さ，再構築性，再構成、拡張性をもつ新たな生産システムを制御するための手法を導入しなければならない．加えて，それら手法は，動的変化に対応するために高いフレキシビリティと適応性の能力を兼ね備えて置く必要がある．生産システムの計画である生産プランニング分野では，その問題構造の複雑性から経験的に部分問題へと分割し，項目間の優先度を設定して決定することが多く行われてきた．しかし，市場の変化や動的変化に柔軟に対応するためには，順次段階的なアプローチではなく，統合的なアプローチにより計画に費やすスパンを短くし全体としての最適化を行うことが求められる．このような統合的なアプローチは，全体の決定変数を同時に考慮する必要があり，組合せ数が膨大になるという課題がある．本論文では，このような課題に資する提案として，最適化を行う前段階としてゼロサプレス型 BDD(Zero-Suppressed Binary Decision Diagram，以下 ZDD) を用い，制約を満たす組合せのみを表現する制約充足解空間を作成し，解空間を限定した上で遺伝的アルゴリズム (Genetic algorithm, 以下 GA) による最適化を行う方法を提案する．生産プランニング分野において，そのような研究はこれまでに無く，高い新規性をもつ．これら手法を例題に適用し，ZDD による統合的な生産プランニング問題の制約充足解空間表現，及び ZDD を探索対象とした GA による解探索の有効性を示す．

キーワード: 動的生産プランニング，選択性，統合的計画法，ZDD，遺伝的アルゴリズム

*北海道大学 大学院情報科学研究科 システム情報科学専攻 博士論文, SSI-DT79125038, 2015 年 2 月 16 日.

Comprehensive approach for dynamic production planning by using Zero-Suppressed Binary Decision Diagrams[†]

Keita Takahashi

Abstract

The global market has become increasingly dynamic, unpredictable and customer-driven. Manufacturing industries require new control methods for manufacturing system, which has agility, reconfigurability and extensibility, to maintain the front line in the competition among companies. Moreover, such a control method also needs flexibility to adapt dynamic changes, such as machine breakdowns, order cancellation and so on. Production planning is the planning phase in manufacturing system. In this study, key decisions for the creation of production plans are defined as production-planning attributes. Production-planning attributes correlate complexly in production-planning problems. Traditionally, the production-planning problem splits sub-problems based on experiences, because of the complexity. However, such approaches make solution space over-restricted and make it difficult to find a better solution. The objective of this paper is to propose a representation of combinations of alternatives in production-planning attributes by using Zero-Suppressed Binary Decision Diagrams (ZDDs) before the optimization. The ZDD represents only feasible combinations of alternatives that satisfy constraints in the production planning. Moreover, a genetic algorithm that solves production-planning problems with ZDDs is proposed. Experimental results applied to sample-planning problem demonstrate the proposed methods are efficiently used for production planning and for dealing with dynamic changes.

Keywords: Dynamic production planning, , production-planning attribute, Comprehensive solution space, ZDD, Genetic algorithm

[†]Doctoral Thesis, Division of Systems Science and Informatics, Graduate School of Information Science and Technology, Hokkaido University, SSI-DT79125038, February 16, 2015.

目 次

図 目 次

表 目 次

第1章　序論

1.1　研究背景

世界の市場はよりグローバルに，動的に，予測不可能に，そして顧客第一という状況へと変化している [1]．これは新規製品導入頻度の増加 (製品ライフサイクルの短命化) を導いている．つまり，製造企業は競争市場で生き残るために市場の変化に迅速に，素早く反応する必要がある．企業間競争の優位性は大部分がプロダクトミックスの動的な変化と需要パターンに対応するための素早い反応力に依存する．加えて，急な発注や機械故障といった生産現場における予期しない状況も同様に効率的に対処される必要がある．これらは，生産システムを制御するための手法に深く関わる．

従来の生産システムの制御手法は，開発された時代がコスト，リードタイム，在庫水準といった要因がより重要な問題であったため，それらの要因にウェイトをおいており，変化への反応力を活かすための設計されてはいない [1]．ビジネスライバルとの企業間競争において，第一線を維持するために製造企業は，敏捷さ，再構築性，再構成，拡張性の特徴をもつ新たな生産システムを制御するための手法を導入しなければいけない．

本研究では，生産システムの計画段階である生産プランニングにおける重要な決定要素を生産プランニングの選択性と呼ぶ．生産プランニング問題は，様々な選択性が相互に関連した問題構造をもち，選択性内の選択肢の組合わせにより計画が立案される．そのため，従来ではその複雑性から生産プランニング問題を経験的に部分問題に分割し，項目間の優先度を設定して決定することが多く行われてきた．例を挙げると，生産プランニングの部分問題である工程設計問題をはじめに解き，その解を用いて生産スケジューリング問題を解くといった順次段階的なアプローチがそれにあたる．このようなアプローチは解空間を過剰に抑制し，よりよい解選択を妨げる．また，部分問題の解が全体として制約を満たさず，再び部分問題を解くといった問題が発生する．

市場の変化や動的変化に柔軟に対応するためには，計画のスパンをより短くし，動的に

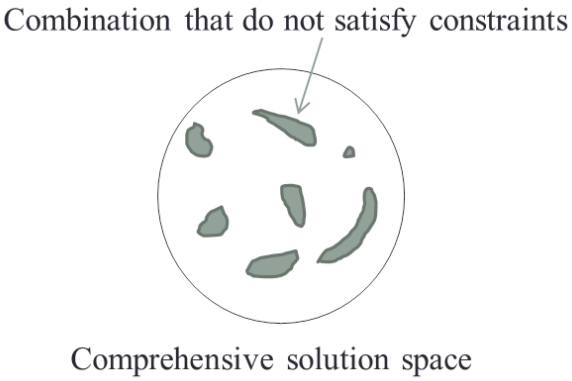


図 1.1: 生産プランニングの統合的解空間の例

生産プランニングを最適化する必要がある．しかしながら，生産プランニング全体の選択肢を決定変数とみたて，同時に考慮した数理モデルを作成し，全体を一括して最適化しようとすれば，考慮すべき組合わせ数が増大し，全体を最適化することは困難であるとされている [2]．生産プランニング問題を統合的なものとして扱う場合，部分問題内及び部分問題間の制約が複雑に絡みあう．そのため，選択肢の組合わせにおいては，図 1.1 に示すように，制約を満たさない組合わせが解空間内に存在する．つまり，最適化を行う前に，それら制約を満たさない組合わせを排除し，制約を充足した組合わせのみで構成された空間を表すことができれば，解探索領域を限定することができると考えられる．

二分決定グラフ (Binary Decision Diagram: BDD)[3] は，VLSI CAD の分野で大規模論理関数データの表現法として広く用いられている．BDD のなかでも，ゼロサプレス型 BDD(Zero-Suppressed Binary Decision Diagrams: ZDD)[4] と呼ばれるデータ構造は，大規模な組合わせ集合データを非明示的に列挙し，効率よく演算処理することができる [5]．また，ZDD では全ての解を過不足無く求められたか否かの論証や，求めた解集合に別の制約を加えた条件下での解の列挙，などといったことが効率行えるという利点がある．生産プランニングは，制約条件を充足する組合わせ集合から最適な組合わせを求める問題である．そのため，生産プランニングの領域においても ZDD の適用可能であると考えられる．また，動的変化により新たな制約が加えられたとしても，ZDD ではその制約を加え制約充足した組合わせを列挙可能であると考えられる．

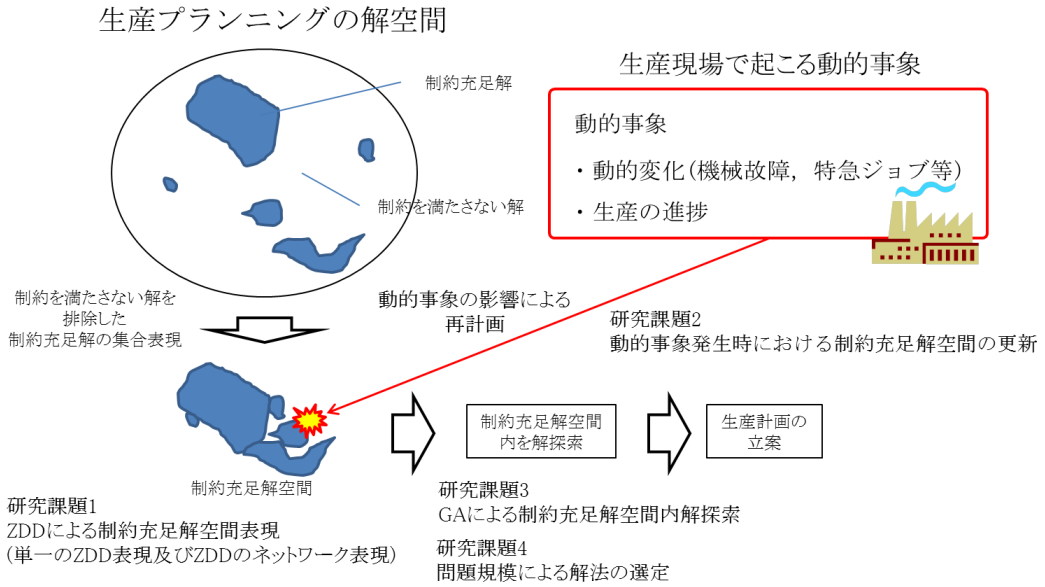


図 1.2: 本論文で提案する手法の枠組みの全体像と研究課題

1.2　研究目的

本論文の目的は，生産プランニングのフレームワークを分析・定式化し，動的変化に適応するために変更する選択性の抽出及び解法の選定を支援する手法の確立することである．図 1.2 は本論文で提案する手法の枠組みの全体像と，取り組むべき研究課題を表している．これを実現するために本論文では，以下の 4 項目について研究を行う．

1. ZDD による制約充足解空間表現の解法

   生産プランニングの問題構造は，各項目が複雑に絡みあう．そのため，従来では問題を部分問題へと分割されて解かれてきた．本論文では，生産プランニングの選択性を分析し，各選択性間の依存関係を明らかにし，解空間を統合的に扱うことの重要性について述べる．本論文では統合的な解空間を扱うために，ZDD を用いて解空間内の制約充足している組合せを 1 つの集合として表現する解法及び，各選択性の選択肢の組合せと選択性間の制約条件を個別の ZDD で管理し，連結したネットワーク表現による解法を提案する．

2. 動的事象に適応するための制約充足解空間の更新の解法

   動的生産プランニング問題では，生産の実績や機械故障などの動的事象が報告されるたびに解空間が変化する可能性がある．これら動的事象を ZDD に対する制約の

追加と捉え，生産実施中に制約充足解空間を表す ZDD を随時更新し，新たな制約条件が発生しても制約充足解空間を常に整合して保持する解法を提案する．

3. 制約充足解空間を探索対象とした解法

   制約充足解空間のみを探索対象とした遺伝的アルゴリズム (Genetic algorithm，以下 GA) を提案する．従来，GA では遺伝子操作の過程にて致死遺伝子が発生する．ZDD は制約充足解空間を表しているため，致死遺伝子は ZDD 上に存在しない解を意味する．提案する GA では，遺伝子操作の過程で致死遺伝子を発生させず，常に ZDD 上の解を探索する．

4. 問題規模による解法の選定

   生産プランニング問題は問題ごとに規模が異なる．そのため，各問題の規模に適した解法は異なる可能性をもつ．制約充足解空間を表す ZDD は，生産プランニングにおける選択肢の組合せを列挙することができ，また ZDD 内の組合せ数を知ることができる．本論文では，組合せ数に応じ GA 及び全探索の選定の検証を行う．

以上の項目に対して実行例を示すことで，有効性を示す．

1.3 本論文の構成

本論文の構成は以下に記す通りである．

第 2 章では，生産プランニングの選択性及び選択性間の関係を分析する．また，生産プランニングに関する研究を調査し，従来手法における問題点を挙げ，生産プランニングの解空間を統合的に扱うことの重要性について述べ，本論文の生産プランニングの分野での位置づけについて明らかにする．

第 3 章では，本論文で用いる BDD 及び ZDD について述べる．ZDD の例題として N クイーン問題を挙げ，制約充足問題への考え方及び，生産プランニング問題への適用の可能性について述べる．

第 4 章では，はじめに本論文で扱う動的生産プランニング問題について述べる．そしてその問題に対して，生産プランニングにおける制約充足解空間を単一の ZDD にて表現するための解法，及び制約充足解空間から解探索を行うために GA を提案する．また，生産実施中に発生する動的事象を ZDD に対する制約と扱い，制約充足解空間を常に整合し

て保持する方法を提案する．これら提案手法は，第 6 章の評価実験にてその有効性を検証する．

第 5 章では，より大規模な生産プランニング問題の制約充足解空間を表現するために，複数の ZDD の連結によるネットワーク表現を用いて制約充足解空間を表現する解法を提案する．ここでは，ZDD は選択性内および間にある制約をそれぞれ表し，それらの連結により選択性相互の制約を満たす解空間を構成する．第 6 章にて単一の ZDD との比較を行い，その有効性を示す．

第 6 章では，例題に対して各提案する手法を適用し，その有効性を示す．はじめに，第 4 章で提案した単一の ZDD における制約充足解空間表現についての検証を行い，生産プランニング問題に対する ZDD の有効性を示す．次にその限定された空間に対して提案する GA と全探索による結果を比較し，ZDD を対象とした GA 探索の有効性を示す．また，ZDD の組合せ数による解探索方法の選定について検証を行い，問題規模に応じた解法の選定について述べる．次に，生産実施中に発生する生産の実績と動的変化として機械故障を対象に，制約充足解空間を常に整合して保持する手法への検討を行い，その有効性を示す．また，単一の ZDD と ZDD のネットワーク表現を例題に適用し，結果の比較を行い，ZDD ネットワーク表現の有効性について述べる．

第 7 章では，本論文の結論と今後の課題について述べる．

第2章　生産プランニングでの本研究の位置付け

2.1　生産プランニングの計画と選択性

2.1.1　生産プランニングの概要

本論文での生産プランニング (production planning) とは，生産を始める以前に，製品設計・材料設計・生産設計に基づいて，製品を生産するための合理的な計画を立案することである．以下では，生産工学入門 [6] の第 8 章　生産における計画と準備を要約し，生産プランニングの役割を明らかにする．

生産プランニングでは，工数計画，負荷計画，工程計画，設備計画，生産スケジューリングなどから構成される．工数計画とは，製品の加工・組立に必要な工数を，工程別または部品別に必要な工数に置き換えることである．工数とは作業者の仕事量又はその単位をいい，仕事に要する人員と時間の積 (人・時 (1 人 1 時間)，人・日 (1 人 1 日)) で表す．

負荷計画は，生産計画によって定められた各製品の納期と生産量から，作業場事の作業量（負荷）と作業の発生時期を見積もり，使用可能な機械設備や作業員の能力を考慮して両者の調整を図った上，製品の完成時期などを推定することである．

工程計画とは，製品・部品設計段階で作成された設計情報を基にして，生産するための工程を計画し，効率よく最適な生産情報を作成することである．製品を得るためにはその素材・生産方法・生産機械・生産順序，さらに各種の作業，コストや納期に至るまでを考えに入れた上で生産情報の作成が要求される．この生産情報を能率よく，低コストの生産を進めることのできるものんでなければならない．このような生産の場におけるものの流れに伴う技術情報を処理して生産工程を計画することを工程計画と称する．工程計画では，加工法やその順序また，機械の選定などが決定される．

また，機械の選定にあたり，設備に関しての設備計画も行う必要がある．設備計画では，主に能力所要量計画 (Capacity Requirements Planning，以下 CRP とする) とレイアウト

設計に分けられる．CRP では，生産設備が持つ能力を基に製品/部品の生産可能性を確認する．レイアウト設計では，決められた品質・生産量・納期で (条件)，高い信頼性を持つ生産・運搬の最小の費用で行うために (目的)，作業者・物的設備・材料などを合理的に配置することである．生産工場の場合には，工場設備配置とも呼ばれることもある．

これら計画の決定に基づいて，生産スケジューリングが行われる．生産スケジューリングは，負荷計画によって平準化された一連の生産設備に割り当てられた一定期間内の作業 (ジョブ : job) に対して，作業者または設備別の作業開始，終了時刻，あるいは作業の順序を決定することである．これらは日程計画とも呼ばれる．日程計画が決まるとこれをガントチャートによって表示する．これは，横方向に日程，縦方向に設備機械をとり，表にしたものである．生産プランニングによりこれら計画が決定し，生産現場に生産の実施を促す．

2.1.2 生産プランニングの計画

本研究では，こうした既存の分類を参考に，生産プランニングを次の 4 つのカテゴリで整理を行う．

- 製品計画 : 製品と工程の情報を計画する．生産システムにおける “縦糸”．
- 資源計画 : 生産を実施するための機械や設備，用具，人員などの生産資源について，数や配置などを計画する．生産システムの “横糸”．
- 実施計画 : 製品計画 (“縦糸”) と資源計画 (“横糸”) を組合わせて生産を実施する計画 (“織物”) を生成する．
- 負荷計画 : オーダーの受注やキャンセル，生産の内外作といった生産プロセスにおける境界条件 (負荷) に関する計画を決定する．

また，計画における決定事項を本研究では生産プランニングの選択性と定義する．上記の各計画は複数の生産プランニングの選択性から構成されている．各選択性は選択肢をもち，それら選択肢を組合わせることで生産計画が作成される．それぞれの計画における選択性について，表 2.1 にもとづいて以下の順に述べていく．

2.1.3　生産プランニングの選択性

本研究では，製品計画において 8 つの選択性を定義した．製造する製品そのものの設計変更を行うことを製品設計 (表 2.1 の 1.1)，異なる製品間で工程を共通化することを工程集約 (1.2)，各製品を製造するために，放電加工や機械加工を用いることを製造方法 (1.3) と定義する．次に，代替工程を考える製造工程 (1.4)，順序が変更可能な場合の工程順序 (1.5)，加工可能な機械種別に関する選択性 (1.6) と工具種別に関する選択性 (1.7)，加工物へのアクセス方向などに関する選択性 (1.8) を定義した．これら選択性の内 (1.4)-(1.8) の 5 つは，フレキシブル工程設計の領域において，それぞれ工程フレキシビリティ，順序フレキシビリティ，機械フレキシビリティ，工具フレキシビリティ，工具アプローチ方向 (TAD) フレキシビリティと呼ばれ研究がなされている [7, 8, 9]．

資源計画は，生産を実施する設備や人員に関するものであり，大きくは能力の増減と構造の変化に分けることができる．能力の増減としては，加工機械の台数を増やしたり，無軌道搬送台車 (AGV) の台数を減らしたりする資源の数 (2.1) と，夜間や休日に操業することでの運転時間 (2.2) を定義する．構造変化としては，ライン構成を変えるといった資源の配置 (2.3) や，AGV の巡回ロジックの修正のような運用ルール (2.4) である．

実施計画においては生産資源に対して時間を考慮して製品を割り当てるスケジュール作成が中心となる．まず，製品を生産システムや各設備への投入順番を変更する順序入替 (3.1) や，設備への割付を変更する割付変更 (3.2)，負荷の平準化や生産性向上のためのロット分割・統合 (3.3) が選択性を定義する．次にスケジュールを作成する手法に関しては，目的関数 (3.4)，設備に割り当てる差立 (dispatching) ロジック (3.5)，計画問題における制約の追加，緩和を含めた制約条件 (3.6) である．最後に生産システムにおいては，仕掛在庫や安全在庫が生産システムの変動に対する感度を決定し，さらにコストに影響する要因として存在する．どこに在庫を置くかという在庫配置 (3.7) と各在庫における容量や計画水準を変更する在庫量 (3.8) を定義した．

最後の負荷計画における選択性とは，生産システムにおいて製造すべき負荷（オーダー）に対する選択性である．まず，オーダーの納期を短縮・延長することで，計画期間における負荷を増減する納期 (4.1)，受注したオーダーをキャンセルして負荷を減らすオーダー取消 (4.2)，オーダーの一部（または全部）を外部委託して行うことで内部において生産する負荷を削減する作業外注 (4.3) を定義した．ただし，これらの選択性は取引先など外

表 2.1: 生産プランニングの計画と選択性と影響関係

			製品計画								資源計画				実施計画								負荷計画		
計画	選択性	ID	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	2.1	2.2	2.3	2.4	3.1	3.2	3.3	3.4	3.5	3.6	3.7	3.8	4.1	4.2	4.3
製品計画	製品設計	1.1			*			*	*	*															
	工程集約	1.2					*	*	*	*					*	*	*				*	*			
	製造方法	1.3				*	*	*	*	*	*												*		*
	製造工程	1.4					*	*	*	*	*														
	工程順序	1.5											*		*	*					*	*			
	機械種別	1.6							*	*	*					*									
	工具種別	1.7						*		*															
	アクセス方向	1.8						*	*				*												
資源計画	資源の数	2.1						*				*	*	*		*					*	*	*		*
	運転時間	2.2																		*			*	*	
	資源の配置	2.3					*								*	*					*	*			
	運用ルール	2.4																*	*	*	*				
実施計画	順序入替	3.1											*				*						*		
	割付変更	3.2											*				*								
	ロット分割･統合	3.3												*	*					*		*			
	目的関数	3.4													*		*		*		*	*	*	*	*
	差立ロジック	3.5													*		*				*	*	*		
	制約条件	3.6													*		*		*		*	*	*		
	在庫配置	3.7												*		*				*		*			
	在庫量	3.8												*											
負荷計画	納期	4.1	*	*	*	*	*	*	*	*	*	*		*	*		*	*	*	*				*	*
	オーダー取消	4.2									*	*										*	*		*
	作業外注	4.3									*	*									*	*	*	*	

部との関係で決定される選択性であり，現実には適用が厳しく制限されることが多い．例えば，納期延長やオーダー取消には違約金支払いが発生し，さらの取引の継続に不利益になりうる．

本研究で定義した生産プランニングの選択性の間には影響関係が存在する．次節からは，その影響関係について詳しく述べる．

2.2 生産プランニングの選択性の波及関係

生産プランニング問題の目的は，制約を満たす組合せ集合から最適解を見つけることである．生産プランニングにおける組合せは，各選択性において選択された選択肢から構成される．図 2.1 は，選択肢の組合せ問題における生産プランニングの解空間を表現した例である．図 2.1 において，$\overline{S_i}, \overline{S_j}$ 及び $\overline{S_k}$ は生産プランニングの選択性を表す．$\overline{S_i}$ の選択肢 p は，$S_{i,p}(S_{i,p} \in \overline{S_i})$ と定義する．また，$\overline{S_j}$ 内の $S_{i,p}$ と組合せ可能な選択肢の集合を $S_j(S_j \subseteq \overline{S_j})$ とすると，$S_{i,p}$ と $\overline{S_j}$ 内の選択肢との関係は以下のように定義できる．

$$f^{i \to j}(S_{i,p}) = S_j \tag{2.1}$$

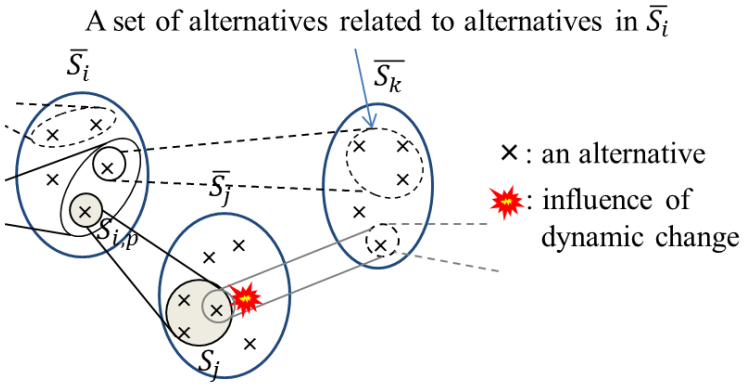


図 2.1: 生産プランニングにおける組合せ問題

したがって，$\overline{S_i}$ と $\overline{S_j}$ 間の関係は式 2.2 のように表すことができる．

$$f^{i\to j}(S_i \subset \overline{S_i}) = \cup f^{i\to j}(S_{i,p}) \tag{2.2}$$

式 2.2 にて，右辺は $S_{i,p}$ と関係を持つ $\overline{S_j}$ 内の選択肢の集合を表す．各選択肢は他の選択性内の選択肢に対してこのような関係をもつ．そのため，ある選択性の選択肢を変更することは，他の選択性内の選択肢に影響を与える可能性がある．

表 2.1 は，それら影響関係を定性的に分析した結果である．表中の行に記された選択性を A，番号で記された列の選択性を B とすると，A の選択肢を変更することで B に何らかの影響を与える場合，A 行 B 列に*を記している．例えば，製品設計は，製造方法，機械種別，工具種別，アクセス方向に影響を与えるとしている．本分析で記した影響関係は，同じ影響あり (*) であっても影響度の大小はある．また，直接の影響ではなく，$A \to B \to C$ のように，B を介して A が C に影響を与えることもあるが，ここでは直接の影響関係のみを対象としている．また，影響関係には $A, C \to B$ のような三者以上が関与する場合も考えられるが，ここでは二者の間の関係に単純化している．

一般的に生産プランニングは，製品計画 → 資源計画 → 実施計画の順で進行することが多く，影響関係においても上流側が下流側に影響を及ぼす割合が多くなっている．ただし，資源計画の資源の配置が工程順序の選択肢に影響を与えるなど，下流が上流を制約する場合も存在する．負荷計画は生産プランニングにおける基本条件を変えるものであり，幅広く影響をもつ傾向がある．

また，本論文では生産プランニングに必要なデータの有効性が，期間内は保証できる場合を「静的生産プランニング」と呼ぶ．一方，それらが変化しうる場合を「動的生産プラ

ンニング」と呼ぶ．計画期間内に発生する様々な変化を内的事象と外的事象の 2 つに分類できる [10]．内的事象は，機械故障，作業員の欠員，原材料の不良，工具の破損等が該当し，外的事象は，計画外の注文の発生や取消，顧客の仕様変更の要求，部品納品遅れなどがある．このような動的に発生する変化が原因で，現行の計画が実行不可能や不適切になることがある．

組合せ問題の観点では動的変化は，いくつかの実施可能な選択肢の組合せを実施不可能にする．もし，動的変化が発生し，図 2.1 の S_j のある選択肢がこの動的変化により直接影響を受け，選択不可能となった場合，この選択肢を含む実施可能であった組合せは実施不可能とある．加えて，生産計画を実施中では，生産計画の修正が必要となる．生産計画の修正にあたっては，他の選択肢が選択される．この選択により，他の選択性内の選択肢に影響を与える．このような影響は生産プランニング全体へと波及する可能性がある．

2.3 本研究の位置付け

従来は製品計画が行われた後に，資源計画が行われ，実施計画を行うといった形で別々に順次段階的に実行される．従来の生産プランニングでは，その計画問題の構造の複雑性から部分問題へ分割する．そして経験則に基づき各部分問題を順に解き，全体としての解を得る．

例えば，製品計画の分野では，工程設計における柔軟性 (フレキシビリティ) を考慮した研究が行われている．Lian らは選択性として，製造工程，機械種別，工具種別，アクセス方向を導入した製品計画に対して，メタヒューリスティクスの一つである Imperialist competitive algorithm[11] を用いて工程設計案を求める解法を提案している [7]．Qiao らは，機械種別，工具種別，アクセス方向を考慮した工程設計の研究を行い，解法として GA を用いた [12]．Ma らは，Qiao らと同様な工程設計問題に対してシミュレーテッドアニーリング (Simulated Annealing，以下 SA)[13] による解法を提案した [14]．また，Guo らは解法として粒子群最適化（Particle Searm Optiomization，以下 PSO）[15] を用いている [9]．動的変化として機械故障を取り扱う事例としては，テヘラニらが，工程順序，機械種別，工具種別，アクセス方法の 4 つの選択性を導入し，マルチエージェント [16] を用いた解法を提案している [17]．これら問題では，生産資源が無限であると考えられているが，資源は有限であり，実際の生産現場では生産資源が使用不可能な状況下にある可能性

がある．

資源計画の分野では，L Wang らが選択性として資源の数と資源の配置を考慮し，動的変化として機械故障を扱ったレイアウト設計問題に対して，GA を用いたレイアウトの再設計の方法を提案している [18]．同様にレイアウト問題に対して，M. A. El-Baz は，GA を用いた解法を，P. Jain らは，蟻コロニー最適化 (Ant Colony Optimization，以下 ACO)[22]，F. Defersha らは，SA を用いた解法を提案している [19, 20, 21]．上記の資源計画に対する研究では製品計画が固定であると考えられている．

実施計画の分野では，Huaccho らは，製品の投入順序 (選択性の順序変更) を入れ替えることで機械故障に適応する方法を提案している [23]．谷水らは，選択性として割付変更を考慮し，スケジューリング問題に対して GA を用いて作業遅延に対する生産スケジュールの変更を提案した [24]．これら研究では，製品計画，資源計画は変わらないものとして捉えられている．

近年では製品計画及び実施計画を統合した (Integration of process planning and scheduling, 以下 IPPS) 研究が活発に行われている．IPPS の分野において機械故障を扱っている研究を挙げる．Rajabinasab らは，エージェントベースの解法を用いて製品計画の製造工程と機械種別，実施計画の順序入替の選択性を導入した問題を扱っている [25]．Y. W. Guo らは，機械種別，工具種別，アクセス方向の選択性を導入した問題に対して PSO を用いた解法を提案している [26] また，Wong らは，同様にエージェントベースの解法を用いて機械種別と順序入替を導入した問題を扱っている [27]．F Wand らは機械種別，工具種別，アクセス方向，順序入替を導入した問題を扱っており，製品計画を SA を用いて解き，実施計画ではディスパッチング・ルールを使い生産スケジュールを作成する方法を提案している [28]．また，W. Guo らは，選択性として機械種別，工具種別，アクセス方向と割付変更を導入した問題に対してメタヒューリスティクスの PSO を用いた解法を提案している．

機械故障を扱っている既存研究が対象としている選択性をまとめたものを表 2.2 に示す．既存研究においては対象するプランニング問題の範囲，解法はそれぞれ異なる．また，部分問題のみを対象とした場合では，機械故障に対処した際に得られる解が全体として実行可能である保証はない．機械故障による影響は，部分問題での選択性だけにとどまらず，最終的に全体へと波及する．そのため，生産プランニングの解空間を統合して扱うべきであると考える．また，提案されている手法が必ずしも扱う問題規模に適切なものであると

表 2.2: 動的変化 (機械故障) を扱う既存研究が導入する選択性

		L.Wang (2011)	Rajabinasab (2011)	Huaccho (2009)	Y. W. Guo (2009)	F Wang (2011)	Wong (2006)	テヘラニ (2009)
製品計画	製造工程		*					
	工程順序							*
	機械種別		*		*	*	*	*
	工具種別				*	*		*
	アクセス方向				*	*		*
実施計画	順序変更		*	*		*	*	
	割付変更	*			*			
Algorithm		Find-route algorithm	Pheromone-based algorithm	Rescheduling strategy	PSO	SA and EDD	oHAN	A*

は限らない．例えば，解空間内の制約充足解が少ない場合，メタヒューリスティクスをはじめとする手法よりも，全探索を行ったほうが効率がよい可能性がある．

W. Shen らは，IPPS に関する研究において，同時に 2 つの領域の制約を考慮しながら，1 つの最適化問題として統合することは，結果として大域的な解を得ることができるが，解空間が膨大になると述べている [29]．この問題点を解決する方法として，各手法による最適化の過程では，問題ごとの制約条件を計算の都度考慮することを提案している．例えば GA を用いた研究では，制約を満たさない解が個体群内に存在する場合，その個体の適応度にペナルティを加え，遺伝子操作の過程で選択されないようにするといった方法が取られる．

本研究では，生産プランニングの解空間に存在する制約を充足していない解に着目し，それら解を排除した制約充足解空間を作成する方法を提案する．この方法では，生産実施中の動的変化による影響を考慮し常に制約充足解空間を整合して更新する．また，制約充足解空間内のみを探索対象とした解探索手法を提案する．それらにより，生産プランニングの統合的な解法を実現する．

2.4 本章のまとめ

本章では，はじめに生産プランニングに関する定義について述べ，本研究で扱う生産プランニングを構成する計画を整理し，生産プランニングの選択性という概念を導入した．4 つの計画が生産プランニングを構成するとし，全部で 23 の選択性を導入した．そして，

生産プランニング選択性間の依存関係について定性的な分析を行った．その結果，1 つの選択性の変更は生産プランニングの上流から下流また下流から上流と全体に影響を与える可能性があることを明らかにした．従来の生産プランニングの研究では，生産プランニング問題を分割して解いているため，このような波及に対して対処できない可能性があり，また，解空間内の制約充足解の規模に応じて解法を選定する必要があることについて言及した．そして，生産プランニングの解空間を統合的な扱うことの重要性について及び，本研究の位置付けについて述べた．

第3章　BDDとゼロサプレス型BDD

3.1　制約充足解空間構成のためのデータ構造

本章では，生産プランニング問題を統合的に扱うために，全体の制約を充足した解の集合表現のためのデータ構造について述べる．生産プランニング問題の解空間の特徴は以下の 2 点である．

- 各選択性の選択肢の組合せにより生産計画が作成される．
- 選択性内と選択性間には制約関係が存在する．例えば，選択性内の制約として，選択性 A の選択肢 a_1, a_2, a_3 が択一の選択条件であることや，選択性間の制約としては，選択性 A と B において，選択肢 a_1 が選択される時，選択性 B の選択肢 b_1 が必ず選択されるといったことが挙げられる．

以上のことから，生産プランニングにおける実行可能解は制約を充足した選択肢の組合せであることが言える．しかし，解空間内には，上記のような制約条件を満たさない選択肢の組合せが多数含まれている．そのような組合せを排除し，制約充足した組合せのみを表現することができれば，生産プランニング問題を統合的に扱うことが可能であると考えられる．

次節からは，本研究で用いる組合せ集合のデータ構造である二分決定グラフ (Binary Decision Diagram, BDD) 及びゼロサプレス型 BDD(Zero-Suppressed BDD, ZDD) について述べる．

3.2　二分決定グラフ (Binary Decision Diagram)

二分決定グラフ (Binary Decision Diagram, 以下 BDD) は，Akers が提案 [30] し，Bryant[3] が効率のよい BDD 処理アルゴリズムを考案したコンパクトな論理関数のグラフ表現方法である．

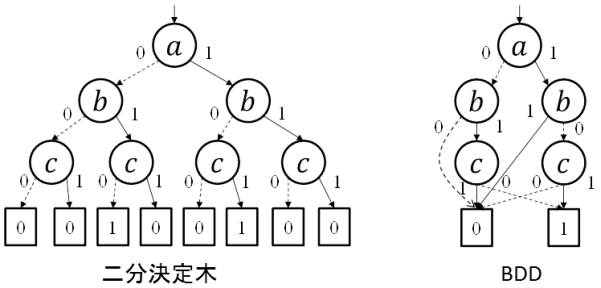


図 3.1: 論理関数 F の二分木と BDD 表現

一般に，論理関数のそれぞれの変数について，0，1 の値を代入した結果を，2 分木の枝 (0 枝/1 枝) で場合分けし，最終的に得られる論理関数の値を 2 値の定数ノード (終端ノード 0/終端ノード 1) で表現すると図 3.1 左のような二分木状のグラフになる ($F(a,b,c) = a\bar{b}c \vee \bar{a}b\bar{c}$)．このとき，場合分けする変数の順序を固定し，

1. 等価なノードを共有する (図 3.2a)

2. 冗長なノードを削除する (図 3.2b)

という 2 つの簡約化規則を可能な限り適用する事により図 3.1 右のような「既約」な形が得られる．この構造により，論理関数をコンパクトかつ一意に表せることが知られている [4]．また，複数の論理関数を表す BDD の間においても，変数の順序を固定すれば，互いにグラフを共有することが可能であり，1 つのメモリ空間の中で多数の論理関数データを扱うことができる．

BDD のサイズは，変数順序により大きく変化する [31]．そのため，最適変数順序を見つける問題は NP 完全であることが示されている [32]．しかし，最適でなくても比較的良い順序が得られれば実用的には有効である．これまでに様々なヒューリスティックスが提案されているが，一般的には，

- 局所計算性のある変数同士は近い順序でおく
- 出力を制御する力の強い変数は上位に置く

という 2 つのヒューリスティクスが知られている [33]．このように変数順序に注意すれば，BDD は多くの実用的な論理関数を比較的コンパクトなデータ構造で表すことができ，変数順序を固定した場合は一意に表現できる．

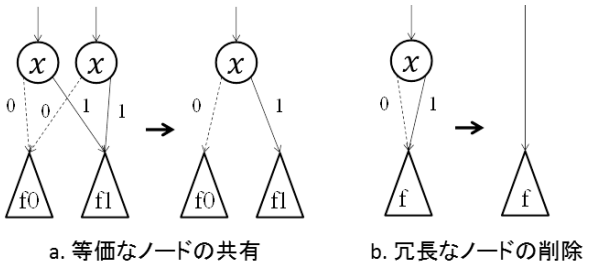


図 3.2: BDD の簡略化規則

3.3　ゼロサプレス型 BDD

BDD は本来，論理関数データを表現するために考案されたものだが，これを用いて「組合せ集合」を表現することもできる [5]．組合せ集合とは，「n 個のアイテムから任意個を選ぶ組合せ」を要素とする集合である．n 個のアイテムから任意個を選ぶ組合せは 2^n 通り存在するので，組合せ集合としては 2^{2^n} 通りの取り方がある．例えば，a, b, c, d, e という 5 つのアイテムに関して $\{ab, e\}, \{abc, cde, bd, acde, e\}, \{1, cd\}, \{\emptyset\}$ は，いずれも組合せ集合の一例である．(本論文では，“1” は空の組合せ要素を表し，$\emptyset$ は空の集合を表すこととする．) 組合せ集合は，組合せ問題の開集合を表現する基本的・汎用的なデータ構造である [5].

例えば，表 3.1 は $F = a\bar{b}c \vee \bar{a}b\bar{c}$ という論理関数を表す真理値表であるが，見方を変えると $S = \{ac, b\}$ という組合せ集合を表現している．ZDD は，このような組合せ集合データの処理に特化した BDD である．ZDD は，通常の BDD とは簡約化規則が異なり，

1. 等価なノードを共有する (図 3.2a，BDD と同様)

2. 1 枝が終端ノード 0 を直接挿しているノードを取り除き，0 枝に直結させる (図 3.3a)

の 2 つの規則を用いることにより，図 3.3b のような形で組合せ集合を表現できる [4].

一般的に，組合せ集合に類似した要素 (部分的組合せ) が多数含まれる場合，BDD では等価なサブグラフが多く出現し，それらが互いに共有され，コンパクトに圧縮された表現となる．更に，ZDD の簡約化規則を用いると，組合せに無関係な変数に関するノードが自動的に削除されることになり，通常の BDD よりも効率よく組合せ集合を表現・操作することができる．この簡約化規則は，特に疎な組合せの集合に対して顕著な効果がある．例えば，組合せ集合の各要素に含まれるアイテムの平均出現頻度が 1%であれば，ZDD は

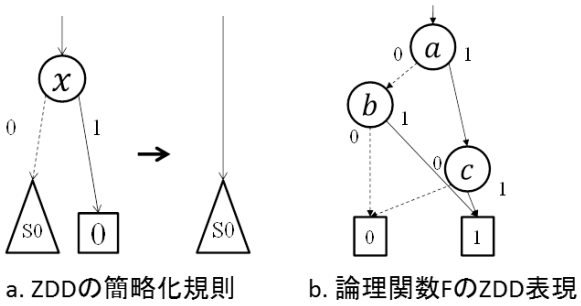


図 3.3: ZDD の簡略化規則と ZDD 表現

表 3.1: 論理関数と組合せ集合の対応

a	b	c	F	
0	0	0	0	$\rightarrow S$
0	0	1	0	
0	1	0	1	$\rightarrow b$
0	1	1	0	
1	0	0	0	
1	0	1	1	$\rightarrow ac$
1	1	0	0	
1	1	1	0	

BDD よりも 100 倍コンパクトになる可能性がある [5]．ZDD の利点はデータの圧縮及び ZDD 同士の様々な集合演算を圧縮されたデータ量にほぼ比例する計算時間で実行できることが挙げられる．つまり，圧縮データを元に戻すことなく，圧縮したままで高速に演算処理できるという優れた特徴がある [5].

また，ZDD は，米スタンフォード大学の Knuth 名誉教授の著書 “The Art of Computer Programming” において，日本人が考案したアルゴリズムとして初めて独立項目として解説されている [34].

3.4 スクリプト処理システム VSOP

VSOP(Valued-Sum-Of-Products)[35] は，スクリプト処理システムの一種であり，組合せ集合を ZDD に基づき処理することができる．集合の各要素には整数値の重みをつける事ができ，四則演算や大小比較の他に，様々な集合演算が実装されている．以下に，本論文で用いる各演算アルゴリズムについて述べる．例題として変数の集合 a, b, c, d を用いて説明する．

- 加算：加算 $(F+G)$ は，F と G の和集合を求める演算である．

  Ex) $F=\{a,b\}$，$G=\{c\}$ の時, $F+G=\{a,b,c\}$

- 減算：減算 $(F-G)$ は，F から G を引いた差分集合を求める演算である．

  Ex) $F=\{a,b,c\}$，$G=\{a,c\}$ の時，$F+G=\{b\}$

- 変数乗算：変数乗算 $(F\times v)$ は，F の各項に v を付加する演算である．

  Ex 1) $F=\{a\}$ の時，$F\times c=\{ac\}$

  Ex 2) $F=\{a,b\}$ の時，$F\times c=\{ac,bc\}$

- 直積乗算：直積乗算 $(F\times G)$ は，F の各項と G の各項から 1 つずつ選ぶ組合せすべてに着いて変数乗算を行い，その総和の式を求める演算である．

  Ex) $F=\{ab,b,c\}$，$G=\{ab,1\}$ の時，

  $F\times G=(ab\times ab)+(ab\times 1)+(b\times ab)+(b\times 1)+(c\times ab)+(c\times 1)$

  $\quad\quad=\{ab,abc,b,c\}$

  G の “1” は，空の組合せ要素を表す．

- 変数除算：変数除算の商 (F/v) は，F の各項から v を含むものを残してそれ以外を削除し，さらに残った各項から v を取り除く演算である．一方，剰余 $(F\%v)$ は，F の各項から v を含まないものを残し，それ以外を削除する演算である．Ex) $F=\{ab,b,c\}$，$v=\{b\}$ の時，$F/v=\{a,1\}$，$F\%v=\{c\}$

  つまり, $F=v\times(F/v)+(F\%v)$.

上記の演算の他に Restrict 演算，Permit 演算，Permitsym 演算，If-Then-Else 演算を実装している．

- Restrict 演算：$F.Restrict(G)$ は，F に含まれる項のうち，G の組合せの少なくとも 1 つを包含するような項だけを残し，それ以外を削除する演算である．

  Ex) $F=\{b,ac,cd,abc,bcd\}$，$G=\{a,bc\}$ の時，$F.Restrict(G)=\{ac,abc,bcd\}$

- Permit 演算：$F.Permit(G)$ は，F に含まれる項のうち，G の組合せ少なくとも 1 つに包含される項だけを残し，それ以外を削除する演算である．

  Ex) $F=\{b,ac,cd,abc,bcd\}$，$G=\{abc,cd\}$ の時，$F.Permit(G)=\{b,ac,cd,abc\}$

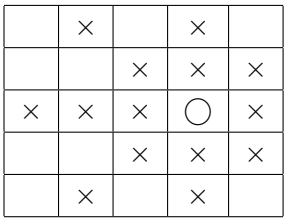


図 3.4: チェスのクイーンの動き

$(0,0)$	$(0,1)$	$(0,2)$	$(0,3)$	$(0,4)$
$(1,0)$	$(1,1)$	$(1,2)$	$(1,3)$	$(1,4)$
$(2,0)$	$(2,1)$	$(2,2)$	$(2,3)$	$(2,4)$
$(3,0)$	$(3,1)$	$(3,2)$	$(3,3)$	$(3,4)$
$(4,0)$	$(4,1)$	$(4,2)$	$(4,3)$	$(4,4)$

図 3.5: 5×5 のチェス盤の座標

- Permitsym 演算：$F.Permitsym(n)$ は，F に含まれる項のうち，n 個以下の変数からなる項だけを残し，それ以外を削除する演算である．

  Ex) $F = \{b, ac, cd, abc, bcd\}$，$n = 1$ の時，$F = Permitsym(n) = \{b\}$

- If-Then-Else 演算：$F?G : H$ は，G から F に含まれる項を取り出し，H から F に含まれない項を取り出す演算である．

  Ex) $F = \{b, ac, cd, abc, bcd\}$，$G = \{ac, cd\}$ の時，$F?G : 0 = \{ac, cd\}, F?0 : G = \{b, abc, bcd\}$

ZDD にて生産プランニングにおける制約充足解空間を表現するために上記の演算を用いる．その他の演算についての詳細は文献 [35, 36] に記載されている．

3.5 ゼロサプレス型 BDD の適用例：N クイーン問題

ここでは，前節で述べた ZDD に基づくスクリプト処理システム VSOP の各種演算を応用して，N クイーン問題を解く方法について述べる．N クイーン問題は，$N \times N$ のチェスの盤上に N 個のクイーンを，お互いに取られないように配置する制約充足問題である．図 3.4 にクイーンの動きを示す．クイーンは，自身を中心に上下左右と斜め 4 方向の計 8 方向に進むことができる．

例えば，5 クイーン問題でのチェス盤の座標を各座標を図 3.5 に示す．ここでは，N クイーン問題を ZDD を用いて解く場合，チェス盤の座標 (i, j) 上にクイーンを置くことを

ZDD の変数 $a_{(i,j)}$ として定義する．まずはじめに，N クイーン問題での全ての組合せ 2^N 通りを全列挙する．全ての組合せを列挙する ZDD を作成するためには，VSOP の乗算演算と加算演算を用いる．全列挙した組合せ集合を all とすると，all は以下のように求めることができる．

$$all = (a_{(0,0)} + 1) \times (a_{(0,1)} + 1) \times \cdots \times (a_{(N,N-1)} + 1) \times (a_{(N,N)} + 1) \tag{3.1}$$

all には，全ての組合せが列挙されており，その中に解が存在する．そこで，次のステップとして，この集合から制約を満たす組合せを選択する必要がある．ここでの制約条件としては以下の 2 つである．

条件 1：クイーン同士がお互いに取られるような組合せを全て削除する．

条件 2：組合せを構成する変数の数が N 変数未満の組合せを削除する．

条件 1 にてクイーン同士がお互い取られるような組合せを全て削除するため，N 個以上の変数から構成される組合せは条件 1 の制約を満たさず削除される．条件 1 では，座標 (i,j) にクイーンがいる場合，

1. 同じ行番号 (i) をもつ変数との組合せ
2. 同じ列番号 (j) をもつ変数との組合せ
3. その座標から右斜め下方向の座標の変数との組合せ
4. その座標から左斜め下方向の座標の変数との組合せ

を全列挙し，該当する各組合せの集合 ng を作成する．例として図 3.4 をあげる．この図では，クイーンの位置が (2,3) である．この位置にクイーンがある場合，同じ行番号を持つ変数との組合せ集合 ng_1 は以下のように定義できる．

$$ng_1 = a_{(2,3)} \times (a_{(2,0)} + a_{(2,1)} + a_{(2,2)} + a_{(2,4)}) \tag{3.2}$$

同様に同じ列番号をもつ変数との組合せ集合を ng_2 とすると，この組合せ集合は以下のように定義できる．

$$ng_2 = a_{(2,3)} \times (a_{(0,3)} + a_{(1,3)} + a_{(3,3)} + a_{(4,3)}) \tag{3.3}$$

次に右斜め下方向の座標の変数との組合せ集合を ng_3 とすると，ng_3 は以下のように表すことができる．

$$ng_3 = a_{(2,3)} \times a_{(3,4)} \tag{3.4}$$

そして左斜め下方向の座標の変数との組合せ集合 ng_4 は以下のように定義できる．

$$ng_4 = a_{(2,3)} \times (a_{(3,2)} + a_{(4,1)}) \tag{3.5}$$

そしてこの座標にクイーンがある際の制約条件を $ng_{(2,3)}$ は以下のように表される．

$$ng_{(2,3)} = ng_1 + ng_2 + ng_3 + ng_4 \tag{3.6}$$

最後に各座標での制約条件を 1 つの集合として表す．

$$ng = ng_{(0,0)} + ng_{(0,1)} + \cdots + ng_{(4,3)} + ng_{(4,4)} \tag{3.7}$$

これにより，ng は組合せとして，含まれてはいけない変数の組合せを表す集合となる．all の中の ng を含む組合せは，制約条件 1 を満たさない組合せとなるため，これら組合せは all から削除されるべきである．ng 内の組合せを包含する all 内の組合せを削除すれば良いので，ここでは Restrict 演算を用いる．

$$selNG = all.Restrict(ng) \tag{3.8}$$

$selNG$ は all から削除されるべき組合せの集合であり，削除するため If-Then-Else 演算を用いる．

$$selOK = selNG?0 : all \tag{3.9}$$

また，条件 2 は Permitsym 演算を用いて以下のように定義される．

$$selLT = selOK.Permitsym(N-1) \tag{3.10}$$

$selLT$ は N 以下の変数で構成される組合せの集合となる．最後に変数の数が N の組合せの集合が作成するためには，$selOK$ から $selLT$ を削除すればよいため，同様に If-Then-Else 演算を用いて以下のように定義できる．

$$ans = selLT?0 : selOK \tag{3.11}$$

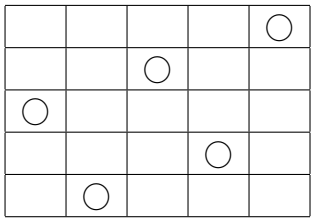


図 3.6: 5 クイーンの解の一例

結果として ans は，N クイーン問題での解の数を表す組合せ集合となる．5 クイーン問題の場合，ans が保持する組合せ数は，10 通りとなる．図 3.6 はその解のうちの 1 通りである．

ZDD では，N クイーン問題における制約条件を組合せとして記述し，解空間内から制約条件を満たす組合せのみを 1 つの集合表現として保持する．一般的に生産プランニング問題は，選択性内の制約充足した選択肢の組合せから最適解を求める問題である．そこで本研究では，制約充足した解のみの集合である制約充足解空間を表すデータ構造として ZDD を用い，生産プランニングにおける制約充足解空間の表現方法を提案する．

3.6　本章のまとめ

本章では，生産プランニングの制約充足解空間を構成するためのデータ構造として ZDD について述べた．生産プランニングにおける解空間は，選択性の選択肢の組合せから構成され，解空間内の実行可能解は選択性内および選択性間の制約条件を満たす選択肢の組合せである．論理関数のグラフ表現である BDD は，組合せの集合としてみることもできる．その中でも組合せ集合データの処理に特化した ZDD では，組合せに無関係なノードを削除し，データ構造として組合せをコンパクトに表現できる．また，ZDD に基づいたスクリプト処理システムである VSOP を紹介し，VSOP に実装されている各種演算処理について述べた．ZDD を用いた例題として N クイーン問題をとりあげ，制約条件を組合せとして表現し，全組合せを表す ZDD からそれら制約条件を満たす組合せのみを取り出すことにより，全ての解候補を 1 つの ZDD にて保持することができることを示した．また，この考え方は，制約を満たす解から最適解を求める生産プランニング問題に対しても適用できることについて言及した．

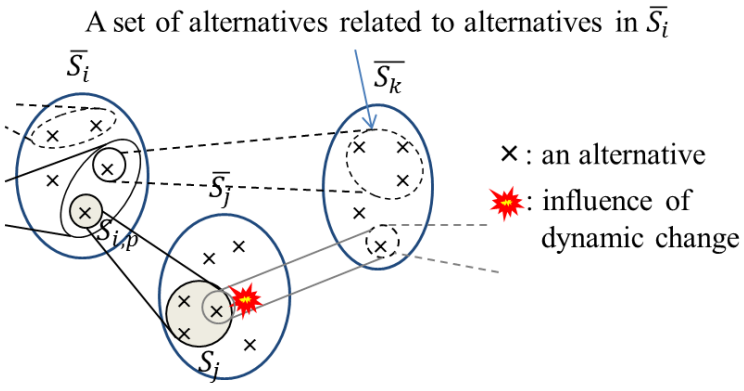
第4章　単一のZDDによる制約充足解空間表現および解探索手法

4.1　ZDDによる制約充足解空間および解探索のためのGAの概要

図2.1に示す通り，生産プランニングの解空間には選択性 $\overline{S}_i$ の選択肢 $S_{i,p}$ と組合わさらない選択肢が存在する．本研究では，そのような組合せを生産プランニングの制約を満たしていない組合せとして定義する．生産プランニングの解空間にはそのような組合せは多く存在している．本研究では，制約を満たさない組合せを排除し，制約を満たす組合せを1つの集合として管理するためにZDDを用い，制約充足解空間を表現する．

しかし，ZDD上では，各組合せに対する目的関数値を保持していない．そのため，ZDD内の目的関数を最小とする解(組合せ)を探索する手法が必要となる．このために生産プランニング分野によく用いられる解法としてメタヒューリスティクスがある．その代表的なものが，遺伝的アルゴリズム（Genetic algorithm, 以下GA）[37]，シミュレーテッドアニーリング(Simulated Annealing，以下SA)[13]やタブーサーチ(Tabu Search, 以下TS)[38]などである．

GAは，ダーウィンの理論にヒントを得た，強力で，適用分野を限定しない確率的最適手法の1つである[39]．GAでは，記号列としてコード化された，個体(解)の集団(集合)



生産プランニングにおける組合せ問題(図2.1の再掲)

を操作する．各個体は目的関数値に応じた適応度を有し，ある世代を形成している個体群において，適合性の高い個体ほど高い確率で生き残るように選択を行う．さらに，交叉，突然変異といった個体生成のオペレーションにより新しい解 (子孫) を生成し，次世代の個体集団を形成する．世代の更新を繰り返し，適合度の高い個体が増加し，徐々に最適解または準最適解に集団を収束させるというのが GA の基本的な枠組みである．

また SA や TS をはじめとする単点探索では，探索空間の 1 点から探索を開始して，ある移動規則に従い次の点まで移動する．点から点まで移動する方法は，頻繁に局所的最適値にとらわれてしまうことになる．それに対して GA は，多点探索法であり，十分な大きさの点集合 (解の集団) が同時に動作してゆく．したがって，偽の谷 (最小化問題の場合) にとらわれてしまう確率が小さくなる．本論文では，多点探索法であることが GA の最も特徴的な点であると捉え，制約充足解空間を表す ZDD に対する解探索法として GA に着目する．

本章では，まずはじめに ZDD を用いて生産プランニングにおける制約充足解空間を表現するための方法について提案する．次に ZDD により表される制約充足解空間を解探索するための GA を提案する．

4.2 動的生産プランニング問題の例題

ここでは，動的生産プランニング問題の例題について述べる．この例題にて扱う選択性は，製品計画から製造工程，工程順序，機械種別，資源計画から資源の数，実施計画から目的関数，差立ロジックである．製品計画において，工程順序では，製品を生産するために必要な工程の種別とその順序を決定する．製造工程では，工程の種別を具体的にどの加工法で処理を行うかを決定し，機械種別では，それら製造工程を処理するための機械の種別を決定する．資源計画では，機械の種別ごとに現場に設置する台数を決定する．実施計画では，製造計画および資源計画の決定された選択肢を用いて，目的関数および差立ロジックに従い生産スケジューリングを行う．

この問題では，1 製品種別に対して，複数の工程順序候補が存在し，工程種別には代替工程がある．工程種別は，複数の製造工程の候補をもつ．また，1 製造工程を処理できる機械種別は複数あり，各機械種別はそれぞれインスタンスをもつ．さらに実際の生産現場では，設置可能な機械台数の上限が存在する．本論文では，製品は生産する 1 製品種別の

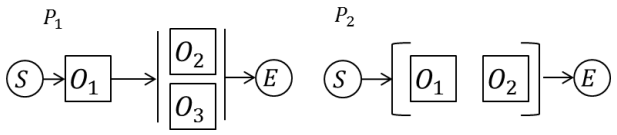


図 4.1: 各製品種別の工程順序と使用工程 (例題)

表 4.1: 工程種別，製造工程および機械種別の組合せと処理時間 (例題)

				Instance		
				1	1	1
	O_1	O_2	O_3	M_1	M_2	M_3
U_1	*	-	-	1	-	-
U_2	-	*	-	-	-	1
U_3	-	-	*	-	3	2
U_4	-	*	-	2	-	-
U_5	*	-	-	-	1	-

1 台を表す．そのため，ある製品種別が 3 台生産される場合，その製品種別に相当する製品の数は 3 となる．

具体的な例題を示す．工程種別 w を O_w，製造工程 k を U_k，機械種別 i を M_i とする．この例題は，製品種別数 2 であり，各製品種別が 1 台ずつ生産されるため製品数は 2 である．また，工程種別数と機械種別数は 3 であり，製造工程数は 5 である．各機械種別のインスタンスは，1 台ずつであり，設置可能機械台数の上限数は 2 とする．図 4.1 は各製品種別の使用工程種別とその順序を示している．図 4.1 には，始点節点，中間節点と終点節点の 3 種の節点がある．図中の開始と終わりを意味する始点節点と終点節点はダミー節点である．中間節点は 1 工程種別を表しており，2 つの節点をつなぐ矢印は，工程種別間の先行関係を表している．さらに図中には二種類の括弧が存在する．角括弧 “[ ]” で囲まれた工程種別は順不同であることを表し，棒括弧 “| |” で囲まれている工程種別は択一の選択を意味する．製品種別 P_1 の工程順序は，$(O_1 \to O_2), (O_1 \to O_3)$ の 3 種の工程順序の内，1 種を選択できることを意味する．また，製品種別 P_2 の場合は，$(O_1 \to O_2), (O_2 \to O_1)$ のどちらかを選択できる．表 4.1 は，工程種別，製造工程と機械種別の組合せを表している．表中の “Instance” は各機械種別のもつインスタンスの台数である．例えば，表中の製造工程 $U_1\times$ 工程種別 $O_1 = *$ は，製造工程 U_1 は，工程種別 O_1 を処理できる製造工程であることを示す．製造工程 $U_1\times$ 機械種別 $M_1 = 1$ は，製造工程 U_1 は機械種別 M_1 で処理

可能であり，その処理時間は 1 であることを表す．この問題では，製品を処理するための設置された機械間の運搬にかかる時間や，運搬による制約は考慮しないとする．目的関数は，納期遅れ時間に比例したペナルティコストと，計画変更の回数に比例したコストの総和である $Total\ Additional\ Cost\ (TAC)$ とし，最小化問題である．目的関数 TAC は以下のように表すことができる．

$$TAC = \sum_{s=1}^{N}(T_s \times cost_{dd}) + cost_c \times e \tag{4.1}$$

式 4.1 の N は，製品の総数である．右辺の第一項は，納期遅れ時間に比例したペナルティコストを表し，第二項は計画変更に比例したコストを表す．T_s は，製品 J_s の納期遅れ時間を表す $(0 <= T_s)$．$cost_{dd}$ は，納期遅れ時間に比例した係数であり，$cost_c$ は計画変更に比例した係数である．e は，計画変更を行った回数を表す．ここでの計画変更とは，各製品の工程種別と順序，処理する製造工程及び機械種別が変更された場合を意味する．初期の生産計画作成時では，第二項の値は 0 となる．

また，動的変化として機械故障を扱い，故障した機械は生産終了まで復旧しないとする．機械故障は，機械上で製造工程が処理開始前または終了後に発生するとする．さらに，一度生産が開始されると，設置された機械の変更は行うことができないとする．動的生産プランニングでは，生産実施中にある製品の 1 製造工程が完了すると，計画を変更する際の代替案の候補が制限される．同時に機械故障により，実施中の生産計画のみならず代替候補であった生産計画が実行不可になる可能性がある．このような条件において，ZDD による生産プランニング問題の制約充足解空間の表現方法及び GA による解探索手法について次節から述べる．

4.3 生産プランニングにおける制約充足解空間

ここでは，単一の ZDD による制約充足解空間表現方法について述べる．作成される制約充足解空間は製品計画と資源計画の選択肢の組合せを表す．この際，変数の表記として用いるものを以下に示す．

i：機械種別のインデックス番号

j：機械インスタンスのインデックス番号

k：製造工程のインデックス番号

r：工程順序のインデックス番号

s：製品のインデックス番号

t：実施可能な工程種別と機械種別の組合せのインデックス番号

w, α, β, γ：工程種別のインデックス番号

F：生産現場に設置可能な機械台数の上限数

m：全ての機械インスタンスの変数の全組合せの集合

m_i：機械種別 M_i のインスタンスの組合せ集合

m_{ij}：機械種別 M_i のインスタンス j の設置することをを表す変数

m_i^{sr}：製品 J_s の r 番目の製造工程を機械種別 M_i にて処理することを表す変数

o_{wr}^{s}：製品 J_s にて，工程種別 O_w を r 番目に行うことを表す変数

u_k^{sr}：製品 J_s の r 番目に製造工程 U_k を割り当てることを表す変数

M_i：機械種別 i

O_w：工程種別 w

U_k：工程種別 w

J_s：製品 s

X：統合的な制約充足解空間を表す組合せ集合

X^s：製品 J_s の工程順序の組合せ集合

Z^{swr}：製品 J_s の r 番目の工程種別 O_w を処理できる製造工程の組合せ集合

$|Z^{swr}|$：Z^{swr} 内の組合せ数

Z_h^{swr}：製品 J_s の r 番目の工程種別 O_w を処理できる製造工程の h 番目の組合せ

Z^{sru}：製品 J_s の r 番目の製造工程 U_k を処理できる機械種別の組合せ集合

$|Z^{sru}|$：Z^{sru} 内の組合せ数

Z_a^{sru}：製品 J_s の r 番目の製造工程 U_k を処理できる製造工程の a 番目の組合せ

図 4.2 に単一の ZDD による制約充足解空間表現方法の概要を示す．作成方法は，はじめに製品ごとに工程順序の組合せ集合を表す ZDD を作成する．次に，この ZDD をベースに工程順序と製造工程間の制約関係を満たす組合せ集合を表す ZDD を作成する．そして，同様に工程順序と製造工程の選択肢の組合せ集合を表す ZDD をベースに，機械種別との制約関係を考慮し，3 つの選択性を考慮した組合せ集合を表す ZDD を作成する．こ

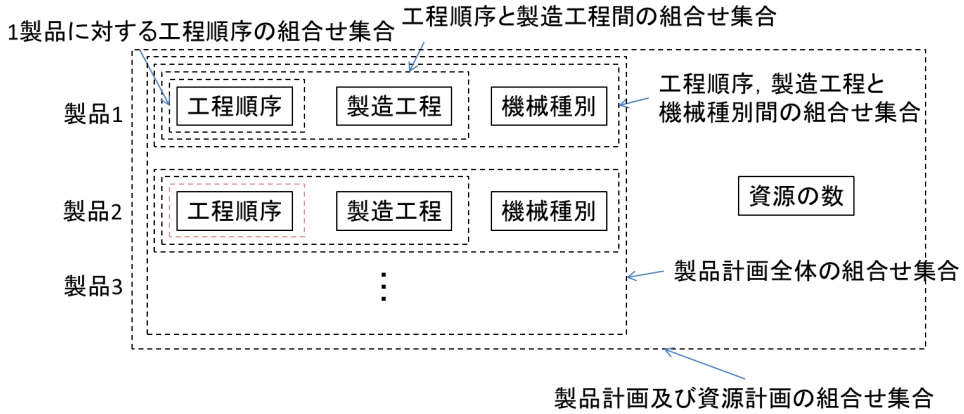


図 4.2: 単一の ZDD による制約充足解空間表現の概要

こまでに作成された ZDD は，製品単体における製品計画での制約充足した組合せを表す．製品ごとの製品計画での組合せ集合の直積を求めることにより，製品計画全体としての組合せ集合を表す ZDD を作成する．そして，製品計画全体の組合せ集合に対して，資源計画の資源の数の選択肢との制約関係を考慮し，製品計画および資源計画における制約充足解空間を表す ZDD を作成する．例えば，工程種別 O_1 が製造工程 U_1 または U_2 で処理される時，o^s_{1r} が含まれる組合せには，u^{sr}_1 もしくは，u^{sr}_2 が含まれる．このような制約条件を，考慮しながら ZDD の拡張を行う． はじめに工程順序の選択肢の組合せを表す ZDD を作成する．具体的には，各製品における工程種別の先行関係を満たす工程順序を表す組合せ集合を作成する．製品 J_s の 1 つの工程順序の長さが r 番目までの時，実施可能な工程順序を表す組合せ X^s_u は式 4.2 のように定義できる．

$$X^s_g = o^s_{\alpha 1} \times o^s_{\beta 2} \times \cdots \times o^s_{\gamma r} \tag{4.2}$$

図 4.1 の製品種別 P_1 を 1 台生産し，それが製品 J_1 であるとすると，上記の式を用いると，$X^1_1 = \{o^1_{11}o^1_{22}\}, X^1_2 = \{o^1_{11}o^1_{32}\}$ となる．このような組合せの数が $G(= |X^s|)$ 通りあるとすると，この和集合 X^s は以下のように定義できる．

$$X^s = X^s_1 + X^s_2 + \cdots + X^s_U \tag{4.3}$$

X^s は製品 J_s における工程順序の解候補を表す ZDD となる．先ほどの例では，$X^1 = \{o^1_{11}o^1_{22}, o^1_{11}o^1_{32}\}$ となり X^1 は 2 通りの組合せを表す．

続いて，工程順序の解候補集合である X^s を製造工程へと拡張を行う．ここでは，選択

性間の制約条件を考慮する．例えば，選択性 A と選択性 B 間において制約条件があるとする．仮に制約条件として，選択性 A の選択肢 a_x が選択される時，選択性 B の選択肢 b_y と b_z が同時に選択されるとする．この制約条件を組合せとして ZDD の算術式で表すと以下のように定義できる．

$$C_{x1} = a_x \times b_y \times b_z \tag{4.4}$$

また，制約条件が選択肢 b_y と b_z のどちらか一方が選択されるという択一の場合では，以下のように定義できる．

$$C_{x2} = a_x \times (b_y + b_z) \tag{4.5}$$

選択性 A の組合せ集合を X とすると，制約条件が C_{x1} の関係である時，制約を満たす選択性 A と B の間の組合せ集合は以下の式を用いて表すことができる．

$$X' = (X/a_x) \times C_{x1} + X\%a_x \tag{4.6}$$

この処理について具体的な例を挙げる．X が選択性 A の組合せ集合 $\{a_1a_2, a_3\}$ を表すとする．選択肢 a_1 が選択される時，選択性 B の選択肢 b_1 または b_2 が選択されるとする．この関係を表す組合せ集合を Y とする．この際，式の各項の処理を図 4.3 に示す．(X/a_1) では X 内の a_1 を含む組合せを取り出し，a_1 を削除する．そのため，出力は a_2 となる．この結果と，Y の乗算では組合せ $\{a_1a_2b_1, a_1a_2b_2\}$ を表す ZDD が作成される．更に $X\%a_1$ では，a_1 を含まない組合せを取り出すため，a_3 が得られる．第 1 項と第 2 項の結果を加算を用いて 1 つの集合として表すと，組合せ集合 $\{a_1a_2b_1, a_1a_2b_2, a_3\}$ となる．考慮するべき選択性の範囲が広がり，選択性間の制約条件を満たす組合せ集合を作成する際に，基本として 4.3 を用いる．

工程順序と製造工程間では，各工程種別には，処理可能な製造工程が存在する．工程種別 O_w が製造工程 U_k で処理可能で，製品 J_s で工程種別 O_w が r 番目に行う時，ZDD ではこの関係を以下のように定義する．

$$Z_h^{swr} = o_{wr}^{s} \times u_k^{sr} \tag{4.7}$$

式 4.7 は，製品 J_s の r 番目に実施可能な工程種別と製造工程の 1 通りの組合せを表す．また，Z_k^{swr} の集合を Z^{swr}，$|Z^{swr}|$ の要素数を H とすると，Z^{swr} は式 4.8 のように表現できる．

$$Z^{swr} = Z_1^{swr} + Z_2^{swr} + \cdots + Z_H^{swr} \tag{4.8}$$

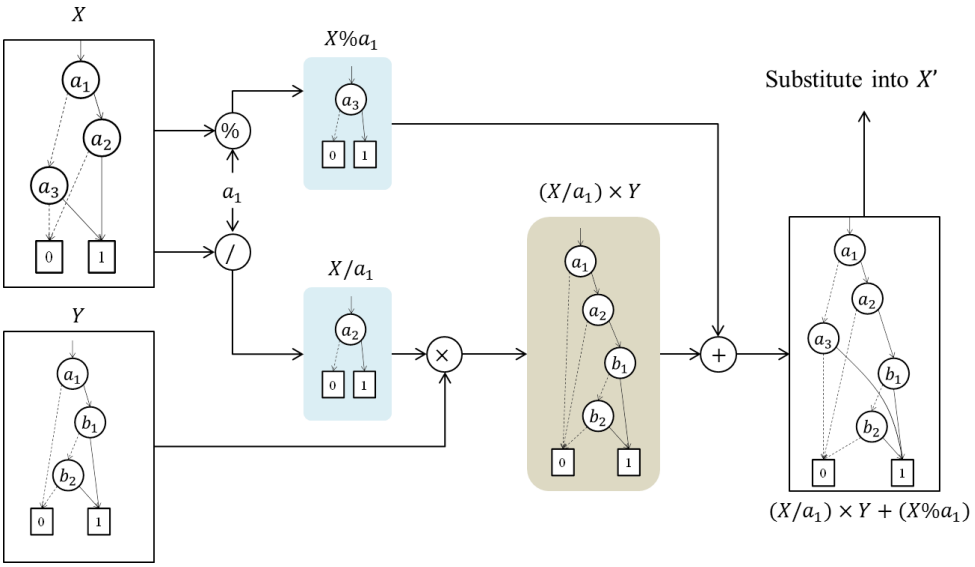


図 4.3: 選択性間の制約条件を満たす組合せを作成する際の演算

アルゴリズム 1 工程順序と製造工程間の解候補集合の作成

1: $s, w, r1 \leftarrow 1$
2: **while** $s \leq N$ **do**
3: **while** $w \leq W$ **do**
4: **while** $r \leq R$ **do**
5: $X^s \leftarrow (X^s/o^s_{wr}) \times Z^{swr} + (X^s\%o^s_{wr}))$
6: $r \leftarrow r + 1$
7: **end while**
8: $r \leftarrow 1, w \leftarrow w + 1$
9: **end while**
10: $r \leftarrow 1, s \leftarrow s + 1$
11: **end while**

ここで作成される Z^{swr} は，製品 J_s の r 番目に実施可能な工程種別と製造工程の和集合を表す ZDD は，工程順序と製造工程間の制約条件となる．この制約条件を用いて，式 4.3 で作成した X^s の拡張を行う．同様に先ほどの例に挙げると，図 4.1 から，工程種別 O_1 は機械種別 M_1 もしくは M_2 にて処理可能であることがわかる．製品 J_1 においては，工程種別 O_1 は 1 番目に処理可能であるため，$Z^{111} = \{o^1_{11}u^{11}_1\}$ となる．

式 4.3 の X^s と式 4.10 で作成される制約条件を用いて，X^s を拡張する．拡張を行うための処理をアルゴリズム 1 に示す．

アルゴリズム 1 の 5 行目は，式 4.3 と同様なものである．X^s/o^s_{wr} は，X^s から o^s_{wr} を含む組合せを取り出し，それら組合せから o^s_{wr} を削除する．o^s_{wr} が削除された組合せに対して，

Z^{swr} との乗算を行う．これにより，製品 J_s の工程種別 O_w が r 番目に処理される際に実施可能な製造工程を表現した組合せが作成される．具体例として先ほどの例題を用いると，アルゴリズム 1 にて $s = 1, w = 1, r = 1$ の時，はじめ X^1 内の組合せは $\{o^1_{11}o^1_{22}, o^1_{11}o^1_{32}\}$ である．また，Z^{111} には $\{o^1_{11}u^{11}_1\}$ の組合せが存在する．5 行目の処理にて，X^1 内の組合せは $\{o^1_{11}o^1_{22}u^{11}_1, o^1_{11}o^1_{32}u^{11}_1\}$ へと変わり，製造工程の選択肢を含む 2 通りの組合せ集合が作成される．最終的に X_s は，工程順序と製造工程の 2 つの選択性の選択肢の組合せを表す ZDD となる．

続いて，機械種別への拡張を行う．各製造工程には処理可能な機械種別が存在する．製造工程 U_k が機械種別 M_i で処理可能で製品 J_s にて製造工程 U_w が r 番目に行う時，ZDD ではこの関係を以下のように定義する．

$$Z^{sru}_a = u^{sr}_k \times m^{sr}_i \tag{4.9}$$

式 4.9 は，製品 J_s の r 番目に実施可能な製造工程と機械種別の 1 通りの組合せを表す．工程種別と製造工程の拡張の際と同様に，Z^{sru}_a の集合を Z^{sru}，$|Z^{sru}|$ の要素数を A とすると，Z^{sru} は式 4.10 のように表現できる．

$$Z^{sru} = Z^{sru}_1 + Z^{sru}_2 + \cdots + Z^{sru}_A \tag{4.10}$$

ここで作成される製品 J_s の r 番目に実施可能な製造工程と機械種別の和集合を表す ZDD は，工程順序，製造工程と機械種別間の制約条件となる．この制約条件を用いて，アルゴリズム 1 で作成した X^s の拡張を行う．同様に，図 4.1 から，製造工程 U_1 は機械種別 M_1 にて処理可能であることがわかる．製品 J_1 においては，製造工程 U_1 は 1 番目に処理可能であるため，$Z^{11} = \{u^{11}_1 m^{11}_1\}$ となる．

拡張を行うための処理をアルゴリズム 2 に示す．

アルゴリズム 2 の 5 行目の処理は式 4.3 と同様である．アルゴリズム 1 と同様に，u^{sr}_k を含む組合せを取り出し，それら組合せから u^{sr}_k を削除する．そして，削除された組合せに対して，Z^{sru} との乗算を行う．これにより，各製品に対して製品計画における 3 つの選択性間にて実施可能な組合せ (解候補) を表現することができる．

次に，製品計画の統合的な解候補集合を作成する．製品計画の統合的な解候補を X とすると，以下のように定義できる．

$$X = X^1 \times X^2 \times \cdots \times X^N \tag{4.11}$$

アルゴリズム 2 製造工程と機械種別間の解候補集合の作成

```
1: s, r1 ← 1
2: while s ≤ N do
3:   while r ≤ R do
4:     while k ≤ K do
5:       X^s ← (X^s / u_k^sr) × Z^sru + (X^s % u_k^sr))
6:       K ← K + 1
7:     end while
8:     k ← 1, r ← r + 1
9:   end while
10:  r ← 1, s ← s + 1
11: end while
```

製品計画での全ての製品の解候補を乗算すると，製品計画での統合的な解候補集合 X が作成される．X は製品計画単体でプランニングを行った際の，実行可能な解の組合せを表す．

続いて，資源計画の資源の数を包含するように式 4.11 で作成してた X の拡張を行う．各機械種別はインスタンスをもつ．生産現場に設置できる機械の台数は有限である．仮に，機械種別 i のインスタンス数が $T_i(1 \leq T_i \leq J)$ とすると，機械種別 i のインスタンスで構成される全ての組合せ集合 m_i は以下の通り定義される．

$$m_i = (m_{i1} + 1) \times (m_{i1} + 2) \times \cdots \times (m_{iT_i} + 1) - 1 \tag{4.12}$$

m_i は，設置可能なインスタンスの組合せ集合を表す．

製品計画にて，製品 J_s の r 番目の順序で製造工程 U_k を機械種別 M_i にて処理する際，機械種別 M_i のインスタンスは 1 台以上設置される．これは，製品計画の選択性である機械種別と資源計画の資源の数との間の制約条件である．ZDD の観点では，製品計画の解候補 X の中で，m_i^{sr} を含む組合せは必ず $m_{i1}, m_{i2} \cdots m_{iT_i}$ を 1 つもしくは複数含むという制約条件となる．この制約を ZDD で表し，X の範囲を資源計画へ拡張する．そのための処理手順をアルゴリズム 3 に示す．アルゴリズムの 6 行目の $Y \leftarrow X.Restrict(m_i^{sr})$ は，X から m_i^{sr} を含む組合せを取り出し，Y に返す．もし，Y が空集合である場合，m_i^{sr} を含む組合せは X 内に存在しないことを意味する．7 行目の $Z \leftarrow Y.Restrict(m_i)$ は，1 通りでも m_i 内の組合せを包含する Y 内の組合せを取り出し，Z に返す．8 行目の $(Z?0 : Y) \times m_i$ は Y の組合せの中で Z の組合せを含まないものを取り出し，それらに対して m_i と乗算を行う．また，$X\%m_i^{sr}$ は，X の組合せの中で m_i の組合せを含まないものを取り出す．そこで得られた結果と Z の和集合を作成する．これにより，機械種別 M_i 上で任意の工程種別が処理

アルゴリズム 3 製品計画及び資源計画間の解候補集合の作成

```
1: s, w, r, i ← 1, ZDD Y, Z = ∅
2: while s ≤ N do
3:   while r ≤ R do
4:     while i ≤ I do
5:       Y ← X.Restrict(m_i^sr)
6:       Z ← Y.Restrict(m_i)
7:       X ← (Z?0 : Y) × m_i + X%m_i^sr + Z
8:       i ← i + 1
9:     end while
10:    i ← 1, r ← r + 1
11:  end while
12:  r ← 1, s ← s + 1
13: end while
```

されることを表す製品計画の組合せに対して，機械種別 M_i のインスタンスの設置を表す変数を追加する．これまでと同様に例題に当てはめて説明する．製品 J_1 の製品計画の解候補内の 1 つである，$\{o_{11}^1 o_{22}^1 u_1^{11} u_2^{12} m_1^{11} m_3^{12}\}$ を例に挙げる．$s, r, = 1, i = 1$ の時，まず 6 行目のステップにて $Y = \{o_{11}^1 o_{22}^1 u_1^{11} u_2^{12} m_1^{11} m_3^{12}\}$ となる．この時，$m_1 = \{m_{11}\}$ であることから，次に 7 行目では $Z = \emptyset$ となる．つまり，8 行目にて $(Z?0 : Y)$ は Y であり，$X\%m_1^{11} = \{\emptyset\}$ であることから，$X \leftarrow Y \times m_i$ となる．ここで $X = \{o_{11}^1 o_{22}^1 u_1^{11} u_2^{12} m_1^{11} m_3^{12} m_{11}\}$ となり，機械種別 M_i が必要な組合せに対して，そのインスタンスの設置を表す変数が追加される．このように製品計画で求められる機械種別のインスタンスが組合せに含まれていない組合せに対してのみ拡張を行う．このアルゴリズムを経て拡張された X は，設置機械台数の上限の制約を考慮しない製品計画と資源計画の組合せ集合を表す．次にこの制約を満たす組合せのみを抽出する．

生産現場に設置可能な機械台数の上限を F とする．組合せの観点では，設置機械台数の上限の制約は，機械のインスタンスを表す変数が F 個より多く組合せに含まれている場合，それら組合せは制約を満たさず，実行不可能な解であることを意味する．まずはじめに，設置する機械のインスタンスの全て組合せを表す ZDD を作成する．

$$m = (m_1 + 1) \times (m_2 + 1) \times \cdots \times (m_I + 1) - 1 \tag{4.13}$$

式 4.13 の m に対して，インスタンス数が F 個より多い組合せを取り出す．

$$m_{ng} = m.Permitsym(F)?0 : m \tag{4.14}$$

$m.Permitsytm(F)$ は m 内の変数 F 以下の組合せを取り出す．そして，If-Then-Else 演算により変数の数が F 個より多い組合せを m_{ng} に格納する．結果として，m は設置機械台数の上限の制約を満たさない機械インスタンスの組合せとなる．

例えば，$m = \{m_{11}m_{12}m_{13}, m_{11}m_{12}, m_{11}m_{13}, m_{12}m_{13}, m_{11}, m_{12}, m_{13}\}, F = 1$ の時，$m.Permitsytm(F)$ は，$\{m_{11}, m_{12}, m_{13}\}$ となる．そして，$\{m_{11}, m_{12}, m_{13}\}$ 以外の組合せが If-Then-Else 演算により m から取り出され，$m_{ng} = \{m_{11}m_{12}m_{13}, m_{11}m_{12}, m_{11}m_{13}, m_{12}m_{13}\}$ となる．

X 内の組合せにおいて，m_{ng} の少なくとも 1 通りを包含する組合せは設置機械台数の上限の制約を満たさず，削除されるべきである．この操作を ZDD 上では以下のように定義する．

$$X \leftarrow X.Restrict(m_{ng})?0 : X \tag{4.15}$$

$X.Restrict(m_{ng})$ は，m_{ng} の組合せを少なくとも 1 通り包含する X 内の組合せを取り出す．さらに If-Then-Else 演算にてこの取り出された組合せ以外の組合せを X 内から取り出す．これにより，X は設置機械台数の上限の制約を満たす組合せ集合となる．

最後に，機械インスタンスの組合せが重複した組合せを削除する．各機械種別のインスタンスは全て同じ性能を持つ．例えば，機械種別 M_i がインスタンス 3 台持つとし，2 台が生産現場に設置されるとする．この時，機械種別 M_i のインスタンスを表す 2 つの変数から構成される組合せは，$\{m_{i1}m_{i2}, m_{i1}m_{i3}, m_{i2}m_{i3}\}$ として表される．機械の性能の観点では，各組合せは同じことを表しているため重複した組合せとなる．このような場合では，早く宣言された変数の組合せが残り，他は削除される．ここでの変数の宣言順序を $m_{i1} \rightarrow m_{i2} \rightarrow m_{i3}$ とすると，上記の例では，$\{m_{i1}m_{i2}\}$ を含む組合せが残り，他は削除される．アルゴリズム 4 は，このような組合せを削除する．

アルゴリズム 4 の 6 行目は，m_i から $count$ 個の変数からなる組合せを取り出し A に代入する．もし A が 1 通り以上の組合せを保持しているならば，これは重複を意味する．8，9 行目では，A の各組合せを 1 通りでも包含している組合せを X から取り出し X から削除する．さらに，10 行目では宣言の早い変数の組合せを A から取り出し C に代入する．そして，C 内の組合せを包含する組合せを B から取り出し B に代入する．最後に X と B の和集合を作成し，X に代入する．例えば，機械種別 M_i が 2 台のインスタンスを保持し，変数 m_{i1} と m_{i2} では m_{i1} が先に宣言された変数であるとする時，$m_i = \{m_{i1}m_{i2}, m_{i1}, m_{i2}\}, T_i = 2$ とな

アルゴリズム 4 重複した機械インスタンスを含む組合せの削除

```
1: ZDD A, B, C = ∅, i = 1
2: while i ≤ I do
3:    count ← T_i
4:    if count > 1 then
5:       while count ≥ 1 do
6:          A ← m_i.Permitsym(count) − m_i.Permitsym(count − 1)
7:          if the number of combinations in A is more than 1 then
8:             B ← X.Restrict(A)
9:             X ← X − B
10:            C ← a combination consisting of earlier declared variables in A
11:            B ← B.Restrict(C)
12:         end if
13:         X ← X + B
14:         count ← count − 1
15:      end while
16:   end if
17:   i ← i + 1
18: end while
```

る．$count = T_i$ なので，まず 6 行目にて $A = \{m_{i1}m_{i2}\}$ となる．A の組合せ数は 1 なので，$B = \{\emptyset\}$ であり，X は変化しない．次に，$count = 1$ となり，6 行目にて $A = \{m_{i1}, m_{i2}\}$ となる．A の組合せ数は 2 なので，これは重複している．次に X から m_{i1} または m_{i2} を含む組合せを取り出し B に代入する．さらに X から B の組合せを削除する．そして，C に宣言順序が早い m_{i1} が代入され，B から m_{i1} を含む組合せを取り出し B に代入する．最後に X と B の和をとり，X に代入する．このアルゴリズムの出力は X であり，X は製品計画と資源計画の制約充足解空間を表す．

4.4　動的事象に伴う制約充足解空間の更新

4.4.1　動的事象と制約充足解空間の関係

動的生産プランニングでは，動的事象が生産実施中に発生する．ここでの動的事象は，以下の 2 つの事象から構成される．

- 生産の実績
- 動的変化（機械故障，特急ジョブ，作業員の欠員など）

ここでの生産の実績は，予定されている作業が完了した際の成果を表す．例えば，生産実施中にある製品の 1 工程の完了する．これが生産の実績に該当する．実績が積まれると，これまで実行可能であった代替候補が実行不可能になる．つまり，生産の実績は，解空間に対しての制約として捉えることができる．そこで本研究では，生産の実績を制約条件として ZDD 上で扱う．

また，本論文では動的変化として機械故障を扱う．機械故障は，生産の実績と同様に解空間に対して代替計画候補を制限するため，同様に ZDD 上では制約として扱う．つまり，これら動的事象は，ZDD に新たな制約条件を付加することとなる．次項からは，ZDD の演算を用いて，これら制約条件を定式化し，動的事象が発生するつど，ZDD が表す制約充足解空間を更新する方法について述べる．

4.4.2 動的事象に適応する制約充足解空間の更新

動的生産プランニングでは，生産実施中にある製品の 1 工程が完了すると，計画を変更する際の代替案の候補が制限される．同様に，機械故障により実施中の生産計画のみならず代替候補であった生産計画が実行不可になる可能性がある．つまり，生産の実績と動的変化は解空間に新たな制約を与えると捉える事ができる．また，本研究が対象とする生産プランニング問題は，一度生産が開始されると，機械のレイアウトが固定される．機械故障と，生産の実績と同様に，これも制約充足解空間に対して制約として扱われる．以降は，レイアウトの固定の制約，生産の実績の制約，機械故障による制約の順に，制約充足解空間を更新する手法について述べる．

(a) レイアウトの固定

本論文で扱う動的生産プランニングの問題では，一度生産計画が決定すると設置された機械インスタンスは固定され，故障した機械は生産終了まで修復されないこととする．それ故，一度生産が開始されると，ZDD の観点では生産現場に設置された各機械インスタンスの台数は制約となり，最初に実施される生産計画と同じ機械インスタンスの設置パターンをもつ生産計画の候補が残る．初期の制約充足解空間を表す ZDD を X とすると，この制約を満さない ZDD 上の組合せは以下の順序で削除される．

$Step.1$ $Q_{ng} = Q_{all} - Q_{all}.Permitsym(q_c)$

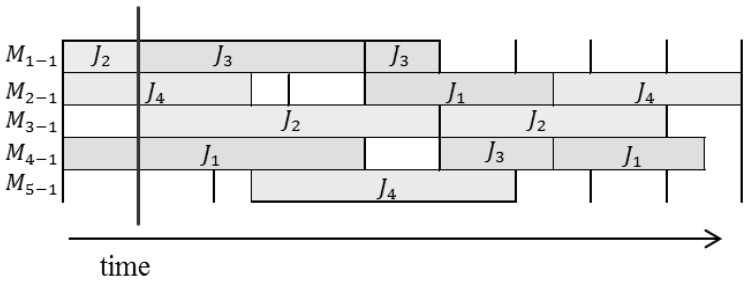


図 4.4: 生産実施中の生産計画のガントチャート (例)

Step.2 $X \leftarrow X - X.Restrict(Q_{ng})$

Step.3 $X \leftarrow X.Restrict(Q_{now})$

第 1 ステップにおいて，Q_{all} は設置機械の全組合せ集合を表し，Q_{ng} は決定された生産計画で使用する機械の台数 q_c 以上の機械の組合せを含む組合せを Q_{all} から除いた組合せ集合を表す．第 2 ステップでは，制約充足解空間 X から Q_{ng} を含む組合せを取り除く．最後に第 3 ステップにおいて，現行の生産計画での設置されている機械の組合せ Q_{now} を含む組合せが取り出される．これにより，ZDD が保持する組合せは，現行の生産計画と同様の機械レイアウトを用いる組合せの集合となる．

(b) 生産の実績

動的生産プランニング問題では，生産計画実施中に動的変化と生産の実績が動的に報告される．ここでの生産の実績は，ある製品の 1 工程が完了することを意味する．ZDD において，生産の実績は完了した工程を表す変数は必ず組合せに含まれる．図 4.4 は，生産計画のガントチャートの例である．図中の $M_{1-1} \cdots M_{5-1}$ は機械種別 $M_1 \sim M_5$ がそれぞれ 1 台ずつ配置されていることを意味し，$J_1 \sim J_4$ は製品を表す．例えば，J_2 の 1 番目に処理される工程種別が完了したとする．この際，解候補として ZDD 上に残る組合せは必ず J_2 の完了した工程種別及び製造工程を表す変数が含まれている．生産実施中に製品 J_s の r 番目に処理される工程種別 O_w に製造工程 U_k を割り当て，機械種別 M_i 上で完了した時，ZDD で表されている制約充足解空間 X とすると，この生産の実績の制約を満たす組合せは以下の方法で取り出すことができる．

$$X \leftarrow X/(o_{wr}^{s} \times u_{k}^{sr} \times m_{i}^{sr}) \tag{4.16}$$

式 4.16 は，X の各組合せから $\{o^s_{wr}u^{sr}_k m^{sr}_i\}$ の組合せを含むものを残してそれ以外を削除し，残った組合せから変数 o^s_{wr}，u^{sr}_k と m^{swr}_i を取り除く．例えば，図 4.4 に示すガントチャートにて J_2 の 1 番目に処理される工程種別が O_2 であり，製造工程 U_1 に割り当てられていたとすると，$X \leftarrow X/(o^2_{21} \times u^{21}_1 \times m^{21}_1)$ となる．これにより，X は，常に生産の実績を満たし，残りの生産計画の候補を表す ZDD となる．

(c) 機械故障

機械故障は，現行の生産計画を実行不可にする可能性がある．もし，ある機械種別が現場に 1 台のみ設置されており，これが故障した場合，計画を変更する必要がある．2 台以上ある場合は，残りの機械に割り付けることで実行可能であるが，新たに割り付けられた機械の仕事が多くなり納期に遅れる可能性はある．例えば，機械種別 M_z が現場に 1 台のみ設置されており，故障したとする．この場合，機械種別 M_z を用いる工程は，生産計画の代替案として候補から外れる．機械故障による制約を満たす組合せを求める式を以下に示す．

$$X \leftarrow X - X.Restrict(m^{sr}_z) \tag{4.17}$$

式 4.17 は，制約充足解空間から，製品 J_s の r 番目に製造工程を機械種別 M_z にて処理する組合せを取り出し削除する．この式を s, r の数だけ繰り返すことで，機械故障による制約を満たす組合せを導出する．得られた ZDD が空集合を表す場合，生産プランニング全体として実行可能な解はないという結果が得られる．次章からは，制約充足解空間を表す ZDD から GA を用いて解を探索する手法について述べる．

4.5 ZDD 内の解探索用遺伝的アルゴリズム

4.5.1 遺伝的アルゴリズムを用いたゼロサプレス型 BDD 内の解探索

本論文では制約充足解空間を表す ZDD の解探索方法として GA を用いる．対象とするプランニング問題が，N 個の論理変数 $x_1, x_2, ..., x_N$ により与えられているとする．これを論理変数順に並べた N ビットの 2 進数としてエンコーディングした場合，膨大な取り得ない組合せ (致死遺伝子) を解空間表現に含んでしまう．このエンコーディング方法では，ZDD の制約充足解空間内に存在しない解を GA で探索してしまう．そのため，本論文では

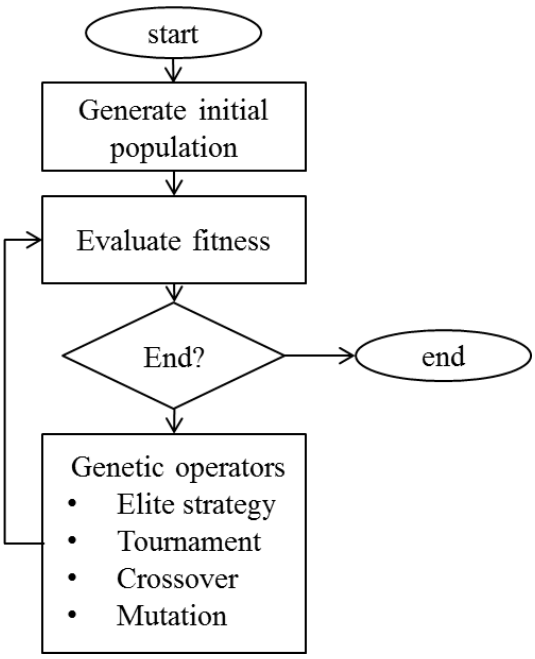


図 4.5: 提案する GA の流れ

遺伝子操作を行っても常に制約充足解空間内を探索するエンコーディングを行う．図 4.5 に示すように提案する GA では，以下の処理を行う．

- 初期個体群生成：初期世代での個体の作成
- 適応度評価：各個体が表す組合せの評価
- エリート戦略：遺伝子操作の選択その 1
- トーナメント選択：遺伝子操作の選択その 2
- 交叉及び突然変異

各処理について解説するに辺り，まずはじめに，エンコーディングについて述べる．

エンコーディング

提案する GA のエンコーディングでは，以下のルールに従い制約充足解空間の記号列で表現している．

1. 遺伝子の値を 0/1 とし，長さ L の 2 進数で符号化する (ex.1011...00)．L の大きさは，パスの最大長さが不明なため初期個体では予めランダムに複数選択された 1-パスの

中の最大の長さと定数 α の和とする．初期個体以降では，p 世代の遺伝子長 L_p は，$p-1$ 世代までの 1-パスの最大の長さ $Pass_{max}$ と α の和とする．もし，p 世代で探索した 1-パスが $Pass_{max}$ よりも長い場合は，$Pass_{max}$ を更新する．

2. 各遺伝子は論理変数の値の割り当てではなく，ZDD 上の 1-パスにおけるノード出現順での 0/1 枝の選択を表す．

3. 0 はノードの選択肢の内の 0 枝を選ぶ．(ただし，終端ノード 0 を直接指している場合，1 枝を選ぶ．)

4. 1 はノード選択肢の内の 1 枝を選ぶ．(ZDD には必ず選択肢が 1 枝に存在する．)

5. 終端ノードに達した後の遺伝子は無視する．

こうしたエンコーディングにより，ランダムに発生させた遺伝子配列は必ず ZDD 上の頂点ノードから終端ノード 1 のパス (1-パス) に対応する．また，交叉や突然変異といった遺伝子操作を行っても致死遺伝子は生じない．これにより GA は常に ZDD 上に存在する組合せのみを探索する．

図 4.6 の宣言した論理変数の数が 5 である ZDD を用いて説明する．ここでは定数 α を 2 とし，遺伝子長 L を 5 とする．個体の遺伝子配列が (10000) を表す時，ZDD 上の 1-パスでは，頂点ノードの変数の取る値は 1 となり 1 枝を選択する．次に ZDD 内に現れるノードの変数 x_2 の取る値は 0 となり 0 枝を選択する．この 0 枝は変数 y_1 のノードを指す．変数 y_1 の取る値は 0 となる．そして，変数 y_3 の取る値は 0 となるが，0 枝は終端ノード 0 を指しているため 1 枝を選択する．そして 1 枝は終端ノード 1 を指しているため残りの遺伝子は無視される．したがって，この個体は組合せ $\{x_1y_3\}$ を表す 1-パスを意味する．

このエンコーディング方法では，遺伝子配列と ZDD 上の 1-パスの対応関係が 1:1 ではなく 1 : N といった関係をもち，遺伝子配列が異なっていても同じ 1 パスを指すことがある．

初期個体生成

はじめにランダムに複数の 1-パスを選択し，その中から頂点ノードから終端ノード 1 までに通過したノード数の最大数と定数 α の和を遺伝子長として設定する．初期個体は，得られた遺伝子長の遺伝子配列が用意され，配列内の各遺伝子の値 (0/1) はランダムに決定される．

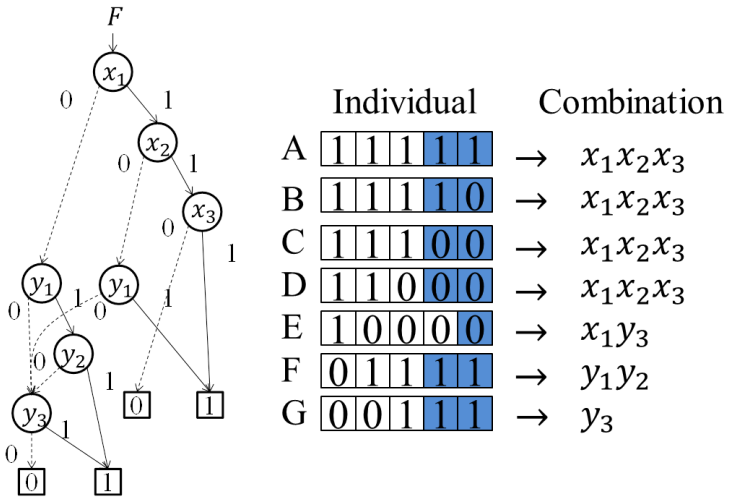


図 4.6: 個体が表す ZDD 上の組合せ表現

デコーディング

ZDD 上の論理変数は宣言した順に変数番号が与えられる．図 4.6 の ZDD 上の変数が $(x_1, x_2, x_3, y_1, y_2, y_3)$ の順に宣言されたとするとそれに対応する変数番号は $(1, 2, 3, 4, 5, 6)$ となる．デコーディングでは，各変数の変数番号を用いる．変数番号は選択性 A の選択肢，選択性 B の選択肢の様に選択性の選択肢を順に一定の規則で宣言している．そのため，変数番号がどの選択性のどの選択肢を意味しているのか判断でき，工程設計案，設備レイアウトといったスケジューリングを行うために必要な情報を読み取ることができる．

適応度の算出

各個体は適応度をもつ．デコーディングにより得られた情報に基づきスケジューリングが行われる．本研究では，スケジューリング手法としては，ディスパッチング・ルール (差立ロジック) と呼ばれるヒューリスティックを用いる．ディスパッチング・ルールは，処理可能な機械に，着手可能な製品の中から，次にその機械が割り当てる製品を決定するために用いられる規則である．これまでに様々なディスパッチング・ルールが提案されている [40]．これらディスパッチング・ルールの例として，

- FIFO: 製品の到着時刻の早いものから処理する先着順の規則 (First-In First-Out の略称)

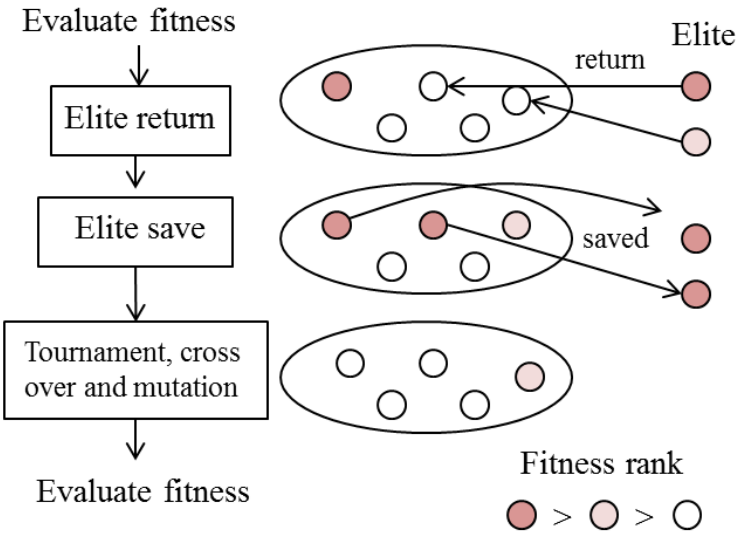


図 4.7: 提案する GA におけるエリート戦略

- SPT: 処理時間の短いものから順に処理をする規則 (Shortest Processing Time の略称)
- EDD: 納期の近いもの順に処理をする規則 (Earliest Due Date の略称)

などが挙げられる．本論文では，スケジューリング手法としてディスパッチング・ルールの SPT と EDD ルールを用いる．スケジューリングにより得られた目的関数 TAC の値を適応度として扱う．

選択と遺伝子操作

GA の選択方法としてはルーレット戦略，トーナメント戦略，ランキング戦略など様々な方法があり，それぞれの手法ごとに依存するパラメータによって操作される．この選択操作におけるランダム性を排除して適応度の高い個体を強制的に残すのがエリート戦略である．本 GA で用いるエリート戦略の流れを図 4.7 に示す．本研究で用いるエリート戦略では，ある一定数のエリートを個体群から選ぶエリート保存とエリート戻しから構成される．エリート戻しは，遺伝子操作後に最も適応度が悪い個体と入れ替わり，エリート保存は各世代ごとに行われる．

一般的にエリート保存戦略は，解の収束性を飛躍的に増大させるが，局所解に陥りやすいとされている．そこで，選択方法としてトーナメント選択を用いる．トーナメント選択

では，エリートを含む一定数の個体がランダムに選択される．選択された個体はトーナメントへ参加し適応度の比較により，勝者が決まる．これらは，最大化問題であれば適応度が高い値をもつ個体，最小化問題であれば適応度が低い値をもつ個体が勝者となる．勝者は遺伝子操作の親として選択される．

親が選択された後，遺伝子操作が行われる．遺伝子操作においては，交叉，突然変異を用いる．図 4.8 は，図 4.6 の個体 D と E，D と F が親 1，2 として選ばれた際の交叉及び突然変異の操作の例を示す．交叉は，一点交叉を行う．一点交叉は，ランダムに切断する箇所を選択する．切断箇所の範囲は，親 1，2 の遺伝子配列にてデコーディングした際 1-パスの通過したノードの数が少ない方が選択される．図 4.8 の例 1 では，親 2 の方が，1 よりも 1 -パスで通過したノード数が少ないため，選択範囲は 1 ～ 3 の遺伝子番号の範囲で選択される．例 2 では，選択範囲は 1 ～ 4 となる．ここでは，ランダムに選択された切断箇所の範囲内の遺伝子が入れ替わり，子として生成される．これにより子 1,2 は最初の切断箇所の遺伝子までは親 1,2 と同じ 1-パスをたどる．例 1 では，親として遺伝子配列 (10000) と (01111) を例にあげている．この際，遺伝子番号が 1 と 2 の間が選択された場合，子 1 は (11100)，子 2 は (00011) の遺伝子配列をもち，それぞれ $\{x_1, x_2, x_3\}$ と $\{y_3\}$ の組合せを表す．

突然変異は，遺伝子 (0/1) を他の対立遺伝子 (1/0) に置き換える方法を取る．エンコーディングにおいて，遺伝子の値が 0 で，0 枝が終端ノード 0 を指している時は，1 枝に進む規則があるため，1-パスと遺伝子配列の関係が 1:N の関係となっている．そのため，この規則に該当する遺伝子に対して突然変異が行われると，子は親と同じ 1-パスを選択する．図 4.8 の突然変異の例では，親 (10000) が選択されている．この遺伝子配列が辿る 1-パスは，出現ノード順に $(x_1 \to x_2 \to y_1 \to y_3)$ となっている．ZDD では，1 枝の先は必ず終端ノード 1 へつながっており，0 枝の先が直接終端ノード 0 を指していない場合も必ず終端ノード 1 へつながっている．そこで，突然変異が発生した際に，各ノードで選択した枝とは反対の枝の先が終端ノード 0 を直接指していない場合，分岐点として対象となる遺伝子番号を配列に格納する．親 (10000) では，1，2，3 番目のノードは反対の枝が選択された場合でも終端ノード 1 へと繋がる．これら番号を配列に順に格納する．突然変異では，1 から分岐点の総数までの範囲で値がランダムに選ばれる．この例では，選択範囲は 1 ～ 3 となる．仮に 2 が選択された場合，分岐点配列の 2 番目の値は 2 であることから，親の遺

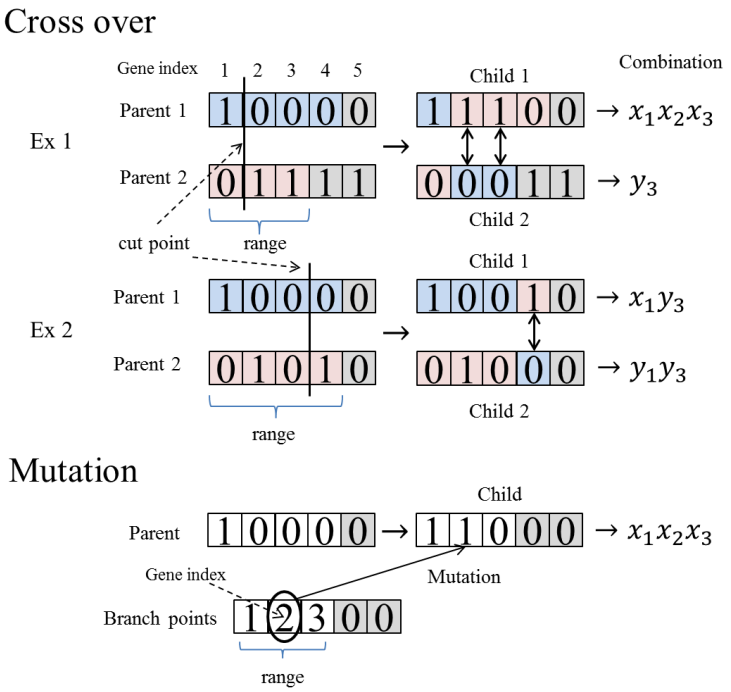


図 4.8: 提案する GA における交叉と突然変異

伝子配列の 2 番目の遺伝子の値が反転する．最終的に遺伝子配列 (11000) をもつ子が生成される．この操作により，突然変異では対象となる親とは異なる 1-パスを表す遺伝子配列をもつ子が生成される．

ZDD の 1-パスを表す遺伝子配列では，仮に終端ノード 1 に近い遺伝子が突然変異により変更されると，子の遺伝子配列は親の 1-パスの近傍を探索する．一方，頂点ノードに近い遺伝子が変更されると，親の 1-パスと ZDD 上の距離として離れた 1-パスを表す遺伝子配列となる可能性をもつ．

4.5.2 探索済み 1-パスを考慮した遺伝子配列の変更

提案する GA では遺伝子操作において致死遺伝子を発生させないために，個体の遺伝子配列は，ZDD 上のノードの枝の選択を表し，終端ノード 0 に到達する場合，1 枝を辿ることにより必ず 1-パスを表す．それ故，1-パスと遺伝子配列は 1:N の関係をもつ．そのため，交叉や突然変異により，解の収束が過剰に行われ，初期収束する可能性をもつ．これを回避するために，探索済みの 1-パスをキャッシュし，未探索の 1-パスのみを対象とする解探索方法について提案する．この方法では，エリートが表す 1-パスを除き，一度探索された

1-パスは再度評価されない．

重複回避のためのキャッシュと遺伝子配列の変更方法

図 4.6 の ZDD において，遺伝子配列 (01111) と (00111) では，出現するノードの 0/1 枝を (01) と選択した後終端ノード 1 に到達し同じ 1-パスを表す．本論文では，遺伝子配列から得られる ZDD 上の 1-パスを 0/1 の文字列で表したものを実パスと呼ぶ．実パスをキャッシュし，新たに遺伝子配列から実パスへ変換された際にキャッシュを参照し，1-パスが探索済みか否かの判定を行う．実パスがキャッシュに存在しない場合は，実パスを新たにキャッシュに追加され，探索済みの 1-パスとして扱われる．探索済みの 1-パスに対しては，実パスの遺伝子配列の一部を変更する．そして，再び実パスに変換し，キャッシュ内を調べる．遺伝子配列の変更方法として，以下の 3 つをテストする．

- top : 実パスを左から右へ進んだ時に，最初に現れる値が 0 の遺伝子を 1 に変更する．
- bottom : 実パスの右から左に進んだ時に，最初に現れる値が 0 の遺伝子を 1 に変更する (ただし，開始位置は，終端ノード 1 を指す枝の選択に対応する遺伝子である)．
- random : 遺伝子をランダムに 1 つ変更し，遺伝子の値を反転する (突然変異と同様)．

ZDD の簡略化規則では，ノードの 1 枝が終端ノード 0 を指す時，ノードを削除するという規則がある．つまり，ZDD 内の 1 枝は必ず終端ノード 1 のパス上に存在する．遺伝子 0 は 0 枝を選択しているため，この値を 1 に変更することで元の遺伝子配列が指す 1-パスとは異なる他の 1 -パス上のノードを選択する．もし，遺伝子配列を 1 から 0 に変更した場合，その変更により 0 枝が直接終端ノード 0 を指す可能性があり，GA のルールにより元の遺伝子配列に戻ってしまう．そのため，値が 0 である遺伝子を変更している．top では，ZDD の頂点ノードから指し夜に 0 枝を選択しているノードに対して 1 枝を選択するように変更する．逆に bottom では，終端ノード 1 から頂点ノードへと遡り，最初に 0 枝を選択しているノードに対して 1 を選択するように変更する．random は，突然変異と同様に，分岐点配列を用いて，変更する遺伝子をランダムに 1 点指定する．そして，遺伝子配列の該当する遺伝子の値を反転する．仮に，top と bottom において ZDD 上の最右の 1-パスが探索済みの場合は，遺伝子配列は方法 3 により変更される．

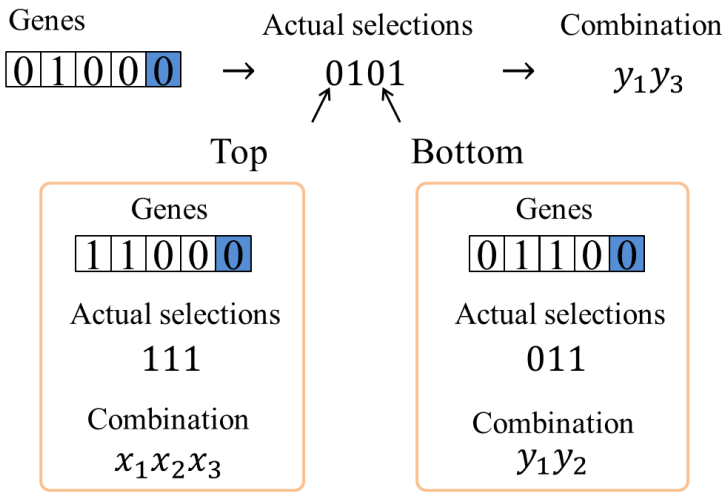


図 4.9: 探索済み 1-パス時の遺伝子配列変更方法 1,2 の例

具体例を図 4.9 に示す．対象とする ZDD は図 4.6 の ZDD を用いる．遺伝子配列として (01000) があるとする．この遺伝子配列の ZDD 上の 1-パスは組合せ $\{y_1y_3\}$ を表し，実パス (0101) である．この時，この 1-パスが探索済みである場合，top では実パスの左から最初の 0 を選択する．最初の 0 は一番目にあることから，遺伝子配列の左から一番目 0 を 1 に変更し，遺伝子配列 (11000) が作成される．遺伝子配列 (11000) は，組合せ $\{x_1x_2x_3\}$ を表し，実パスは (111) となる．bottom の場合は，実パスの右から最初の 0 を変更するため，遺伝子配列の左から 3 番目の 0 が選択される．つまり，遺伝子配列は (01100) に変わり，実パス (011) である組合せ $\{y_1y_2\}$ となる．

4.6 本章のまとめ

本章では，生産プランニングの制約充足解空間を ZDD にて表現するための解法を提案した．具体的には，はじめに静的生産プランニング問題における制約充足解空間の表現方法ついて提案した．次に，生産実施中に発生する動的事象について述べた．本研究では，動的事象として生産の実績と機械故障をとりあげ，それら事象を ZDD に対する制約として扱い，新たに制約が発生した際に常に制約充足解空間を整合して更新する方法を提案した．

また，制約充足解空間を表す ZDD を対象とした GA による解探索手法を提案した．GA のエンコーディングにおいて，個体が常に ZDD 上の 1-パスを表す方法について提案し，解探索中は常に制約充足解のみを得ることができる．また，世代間や世代内で発生する探

索済みの 1-パスに対して，遺伝子配列が表す実パスをキャッシュし，常に未探索の 1-パスのみを対象に解探索を行う方法を提案した．これらを用いた評価実験については，第 6 章にて述べる．

第5章　ゼロサプレス型BDDネットワークによる制約充足解空間表現

5.1　ZDDのネットワーク表現の概要

単一のZDDによる制約充足解空間表現では，製造工程，工程順序，機械種別，資源の数の4つの選択性を扱う問題に対して，4種 $(o_{wr}^{s}, u_{k}^{sr}, m_{i}^{sr}, m_{ij})$ のZDDの変数を定義した．製品計画で用いた変数は，製品，工程を処理する順序に基づき定義されているため，大規模の問題ではZDD上に宣言する変数の数が増える．そのような大規模問題にて単一のZDDによる表現を用いるとZDDのノード数が増加に伴い，計算時間や計算機への負荷が増大する可能性がある．それのため，本節ではZDDを連結したネットワーク表現を用いて制約充足解空間を表す方法について述べる．

ZDDネットワークのアプローチ

図5.1はZDDネットワークの全体像を表す．この手法では，ZDDは各選択性の選択肢の組合せ，または選択性間の制約条件を表現し，個別に管理する．そして，選択性の選択肢の組合せを表すZDDの構造が変化すると，変化に伴い他の選択性のZDDが選択性間のZDDが表す制約条件を連動して満たすネットワーク構造を形成する．ZDDネットワークを形成するためには大きく分けて3つの段階を経る．

1. 各選択性の選択肢の組合せ集合をZDDにて作成
2. 各選択性内の制約条件のZDD表現及び各選択性のZDDに対してそれら制約を満たす組合せの抽出
3. 選択性間の制約条件のZDD表現及び各選択性のZDDに対して選択性間の制約を満たす組合せの抽出

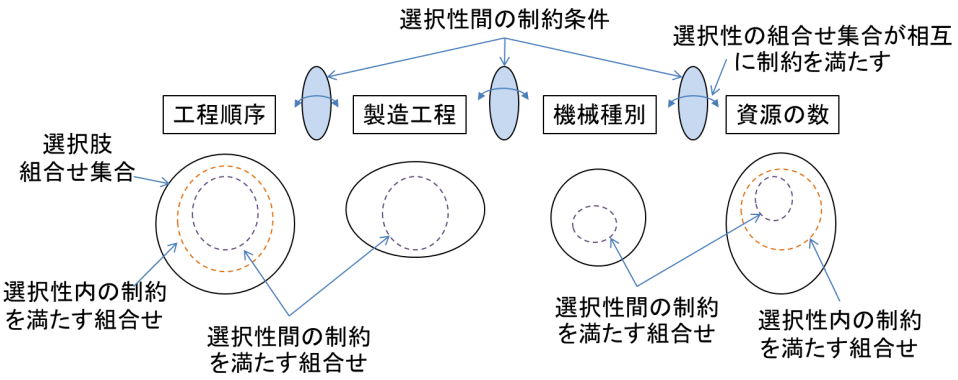


図 5.1: ZDD ネットワークの全体像

もし，ある選択性の選択肢の組合せを表す ZDD の組合せ数が変化すると，他の選択性の ZDD が相互作用して変化する．これにより，各選択性の組合せを表す ZDD は全体としての制約条件を満たす集合をそれぞれ表す．

5.2 ZDD ネットワークの解法

5.2.1 選択性内の組合せ集合

ZDD ネットワークでは，各選択性は，選択肢の組合せ集合を表す ZDD を保持する．ZDD ネットワークにおいて用いる ZDD の変数を以下に示す．

o^s_{wr}：製品 J_s の r 番目に工程種別 w を用いる．

u_k：製造工程 U_k を用いる．

m_i：機械種別 M_i を用いる．

m_{ij}：機械種別 M_i のインスタンス j を設置する．

これら 4 種類の変数を用いて本論文で扱う生産プランニング問題の制約充足解空間を表す．

まずはじめに，各選択性の選択肢の全組合せを表す集合を作成する．仮に，ある選択性内の変数が $a_1, a_2, a_3, \cdots, a_x$ とあるとすると，全ての組合せを表す ZDD X は以下のように定義できる．

$$X = (a_1 + 1) \times (a_2 + 1) \times (a_3 + 1) \times \cdots \times (a_x + 1) \tag{5.1}$$

式 5.1 では，ZDD X が空の組合せである $null$ を含む全ての組合せを表す集合となる．例えば，変数の数が 3 の時では，ZDD X は $\{a_1a_2a_3, a_1a_2, a_1a_3, a_2a_3, a_1, a_2, a_3, 1\}$ となる．しか

し，選択性はその選択性内での選択肢の制約関係をもつ．例えば，選択性内において変数が必ず 1 つ選択され，また排他的に選択される制約がある場合は，X の $\{a_1a_2a_3, a_1a_2, a_1a_3, a_2a_3, 1\}$ はこの制約を満たさない．それ故，次の段階では，それら制約関係を ZDD で表し，それら制約を満たさない組合せを排除する．

5.2.2　選択性内の制約関係

選択性には自身の選択肢同士の制約関係をもつものがある．本論文で扱う選択性では，工程順序と資源の数がこれらに該当する．

工程順序では，単一の ZDD の同様の方法に式 4.2，4.3 用いる．式 4.3 で得られる X^s を用いて，全製品に対しての制約条件を ZDD で表すと，以下のように定義できる．

$$C_{all} = X^1 \times X^2 \times \cdots \times X^N \tag{5.2}$$

A_s が工程順序の選択肢の全ての組合せの集合を表している時，この制約を満たす組合せは If-Then-Else 演算を用いて以下のように得られる．

$$A_s \leftarrow A_s?C_{all} : 0 \tag{5.3}$$

式 5.3 では，C_{all} の組合せの内 A_s に含まれる組合せを取り出し A_s に代入する．これにより，A_s から制約を満たさない組合せは排除される．

資源の数では，機械種別 M_i のインスタンスは同じ性能であるという条件がある．例えば，機械種別 M_2 のインスタンスが 3 台あるとする．そのうち 2 台を設置するとした場合，組合せとして $\{m_{21}m_{22}, m_{21}m_{23}, m_{22}m_{23}\}$ の 3 通りが考慮される．各インスタンスは同じ性能であることから，これら組合せは重複した表現となる．そのため，このような重複した組合せを削除する必要があり，3 通りの内 1 通りの組合せが選択肢の組合せとして含まれていればよい．ここでは，2 台設置する場合，インスタンスの番号が変数の組合せを残すとする．つまり，組合せとして $\{m_{21}m_{22}\}$ のみが残ればよい．A_r が資源の数での選択肢の全ての組合せの集合であり，機械種別の種類が I とすると，アルゴリズム 5 を用いて重複した組合せを排除する．このアルゴリズムでははじめに，機械種別 M_i の J 個のインスタンスからなる組合せを取り出す (5 行目)．そして取り出した組合せを ZDD t へと代入する (7 行目)．さらに，9 行目にてインスタンス番号が遅い変数を 1 つ削除した組合せを作成し c に代入

アルゴリズム 5 資源の数における重複した組合せの削除

```
1: I, J, ZDD A_r
2: i ← 1, j ← 1, ZDDb, c ← {1} //{1} is "null"
3: while i ≤ I do
4:    while j ≤ J do
5:       c ← c × m_ij, j++
6:    end while
7:    ZDD t ← c, l ← J
8:    while l > 1 do
9:       c ← c/m_il
10:      t ← t + c
11:      l−−
12:   end while
13:   b ← b × (t + 1)
14:   c ← {1}, j ← 0, i++
15: end while
16: A_r ← A_r?b : 0
```

する．そして，c と t を加算し t に代入する．これを繰り返し，重複する組合せを排除する．最後に，インスタンスが重複しない組合せのみで作成されている集合 b を作成し，A_r から b に含まれる組合せを取り出す．例として，先ほどの機械種別 M_2 を挙げる．M_2 のインスタンスは 3 台あるため，はじめに $t = \{m_{21}m_{22}m_{23}\}$ が作成される．次に，$c = \{m_{21}m_{22}\}$ が作成され，t に加算される．これにより $t = \{m_{21}m_{22}m_{23}, m_{21}m_{22}\}$ となる．さらに，$c = \{m_{21}\}$ が作成され，同様に加算される．最終的に $t = \{m_{21}m_{22}m_{23}, m_{21}m_{22}, m_{21}\}$ となり，b との乗算が行われる．そして，A_r から重複しない組合せが取り出され，A_r に代入される．

また，機械種別 α の利用可能なインスタンスの数が $\beta(0 \leq \beta \leq J)$ とすると，アルゴリズム 6 を用いて利用不可能なインスタンスを含む組合せを排除する．このアルゴリズムでははじめにインスタンス番号が β 以上の変数の和集合を作成する．そして，それら変数を含む組合せを A_r から取り出し，A_r から削除する．例えば，$A_r = \{m_{11}m_{12}m_{13}, m_{11}m_{12}, m_{11}\}$ で $\beta = 2$ の時，$ng = \{m_{13}\}$ となる．そして，m_{13} を含む組合せが A_r から削除され，$A_r = \{m_{11}m_{12}, m_{11}\}$ となる．

さらに，資源の数では生産現場に設置可能な機械の台数は有限であるため，その台数以上の機械インスタンスを設置することはできない．その上限数を F とすると，F 以上の

アルゴリズム 6 利用不可能な機械インスタンスを含む組合せの削除

1: β, J, ZDD A_r
2: $j \leftarrow \beta + 1$, ZDD $ng \leftarrow \emptyset$
3: **while** $j \leq 1$ **do**
4: $\quad ng \leftarrow ng + m_{\alpha j}, j++$
5: **end while**
6: $A_r \leftarrow A_r.Restrict(ng)?0 : A_r$

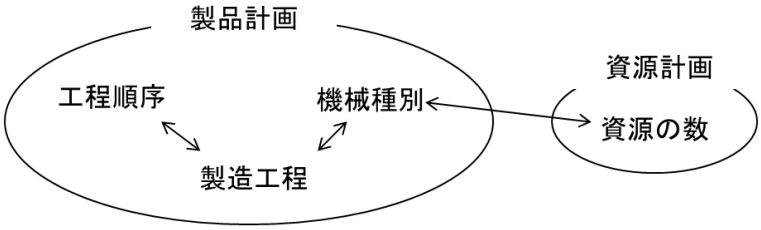


図 5.2: 選択性間の関係性

機械インスタンスの設置を表す組合せは以下の式を用いて削除される．

$$A_r \leftarrow A_r.Permitsym(F) \tag{5.4}$$

これにより，資源の数の選択肢の組合せ集合 A_r は選択性内の制約を満たす組合せ集合となる．

5.2.3　選択性間の制約関係と ZDD の連結によるネットワーク

各選択性の選択肢は他の選択性の選択肢と関係をもつ．例えば，工程種別を限定すると，その選択に伴い製造工程の選択肢が限定される．生産プランニングには，このような選択性間関係が存在し，この関係が選択による波及を生む．図 5.2 は，本論文で扱う選択性の定性的な関係を表している．図中の矢印は直接的な波及関係を表す．工程順序の選択肢の組合せ数が変わると，製造工程の選択肢が影響を受け，選択可能な組合せが変わる可能性がある．ZDD ネットワークでは，このような関係を ZDD 上にて制約条件として扱う．

選択性間の制約条件を ZDD にて管理，表現する方法を次から述べる．

工程順序と製造工程

まずはじめに，工程順序と製造工程間の制約について述べる．工程種別は，割り当て可能な製造工程が存在する．工程種別 O_w に割り当て可能な製造工程を u_x, と u_y とすると，

アルゴリズム 7 製造工程から工程順序に対する制約条件の作成

```
1: s ← 1, r ← 1, w ← 1, k ← 1
2: ZDDY ← ∅
3: for k = 1 to K do
4:   for w = 1 to W do
5:     ZDD A ← 1
6:     if U_k is available for O_w then
7:       for s = 1 to N do
8:         for r = 1 to R do
9:           if o^s_wr included in the ZDD of the process sequence then
10:            A ← A × (o^s_wr + 1)
11:          end if
12:        end for
13:      end for
14:    end if
15:    A ← A − 1
16:    Y ← Y + A
17:  end for
18: end for
```

製品 J_s の r 番目の工程種別 O_w に対しては以下のように定義できる．

$$Z_l^s = o_{wr}^s \times (u_x + u_y) \tag{5.5}$$

式 5.5 は，o_{wr}^s が選択されている時は，製造工程 u_x もしくは u_y が必要であることを表す．この関係を工程種別及び順序ごとに作成し，それら組合せの和集合 Z^s を作成する．Z^s の要素数を L とすると，以下のように定義される．

$$Z^s = Z_1^s + Z_2^s + \cdots + Z_L^s \tag{5.6}$$

さらに，各製品の Z^s を加算する．

$$Z = Z^1 + Z^2 + \cdots + Z^N \tag{5.7}$$

Z は，選択性の工程順序から製造工程に対しての制約の和集合をあらわす．

製造工程から工程順序に対しての制約条件では，製造工程 u_x が選択される時，工程種別 O_1 を用いることを表す変数は選択される．アルゴリズム 7 にこの制約を作成するための手続きを示す．各製造工程の選択肢に対して，選択の可能性のある工程順序の変数の全組合せ A を作成し，それの和集合 Y を作成する．9 行目の条件は，工程種別 O_w の変数

o^s_{wr} が工程順序の ZDD に含まれているかを判定する．．工程順序の ZDD を A_r とすると，$A_r.Restrict(o^s_{wr})$ が空集合でない時組合せを含むことを意味する．

製造工程と機械種別

製造工程と機械種別間の制約は,「工程種別は処理可能である機械種別にて処理される」及び「機械種別は処理可能な工程種別を処理する」の 2 つである．前者は，製造工程からの観点の制約であり，後者は機械種別の観点である．まず製造工程からの観点での制約条件を ZDD にて記述する．製造工程 U_k が機械種別 M_x または M_y もしくは両方にて処理可能であるとすると，この制約は以下のように定義できる．

$$E_k = u_k \times (m_x + m_y + m_x \times m_y) \tag{5.8}$$

E_k は，工程種別 O_w は機械種別 M_x か M_y のみ，または M_x と M_y どちらでも処理されることを表す．そして，この制約を表す ZDD が製造工程の数 (K) だけあるので，それらを和集合にて表現すると以下ようになる．

$$E = E_1 + E_2 + \cdots + E_K \tag{5.9}$$

E は，製造工程が処理される機械種別の制約を表す ZDD となる．

続いて，機械種別の観点では，機械種別 M_i で処理可能な製造工程は U_α, U_β であるとした場合，以下のように定義できる．

$$H_i = m_i \times (u_\alpha + u_\beta + u_\alpha \times u_\beta) \tag{5.10}$$

H_i は，E_k と同様に機械種別 M_i は工程種別 U_α か U_β のみ，もしくは両方を処理することを表す．機械種別数は I であることから，この和集合 H は以下の通り定義できる．

$$H = H_1 + H_2 + \cdots + H_I \tag{5.11}$$

E は，機械種別が処理可能である製造工程の制約を表す ZDD となる．

機械種別と資源の数

機械種別と資源の数間では,「機械種別 M_i を用いる時は，少なくとも M_i のインスタンスが 1 台以上設置される」という制約が存在する．この制約を ZDD で表すと以下のとお

りとなる．

$$D_i = m_i \times V_i V_i = (m_{i1} + 1) \times (m_{i2} + 1) \times \cdots \times (m_{iJ} + 1) - 1 \tag{5.12}$$

V_i は機械種別 M_i のインスタンスの全組合せの集合を表す．D_i は機械種別 M_i が求められる時，機械種別 M_i のインスタンス必ず設置されることを表す制約条件である．D_i は，機械種別の数だけあるので，制約条件を加算を用いて 1 つの集合として表す．

$$D = D_1 + D_2 + \cdots + D_I \tag{5.13}$$

D は機械種別から資源の数に対する制約を表す組合せ集合を表す．

次に，資源の数から機械種別に対する制約を表す組合せ集合を作成する．そのために，式 5.13 で作成した D を用いる．D 内の組合せにおいて，2 つの変数で構成される組合せを Permitsym 演算を用いて取り出す．

$$D' = D.Permitsym(2) \tag{5.14}$$

D から取り出された 2 つの変数からなる組合せ D' には，必ず機械種別の選択肢と資源の数の選択肢の組合せとなる．これは資源の数から機械種別に対する制約条件となる．

これまで作成した選択性の ZDD 及び選択性間の ZDD を用いて，制約充足解空間を構成する．

選択性の ZDD と選択性間の制約条件を表す ZDD の連結による組合せ削除

選択性の ZDD は，選択性内の制約を満たす組合せの集合を表す．各選択性の ZDD に図 5.2 の選択性間の制約条件を与えることにより，選択性間の制約を満たす組合せのみが残り，制約充足解空間を構成する．

はじめに，各選択性間の制約を満たすために行う演算処理について述べる．選択性 A が選択性 B と関係を持つとする．また，選択性 A の選択肢 a_i と，選択性 B の選択肢との間の制約関係を表す ZDD を C_i とし，選択性 A の選択肢が n 個，選択性 A, B の選択肢の組合せ集合を表す ZDD を A_z, B_z とする．この時，選択性 A の選択肢の組合せを満たす選択性 B の組合せをアルゴリズム 8 を用いて求める．このアルゴリズムの 3 行目の式の (t/a_i)，選択性 A の組合せ集合 A_z を代入した ZDD t から a_i を含む組合せ集合を取り出し，a_i を削除する．そして，(C_i/a_i) は，選択肢 a_i と選択性 B の選択肢間の制約関係を

アルゴリズム 8 選択性 A から選択性 B に対する制約を満たす組合せ集合の作成

1: ZDD $t \leftarrow A_z$
2: **for** $i = 0$ to N **do**
3: $\quad t = (t/a_i) \times (C_i/a_i) + t\%a_i$
4: **end for**
5: $B_z \leftarrow B_z?t : 0$

表す組合せ集合から，a_i を取り除き，a_i と組合さる選択性 B の選択肢を取り出す．これらを乗算することにより，t にて a_i を含んでいた組合せと，a_i が選択されるときに選択される選択性 B の選択肢との組合せを表す集合が作成される．この集合と，$(t\%a_i)$ にて得られる t から a_i を含まない組合せとの和集合を作成し，t を更新する．この結果，t からは a_i が削除され，その代わりに a_i と制約関係をもつ選択性 B の選択肢が含まれる組合せ集合が作成される．これを選択性 A の選択肢の数だけ繰り返すことにより，t は，選択性 A 内の制約，選択性 A から選択性 B に対しての制約を満たす組合せ集合となる．5 行目にて If-Then-Else 演算を用いて，t から B_z に含まれる組合せを取り出し，B_z を更新する．これにより，B_z は，選択性 A の制約を満たす組合せ集合となる．このアルゴリズムは，各選択性に対して適用できる．工程順序 → 製造工程，製造工程 → 機械種別，機械種別 → 資源の数，資源の数 → 機械種別，機械種別 → 製造工程，製造工程 → 工程順序の順にこのアルゴリズムを適用し，各選択性の ZDD の構造の変化が止まるまで繰り返す．これにより，各選択性は，選択性間の制約を満たす組合せ集合を表す ZDD を保持する．

このようなネットワークを形成し各選択性の ZDD が変化すると連鎖して他の ZDD が変化させ，常に各選択性の ZDD が選択性間の制約を充足する．

5.3　遺伝的アルゴリズムを用いた ZDD ネットワーク表現内の解探索

単一の ZDD に対する解探索と同様に，ZDD のネットワーク表現による制約充足解空間から解探索のために GA を用いる．エンコーディング方法は，同様に ZDD の頂点ノードから終端ノード 1 への枝の選択を遺伝子として扱う．しかし，ネットワーク表現では，複数の ZDD から構成されるため，遺伝子配列は，1 つの ZDD の 1-パスを表すのではなく，各選択性の 1-パスを表す．図 5.3 にデコーディングの概要を示す．ここでは，工程順序 → 製造工程 → 機械種別 → 資源の数の順に 1-パスを探索する．まずはじめに，工程順

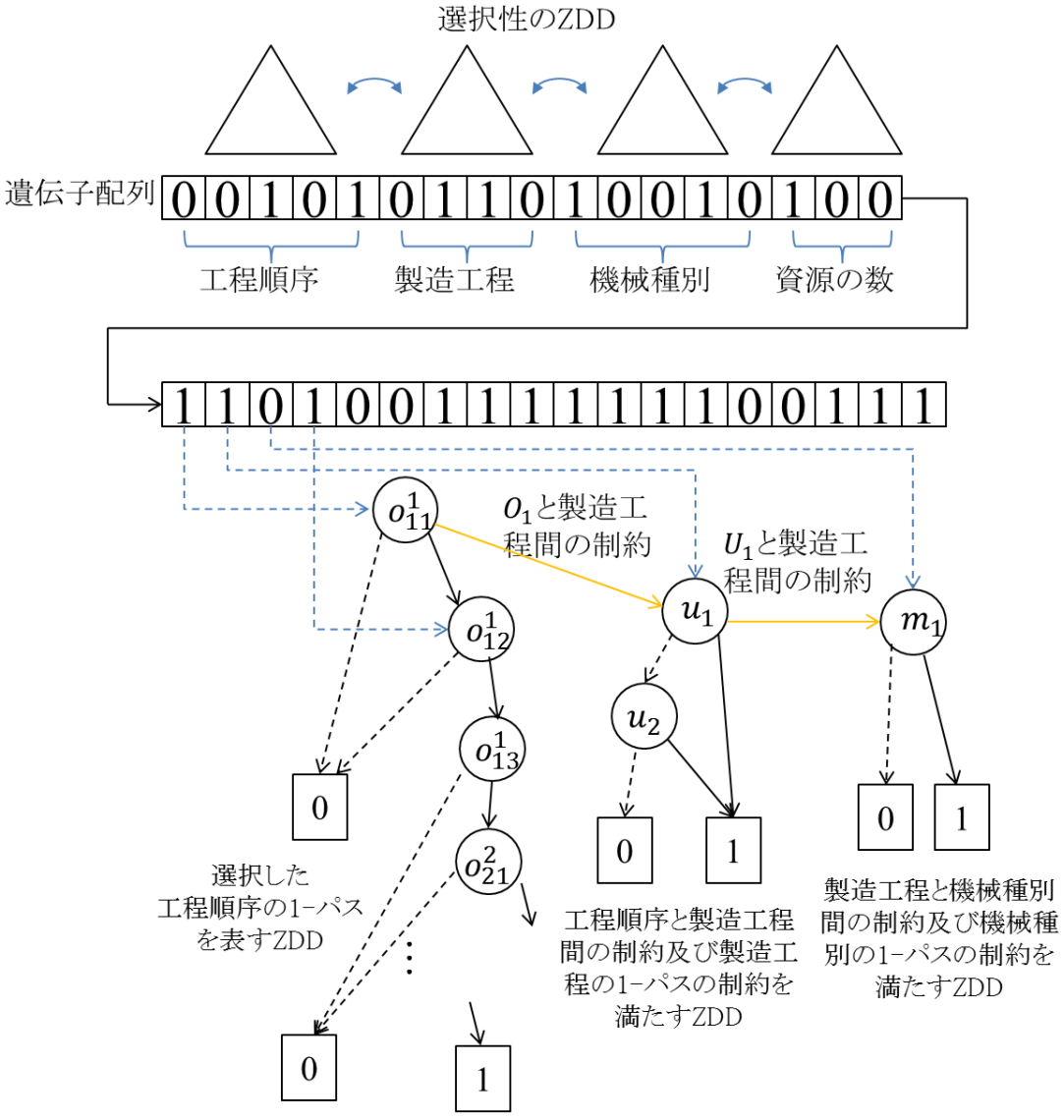


図 5.3: ZDD ネットワークを対象とした GA のデコーディング方法

序の ZDD に対して，個体のもつ遺伝子配列の先頭の遺伝子から順に遺伝子の値に従い 1-パスを辿る．得られた 1-パスは，工程順序で用いる選択肢の組合せを表す．同時にこの組合せは，他の関係をもつ選択性に対しての制約条件となる．つまり，1-パスが得られるつど，1-パスが制約となり他の ZDD が更新される．更新には，アルゴリズム 8 が適用される．この際に，工程順序で得られた 1-パスを表す組合せは，必ず全体として実行可能解をもつことから工程順序から製造工程に対してアルゴリズム 8 は適用されない．ネットワーク全体の更新が終了すると，続いて製造工程の 1-パスを遺伝子配列の遺伝子の値に従い辿る．工程順序の ZDD を辿った際，終端ノード 1 を指す枝の選択の遺伝子が p 番目だとすると，製造工程の頂点ノードからの枝の選択を表す遺伝子は，$p+1$ 番目の遺伝子となる．そして，製造工程の 1-パスが選択されると，工程順序と同様にネットワークの更新が行われる．この処理を繰り返すことにより，各選択性の ZDD から選択肢の組合せを表す 1-パスが得られる．

これまでに得られた組合せが表すのは，製品ごとに使用する工程種別とその順序 (工程順序)，生産計画として使用する製造工程及び機械種別，そして設置する機械インスタンスである．次の段階では，各製品の r 番目に割り当てる製造工程及び機械種別の選択を行う．これは，各選択性で得た組合せ及び，選択性間の制約条件を表す ZDD を用いる．この選択においても，遺伝子配列を用い，資源の数の終端ノード 1 を指すノードの枝の選択を表す遺伝子の 1 つ隣の遺伝子の値が始点となる．はじめに，工程順序から得られた 1-パスを遺伝子の値にに従い辿る．1 枝を選択した際，対象のノードが表す工程順序の変数 o^s_{wr} とする．次に，式 5.7 で作成した工程順序と製造工程間の制約を記述した ZDD Z から以下の式を用いて o^s_{wr} を含む組合せを取り出す．

$$Z_1 = Z/o^s_{wr} \tag{5.15}$$

Z_1 は Z から o^s_{wr} を含む組合せを取り出し，o^s_{wr} を削除する．製造工程の選択性の ZDD から得た 1-パスが表す組合せを B_p とすると，Z_1 は B_p を構成する変数に含まれないものを含む可能性があるため，以下の式にてそれらを削除する．

$$Z_2 = Z_1.Permit(B_p) \tag{5.16}$$

Permit 演算により，B_p を構成する変数の組合せに包含されない Z_1 内の組合せは削除される．ここで得られた Z_2 に対して遺伝子の値に従い 1-パスを辿る．

1 枝が選択された際に，対象のノードが表す製造工程の変数を u_k とする．続いて，製造工程と機械種別間の制約を表す ZDD D を用いる．

$$D_1 = D/u_k \tag{5.17}$$

上記の式は，Z_1 の作成と同じ手順である．また，D_1 も Z_1 と同様に，機械種別の選択性の ZDD から得た 1-パスが表す組合せを C_p とすると，D_1 は C_p を構成する変数に含まれないものを含む可能性があるため，それを削除する．

$$D_2 = D_1.Permit(C_p) \tag{5.18}$$

ここで得られた D_2 に対して遺伝子に従い 1-パスを辿る．1 枝選択後，再び工程順序の 1-パスに戻り，次の工程順序のノードの枝選択を行う．これを工程順序から得られた 1-パスが終端ノード 1 に到達するまで繰り返す．選択された 1-パスと選択性間の制約を表す ZDD を用いて，間接的に ZDD を連結させて，製造工程と機械種別の割り当てを行う．ここでの繰り返しにて 1 枝を選択したノードの変数番号を順次配列 (出現順変数番号配列) に格納する．図 5.3 の例を用いて説明すると，はじめに o_{11}^1 が選択され，次に u_1，そして m_1 が選択され，工程順序の ZDD へと戻る．この際，工程順序 $\rightarrow$ 製造工程 $\rightarrow$ 機械種別の変数番号順に配列に格納される．そのため，工程順序の変数番号の次は，選択された工程種別を割り当てる製造工程，さらにその製造工程を処理する機械種別の情報を得ることができる．

遺伝子操作及び選択は，単一の ZDD と同じ方法を用いる．探索済み 1-パスのキャッシュは，複数の ZDD の実パスを 1 つの 1/0 配列として管理する．以上の方法を用いて，ZDD ネットワーク内の解探索を行う．

5.4 本章まとめ

本章では，生産プランニングの制約充足解空間表現の 1 つの解法として ZDD の連結によるネットワーク表現について提案した．ネットワーク表現では，単一の ZDD に比べ変数の数およびノード数が減少し，ZDD の作成時間が短縮されることが期待される．ネットワーク表現作成には，各選択性が自身の選択肢の組合せを表す ZDD と，選択性間の制約関係表す ZDD を連結を行う．この連結により，任意の選択性の ZDD に変化が起こると，その選択性と関係をもつ選択性の ZDD が連動して変化するシステムを提案した．ま

た，第 4 章で提案した GA を応用し ZDD のネットワーク表現による制約充足解空間から解探索する方法を提案した．本章で提案した ZDD のネットワーク表現及び解探索の検証実験結果は第 6 章で言及する．

第6章　ZDDによる制約充足解空間及びGA探索の評価実験

6.1　評価実験の概要

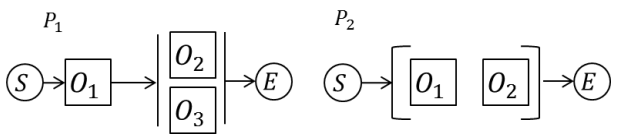
本章では，第4章で提案した単一のZDDによる生産プランニングの制約充足解空間，制約充足解空間の解探索のGA及び第5章でのZDDの連結によるネットワーク表現の評価実験を行う．

まずはじめに，ZDDが生産プランニングにおける制約充足解を正確に表現しているか確認を行う．第4章の図4.1と表4.1を用いる．ここでは，製品種別数は2種 ($P_1 \sim P_2$)，工程種別数は3種 ($O_1 \sim O_3$)，製造工程は5種 ($U_1 \sim U_5$)，機械種別数は3種 ($M_1 \sim M_3$) である．各機械種別は，1台ずつインスタンスをもつ．各製品種別は1台生産されるとし，設置可能な機械台数は2台とする．このような条件のもと制約充足する組合せ数は32通りである．この問題の制約充足解空間を単一のZDDを用いて表現する．表6.1にその結果を示す．単一のZDDによる制約充足解空間表現は48の変数を用いて32通りの制約充足解を表現することを確認した．

評価実験概要

本実験では，2種類の生産プランニング問題 (例題1と2) を用意した．例題1に対しては，以下の評価を行う．

- 単一のZDDによる制約充足解空間表現
- GAによる解探索および変更方法の影響



製品種別の工程順序と使用工程の例 (図4.1の再掲)

工程種別，製造工程および機械種別の組合せと処理時間 (表 4.1 の再掲)

				Instance		
				1	1	1
	O_1	O_2	O_3	M_1	M_2	M_3
U_1	*	-	-	1	-	-
U_2	-	*	-	-	-	1
U_3	-	-	*	-	3	2
U_4	-	*	-	2	-	-
U_5	*	-	-	-	1	-

表 6.1: 第 4 章の例題の単一の ZDD による制約充足解空間表現

組合せ数	変数の数	計算時間 [s]
32	48	0.01

- 全探索による解分布の分析及び解法選択

例題 2 に対しては，以下の評価を行う．

- 単一の ZDD のによる制約充足解空間表現及び動的事象による制約充足解空間の更新
- ネットワーク表現による制約充足解空間表現
- ネットワーク表現を対象とした GA による解探索

以上の点についてそれぞれ例題に対して検討を行う．

例題 1 に用いる製品の種類，工程種別及び工程順序を図 6.1 に示す．表 6.2 は機械種別が処理可能な工程種別とその処理時間を示している．例題 1 では，製品種別数は 4 種 ($P_1 \sim P_4$)，工程種別数は 12 種 ($O_1 \sim O_{12}$)，製造工程は 20 種 ($U_1 \sim U_{20}$)，機械種別数は 10 種 ($M_1 \sim M_{10}$) である．表 6.2 に示されている通り総機械インスタンス数は 14 台である．この例題では，生産現場に設置可能な機械インスタンスの数を 6 台と設定した．各製品種別から 1 台ずつ生産されるとし製品数は 4 ($J_1 \sim J_4$) である．各製品の納期は表 6.3 に示す．目的関数は，式 4.1 の TAC であり，最小化問題である．

$$TAC = \sum_{s=1}^{N} (T_s \times cost_{dd}) + cost_c \times e \tag{4.1}$$

ただし，本問題では，式 4.1 の第二項は 0 とし，$cost_{dd} = 1$ の総納期遅れ時間とする．本例題では，スケジューリング作成にあたり，解法としてディスパッチング・ルールの Shortest Processing Time(SPT) または Earliest Due Date(EDD) を用いるとする．

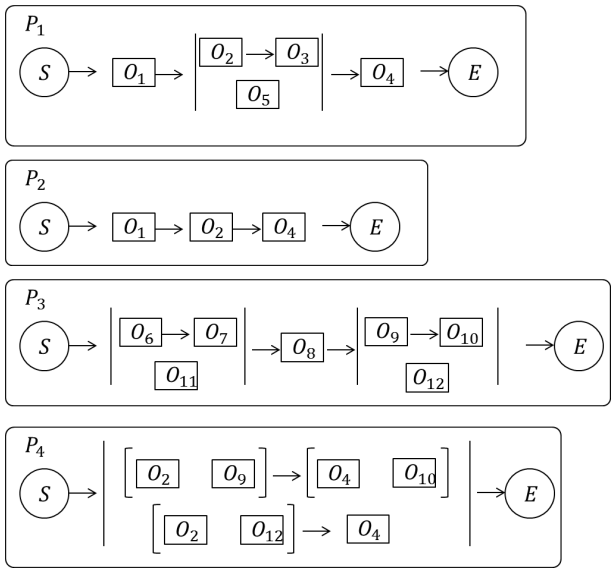


図 6.1: 各製品種別の工程順序と使用工程種別 (例題 1)

表 6.2: 工程種別，製造工程と機械種別の組合せと処理時間 (例題 1)

		機械種別									
		Instance									
		2	1	1	1	1	1	2	2	1	2
	工程種別	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
製造工程	O_1	*									
U_1		3					1	2			
U_2	O_1	*									
U_2		2									
U_3	O_2		*								
U_3			2				1				
U_4	O_2		*								
U_4			1					2			
U_5	O_3			*							
U_5				3							
U_6	O_3			*							
U_6				2							
U_7	O_4				*						
U_7					5		3				
U_8	O_5				*						
U_8					4						
U_9	O_6					*					
U_9						4		2			
U_{10}	O_6					*					
U_{10}						6					
U_{11}	O_7						*				
U_{11}									3		
U_{12}	O_7						*				
U_{12}				4							
U_{13}	O_8							*			
U_{13}				1					1		
U_{14}	O_9							*			
U_{14}										7	
U_{15}	O_9							*			
U_{15}											7
U_{16}	O_9							*			
U_{16}	O_{10}	4					2				
U_{17}	O_9							*			
U_{17}				3							
U_{18}	O_{11}							*			
U_{18}					6						
U_{19}	O_{12}							*			
U_{19}								4			6
U_{20}	O_{12}							*			
U_{20}						8					

表 6.3: 各製品の構成及び納期

製品	J_1	J_2	J_3	J_4
納期	10	8	15	7
製品種別	P_1	P_2	P_3	P_4

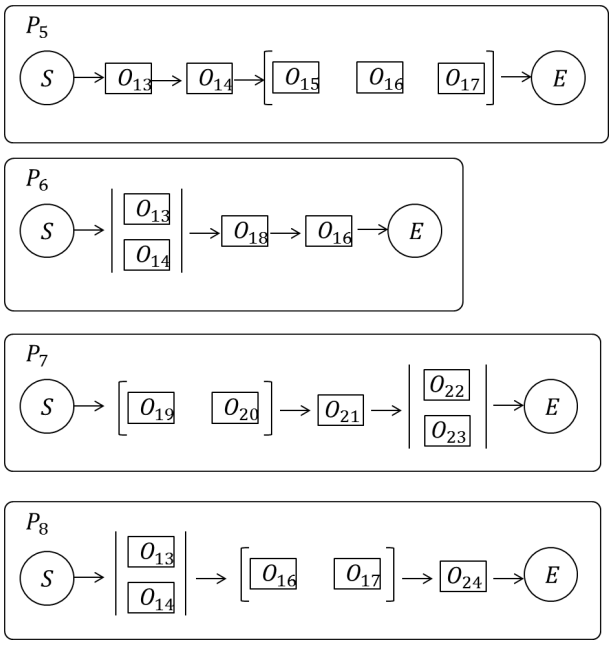
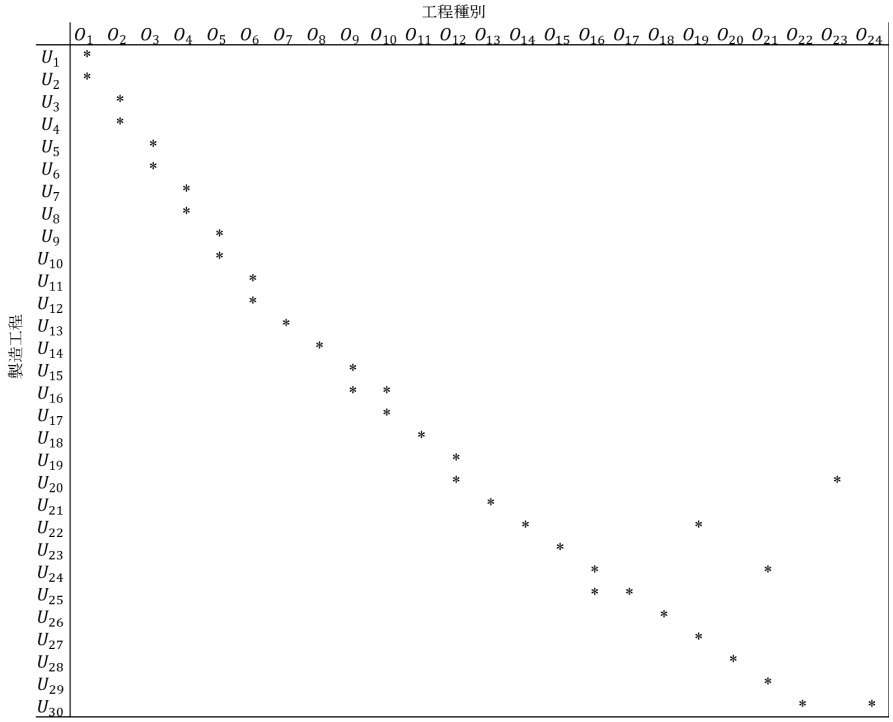


図 6.2: 各製品種別の工程順序と使用工程種別 (例題 2)

例題 2 に用いる製品の種類，工程種別及び工程順序は図 6.1 の 4 種類と図 6.2 の 4 種類である．工程種別，製造工程及び機械種別の組合せと処理時間を表 6.5 に示す．例題 2 では，製品種別数は 8，工程種別数は 24 種，製造工程数は 30 種，機械種別数は 15 種である．表 6.2 に示されている通り総機械インスタンス数は 21 である．例題 2 では，生産現場に設置可能な機械インスタンスの数を 9 台と設定した．製品種別 $P_1 \sim P_4$ は 1 台，製品種別 $P_5 \sim P8$ は 2 台ずつ生産されるとし，製品数は 12 である．各製品の納期は表 6.6 に示す．目的関数は，同様に式 4.1 の TAC であり，最小化問題である．本例題では，スケジューリング作成にあたり，解法としてディスパッチングルールの EDD を用いるとする．例題 2 では，$cost_{dd} = 1, cost_c = 0.2$ と設定した．

ZDD 作成には Ruby 版 VSOP[41] を用いる．GA は C++言語で実装され，GA 実行中の ZDD の操作は湊 [4] によって開発された SAPPOROBDD パッケージを用いる．用いた計算機は，CPU が Xeon E5-2643 3.30GHz，主記憶装置が 256GB で OS は Linux である．

表 6.4: 工程種別と製造工程の組合せ (例題 2)



6.2　単一の ZDD 表現での制約充足解空間表現及び GA 探索の評価（例題 1）

6.2.1　ZDD による制約充足解空間表現

はじめに，ZDD の変数順序を定義する．最適な変数順序を見つけることは NP 完全である．そこで，局所計算性のある変数同士は近い順序でおくというヒューリスティクスに従い，次のように変数を宣言した．はじめに，製品計画に関する変数を製品ごとに宣言する．製品は $s = 1$ から順に宣言される．更に，順序を優先し，r 番目の工程順序の変数を $w = 1$ から順に宣言後，製造工程，そして機械種別の順に宣言し，$r + 1$ 番目の各選択性の変数を宣言する．製品計画の変数を宣言後，資源計画の変数を宣言する．ここでは，機械種別ごとに，インスタンスの設置を表す変数を宣言する．

表 6.7 は，ZDD における各製品の製品計画での組合せ数，ノード数及び計算時間を示す．製品計画における各製品の工程順序，製造工程及び機械種別の組合せは非常に短い時間で得ることができる．また，製品 J_4 が最も代替案をもつ製品であることが組合せ数からわかる．

表 6.5: 製造工程と機械種別の組合せ及び処理時間 (例題 2)

	機械種別 Instance														
	2	1	1	1	1	1	2	2	1	2	2	1	3	2	1
製造工程	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}	M_{11}	M_{12}	M_{13}	M_{14}	M_{15}
U_1	3					1	2								
U_2	2														
U_3		2				1									
U_4		1					2								
U_5			3												
U_6			2												
U_7				5		3									
U_8				4											
U_9					4		2								
U_{10}					6										
U_{11}								3							
U_{12}			4												
U_{13}			1					1							
U_{14}									7						
U_{15}										7					
U_{16}	4					2									
U_{17}			3												
U_{18}				6											
U_{19}							4			6					
U_{20}					8										
U_{21}											5				
U_{22}											3				
U_{23}												3			
U_{24}													10		
U_{25}												6		4	
U_{26}															2
U_{27}													7		
U_{28}											7			8	
U_{29}															8
U_{30}										9					

表 6.6: 製品の構成と納期

製品	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8	J_9	J_{10}	J_{11}	J_{12}
納期	10	8	15	7	9	21	17	23	20	28	38	30
製品種別	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_5	P_6	P_7	P_8

表 6.7: 製品計画における各製品の工程順序，製造工程及び機械種別の組合せ

製品	組合せ数	ノード数	計算時間 [s]
J_1	132	36	0.01
J_2	48	20	0.01
J_3	60	54	0.01
J_4	504	73	0.01

表 6.8: 例題 1 の ZDD による制約充足解空間

計画	組合せ数	ノード数	計算時間 [s]
製品	191,600,640	644	0.01
製品+資源 1	6,471,482,688	69,670	1.13
製品+資源 2	44,441,064	28,132	0.42
製品+資源 3	13,944,540	28,061	0.15
製品+資源 3+実施	27,889,080	28,063	0.1

表 6.8 は，制約充足解空間を ZDD で表す際の各段階における組合せ数，ノード数及び計算時間である．表中の計画の製品は，表 6.7 の各製品の ZDD を 1 つにまとめた際の ZDD での組合せ数，ノード数及び計算時間を示す．製品計画での組合せ数は 1.9×10^8 である．この ZDD 作成時間は 0.01 秒であった．製品+資源 1 は，製品計画単体を表す ZDD を資源計画へと拡張し，設置可能機械台数の上限の制約を与える前の ZDD の計算結果を表す．ここでの計算時間は，製品計画単体の ZDD から製品+資源 1 の ZDD を作成するまでの時間であり，1.13 秒である．作成された ZDD は，組合せ数 6.5×10^9 でノード数は 69,670 となった．製品+資源 2 は，製品+資源 1 の結果で得られた ZDD に対して，設置可能機械台数の上限の制約を与え，この制約を満たす組合せのみで表現される ZDD の計算結果である．ここでは，組合せ数が 4.4×10^7 となり，製品+資源計画 1 の ZDD と比べると，約 99.3%組合せ数が減少した．計算時間は，製品+資源 1 の ZDD 作成後から製品+資源 2 の ZDD 作成までの時間であり，0.42 秒である．製品+資源 3 は，製品+資源 2 の ZDD から冗長な機械インスタンスの設置の組合せを削除した ZDD であり，この ZDD は制約充足解空間を表す．冗長な組合せを排除したことから組合せ数は 1.4×10^7 となり，製品+資源計画 2 の ZDD と比べ組合せ数は約 68.6%減少した．計算時間も今までと同様に，製品+資源 2 の ZDD 作成後から製品+資源 3 の ZDD 作成までの時間は，0.15 秒である．製品+資源 3+実施は，製品+資源 3 の ZDD から実施計画でのディスパッチング・ルールの選択を考慮した ZDD である．ここでは，EDD ルールもしくは SPT ルールの 2 通りであるた

表 6.9: 交叉及び突然変異確率のパターン

パターン 1	交叉	突然変異
1	0.5	0.5
2	0.7	0.3
3	0.9	0.1

表 6.10: 各 GA パラメータの最良値の平均と標準偏差の値 (50 回試行)

変更方法	交叉確率 & 突然変異確率	最良値の平均	標準偏差
top	0.5 & 0.5	2.02	0.55
	0.7 & 0.3	2.02	0.58
	0.9 & 0.1	2.18	0.62
bottom	0.5 & 0.5	1.56	0.64
	0.7 & 0.3	1.88	0.59
	0.9 & 0.1	1.76	0.59
random	0.5 & 0.5	1.62	0.66
	0.7 & 0.3	1.62	0.60
	0.9 & 0.1	1.44	0.61

め，単純に製品+資源 3 の ZDD の組合せの 2 倍の組合せを表す．この時，制約充足解空間作成の計算時間は，1.75 秒であった．例題 1 では，862 の ZDD の変数を用いた．つまり，解空間の大きさは $2^{862} \approx 10^{259}$ であり，ZDD は，制約を満たす組合せのみに限定し解空間を 99.9%以上縮約したことがわかる．また，製品計画単体で実行可能であった 1.9×10^8 の組合せのうち，生産プランニング全体として実行可能である組合せは，12,876,740 通りであり，約 6.7%が実行可能であるという結果を得た．

6.2.2 GA の解探索能力と評価

探索済み 1-パスの変更方法の検証

表 6.8 における制約充足解空間を表す製品+資源 3 の ZDD を入力とし，GA による解探索を行う．GA のパラメータは，個体数を 100，世代数を 100，エリート数を 2，トーナメントサイズを 2 と設定した．表 6.9 に示すように交叉確率，突然変異確率のバランスは 3 種類用意した． まずはじめに，GA 探索において遺伝子配列が探索済みの 1 パスを表す際の変更方法として，top，bottom，random の計 3 種類について検証する．GA のパラメータは計 9 パターンあり，各パターンに対して GA を 50 回試行し，得られた結果を比較する．各パターンの最良値の平均と標準偏差の値を表 6.10 に示す． ここでの最良値と

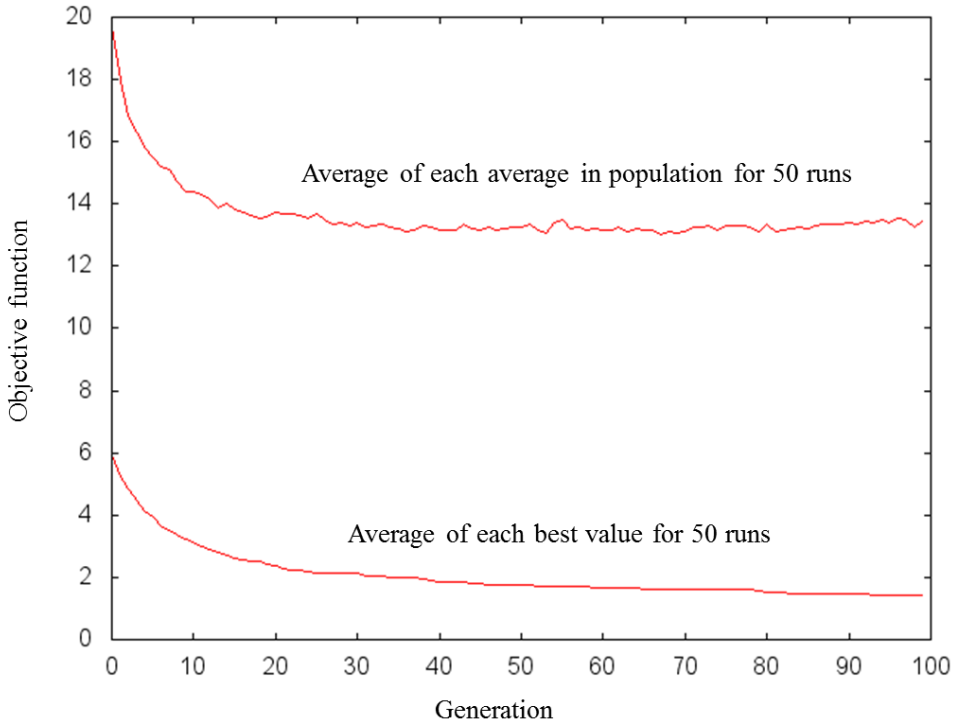


図 6.3: 50 試行での最良値の平均と，各世代の個体の適応度平均の推移

は，最終世代に到達した際に個体がもつ最小の目的関数の値を表す．まず変更方法から見ると，random が最も良い値を得ており，次に bottom そして top と続く．本実験から得た結果から，変更方法として random が交叉，突然変異確率を問わず評価関数値が良い値の解を探索する傾向があることがわかる．

図 6.3 は，表 6.9 のパターン 3 における変更方法 random の 50 回試行した際に得られた各試行の最良値の平均と，各世代での個体の適応度平均を示す．個体は徐々に収束をはじめ，約 30 世代目付近で適応度の平均が 14 付近に収束していることがわかる．

ZDD 上を頂点ノードから出来る限り 0 枝を進み終端ノード 1 へ到達すると，ZDD 上の最左の 1-パスを得ることができる．また，頂点ノードから出来る限り 0 枝を進み，最左の 1-パスを通らないようにすると，ZDD 上の左から 2 番目の 1-パスを得る．この方法を用いて，1 試行における各変更方法の世代ごとの探索領域を図 6.4，6.5，6.6 に示す．横軸は ZDD 上の 1-パスのインデックス番号，縦軸は世代数を表す．1-パスのインデックス番号は，ZDD の最左の 1-パスを 1 とし，最右の 1-パスのインデックス番号は組合せ数と同数である．世代数は第 1 世代から降順に並んでいる．top では，頂点ノードから最初の 0 枝

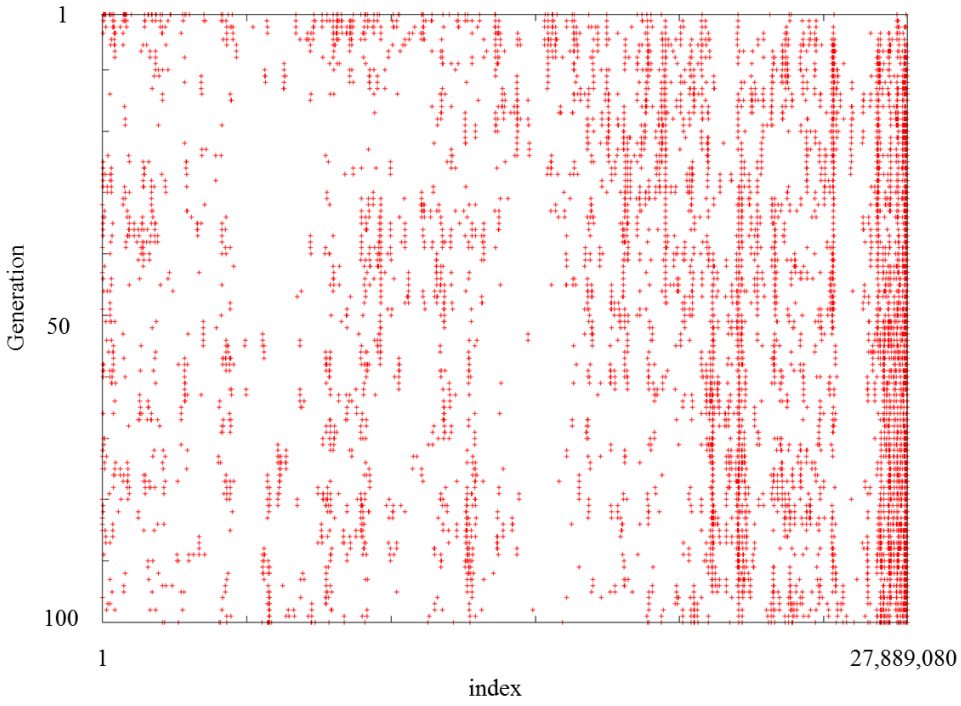


図 6.4: 変更方法 top における世代ごとの探索領域 (交叉 0.9，突然変異 0.1)

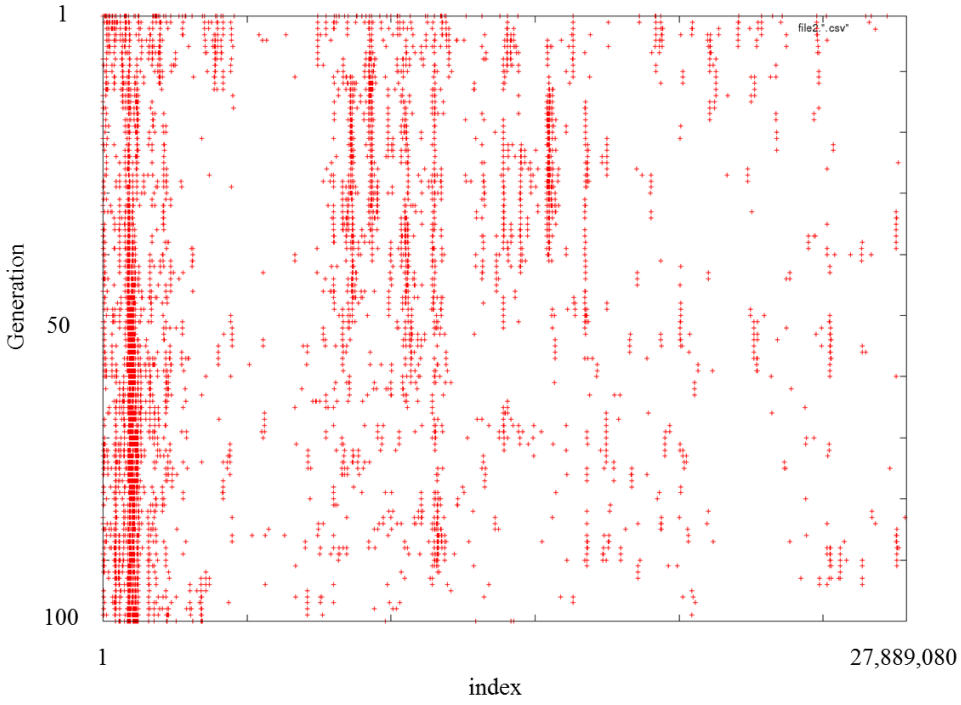


図 6.5: 変更方法 bottom における世代ごとの探索領域 (交叉 0.9，突然変異 0.1)

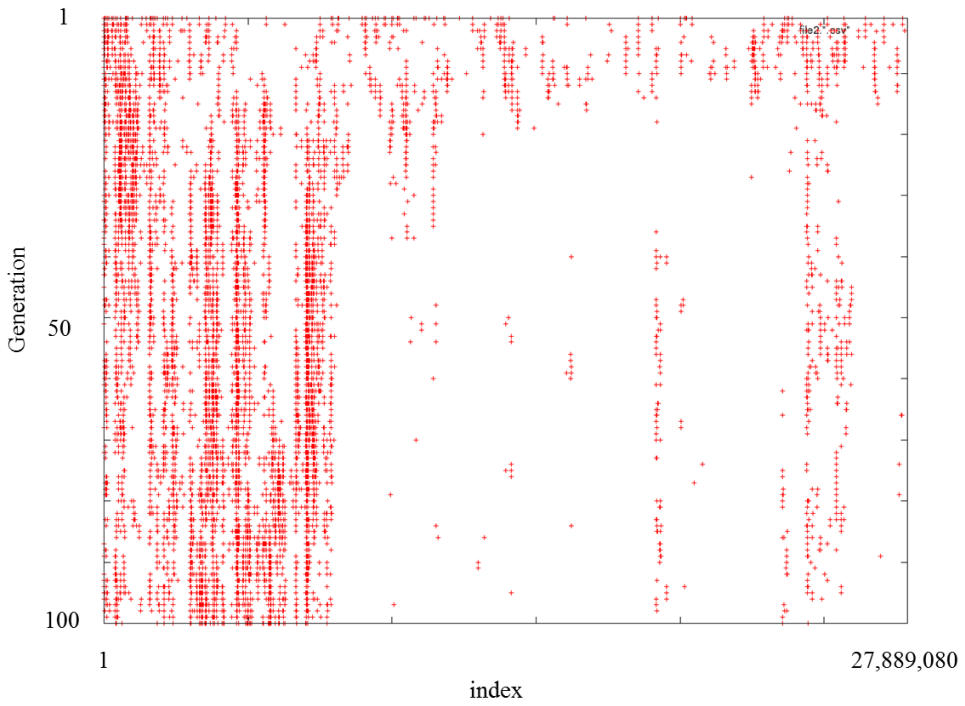


図 6.6: 変更方法 random における世代ごとの探索領域 (交叉 0.9，突然変異 0.1)

の選択を 1 枝に変更する．そのため，探索済み 1-パスを表す遺伝子配列は，ZDD 上の右側の組合せを表す 1-パスへ変更される．図 6.4 から，組合せ番号が 2.7×10^7 に近い領域を重点的に探索していることがわかる．逆に bottom は，終端ノード 1 から遡り最初の 0 枝の選択を 1 枝に変更するを探索するため，探索済み 1-パスに近い 1-パスへ変更される．図 6.5 から，区間 1,2 付近を重点的に探索していることがわかる．これは，探索済みの 1-パスが区間 1,2 に多く，遺伝子操作により生成される子孫は，この領域を表す遺伝子配列をもつ傾向があることを示している．random では，top と bottom の他にも，1 枝の選択を 0 枝に変更する可能性をもち，幅広く探索を行う可能性をもつ．図 6.6 から，区間 1 から 7 付近に対して幅広く探索を行っていることがわかる．本研究では，上位のノードは製品 1,2 の製品計画に関する変数が並んでおり，下位のノードでは資源計画に関する変数が並んでいる．top や bottom では，そのようなノードの選択のみが変更されるため探索に偏りが生じる．各変更方法での解探索を行った結果，top や bottom といった一定の規則に従って変更するのではなく，探索済み 1-パス上の全てのノードに対してランダムに変更した方が探索効率が良くなるという結果が得られた．

エリート数による局所解探索能力の評価

提案する GA は，エリート個体を除く個体は常に未探索の 1-パスを表す．個体群にエリート個体が少ない場合，解探索のランダム性が高まり，解の収束を悪くする場合がある．そこで，ZDD を探索する GA においてエリート個体数が解探索にあたえる影響について評価を行う．変更方法を random に固定し，エリートの数を個体数の 2%，5%，10%，そして 15%のパターンを検証する (小数点切り上げ)．GA パラメータは，交叉及び突然変異確率は 0.9 と 0.1 とする．個体数と世代数の組合せを 3 種類用意した．個体数 50 と世代数 200，個体数 100 と世代数 100，個体数 200 と世代数 50 である．これらは延べ 10,000 通りの組合せを探索する．各種パラメータにて GA をそれぞれ 50 回試行し，ZDD 上の解を探索する GA でのエリート個体数が解探索能力にあたえる影響の評価を行う．表 6.11 は，50 回試行での最良値の平均と標準偏差を示す．3 種の個体と世代の組合せの比較を行ったが，どの組合せにおいてもエリート個体が個体群 2%では他のエリート個体の割合に比べ，最良値の平均が低い結果となった．また，最良値の平均が最も高かったのは，個体数 200，世代数 50 の時にエリート個体数を 5%及び 15%の組合せであった．得られた結果から探索する 1-パスの延べ数が同じ時は，個体の数を延べ数の半分以上に設定した方が，良解を得やすい傾向があることがわかった．

表 6.11: 個体数，世代数及びエリート数の関係

個体数 & 世代数	エリート個体の割合	最良値の平均	標準偏差
50 & 200	2%	1.74	0.80
	5%	1.8	0.8
	10%	1.58	0.72
	15%	1.68	0.68
100 & 100	2%	1.44	0.61
	5%	1.6	0.69
	10%	1.32	0.47
	15%	1.3	0.54
200 & 50	2%	1.54	0.57
	5%	1.2	0.4
	10%	1.34	0.47
	15%	1.2	0.4

表 6.12: 例題 1 の ZDD 上の最適解，最悪解及び平均

最適解	平均	最悪解
1	17.36	51

6.2.3　全探索による評価関数値の分布

提案する GA の有効性を検証するために全探索との比較を行う．しかしながら，10^{259} 規模の解空間に対して探索を行うことは実時間内では困難であるため，制約充足解空間である ZDD に対して全探索を行う．ZDD は，組合せを枚挙することができる．終端ノード 0 に到達しないように 0 枝を選択していくと，ZDD から見て最も左にある組合せを取り出すことができる．この組合せ番号を 1 とすると，1 つ隣の組合せ番号 2 となる．つまり，組合せ番号の最大はその ZDD の組合せ数と同数となる．この方法を用いて，得られた制約充足解空間を表す ZDD に対して全探索を行った．ZDD 上の解を全探索した際の計算時間は，約 888 秒であった．表 6.12 に最適解，最悪解及び解の平均を示す．この探索により最適解は，1 であり，最悪解は 51 であることが判明した．つまり，実行可能解の中に全ての製品が納期を遵守する組合せは存在しない．解の持つ評価関数値の平均は，17.36 であった．組合せ数 2.8×10^7 内に最適解を持つ組合せは 36 通りであり，全体の $1.3 \times 10^{-6}\%$ の組合せにあたる．また，評価関数値順にランク付けを行うと全部で 50 の値が存在することがわかった．最適解の順位を 1 位，最悪解の順位を 50 位と設定し，各順位の度数分布を図 6.8 に示す．最も多い評価関数値は 15 であり，全体の 7.46%を占める．

次に，組合せ番号を 1.0×10^6 ごとに区切り，それぞれに区間番号を与える．例えば，区間番号 1 は組合せ番号 $1 \sim 1.0 \times 10^6$ の領域を表す．各区間での目的関数値の平均値，最大値及び最小値を図 6.8 に示す．また，最も平均が高かったのは区間 15 であり，15.85 であった．しかしこの区間には最適解が存在しておらず，最適解は区間 1,4,5 と 20 に存在している．表 6.13 は最適解の分布を表す．最も最適解が多いのは区間 4 であり，その数は 24 通りである．最適解をもつ組合せの約 66%がこの区間に存在している．図 6.5 から bottom は，区間 1,2 付近を重点的に探索している原因は，この領域に最適解があるからでることがわかる．また，図 6.6 から random は，区間 4,5 付近を重点的に探索しており，この区間は最適解が最も多く存在する領域を探索していることがわかる．最適解は，全体の全体の $1.3 \times 10^{-6}\%$ の組合せだけ存在する．また，解が 2 位の組合せを含めると，全体の $4.0 \times 10^{-6}\%$ の組合せとなる．表 6.11 の結果の中で最も良いパラメータの組合せでは，

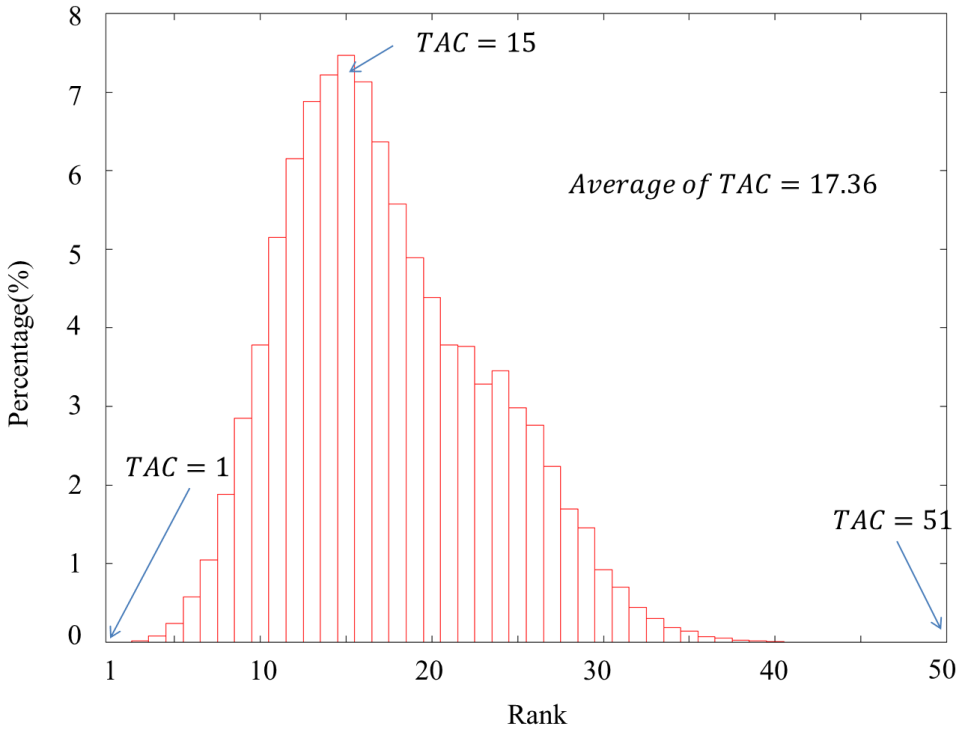


図 6.7: 目的関数値のランキングによる解分布

表 6.13: 例題 1 の最適解の分布

区間	1	4	5	20
最適解の数	4	24	4	4

GA では延べ 10,000 通りを探索すると，最適解に近い値が得られ，解探索が効果的に行われたことがわかる．

ZDD では，変数の宣言順序により組合せ集合を構成するノードの数が異なる．また，宣言順序が異なれば，任意の組合せ番号は異なり，区間ごとの解の分布が異なる．そのため，制約充足解空間を表す ZDD とその空間を探索する GA では，制約充足解空間を短時間で構成し，なるべく最適解が同じ区間に集まるような変数の宣言順序が求められることがわかった．

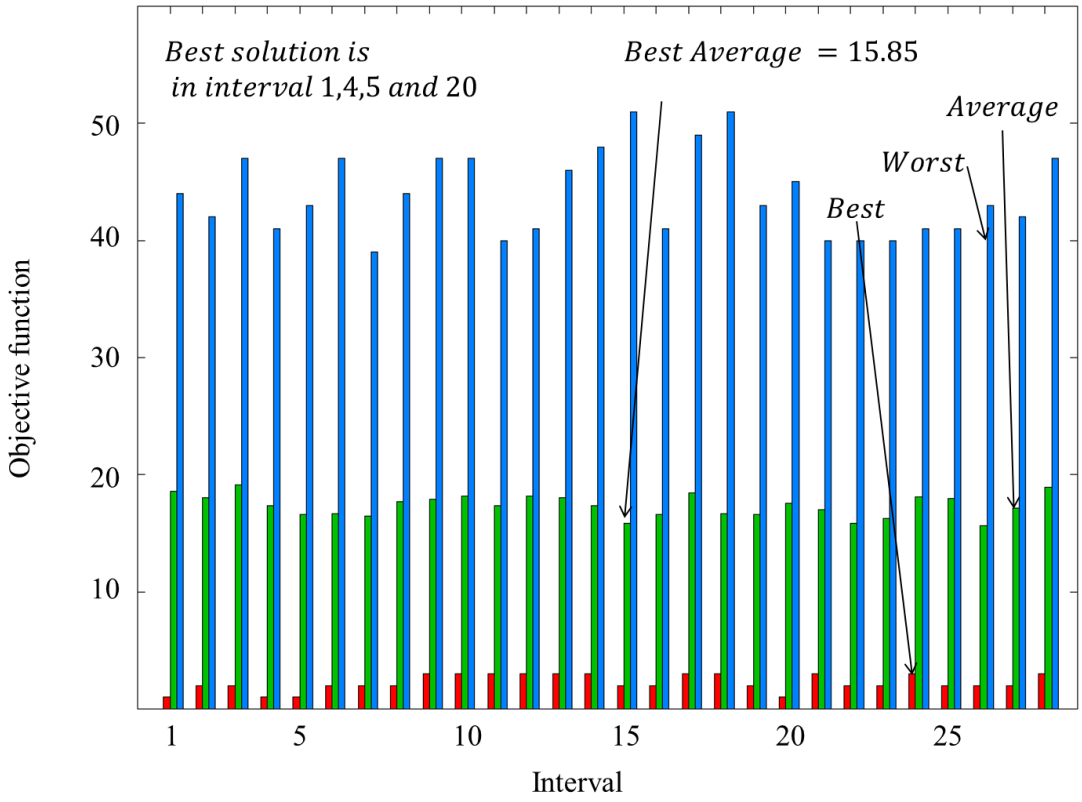


図 6.8: 区間ごとの評価関数値の平均と最大値

全探索と GA の結果比較

GA のパラメータとして，もっともよい最良値の平均を出したパラメータのパターンでは，計算時間は 1 試行につき平均 0.46 秒であった．ZDD 上の解の全探索では計算時間は約 888 秒である．ZDD にて宣言した変数の数は 862 であることから，もとの解空間の大きさは $2^{862} \approx 10^{259}$ である．この解空間に対して全探索を行った場合に，単純に計算すると 10^{248} 年かかるため，実時間内に解を得ることは不可能である．そのため，ZDD にて制約を充足した解のみで構成される制約充足解空間を表現した上で，解探索を行うことは非常に有効である．また，ZDD は解の数を知ることができるため，その規模に応じて GA もしくは全探索の選定を行うことができる．

表 6.14: ZDD による初期制約充足解空間

計画	組合せ数	ノード数	計算時間 [s]
製品	5,272,625,161,080,668,160	1,162	0.21
製品+資源 1	16,046,558,698,708,936,949,760	220,921	40.93
製品+資源 2	103,096,650,092,544	32,979	0.1
製品+資源 3	5,122,042,970,112	32,922	0.25

6.3 動的事象による制約充足解空間の変化と計画の修正（例題 2）

6.3.1 初期の制約充足解空間の作成と生産計画

まずはじめに，例題 1 と同様に制約充足解空間を作成する．この問題では，差立ロジックの選択肢が EDD ルールのみであり，資源計画までの制約充足解空間内の解の数が変化しないことから，EDD を表す変数は宣言しない．表 6.14 は，初期の制約充足解空間を表す ZDD を作成した結果である．表中の製品計画の ZDD は，製品計画における実行可能な組合せ数を表す．ここでの計算時間は製品計画における実行可能な組合せ集合を表す ZDD を作成するために費やした時間であり，0.21 秒であった．製品+資源計画 1 は，製品計画の ZDD を基に資源計画の選択肢を包含するように ZDD を拡張した結果である．さらに製品+資源計画 2 は，拡張された ZDD に対して設置可能機械台数の制約を満たす組合せのみを抽出した結果である．また，製品+資源計画 3 は，機械インスタンスが重複した組合せを排除した結果を示す．この制約により，製品+資源計画 I の組合せの内，99.9%の組合せが制約を満たしていないことがわかる．全ての処理に費やした計算時間は，41.49 秒である．またこの問題に対して用いた変数の数は，4,185 個である．ここで得られた ZDD は，この問題における制約充足解空間を表す．

本実験の条件では，機械の種別数は全部で 15 種類あり，設置可能な機械の台数は 9 台が上限である．この制約により，表 6.14 における製品計画の ZDD 上の組合せにおいて，9 種類以上必要とする製品計画の候補は実施不可能となる．製品計画の ZDD において，実行可能な組合せ数は，9.4×10^{18} であるが，製品計画+資源計画 3 の ZDD に含まれる製品計画の組合せ数は，8.0×10^{12} であった．つまり，製品計画のみを考慮した場合，製品計画における組合せでは約 0.01%以下の組合せのみが資源計画を考慮した場合の実行可能な組合せになる．さらに，製品+資源 3 の結果である資源計画における機械の設置台数の制約を満たす組合せは 8.0×10^{12} 通りであり，各組合せは生産プランニングとして実行可能

表 6.15: 例題 2 の制約充足解空間探索のための GA パラメータ

個体数	500
世代数	200
エリート個体数	25(5%)
交叉確率	0.9
突然変異	0.1
変更方法	random

な解であることを意味する．この ZDD は初期の制約充足解空間を表す.

次に初期の制約充足解空間から GA を用いて解探索を行う．例題 2 の制約充足解空間の規模は，例題 1 に比べると大きく，全探索を行うことは困難である．GA のパラメータは，例題 1 の結果から，表 6.15 のように設定した．また，探索済み 1-パスの際の変更方法は random を用いる．本例題の目的関数は，納期遅れ時間に比例したペナルティコストと計画変更の回数に比例したコストの総和であり，最小化問題である．初期の生産計画作成時には，計画変更コストは 0 となる．この条件下で 10 回試行を行う．10 回の試行において，最良値の平均は 125.2，標準偏差は 2.32，平均計算時間は約 9.56 秒であった．この試行で最も良い値であった総納期遅れ時間 121 の組合せを初期の生産計画として生産の実施を行う．この計画では，納期を遵守している製品は J_2 のみである．

6.3.2　機械故障の発生による制約充足解空間の更新及び計画の修正

図 6.9 は生産計画のガントチャートを示している．図中の M_{4-1} は，機械種別 M_4 のインスタンス 1 を示す．図 6.9 では，各インスタンスの上段を初期生産計画のガントチャートとしている．この生産計画では各機械種別が 1 台ずつ機械を設置している．このガントチャートでは，M_{4-1} の最初の処理として，$3-18$ が割り当てられている．これは，この機械上にて製品 J_3 の製造工程 U_{18} を処理することを意味する．この生産計画を実施中に時刻 12 にて機械 M_{14-1} が故障するとする．この時，生産の実績の数は 14 である．また，他の機械上にて処理中の製品数は 4 である．この機械故障により，機械種別 M_{14} を必要とする工程種別は実施不可能となり，現行の生産計画は実行不可となる．この際，本実験では処理中の製品は取り外しできないとする．そのため，処理中の製品の工程は，変更できないものとして考え，ZDD 上では制約として扱う．この制約は生産の実績と同じ扱いとなる．設置機械固定，生産の実績及び機械故障の制約を満たす組合せを ZDD にて表現し，

表 6.16: 設置機械固定，生産の実績及び機械故障による ZDD の変化

ZDD の範囲	組合せ数	ノード数	計算時間
設置機械固定	780, 151, 357, 440	685	0.01[s]
生産の実績	12, 582, 912	176	0.01[s]
機械故障	6, 144	149	0.01[s]

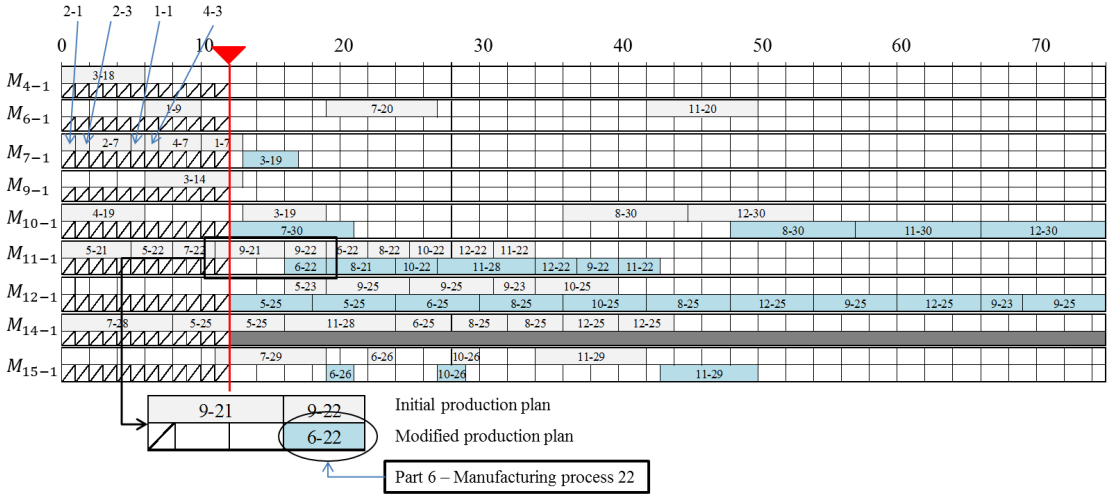


図 6.9: 生産計画のガントチャート (上段：初期生産計画，下段：修正後の生産計画)

その中から解探索を行い計画を変更する．

表 6.16 に設置機械固定，生産の実績及び機械故障の制約を ZDD に与えた結果を示す．表中の設置機械固定は，一度設置した機械は変更不可の制約を ZDD に与えた結果である．この制約により，生産開始の段階で ZDD の組合せ数は，7.8×10^{11} に減少し，初期の生産計画の際の ZDD と比較すると組合せ数が約 99.9%以上減少している．また，生産の実績の制約により，時刻 12 では ZDD の組合せ数は 1.3×10^{7} へ縮約した．最後に，機械故障による制約を付加することで，ZDD の組合せ数は，6, 144 へと減少した．生産の実績と機械故障を ZDD に対する制約として扱うことにより，制約充足解空間を縮約することができることがわかる．

この縮約した制約充足解空間は 6,144 通りであるため，全探索を行った．計算時間は約 2.74 秒である．残された制約を満たす代替候補の中で最小の TAC は，181.4 であった．内訳は，計画の変更が 17 箇所．そして，総納期遅れ時間は，178 である．この目的関数値が最小の生産計画に変更するとする．図 6.9 のガントチャートの各機械インスタンスの下の列は，機械故障後の生産スケジュールを表す．

6.4　単一の ZDD と ZDD ネットワークの比較

ZDD ネットワークの変数の宣言順序を定義する．単一の ZDD と同様にはじめに製品計画の変数を宣言する．工程順序の変数を単一の ZDD と同じ方法で宣言した後，製造工程の変数を $k = 1$ から順に宣言する．そして，機械種別の変数を $i = 1$ から順に宣言し，資源計画の資源の数の変数を宣言する．資源の数は，単一の ZDD と同じ方法で宣言する．

ZDD ネットワークを例題 2 に適用し，得られた結果を表 6.17 に示す．表中の Phase I は初期解空間としての選択肢の全組合せ集合の作成を意味する．Phase II は，選択性内の選択肢の制約を満たす組合せを Phase I の結果から取り出した結果である．Phase III は，選択性間の制約を表す ZDD と各選択性の ZDD を連結して得られた結果である．例題 2 では，用いた変数の総数は 1,530 である．

Phase II では，各選択性内の制約を満たす組合せを導出する．ここでは，工程順序及び資源の数にそれぞれ選択性内の制約が存在する．そのため，それら選択性の ZDD は制約を満たさない組合せを排除し，制約を満たすもののみで表現される．工程順序での制約条件により，99.9%以上の組合せが削除され，1,769,472 通りの組合せが残った．また，資源の数でも同様に，設置台数の制約及び機械インスタンスの重複の制約により，172,284 通りの制約を満たす組合せが取り出された．

Phase III では，選択性間の制約を ZDD で作成しそれらを連結することにより，制約を満たさない組合せを排除する．ここでは，工程順序を除く選択性の ZDD の組合せ数が減少している．工程順序のみでは，Phase II で得られた組合せは実行可能であるが，選択性間の制約を考慮しても PhaseII の制約を満たす組合せは全て実施可能であることがわかる．製造工程では，組合せが減少し，約 99.9%以上縮約した．機械種別では，32,768 通りの組合せのうち 23 通りが制約を満たすものとして取り出された．資源の数では，Phase II で縮約した結果に加えて，さらに 99%の組合せが削除された．選択性の選択肢の組合せを表す ZDD と，関連性をもつ選択性間の制約が記述された ZDD を用いて，選択性の ZDD が変更されると選択性間の制約により，連鎖的に各選択性の ZDD が更新される．この結果から，ZDD の連結によるネットワークは機能したということがわかる．

次に単一の ZDD の例題 2 の結果との比較を行う．その結果を表 6.18 に示す．用いた変数の数は，ZDD ネットワークでは，単一の ZDD に比べ，63.4%削減された．また，ZDD ネットワークでは，最終的に用いる選択性間の制約及び選択肢の組合せを表す ZDD の総

表 6.17: ZDD ネットワークによる各選択性の組合せ集合 (例題 2)

選択性	Phase	ノード数	組合せ数
工程順序	I	1,440	Approx.. 10^{433}
	II	96	1,769,472
	III	96	1,769,472
製造工程	I	30	1,073,741,824
	II	30	1,073,741,824
	III	255	5,545
機械種別	I	15	32,768
	II	15	32,768
	III	59	92
資源の数	I	45	35,184,372,088,832
	II	131	172,284
	III	73	159

表 6.18: 単一の ZDD と ZDD ネットワークによる比較

解法	変数の数	ノード数	計算時間 [s]
単一の ZDD	4,185	32,922	41.49
ZDD ネットワーク	1,530	総和:1,004	3.65

ノード数は 1,009 である．更に，計算時間は単一の ZDD と比べ ZDD ネットワークでは 91.4%短縮した．

ZDD ネットワーク表現では，各選択性の ZDD は実行可能解の候補を表している．1 つの選択性が新たに制約を受けると，連鎖して関係をもつ選択性の ZDD が相互に更新される．製品計画において，生産計画に用いる製造工程と機械種別の組合せを選択できるが，ある製品の何番目のどの工程種別に対して用いるかまで限定しておらず，単一の ZDD 表現と比べると抽象的な表現となっている．そのため，ZDD ネットワークは，全体の抽象的な表現までは単一の ZDD よりも素早く表現可能であるが，厳密な実行可能解を得るためには更に計算が必要である． 次に ZDD ネットワークによる制約充足解空間から解探索を行う．GA のパラメータは，単一の ZDD の際の GA 探索と同様に表 6.15 のものに設定した．目的関数 TAC の第二項は 0 とする．10 回試行を行い，単一の ZDD に対しての GA 探索との比較を行う．その結果を表 6.19 に示す．10 回の試行では，最良値の平均は僅かながら単一の ZDD を対象とした GA の方がよい結果が得られた．また標準偏差から，ZDD ネットワークを対象とする GA ではバラつきが多い．しかし，最良値は単一の ZDD より

表 6.19: 各制約充足解空間表現を対象とした GA による探索結果

探索対象	最良値の平均	標準偏差	平均計算時間
単一の ZDD	125.2	2.32	9.56
ZDD ネットワーク	125.8	21.53	80.1

も良い値を得ることもあった．計算時間では，単一の ZDD を対象とする GA の 8.38 倍長い．ZDD ネットワークでは，ある選択性の選択肢が選択されると，関係をもつ選択性間の制約条件により，他の選択性の ZDD が更新される．そのため，計算時間の約 93%程がデコーディングにかかる時間となった．

6.5　本章のまとめ

本章では，第 4 章で提案した単一の ZDD による制約充足解空間とその空間を探索する GA の検証，及び 5 章で提案した ZDD ネットワークの検証実験として，それら手法を用意した例題に適用し，有効性を示した．例題 1 では，2^{862} の解空間に対して選択性及び選択性間にある制約関係を ZDD の演算処理を用い，制約を満たさない組合せを排除することにより，単一の ZDD による制約充足解空間が作成されることを示した．ZDD で表される制約充足解空間は，もとの解空間に対して 99.9%以上縮約した．

また例題 1 では，作成された制約充足解空間に対して，提案する GA による探索を行った．探索済みの 1-パスを表す遺伝子配列の変更方法では，random がもっとも効果的であることを明らかにした．そして，提案した GA により得られた結果を全探索による結果と比較し，提案した GA は効率よく良解を得られることを明らかにした．さらに，制約充足解空間は ZDD の変数の宣言順序により解の分布が異なり，解探索に適した変数の宣言順序が必要であることについて述べた．

例題 2 では，生産実施中の動的事象を取り入れ，制約充足解空間の変化に対しての検証を行った．動的事象を ZDD で制約として扱うことで制約充足解空間を常に整合して保持することを明らかにした．そして，動的事象により更新された制約充足解空間に対して全探索を行い，計画の変更を行うことで計画の最適化が行われることを示した．また，例題 2 に対しては ZDD ネットワークによる制約充足解空間表現を行い，単一の ZDD 表現との

比較を行った．ZDD ネットワークは，単一の ZDD と比べ短い時間で制約充足解空間を作成可能であるが，GA 探索を用いて厳密な制約充足する組合せを導出するためにさらに計算が必要であるという結果を得た．

第7章　結論

7.1　本論文の成果

製造業は，企業間競争において第一線を維持するために，敏捷さ，再構築性，再構成，拡張性の特徴をもつ新たな生産システムを制御するための手法を導入することが求められる．そこで本研究では，生産プランニング問題を統合的に扱い，解空間内の制約を充足した解のみで構成される制約充足解空間を ZDD を用いて表現するための解法，および制約充足解空間を探索対象とした GA を提案し，各章にて以下の結論を得た．

第 2 章では，生産プランニングの概要を述べた上で，本研究で扱う生産プランニングの構成および選択性を定義し，それら選択性の関係性による波及を明らかにし，生産プランニングの解空間を統合的に扱うことの重要性について述べた．そして，生産プランニングを扱う既存研究における研究動向および問題点について論じた．得られた結論は以下のとおりである．

1. 生産プランニング問題を統合的に解くことは，組合せの爆発を引き起こすため，従来では部分問題として段階的に解く方法が取られているが，これら順次段階的なアプローチは，よりよい解の選択を妨げることを明らかにした．

2. 生産プランニングにおける解空間では，制約を満たさない組合せが存在し，それらを含めて解空間をサンプリングして最適化計算が行われている．最適化を行う前段階として，その解空間を制約を充足した解のみで表現することが必要であることを明らかにした．

第 3 章では，第 2 章で明らかにした生産プランニングの解空間の制約充足した解の表現として，組合せ集合データ処理に特化したグラフ表現である ZDD について述べた．得られた結論は以下のとおりである．

1. ZDD を用いることにより組合せ集合をデータ構造としてコンパクトに保持すること

ができることについて述べた．また，ZDD にもとづいてスクリプト処理システムである VSOP について言及し，各演算処理について述べた．

2. ZDD の適用例として N クイーン問題について述べた．N クイーン問題は制約充足問題のひとつであることについて触れ，制約充足した解から最適解を求める問題である生産プランニング問題にも適用可能であることについて言及した．

第 4 章では，第 3 章の ZDD を用いて生産プランニングの制約充足解空間を構成する解法及び制約充足問題を探索対象とした GA を提案した．得られた結論は以下のとおりである．

1. ZDD による制約充足解空間を表現するための解法を提案した．ここでは，工程順序と製造工程の選択肢の組合せ集合を表す ZDD に作成し，そこから機械種別や資源の数を包含するように ZDD の適用範囲を拡大し，その都度選択性間の制約関係を付加することとにより制約を充足した組合せのみの集合が得られることについて述べ，各プロセスを ZDD の演算処理を用いた解法を提案した．

2. 動的事象である機械故障と生産の実績を制約充足解空間に与える制約として捉え，それら影響を ZDD の演算処理を用いて表現し，制約充足解空間が常に整合して保持される解法を提案した．

3. 制約充足解空間を対象とした解探索手法として GA を提案した．提案手法では，遺伝子操作により致死遺伝子が発生させず，常に ZDD 内を探索するようにエンコーディングでは頂点ノードから出現するノードが表す変数の選択 (0/1) を遺伝子として表現する方法を提案した．

4. GA は ZDD 内を探索する過程にて生じる探索済みの 1-パスが発生した際に遺伝子配列の一部を変更する方法を提案した．提案方法では，top，bottom，random の 3 種類の変更方法について，それぞれの変更方法の特徴を示した．

第 5 章では，第 4 章で提案した ZDD の制約充足解空間とは異なり，ZDD の連結によるネットワーク型の制約充足解空間表現について提案した．得られた結論は以下のとおりである．

1. 単一の ZDD における問題点を挙げ，それらを解決するために ZDD ネットワークを提案した．

2. ZDD ネットワークは，各選択性内の選択肢の組合せ集合と，選択性と関係をもつ選択性間の制約関係を表す組合せ集合を ZDD にて管理し，それらを相互作用させ，ネットワーク構造を構築する．任意の選択性の ZDD が変化すると連動して関係をもつ選択性の ZDD が選択性間の制約関係を満たすように更新され，常に制約充足解空間が保持される方法を提案した．

3. 第 4 章で提案した GA を応用し，ZDD ネットワークによる制約充足解空間を探索する方法を提案した．ネットワークで用いる変数は，具体的にどの製品のどの順序に製造工程または機械種別を用いるのかという情報をもたない．この課題の解決策として，GA のデコーディングにおいて，各選択性から得られる 1-パスを表す ZDD と，選択性間の組合せを表す ZDD を用いて，各製品の各順序にて用いる製造工程及び機械種別の割付を行う方法を提案した．

第 6 章では，第 3 章及び第 4 章で提案した手法の検証評価実験を行った．得られた結論は以下のとおりである．

1. 例題 1 では，2^{862} の解空間に対して ZDD を用いて制約充足解空間を表現することにより，考慮するべき解候補が 99.99%削減された．制約充足解空間を探索する GA では，致死遺伝子を発生させず探索が行われ，探索済みの 1-パスに対しての変更方法としては，random が優れていることを示した．全探索による解の分布と GA によって得られた結果の比較では，提案した GA では効率よく良解を得られることを示した．

2. 例題 2 では，生産実施中の動的事象を取り入れ，提案手法による制約充足解空間の更新を検証した．動的事象を ZDD の制約として扱うことで制約充足解空間を常に整合して保持することを明らかにした．そして，動的事象により更新された制約充足解空間に対して全探索を行い，計画の変更を行うことで計画の最適化が行われることを示した．

3. 例題 2 に対しては ZDD ネットワークによる制約充足解空間表現を行い，単一の ZDD 表現との計算時間及びノード数との比較を行った．ZDD ネットワークは，単一の ZDD と比べ短い時間で制約充足解空間を作成可能である．また，ZDD ネットワークによる制約充足解空間に対して GA の解探索を行い，単一の ZDD による制約充

足解空間に対しての解探索結果と比較を行った．ZDD ネットワークでは，選択性内の選択肢の選択を行うと，関係をもつ選択性の ZDD が相互作用して更新が行われるため，GA のデコーディングに費やす時間が多く，単一の ZDD による制約充足解空間に対しての探索時間よりも計算時間が長いことを明らかにした．

以上により，生産プランニングの統合的解空間内の制約を満たさない解を排除し，制約充足解空間を ZDD を用いて作成し，その限定された空間を探索することにより生産プランニングを統合的に解くことができることを示した．本研究の新規性，学術的有効性，産業的有効性は以下のとおりである．

新規性：

従来研究で提案された生産システムの制御手法では，問題を経験則に基づき部分問題に分割し，順次段階的に解く手法が取られているが，このアプローチではよりよい解の選択の妨げや，部分問題の解が全体としての実行可能解ではないといった問題がある．本研究では，生産プランニングの選択性の波及を考慮し，統合的に問題を解くための手法として，ZDD による制約充足解空間表現を提案した．最適化計算を行う前に制約充足解空間を構成することで全体としての実行可能解の候補を ZDD で管理し，新たな制約が発生してもそれらを満たす解を ZDD の演算処理により導出することができる．また，ZDD は保持する組合せ数を知ることができ，その制約充足解空間の規模から解法の選択が可能である．生産プランニング分野において本手法は独創的であり，いままでにないアプローチである．

学術的有効性：

本研究でのアプローチは，生産システムの計画問題を統合的に解くために，ZDD を用いて制約充足解空間を作成し制約を満たさない解を排除した上で最適化問題へ取り組む．生産プランニングでは，選択性の波及関係が上流から下流または下流から上流へと波及する．このような問題構造をもつ他の分野の計画問題を有する他の領域においても，本研究のアプローチは応用可能であると考えられる．例として，建築やロジスティクスなどが挙げられる

産業的有効性：

提案する動的生産プランニングの統合的計画法は，各部分問題を個別に解くのではなく，統合的に扱い解を得ることが可能である．計画に費やすリードタイムの短縮，動的変化への柔軟な対応という点において産業に貢献できる．

7.2　今後の研究課題

本研究において残る研究課題を以下に述べる．

1. 制約充足解空間を構成する ZDD では，変数の宣言順序によりノード数が異なる．加えて，変数の宣言順序により，解の分布も異なる性質をもつ．そのため，問題ごとになるべく少ないノード数で制約充足解空間を構成し，また制約充足解空間内の最適解の分布を一箇所に集めることができる変数の宣言順序を導出する手法の提案が課題として挙げられる．

2. 本論文では，単一の ZDD との比較として ZDD の連結によるネットワーク表現を提案した．ZDD ネットワークを対象とした GA では，デコーディングにかかる時間が全体の計算時間に対して長い．ZDD ネットワークを対象とした GA のエンコーディング方法の改善や，GA のみならず他の手法の提案が課題として挙げられる．

3. 本論文では，動的変化として機械故障を扱い，それを制約充足解空間を表す ZDD に対する制約とみなし，制約充足解空間を常に整合して保持する手法を提案した．生産現場では他にも様々な動的変化が発生する．それら動的変化を ZDD にて扱うための解法の提案及び ZDD ネットワークへの対応が課題として挙げられる．

謝辞

本研究を進めるにあたり，多くの方々にお世話になりました．ここに深く感謝の意を表します．

指導教員であり本研究論文の主査を引き受けて頂いた小野里雅彦教授には，研究テーマの設定，研究の遂行から学生生活に至るまで，熱心な御指導，御鞭撻，多大な尽力を賜ると共に，常に暖かく寛大な心で見守って頂きました．心から御礼を申し上げます．副査を引き受けて頂いた金井理教授には，研究遂行および本論文の執筆にあたり，数多くの有意義な御助言を頂きました．心からお礼申し上げます．副査を引き受けていただいたシステム環境情報学研究室の田中文基准教授にはミーティング等の研究進捗報告の場で積極的な御指導を頂きました．研究内容に対しての指摘は的確なものばかりで，議論を通じて博士論文をより洗練されたものにすることができました．心から御礼を申し上げます．本研究の主要な情報技術である ZDD の提案者であり，情報理工学専攻アルゴリズム研究室湊真一教授には，生産プランニング問題へ応用に対しての御助言，SAPPOROBDD パッケージ及び Ruby 版 VSOP の使用許可を頂きました．心からお礼申し上げます．また，芝浦工業大学安藤公一名誉教授には，博士課程進学に関する進路について御助言を頂きました．心からお礼申し上げます．株式会社ディビジョン・エンジニアリング　大内功氏代表取締役には，SIer の観点からシステムの構成やプログラミング技術等の情報技術を教えて頂きました．また，システムエンジニアとして，活躍の場を頂き，実務を経験させて頂きました．心からお礼申し上げます．私が，システム環境情報学研究室で研究活動を行うに際して，研究室の先輩・後輩の皆様には数々の有意義な御助言を頂きました．また，私生活においても，非常に楽しく有意義な時間を共にし，数々の貴重な体験をさせていただいたことは，今後の人生にとっても大きな糧となります．皆様に心からお礼申し上げます．最後に，長年に渡り学生生活を支えて頂くと共に，暖かく見守り励まして頂いた家族に，心から感謝の意を表します．

参考文献

[1] M.K. Lim, Z. Zhang, “A multi-agent system using iterative bidding mechanism to enhance manufacturing agility” Expert Systems with Applications, Vol.39, No.9, pp.8259-8273 (2012)

[2] 相吉栄太郎, 安田恵一郎, “メタヒューリスティクスと応用”, オーム社 (2007)

[3] R. E. Bryant, “Graph-based algorithms for Boolean function manipulation”, IEEE Trans. Comput., Vol.C-35, No.8, pp.667-691 (1986)

[4] Shin-ichi Minato, “Zero-Suppressed BDDs for Set Manipulation in Combinatorial Problems”, 30th ACM/IEEE Design Automation Conference, pp.272-277 (1993)

[5] 湊　真一, “BDD/ZDD を基盤とする離散構造と演算処理系の最近の展開”, 電子情報通信学会　基礎・協会ソサイエティ Fundamentals Review, Vol.4, No.3, pp.224-230 (2010)

[6] 岩田一明, “生産工学入門”, 森北出版株式会社 (1997)

[7] Kunlei Lian, Chaoyong Zhang, Xinyu Shao, “Optimization of process planning with various flexibilities using an imperialist competitive algorithm”, The International Journal of Advanced Manufacturing Technology, Vol.59, No.5-8, pp.815-828 (2012)

[8] Leung CW, Wong TN, Mak KL, Fung RYK, “Integrated process planning and scheduling by an agent-based ant colony optimization”, Computers & Industrial Engineering, Vol.59, No.1, pp.166180 (2010)

[9] Guo Y, Mileham A, Owen G, Li W, “Operation sequencing optimization using a particle swarm optimization approach”, Proceedings of the Institution of Mechanical

Engineers, Part B: Journal of Engineering Manufacture, Vol.220, No.12, pp.19451958 (2006)

[10] Rakesh Kumar Phanden, Ajai Rain, Rajiv Verma, “Integration of process planning and scheduling: a state-of-the-art review”, International Journal of Computer Integrated Manufacturing, Vol.24, No.4-6, pp.517-534 (2011)

[11] Atashpaz-Gargari and Lucas, “Imperialist Competitive Algorithm: An Algorithm for Optimization Inspired by Imperialistic Competition”, Evolutionary Computation, 2007. CEC 2007. 25-28 Sept. 2007

[12] Qiao L, Wang X-Y, Wang S-C, “A GA-based approach to machining operation sequencing for prismatic parts”, International Journal of Production Research, Vol.38, No.14, pp.32833303 (2000)

[13] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, “Optimization by Simulated Annealing”, Science, Vol.220, No.4598, pp.671-680 (1983)

[14] Ma GH, Zhang YF, Nee AYC, “A simulated annealing-based optimization algorithm for process planning”, International Journal of Production Research, Vol.38 No.12, pp.26712687 (2000)

[15] Russell Eberhart, James Kennedy, “A new optimizer using particle swarm theory”, Proceedings of the Sixth International Symposium on Micro Machine and Human Science, pp.39-43 (1995)

[16] Jacques Ferber, “Multi-Agent System: An Introduction to Distributed Artificial Intelligence”, Addison-Wesley Professional

[17] テヘラニ ニキ ネジャド ホセイン, 杉村延広, 岩村幸治, 谷水義隆, “フレキシブル生産システムにおける動的工程設計のための自律型エージェント”, システム制御情報学会論文誌, 22, pp.260-272 (2009)

[18] Lihui Wang, “Combining Facility Layout Redesign and Dynamic Routing for job-shop Assembly Operations”, 2011 IEEE International Symposium on Assembly and Manufacturing, pp.25-27 (2011)

[19] M. A. El-Baz, “A genetic algorithm for facility layout problems of different manufacturing environment”, Computers and Industrial Engineering, vol.47, pp.233-246 (2004)

[20] P. Jain and P. K. Sharma, “Solving job shop layout problem using ant colony optimization technique”, IEEE International Conference on Systems, Man and Cybernetics, pp.288292 (2005)

[21] F. Defersha and M. Chen, “A simulated annealing algorithm for dynamic system reconfiguration and production planning in cellular manufacturing”, International Journal of Manufacturing Technology and Management, Vol.17, No.1/2, pp.103124 (2009)

[22] M Dorigo, Di Caro G, Gambardella LM, “Ant Algorithms for Discreate Optimization”, Artificial Life, Vol.5, No.2, pp137-172

[23] Luisa Huaccho Huatuco, Janet Efstathiou, Ani Calinescu, Suja Sivadasan, Stella Kariuki, “Comparing the impact of different rescheduling strategies on the entropic-related complexity of manufacturing systems”, International Journal of Production Research, Vol.47, No.15, pp.4305-4325 (2009)

[24] 谷水義隆，阪口龍彦，杉村延広，“遺伝的アルゴリズムを用いたリアクティブスケジューリング”，日本機械学会論文集 (C 編)，Vol.69, No.685, pp.234-239 (2003)

[25] Amir Rajabinasab, Saeed Mansour, “Dynamic flexible job shop scheduling with alternative process plans: an agent-based approach”, International Journal of Advanced Manufacturing Technology, 54:1091-1107(2011)

[26] Y. W. Guo, Q. D. Li, A. R. Mileham, G. W. Owen, “Optimisation of integrated process planning and scheduling using a particle swarm optimisation approach”, International Journal of Production Research, Vol.47, No.42, pp.3775-3796 (2009)

[27] T. N. Wong, C. W. Leung, K. L. Mak, R. Y. K. Fung , “Integrated process planning and scheduling/rescheduling: an agent-based approach”, International Journal of Production Research, Vol. 44, Nos. 18.19, 15 September.1 October(2006)

[28] Y F Wang, T Zhang, J Y Fuh, “Job rescheduling by exploring the solution space of process planning for machine breakdown/arrival problems”, Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture February 1, vol. 225 no. 2 282-296 (2011)

[29] Weiming Shen, Lihui Wang, Qi Hao, “Agent-Based Distributed Manufacturing Process Planning and Scheduling: A State-of-the-Art Survey”, Systems, Man, And Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, Volume 36, Issue 4, 2006

[30] Akers, S.B., “Binary Decision Diagmras”, IEEE Trans. Comput., Vol.C-27, No.6, pp.509-516 (1978)

[31] 鈴木　拡, 湊　真一, “BDD/ZDD を用いたペントミノパズルの解の列挙”, 電子情報通信学会. COMP, コンピュテーション, Vol.109, No54, pp.1-7 (2009)

[32] 奥乃　博，湊真一，“算術論理式システム BEM-II を使ってパズルを解こう”，bit, Vol.29， No.4 (1997)

[33] 湊　真一，“特集「BDD（二分決定グラフ）」，計算機状での BDD の処理技法”，情報処理学会誌，Vol.34，No.5，pp.593-599 (1993)

[34] D. E. Knuth, “The Art of Computer Programming”, Volume. 4A, Combinatorial Algorithms, Part 1. 1st. Addison-Wesley Professional, 2011. ISBN:0321751043

[35] Shin-ichi Minato, “VSOP (Valued-Sum-of-Products) Calculator for Knowledge Processing Based on Zero-Suppressed BDDs”, Lecture Notes in Computer Science, Vol.3847, pp.40-58 (2006)

[36] Shin-ichi Minato, “Zero-suppressed BDDs and Their applications”, International Journal on Software Tools for Technology Transfer, Vol.3, No.2, pp.156-170 (2001)

[37] J. H. Holland, “Adaptation in Natural Artificial Systems”, University of Michigan Press, Ann Arbor, MI., (1975)

[38] F. Glover, M. Laguna, “Tabu Search: A Tutorial” Interfaces, Vol.20, No.4, pp.74-94 (1997)

[39] Sadiq M. Sait, Habib Youssef, “Interactive Computer Algorithms with Application in Engineering -Solving Combinatorial Optimization Problems (組合せ最適化アルゴリズムの最新手法－基礎から工学応用までー，白石洋一訳)”, 丸善株式会社 (2002)

[40] R. Haupt, “ A survey of priority rule-based scheduling”, Operations-Research-Spektrum, Vol.11. No.1, pp.3-16 (1989)

[41] NYSOL, http://www.nysol.jp/

研究業績

[1] Keita TAKAHASHI, Masahiko ONOSATO, Fumiki TANAKA, Comprehensive representation of feasible combinations of alternatives for dynamic production planning using Zero-Suppressed Binary Decision Diagram, Journal of Advanced Mechanical Design, Systems, and Manufacturing, Special Issue on Advanced Manufacturing Technology, Vol.8, No.4 Paper No.14-00102 (2014 年 10 月)

[2] 高橋啓太, 小野里雅彦, 田中文基, ゼロサプレス型 BDD を用いた動的生産プランニングの統合的解候補表現のための解法, システム制御情報学会論文誌, 第 28 巻第 3 号 (2015 年 3 月号掲載)

[3] Keita TAKAHASHI, Masahiko ONOSATO, Fumiki TANAKA, A comprehensive approach for managing feasible solutions in production planning by an interacting network of Zero-Suppressed Binary Decision Diagrams, Journal of Computational Design and Engineering, Vol.2, No.2 (2014 年 12 月採択)

[4] Keita TAKAHASHI, Masahiko ONOSATO, Fumiki TANAKA, Comprehensive representation of feasible combinations of alternatives for dynamic production planning using Zero-Suppressed Binary Decision Diagram, The 7th International Conference on Leading Edge Manufacturing in 21st Century, pp. 559-564 (Miyagi, September 2013)(推薦により，上記論文 [1] として学術雑誌に掲載)

[5] Keita TAKAHASHI, Masahiko ONOSATO, Fumiki TANAKA, A solution method for comprehensive solution candidates by Zero Suppressed Binary Decision Diagram in dynamic production planning, 2014 International Symposium on Flexible Automation, 2014-66L (Hyogo, July 2014)(推薦により，上記論文 [2] として学術雑誌に掲載)

[6] Keita TAKAHASHI, Masahiko ONOSATO, Fumiki TANAKA, A comprehensive approach for managing feasible solutions in production planning by an interacting network of Zero-Suppressed Binary Decision Diagrams, The Asian Conference on Design and Digital Engineering 2014 (Shandong, China)(推薦により，上記論文 [3] として採択，掲載予定)