



# 分散協調型生産スケジューリング手法の半導体製造における実用化に関する研究

黒瀬, 伸二

---

(Degree)

博士 (工学)

(Date of Degree)

2013-03-25

(Date of Publication)

2013-05-08

(Resource Type)

doctoral thesis

(Report Number)

甲5772

(URL)

<https://hdl.handle.net/20.500.14094/D1005772>

※ 当コンテンツは神戸大学の学術成果です。無断複製・不正使用等を禁じます。著作権法で認められている範囲内で、適切にご利用ください。



# 博士論文

分散協調型生産スケジューリング手法の  
半導体製造における実用化に関する研究

平成25年1月

神戸大学大学院工学研究科

黒瀬伸二



## 目次

1	緒論	1
1.1	研究背景	1
1.2	研究目的	2
1.3	本論文の構成	3
2	分散協調型手法を用いた半導体生産スケジューリング方式の提案	5
2.1	緒言	5
2.2	半導体製造におけるスケジューリングの課題	5
2.2.1	半導体製造の概要	5
2.2.2	半導体前工程におけるスケジューリングの課題	5
2.2.3	半導体後工程（組立工程）におけるスケジューリングの課題	9
2.2.4	半導体後工程（テスト工程）におけるスケジューリングの課題	10
2.3	先行研究の現実生産ラインへの適用状況	11
2.4	ラグランジュ分解・調整法	12
2.5	結言	13
3	半導体製造における分散協調型スケジューリング手法の一提案	15
3.1	緒言	15
3.2	対象システム	15
3.3	ラグランジュ分解・調整法の適用	15
3.3.1	記号	15
3.3.2	原問題の定式化	16
3.3.3	緩和問題の作成	17
3.3.4	動的計画法による部分問題の解法（上界値計算）	18
3.3.5	実行可能スケジュールの解法（下界値計算）	18
3.3.6	双対問題の解法	18
3.3.7	最適化アルゴリズム	19
3.4	提案手法（リストスケジューリングのアルゴリズム改善）	21
3.4.1	最も単純なアルゴリズム	21
3.4.2	アルゴリズムの改善	23
3.5	計算機実験	24
3.5.1	実験条件	24
3.5.2	実験シナリオ	27
3.5.3	予備実験	28
3.5.4	実験 I（単純アルゴリズム）	29
3.5.5	実験 II（下界値計算のアルゴリズム改善）	36

3.5.6	実験 III (上界値計算のアルゴリズム改善)	40
3.5.7	実験 IV (LDC の詳細特性確認)	43
3.6	結言	51
4	分散協調型スケジューリング手法の半導体前工程実規模モデルへの適用	53
4.1	緒言	53
4.2	対象システムおよび定式化	53
4.3	計算機実験	53
4.3.1	実験条件	53
4.3.2	実験 I (第 3 章における提案手法の適用)	54
4.3.3	実験 II (リストスケジューリングの更なるアルゴリズム改善)	56
4.3.4	実験 III (高負荷状態における追加検証)	61
4.4	結言	65
5	分散協調型スケジューリング手法の半導体後工程生産ラインへの適用	67
5.1	緒言	67
5.2	対象システム	67
5.3	定式化	68
5.4	計算機実験	68
5.4.1	実験条件	68
5.4.2	実験 I (後工程単純モデルにおける LDC の有効性確認)	70
5.4.3	実験 II (後工程単純モデルにおける LDC の特性確認)	72
5.4.4	実験 III (後工程複雑モデルにおける LDC の有効性確認)	72
5.4.5	実験 IV (更なる解の改善のための方案検証)	75
5.5	結言	80
6	結論	81
6.1	結論	81
6.2	今後の展望	82
	参考文献	83
	本論文の構成	87
	謝辞	89

## 目次

Fig. 2.1	Semiconductor Manufacturing .....	5
Fig. 2.2	Wafer Process .....	6
Fig. 2.3	Production Unit.....	7
Fig. 2.4	Frequent Split and Merge.....	7
Fig. 2.5	Lot Batching .....	8
Fig. 2.6	Various Productions .....	9
Fig. 2.7	Assembly Process .....	10
Fig. 3.1	Re-entrant Flow Shop.....	15
Fig. 3.2	Algorithm with LDC.....	20
Fig. 3.3	Algorithm of Simple List Scheduling .....	21
Fig. 3.4	Original Gantt chart by DP .....	22
Fig. 3.5	Optimized Gantt chart by Simple Algorithm.....	22
Fig. 3.6	Algorithm of Improved List Scheduling.....	23
Fig. 3.7	Optimized Gantt chart by Proposal Algorithm.....	24
Fig. 3.8	Job Type 1 (Mass Production).....	24
Fig. 3.9	Job Type 2 (Mass Production).....	25
Fig. 3.10	Job Type 3 (Pilot Production).....	25
Fig. 3.11	Utilization of Bottle-Neck Machine (#4).....	28
Fig. 3.12	Total Tardiness By Dispatching rule.....	28
Fig. 3.13	Duality Gap I by Iteration in the 33 <sup>rd</sup> trial.....	31
Fig. 3.14	Total Tardiness I by Iteration in the 33 <sup>rd</sup> trial.....	31
Fig. 3.15	Duality Gap I by Iteration in the 73 <sup>rd</sup> trial.....	32
Fig. 3.16	Total Tardiness I by Iteration in the 73 <sup>rd</sup> trial.....	32
Fig. 3.17	Duality Gap I by Iteration in the 92 <sup>nd</sup> trial.....	33
Fig. 3.18	Total Tardiness I by Iteration in the 92 <sup>nd</sup> trial .....	33
Fig. 3.19	Gantt Chart of the feasible schedule I in the 92 <sup>nd</sup> trial.....	34
Fig. 3.20	Optimizing process I with DP in the 92 <sup>nd</sup> trial .....	35
Fig. 3.21	Duality Gap II by Iteration in the 92 <sup>nd</sup> trial .....	37
Fig. 3.22	Total Tardiness II by Iteration in the 92 <sup>nd</sup> trial.....	37
Fig. 3.23	Optimizing process II with improved DP in the 92 <sup>nd</sup> trial .....	38
Fig. 3.24	Gantt Chart of the feasible schedule II in the 92 <sup>nd</sup> trial .....	39
Fig. 3.25	Duality Gap III by Iteration in the 92 <sup>nd</sup> trial.....	41
Fig. 3.26	Total Tardiness III by Iteration in the 92 <sup>nd</sup> trial .....	41
Fig. 3.27	Gantt Chart of the feasible schedule III in the 92 <sup>nd</sup> trial.....	42

Fig. 3.28	Total Tardiness IV in 16 jobs with Subgradient Method.....	44
Fig. 3.29	Total Tardiness IV in 16 jobs with Monotonic Non-Decreasing Method	44
Fig. 3.30	Gantt chart IV of the best case in 16 jobs by LDC.....	45
Fig. 3.31	Gantt chart IV of the best case in 16 jobs by Dispatching rule (EDD) ..	46
Fig. 3.32	Gantt chart IV of the worst case in 16 jobs by LDC .....	47
Fig. 3.33	Gantt chart IV of the worst case in 16 jobs by Dispatching rule (EDD)	48
Fig. 3.34	Completed contidion ratio with Subgradient Method ( $\beta=0.0005$ ) .....	49
Fig. 3.35	Completed contidion ratio.....	49
Fig. 3.36	CPU Time VI with Subgradient Method.....	50
Fig. 3.37	CPU Time VI with Monotonic Non-Decreasing Method.....	50
Fig. 3.38	Average iteration number VI.....	51
Fig. 4.1	Total Tardiness I by Itaration.....	55
Fig. 4.2	Total Tardiness I by Step Size .....	56
Fig. 4.3	Optimized Gantt chart by Proposal Algorithm Type2.....	56
Fig. 4.4	Algorithm of Improved Type2 List Scheduling .....	57
Fig. 4.5	Calculation completed contidion with Simple algorithm .....	59
Fig. 4.6	Calculation completed contidion with Proposal algorithm Type1.....	59
Fig. 4.7	Calculation completed contidion with Proposal algorithm Type2.....	60
Fig. 4.8	Average Iteration Number .....	60
Fig. 4.9	Duality Gap by Iteration in 1/1 scale .....	61
Fig. 4.10	Total Tardiness by Iteration in 1/1 scale .....	62
Fig. 4.11	Calculation completed contidion in the overload conditon.....	63
Fig. 4.12	Average Iteration and the Itertation in solving best result .....	63
Fig. 4.13	Total Tardiness by Iteration in 1/1 scale .....	64
Fig. 5.1	Flow Shop .....	67
Fig. 5.2	Job Type 1 (Mass Production).....	68
Fig. 5.3	Job Type 2 (Mass Production).....	68
Fig. 5.4	Job Type 3 (Pilot Production).....	68
Fig. 5.5	Duality Gap I by iteration in the 7 <sup>th</sup> trial .....	71
Fig. 5.6	Total Tardiness I by iteration in the 7 <sup>th</sup> traial .....	71
Fig. 5.7	Gantt chart by LDC .....	74
Fig. 5.8	Gantt chart by Dispatching rule (FIFO: First-IN, First-OUT) .....	74
Fig. 5.9	Gantt chart by Dispatching rule (EDD: Earliest Due Date) .....	74
Fig. 5.10	Duality Gap IV by iteration in the 1 <sup>st</sup> traial .....	76
Fig. 5.11	Total Tardiness IV by iteration in the 1 <sup>st</sup> traial .....	76
Fig. 5.12	Gantt chart by Improved LDC.....	77

Fig. 5.13	Gantt chart by Dispatching rule (Shortest Setup Time) .....	77
Fig. 5.14	Duality Gap IV by iteration in the 15 <sup>th</sup> trial.....	78
Fig. 5.15	Total Tardiness IV by iteration in the 15 <sup>th</sup> trial.....	78
Fig. 5.16	Total Tardiness IV-2 by iteration in the 15 <sup>th</sup> trial.....	79

## 表目次

Table 1.1	Changes in Semiconductor Manufacturing .....	1
Table 1.2	Scheduling Businesses in Production Line .....	2
Table 2.1	Actual operation in semiconductor fabrication .....	12
Table 3.1	Process Chart of Job Type .....	25
Table 3.2	Job Sample ( $L=16$ ).....	26
Table 3.3	Experimental Parameters.....	27
Table 3.4	Experimental Scenarios .....	27
Table 3.5	Experimental Results I (Total Tardiness) .....	29
Table 3.6	Experimental Results I (CPU Time(sec)) .....	29
Table 3.7	Experimental Results I (CPU Detail Time(sec)) .....	29
Table 3.8	Experimentl Results I in each trial .....	30
Table 3.9	Job Model I in each trial.....	30
Table 3.10	Experimental Results II (Total Tardiness).....	36
Table 3.11	Experimental Results II (CPU Time(sec)) .....	36
Table 3.12	Experimental Results III (Total Tardiness).....	40
Table 3.13	Experimental Results III (CPU Time(sec)) .....	40
Table 3.14	Experiment Results IV (Total Tardiness and CPU Time(sec)) .....	43
Table 4.1	Scale Models .....	53
Table 4.2	Experimental Parameters.....	54
Table 4.3	Experimental Results I (Total Tardiness) .....	55
Table 4.4	Experimental Results II (Total Tardiness).....	58
Table 4.5	Experimental Results II (CPU Time (sec)).....	58
Table 4.6	Experimental Results III (Total Tardiness).....	61
Table 4.7	Experimental Results III (CPU Time (sec)) .....	62
Table 4.8	Experimental Results III (CPU Detail Time (sec)) .....	65
Table 5.1	Process Chart of Job Type .....	69
Table 5.2	Experimental Parameters I .....	69
Table 5.3	Experimental Results I (Total Tardiness) .....	70
Table 5.4	Experimental Results I (CPU Time(sec)) .....	70
Table 5.5	Experimental Results II (Total Tardiness and CPU Time(sec)) .....	72
Table 5.6	Experimental Parameters III.....	73
Table 5.7	Experimental Results III (Total Tardiness).....	73
Table 5.8	Experimental Results III (CPU Time(sec)) .....	73
Table 5.9	Experimental Results IV (Total Tardiness).....	75

Table 5.10	Experimental Results IV (CPU Time(sec)).....	75
Table 5.11	Experimental Results IV -2 (Total Tardnes and CPU Time(sec)).....	79



# 1 緒論

## 1.1 研究背景

1980年代後半から1990年代前半にかけて、計算機の高速・高性能化、並びにウィンドウシステムに代表されるソフトウェアの目覚ましい発展により多種多様なシミュレータが比較的容易に利用できる環境が整った。それ以降現在に至るまで様々な業種・分野の生産スケジューリング問題において、如何に有効的に活用すべきかが盛んに研究されてきた。

半導体製造においても、数千億円規模に及ぶ半導体前工程（ウェハプロセス）への投資を最適化、且つ最大限装置を有効利用することを主目的に実用化の研究を進めたものの、1,000台近い装置に数千個のジョブが組合さるだけではなく、同じ装置で複数の工程を処理するという複雑性も合わせ持ち、組合せ問題としては大規模過ぎるため、その適用範囲は、ボトルネック装置の稼働率最大化などの局所的な最適化に止まり、実用化という観点からも生産ライン建設時に必要な装置台数を検証するなど限定的利用であった。

1990年代初頭にメモリ事業において世界市場を席卷した日本半導体産業であるが、台湾企業を代表とするファウンダリ、欧米企業のファブレスといった新しいビジネスモデルの台頭に押され、その多くが撤退を余儀なくされた。これに対し日本企業は、得意とする複合化技術で差別化できると考える SoC (System on Chip) 事業に将来を見出そうとした。

**Table 1.1 Changes in Semiconductor Manufacturing**

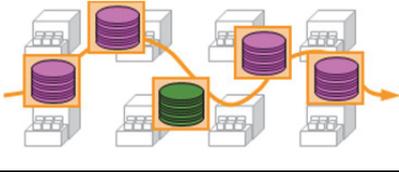
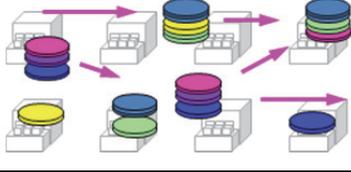
	Previous	Current
Products	Memory	SoC (System on Chip)
Application	Computer	Mobile phone, Automobile, Digital appliance
Life Cycle	4 years	3 - 6 months
Production Type	Low mix and High volume 	High mix and small-lot-size 
Index	Throughput, Machine Utilization	<b>TAT(Turn Around Time), Delivery, Cost</b>

Table 1.1 に示す通り、従来の主要製品であるメモリ製品は少品種多量生産であり、スケジューリング問題の課題もスループットおよび装置稼働率の向上と単純で、生産ラインのボトルネック装置を対象としたディスパッチングルールが有効に機能していた。

しかしながら近年の主要製品である SoC 製品は、ライフサイクルが数ヶ月から半年程度、仕様や需要が短期間に変化するために多品種少ロット生産となり、生産性が著しく低い上に、ボトルネックとなる装置もプロダクトミクス次第で容易に遷移してしまうため、従来の局所的な最適化だけでは、生産効率の最大化、製造コストの削減、TAT (Turn Around Time) の短縮といった課題が解決できなくなってきた。また、ロットが最小化することで装置を有効利用することが更に難しくなっている。

一方で、事業継続のための投資、すなわち新しい工場の建設や生産能力拡張の大部分を占める製造装置の購入費用を抑制することを目指し、従来企業別に特殊仕様のため更に高額となっていた製造装置仕様の世界標準化を推進した結果、現在では半導体製造、特に半導体前工程においては装置を購入して並べれば一般製品が比較的容易に製造できる程、装置利用技術に関する標準化が進むに至った。その上、信頼性をそれ程必要とされない民生品への半導体利用が拡大したことなどにより、2010 年前後には SoC 製品の低価格化が一挙に進み、労働生産性の高い韓国企業や台湾企業にシェアを奪われ、製造業としての SoC 事業そのものが成り立たなくなってきた。欧米企業が一早く工場を切り離しファブレスの設計企業に特化して行ったこともこの流れを加速させた。

こうしたビジネス環境の変化の中、半導体後工程においても、製品の更なる微細化やコストダウンにおいて製造技術力への要求が高まり、顧客に一番近い工程であることから、生産スケジューリングの必要性に対する認識が急速に高まってきた。

## 1.2 研究目的

**Table 1.2 Scheduling Businesses in Production Line**

Scheduling Business	Key Issues	Frequency in use
Constructing Fab.	A number of Equipment	Once in constructing
Equipment Capacity Study in the following three month	<ul style="list-style-type: none"> <li>• A number of Equipment and Jigs</li> <li>• Input Quantity by Products</li> </ul>	A few times a month
Daily Scheduling	Operation Schedule for all equipment	A few times a day
Dispatching (What's Next)	One lot to be processed for the idle equipment	Once per a few seconds
Dispatching (Where Next)	Some equipment in which the completed lot will be processed	Once per a few seconds

本研究は、生産ラインにおけるボトルネックなど一部装置やエリアに着目した最適化だけでは解決が困難な、生産ライン全体を効率化するための生産スケジューリング手法を提案し、Table 1.2 に示す実際の生産ラインでの生産スケジューリング業務において実用化することを目的とする。

### 1.3 本論文の構成

本論文では、前述した内容について以下の構成により述べる。

第 2 章では、対象とするシステムの課題を明確にする。そしてその解決策として、半導体製造への適用に関する先行研究は少ないが、生産スケジューリングに限らず様々な分野の最適化問題での研究が盛んな分散協調型スケジューリング手法の一つであるラグランジュ分解・調整法の適用を提案する。

第 3 章では、第 2 章で提案したスケジューリング手法につき、リエントラントを特徴とする半導体前工程生産ラインを抽象化した小規模モデルを用いて、従来手法であるディスプレイパッチングルールと比較して有効であることを、数値実験により示す。

第 4 章では、第 3 章の小規模モデルにおいて有効性を確認した提案手法を、半導体前工程を抽象化した大規模モデルへ適用する上での課題を明確にする。そしてその解決策を提案し、数値実験により有効性を示す。

第 5 章では、第 4 章までに確立した大規模モデルにおける生産スケジューリングの最適化手法を、生産能力の大きな装置を複数品種が共用し、段取り時間を必要とする品種切り替えが頻繁に行われることを特徴とする現実の半導体後工程生産ラインに適用し、解精度および計算時間の観点から実際の生産スケジューリング業務において実用化可能であることを、数値実験により示す。

第 6 章では、本研究のまとめと今後の展望について述べる。



## 2 分散協調型手法を用いた半導体生産スケジューリング方式の提案

### 2.1 緒言

本章では、対象システムである半導体製造のスケジューリングの課題を明確にする。そしてその解決策として、その高速性および高品質の解により注目を集めているラグランジュ分解・調整法を用いた、生産ライン全体を効率化するための生産スケジューリング方式を提案する。

### 2.2 半導体製造におけるスケジューリングの課題

#### 2.2.1 半導体製造の概要

半導体製造工程は、Fig. 2.1 に示す通り、シリコン単結晶をスライスした1枚のシリコンウェハの上に同時に数百個から数千個の電子回路（ICチップ）を作りこむ前工程と呼ばれるウェハプロセス工程と、1個ずつのICチップを切り離して電子基盤に組み込むためのパッケージに収納、およびそれを検査する後工程と呼ばれる組立・テスト工程から成る。

300mmのシリコンウェハの場合、1枚当たり約1,500個のICチップが作りこまれ、1～数个のチップをリードと呼ばれる電子回路に組み込むためのピンに配線され樹脂などでパッケージされ製品として完成する。

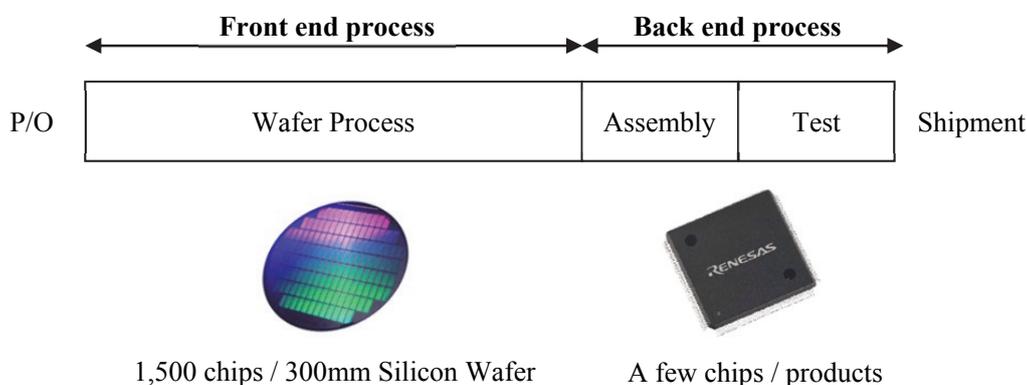


Fig. 2.1 Semiconductor Manufacturing

#### 2.2.2 半導体前工程におけるスケジューリングの課題

##### (1) 生産システムの規模と複雑性に関する課題

半導体前工程における生産スケジューリングの最大の課題は、数百台の装置に数千個のジョブの数百工程が組合さるだけでなく、同じ装置で複数の工程を処理するという複雑性である。Fig. 2.2に示す通り、最初にトランジスタやダイオードなどの部品を形成、近年

では7~8層にもわたり微細に配線を積み上げた後で、縦穴を通して配線間の結線をする。形成するパターンや、回路を形成する金属は工程により異なっても同じ装置で加工が可能な工程が多く、転写→エッチング→成膜→平坦化などの一連の処理を何回も繰り返す、リエントラントフローショップと呼ばれる生産形態をとる。装置1台が数千万円から数十億円と高価であるため高い稼働率が要求されること、数百工程に亘るため生産リードタイムが非常に長いことが生産管理上の基本的な課題となる[1]。

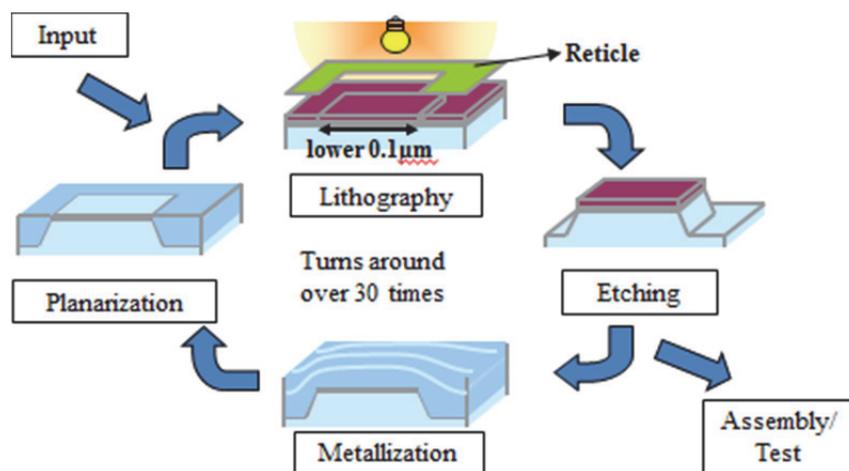


Fig. 2.2 Wafer Process

### (2) 製品品質に関する課題

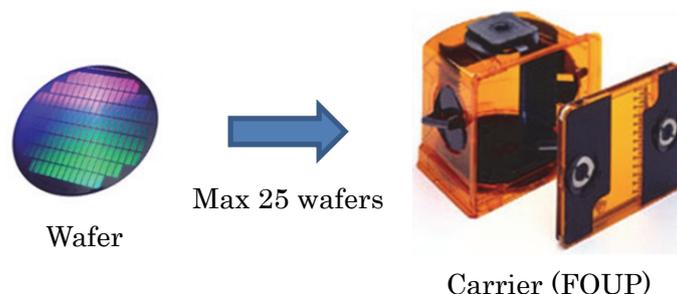
集積度が高まるに連れ多層化が進むことで、各層における配線パターンのズレが大きな問題となる。このため、露光装置では高い重ね合わせ精度が要求され、特に精度が高く要求されるいくつかの工程においては、代替装置の中から最初の工程で利用した露光装置と全く同一の露光装置を後続の工程でも利用しなければならない、一般的に装置限定と呼ばれる制約が発生する

酸化膜が絶縁層となるため、微細化が進むに連れ、自然酸化膜の問題が品質へ与える影響が大きくなってきた。これは、加工せずに放置しておくだけで自然に酸化膜が形成され回路の特性が設計仕様と異なってしまうという問題である。このため、成膜する前の薄い自然酸化膜を除去する工程から成膜までの工程間に、待ち時間の制約が必要となる。すなわち、ある工程の終了時刻から、次の工程の開始時刻に制限があるということである。もしもこの制限時間を超過した場合は、酸化膜除去工程をやり直すか、極めて高品質が要求される製品であれば、廃棄せざるを得ない。これは一般的に待ち時間制約と呼ばれる。

### (3) 多品種化に関する課題

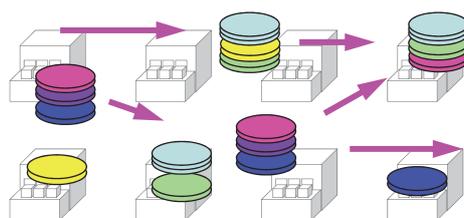
半導体前工程では加工対象となるウェハを、自動化された生産ラインで製造するため、

Fig. 2.3 に示す通り、FOUP (Front Opening Unified Pod) と呼ばれるキャリアに、受注単位に応じて最大 25 枚のウェハを収納、1 生産単位 (ロット) とされる。



**Fig. 2.3 Production Unit**

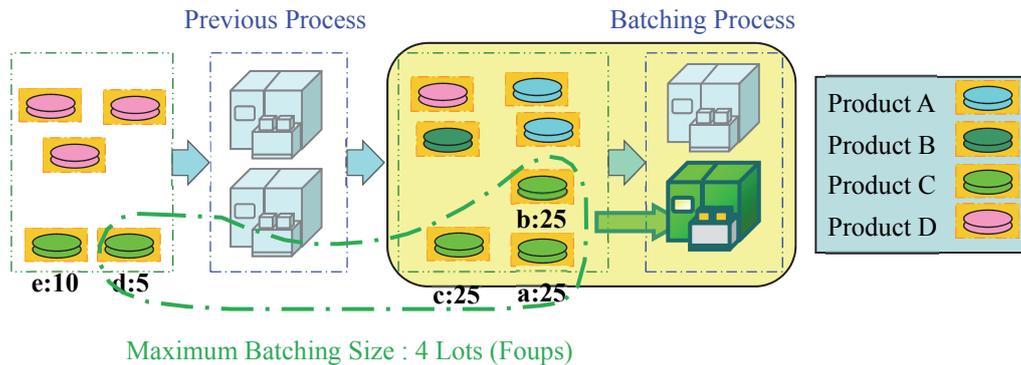
微細化とウェハの大口径化により 1 ウェハ当たりのチップ数が飛躍的に増加したこと、多品種化が進み 1 つの注文 (Purchase Order) の数量が少なくなってきたことにより、1 つの注文に必要なウェハ枚数も 1 枚から 5 枚程度の割合が増加した結果、必要なキャリア数が爆発的に増えた。しかしながら防塵のためにキャリアは特殊な (例えば、窒素でパージされた機密性の高い) 棚に収納しなければならないが、その棚の容量にも物理的な制限がある。また搬送の能力にもおのずと限界があり、生産ライン内のキャリア数を最小に止めるためには、装置や工程の特性に応じて、分割・統合を繰り返す必要がある。高効率生産を達成するためには、どの製品をどのタイミングで分割・統合するべきか、生産単位を臨機応変に変更する必要がある。



**Fig. 2.4 Frequent Split and Merge**

また、半導体前工程の装置には、熱処理など処理時間が数時間と長い工程では、スループット低下と品質バラツキ抑制のために、1 度に 100~150 枚のウェハを同時に処理する装置が存在する。1 枚でも処理は可能であるが、熱分布一様化のため空いているスペースにダミーウェハを装填する準備時間が長くなってしまい更なるスループットの低下を招く。

前述の通り、通常は 1 ロット 25 ウェハを単位として生産しているため、この装置 (工程) の処理前に、Fig. 2.5 に示す通り、バッチ組みと呼ばれる、同じレシピで処理できるロットを揃える作業が発生する。



**Fig. 2.5 Lot Batching**

処理時間とスループットとの関係より、バッチ組みのために処理装置を空けたまま、最後のロットが到着するまでどのくらいの時間待つのが従来からの課題でもあるが、多品種化・小ロット化の流れがこの課題をさらに顕著にしている。

(4) 装置稼働率に関する課題

半導体前工程においては、最先端に微細化が進んだ製品は最新の装置でしか加工できない工程がいくつも存在する。最新の装置は究極に高い加工精度、すなわち高品質を追求するため、スループットは低く、故障やJAMが多い上、加工精度を維持するために高い頻度でメンテナンスが必要となる。前述の通り最新の装置は高価であり最大稼働率での運用が要求されるため、定期メンテナンススケジューリングの最適化のみならず、突発的な装置トラブルをどのようにスケジューリングで考慮するかも重要な課題となる。

(5) 治具に関する課題

リソグラフィ（露光）工程で用いられるレティクルは、微細化に伴い高価格になる上、集積度が高まるに連れ多層化も合わせて進むことで1製品を生産するために20~30種類のレティクルが必要となり、1セットが数千万円から1億円と高コスト化も進んでいるため、製品ライフサイクルの短い近年の製品においては、従来のようにフレキシブルに生産に対応するため複数のレティクルセットを準備するということが困難になっている。リソグラフィ装置にはレティクルを数枚格納可能で、装置選択の場合には該当製品・工程のレティクルがセットされている装置を選ぶ。セットされていない場合、保管庫から搬送する時間が必要なことに加え、どのレティクルを取り外すのか、即ち、最も長く使われないレティクルをスケジューリングと同期して判断しなければならない。

微細化が進んだ近年は、一定期間使用されない（使用前の異物チェックが実施されない）レティクルは洗浄した上で特殊なステーションで別管理しなければならない、製品到着に合わせたレティクルを配膳という更に難しい課題も表面化している。

## 2.2.3 半導体後工程（組立工程）におけるスケジューリングの課題

### (1) 段取り替えに関する課題

半導体前工程においては、Fig. 2.3 に示した通り、生産ライン毎にシリコンウェハの口径は違うものの、同一生産ラインにおいては数百工程あろうとも加工対象物は変化せず、同一のキャリアで搬送される。キャリアおよび、ロードポートと呼ばれる搬送システムと装置加工部との受け渡し部が標準化されていることにより、品種切り替えに伴う装置側加工部の交換作業は発生しない。しかしながら半導体後工程では、Fig. 2.6 に示す通り、外形の大きさ、形、基板に実装するためのピンの数などが異なる品種多数が生産されるため、比較的処理能力の小さな装置は装置台数を揃えることで品種毎に利用装置を固定できるが、処理能力の大きな装置では品種切り替えしながら共用せざるを得ず、その場合には大掛かりな段取り替え作業、すなわち、加工部の交換作業が必要となる。

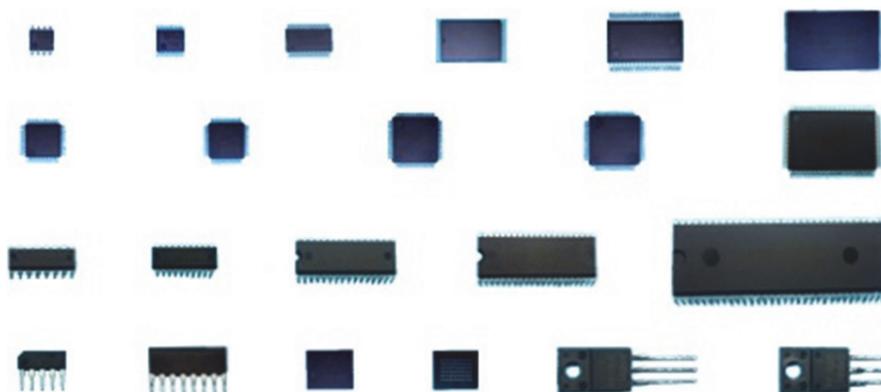


Fig. 2.6 Various Productions

### (2) 治具・金型に関する課題

品種切り替えに関しては、先に示した装置のセットアップ作業（時間）だけではなく、品種に応じた治具や金型を考慮することも必要となる。金型は数が少なく、装置との組合せが限定されるため、スケジューリング上も大きな課題となる。

### (3) 生産単位に関する課題

Fig. 2.7 に半導体後工程のうち組立工程の主な製造工程を示す。半導体前工程を完了したシリコンウェハを IC チップ（Die）に切り離すダイシング工程から始まり、良品チップのみをリードフレームと呼ばれる台の上に固定、チップの電極とリードフレームを金線などで接続、樹脂で封止してメッキ加工、リードを切断し成型した後、表面に型名や製造国などをマーキングして完成となる。

この間、加工対象物が、ウェハ→リードフレーム→製品と移行していくが、それぞれの加工工程における処理時間の差が大きい。例えば、80 ピンの比較的ピン数の多い製品では、

ピン数に関係ないメッキ工程が 1,000 個当たり約 10 分とすれば、ワイヤ接続工程では約 13 時間と、およそ 80 倍もの差となり、これはそのまま、各工程における必要装置台数の差となって現れる。作業時間が短いため装置台数が少ないが、品種切り替えの発生により装置のセットアップ作業が必要な共有装置をいかに効率よく利用するか、その装置に機会損失を発生させないために他の工程（装置）とどのように連携させるのが、重要な課題となる。

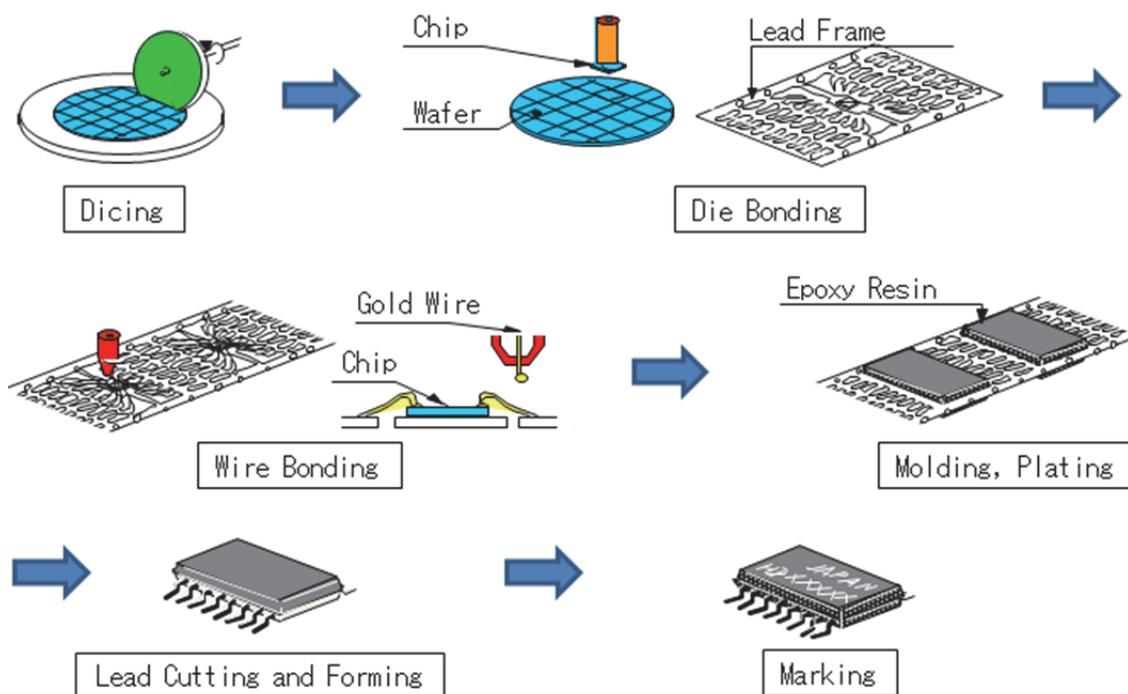


Fig. 2.7 Assembly Process

## 2.2.4 半導体後工程（テスト工程）におけるスケジューリングの課題

### (1) 製造（テスト）工程に関する課題

パッケージ化された IC を選別するテスト工程においては、初期不良を除くため、機能テストを行いながら温度や電圧ストレスの加速試験を行う。この工程は数十時間にも及ぶため、成熟した品種では省略されたり試験時間が短くなったりする。逆に、急に歩留が下がると成熟した品種でも追加することがあり得る。このように、特定の工程におけるテスト結果、すなわち歩留によりテスト工程が動的に変化することが、スケジューリングを困難にしている要因の一つである。

### (2) 製造（テスト）時間に関する課題

メモリ製品においては、読み書きの可否およびスピードなどの単純なテストであったが、マイコンや SoC においてはロジックテストが必要となるため、回路の高集積化と合わせ、

テスト時間も飛躍的に長くなっている。1ロットあたり24時間を超えるテストも少なくない。製品品質が安定してくるに連れ、テスト時間の短縮が行われる。日次計画策定においては問題にならないが、中長期のスケジューリングにおいては考慮が必要となる。

### (3) 生産単位に関する課題

組立工程と違いテスト工程では、ICは既にパッケージ化されており、生産（テスト）対象物の変化はないが、前述のテスト時間の増加および短納期化に対応するため同一テスト工程において、生産単位が複数に分割される場合がある。装置（テスト）の利用状況も考慮した上で、何分割するのが効率的であるのか判断が困難な課題の一つである。

### (4) 段取り替え・治具に関する課題

外形の違いによる品種切り替えに関する課題は、組立工程と同じであるが、テスト工程においては、同じテストを複数の工程、例えば、低温テストと高温テストで利用するため、前工程のリエントラントの課題に加え、工程が違うことによる段取り替え、すなわち、装置のセットアップ時間を考慮しなければならないという課題が追加される。さらに、治具と装置の組合せが限定されるという困難な課題も併せ持つ。

## 2.3 先行研究の現実生産ラインへの適用状況

先行研究において数多の生産性向上が提案され、2.2.2に示した半導体前工程の課題においては、製品品質に関するリソ装置の限定制約[2]や待ち時間制約[3]、リソ工程の治具であるレティクル[2][4]の最適化、段取り替えや装置故障発生時の効率的運用方法[5][6][7]、2.2.3に示した段取り替えの課題に関しては半導体後工程（組立工程）[8][9]および一般的なフローショップ[10]に適用し効果を示しているが、何れもモデルの規模が小さい。また、生産規模拡大のみによる生産効率化を達成するための意思決定支援の提案[11]もあるが、生産ライン全体を効率化するものではない。

このように特定の課題解決のため一部エリアに限定した研究は半導体製造に限定しない一般的な手法を適用することで今後も解決が図られていくであろうが、生産ライン全体を効率化する大規模モデルにおける実用化に関する課題は未だ残されている。

Table 2.1に現実の生産ラインにおける実用化状況を示す。前述の通り、ほとんど唯一実用化されていると言えるのが、ディスパッチングルールである。対象とする装置、作業エリア単位においては、装置の故障や品質問題発生など刻一刻と変化する状況に瞬時に対応できることが最大の理由である。また、実際に処理開始を待っているジョブだけではなく、間もなく上流工程を完了して処理開始待ち行列に加わるであろうジョブも対象にすることで、その状況に応じたディスパッチングルールにより処理順序を並べた結果が、作業員が見ても理解し易い短期的な作業日程計画として有用であるなど、現実の生産ライン運用に適している。

**Table 2.1 Actual operation in semiconductor fabrication**

Methodology	Cases	Remarks
Dispatching rule	Many	• Partial effectiveness for special machines
Mathematical Programming	Few	• Effective in constructing Fab.
Genetic Algorithm(GA)	No	• Difficult to cope with frequent changes
<b>Lagrange Decomposition and Coordination (LDC)</b>	<b>No</b>	• <b>Few previous researches in Semiconductor Manufacturing</b>

数理計画法[12]や GA[13][14]についても幾度と無く適用が試みられているが、モデル規模が大きくなるに連れ計算時間が指数関数的に増加することより、生産ライン建設時など、長期時間掛けて検証できる業務以外への適用は難しく、半導体製造においては、事実上実用化されているとは言えない。

生産ライン全体を最適化するためには、臨機応変なまとめ生産、木目細かなディスパッチング、生産ライン全体を対象とする大規模スケジューリングへの対応など、生産効率の最大化を達成するためには様々なスケジューリング技術を融合させることが不可欠である[15][16]。

#### 2.4 ラグランジュ分解・調整法

ラグランジュ分解・調整法は分散協調型スケジューリング手法の 1 つである。複雑な原問題の制約条件を緩和し目的関数に組み込んだラグランジュ緩和問題を、加法性を利用して部分問題に分解、個々の部分問題単位に分散して最適化を行い、高速に下界値を計算する。原問題の定式化や緩和において、緩和問題を部分問題へ分解可能にする工夫が必要となる。さらに個々の部分問題の最適解の間で協調するよう調整して上界値を計算、すなわち実行可能スケジュールを生成する[17]。本提案手法でも用いる装置能力制約の緩和においては、装置干渉が生じた場合のペナルティを、ある装置においてある時刻にあるジョブを処理する場合に支払う使用料と経済学的な観点から解釈[18]して目的関数に組み込み、このペナルティが最小となるよう最適化する。

生産スケジューリング問題に対して、並列機械問題に導入[19]された後、ジョブショップ問題へと拡張された[20][21]。その後、フローショップモデル[22]、半導体製造に代表されるリエントラントフローショップ[23][24]へ適用されるなど、その高速性、高品質の解が注目を集めている[25]。近年、ロット編成を有するフレキシブル・フローショップ[26] や企業経営レベルでの部門間にわたる広範囲なスケジュール[27] などの大規模で複雑な生産スケジューリング、あるいはメンテナンス計画の最適化[28] などに利用されている。

## 2.5 結言

本章では、対象システムにおけるスケジューリングの課題は数多くあり、生産ラインにおける一部の装置や、一部の工程に特化された最適化の研究は盛んに行われているが、現実の半導体生産ラインが装置約 1,000 台から構成されるという大規模性により、生産ライン全体を効率化する課題が最も困難であることを明確にした。また、その課題を解決するために、大規模かつ複雑なモデルを、高速かつ高品質に解くことが期待されるラグランジュ分解・調整法を適用することを提案した。

次章以降において、現実の半導体生産ラインにおける生産スケジューリング業務への実用化を目的として、まずは半導体前工程の小規模モデルにおいて提案手法の有効性を検証、大規模モデルへ適用した後、最後に現実生産ラインを対象とした生産スケジューリングの最適化について述べる。



### 3 半導体製造における分散協調型スケジューリング手法の一提案

#### 3.1 緒言

本章では、第2章で提案したラグランジュ分解・調整法(LDC)につき、実際の半導体生産ラインを抽象化したモデルに適用し、数値実験により多品種生産における生産ライン負荷、すなわち、ボトルネック装置の稼働率の変化に対する特性を評価し、従来手法であるディスパッチングルールと比較してその有効性を考察した。

このとき、代替装置が増えるに従い解精度が悪化することが確認された先行研究においては、目的関数の一つである治具の最少化を図ることで解精度の悪化を防ぐことができた[2]が、対象システムの規模を拡張し、あらゆる工程において代替装置を増やした場合には問題となることが容易に予見された。

数多くの先行研究において一般的に上界値計算に用いられている、一種のディスパッチングルールであるリストスケジューリングは、その手法が単純で、問題によっては良い解が得られない場合もあることが知られている[17]。しかしながら、単純であるが故に他のあらゆるスケジューリング手法と比較しても非常に高速であるという点が最大のメリットでもあり、実用化という観点からも本手法の利用が望ましいと判断、アルゴリズムを工夫することで解の改善を図り、有効性を確認した。

#### 3.2 対象システム

Fig. 2.2 で示した、同じ装置で加工を行う工程が複数存在する、代替装置が複数ある工程が存在するなど、今後大規模モデルに拡張する際に研究成果を可能な限り継承するために、半導体前工程の特徴を残して抽象化した対象システムを Fig. 3.1 に示す。

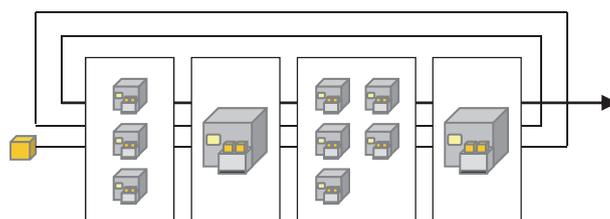


Fig. 3.1 Re-entrant Flow Shop

#### 3.3 ラグランジュ分解・調整法の適用

##### 3.3.1 記号

本論文で使用する記号を以下に示す。

$i$  : ジョブ番号 ( $i=1, 2, \dots, I$ )

$j$	:	工程番号 ( $j=1, 2, \dots, J_i$ )
$m$	:	装置番号 ( $m=1, 2, \dots, M$ )
$t$	:	時刻 ( $t=1, 2, \dots, T$ )
$\delta_{i,m,t}$	:	ジョブ $i$ の装置 $m$ 時刻 $t$ における加工状況
$c_i$	:	ジョブ $i$ の最終工程の完了時刻
$d_i$	:	ジョブ $i$ の納期
$st_{i,j,m}$	:	ジョブ $i$ 工程 $j$ 装置 $m$ における開始時刻
$ed_{i,j,m}$	:	ジョブ $i$ 工程 $j$ 装置 $m$ における終了時刻
$pt_{i,j}$	:	ジョブ $i$ 工程 $j$ の処理時間
$U_{ij}$	:	ジョブ $i$ 工程 $j$ で使用可能な装置の集合

### 3.3.2 原問題の定式化

本章で対象とするのは、 $M$  台の装置で構成されるリエントラントフローショップにおける、 $I$  個のジョブごとの納期遅れ時間の総量を最小化するスケジューリング問題である。納期遅れ時間の総量を最小化する目的関数は、ジョブ  $i$  の納期遅れ量  $D_i$  を用いると式 (3.1) となる。

$$\min \sum_{i=1}^I D_i \quad (3.1)$$

subject to

$$D_i = \max \{c_i - d_i, 0\} \quad (3.2)$$

式 (3.3) で示す、ジョブ  $i$  が装置  $m$  で時刻  $t$  において加工中である場合に 1、さもなければ 0 であるような 0-1 変数である決定変数  $\delta_{i,m,t}$  を用いると、装置能力制約は式 (3.4) となる。

$$\delta_{i,m,t} = \begin{cases} 1 & (m \in U_{i,j}, st_{i,j,m} \leq t \leq ed_{i,j,m}) \\ 0 & otherwise \end{cases} \quad (3.3)$$

$$\sum_{i=1}^I \delta_{i,m,t} \leq 1 \quad (3.4)$$

このとき、決定変数  $\delta_{i,m,t}$  とジョブ  $i$  の処理時間の総和との間に式 (3.5) が成立する。

$$\sum_{m=1}^M \sum_{t=1}^T \delta_{i,m,t} = \sum_{j=1}^{j=J_i} pt_{i,j} \quad (3.5)$$

ジョブの先行関係制約は式 (3.6) となり, ジョブの開始時刻と終了時刻との間には式 (3.7) が成立する.

$$ed_{i,j,m} < st_{i,j+1,m'} \quad (m \in U_{i,j}, m' \in U_{i,j+1}) \quad (3.6)$$

$$ed_{i,j,m} = st_{i,j,m} + pt_{i,j} - 1 \quad (m \in U_{i,j}) \quad (3.7)$$

### 3.3.3 緩和問題の作成

装置能力制約式 (3.4) を緩和し, ラグランジュ乗数  $\lambda_{m,t}$  を用いると緩和問題のラグランジュ関数は式 (3.8) となる.

$$\min L(\lambda) = \sum_{i=1}^I D_i + \sum_{m=1}^M \sum_{t=1}^T \lambda_{m,t} \left( \sum_{i=1}^I \delta_{i,m,t} - 1 \right) \quad (3.8)$$

この式 (3.8) は以下のように展開することができる.

$$\min L(\lambda) = \sum_{i=1}^I \left( D_i + \sum_{m=1}^M \sum_{t=1}^T \lambda_{m,t} \delta_{i,m,t} \right) - \sum_{m=1}^M \sum_{t=1}^T \lambda_{m,t} \quad (3.9)$$

この式 (3.9) は第二項が定数となり, 第一項はジョブごとに加法性を有するため, 以下のジョブごとに分解した部分問題を解けばよいことになる.

$$\min \left( D_i + \sum_{m=1}^M \sum_{t=1}^T \lambda_{m,t} \delta_{i,m,t} \right) \quad (3.10)$$

s.t. (3.2) ~ (3.7)

### 3.3.4 動的計画法による部分問題の解法（上界値計算）

式 (3.10) により定式化されたジョブごとの部分問題の最適化には動的計画法 (Dynamic Programming) を用いて厳密な最適解を用いる. 式 (3.10) の第二項すなわちペナルティの総和に相当する部分を  $R_{i,j,m}$ , ジョブ  $i$  工程  $j$  の開始時刻を  $st_{i,j,m}$ , 終了時刻を  $ed_{i,j,m}$  とすると再帰方程式は以下のように記述できる. このとき  $R^*_{i,j-1,m}$  は,  $j-1$  工程までの最適値である.

$$R_{i,0,m} = 0 \quad (3.11)$$

$$R_{i,j,m} = \min \left( R^*_{i,j-1,m'} + \sum_{t=st_{i,j,m}}^{ed_{i,j,m}} \lambda_{m,t} \delta_{i,m,t} \right) \quad (3.12)$$

まず再帰方程式  $R_{i,j,m}$  を工程 1 から工程  $J_i$  まで順に解き,

$$\min(D_i + R_{i,J_i,m}) \quad (3.13)$$

となるようなスケジュールを求めることで, ジョブ  $i$  の最適スケジュールが求まる.

### 3.3.5 実行可能スケジュールの解法（下界値計算）

部分問題の解を元にリストスケジューリングにより実行可能スケジュールを導出する. 一般的に利用されている最も単純なアルゴリズム (FIFO : 先入先出) を以下に示す.

STEP1 部分問題の解を開始時刻順に並べる.

STEP2 リストの先頭から先行関係制約, 装置能力制約を満たすようにジョブを割当てる.

STEP3 リストが空になるまで STEP2 を繰り返す.

### 3.3.6 双対問題の解法

式 (3.9) に示した緩和問題のラグランジュ双対問題は以下の式となる.

$$\max_{\lambda} \left\{ \sum_{i=1}^I \min_{\delta} \left( D_i + \sum_{m=1}^M \sum_{t=1}^T \lambda_{m,t} \delta_{i,m,t} \right) - \sum_{m=1}^M \sum_{t=1}^T \lambda_{m,t} \right\} \quad (3.14)$$

ラグランジュ双対問題の解法には, 少ない協調回数で打ち切っても十分良い下界値が求まることが様々な問題で報告されている劣勾配法を用いる. アルゴリズムを以下に示す.

STEP1 ラグランジュ乗数を初期化する.

STEP2 協調回数  $v$  回目のラグランジュ乗数  $\lambda_{m,t}$  でラグランジュ緩和問題を解き, 最適な決定変数値を得ることでの下界値 **LB** を得る.

STEP3 現在の解について, 式 (3.15) を用いて緩和した装置能力に対する劣勾配  $S_{m,t}$  を求める.

$$S_{m,t} = \sum_{i=1}^I \delta_{i,m,t} - 1 \quad (3.15)$$

STEP4 式 (3.16) によりステップサイズ  $\alpha$  を求める. このとき  $\beta$  は初期パラメータ, **UB** は上界値の最適値である.

$$\alpha = \frac{\beta(UB - LB)}{\sum_{m=1}^M \sum_{t=1}^T S_{m,t}^2} \quad (3.16)$$

式 (3.17) によりラグランジュ乗数を更新して **STEP2** へ戻る.

$$\lambda_{m,t}^{v+1} = \max\left(0, \lambda_{m,t}^v + \alpha S_{m,t}\right) \quad (3.17)$$

ここで比較検証のため, 解の振動が少ないことで知られる単調非減少法[29]による双対問題の解法も試みる. ラグランジュ乗数の更新幅を示すステップサイズを  $\gamma$  とすると以下の式となる.

$$\lambda_{m,t}^{v+1} = \lambda_{m,t}^v + \max\left\{0, \gamma\left(\sum_{i=1}^I \delta_{i,m,t} - 1\right)\right\} \quad (3.18)$$

式 (3.18) は, 装置干渉が起こっている場合にラグランジュ乗数  $\lambda_{m,t}^v$  の値を増加させ, 装置干渉がない場合は値を変化させないといった更新ルールである.

### 3.3.7 最適化アルゴリズム

ラグランジュ分解・調整法 (LDC) を用いた最適化アルゴリズムを以下および Fig. 3.2 に示す.

STEP1 ラグランジュ乗数を初期化する.

STEP2 ラグランジュ緩和問題を最適に解く. (下界値)

- STEP3 ラグランジュ緩和問題の最適解の情報を参照して，実行可能解を得る．（上界値）  
STEP4 終了条件を満たしていれば終了．満たしていなければラグランジュ乗数を更新してSTEP2に戻る．

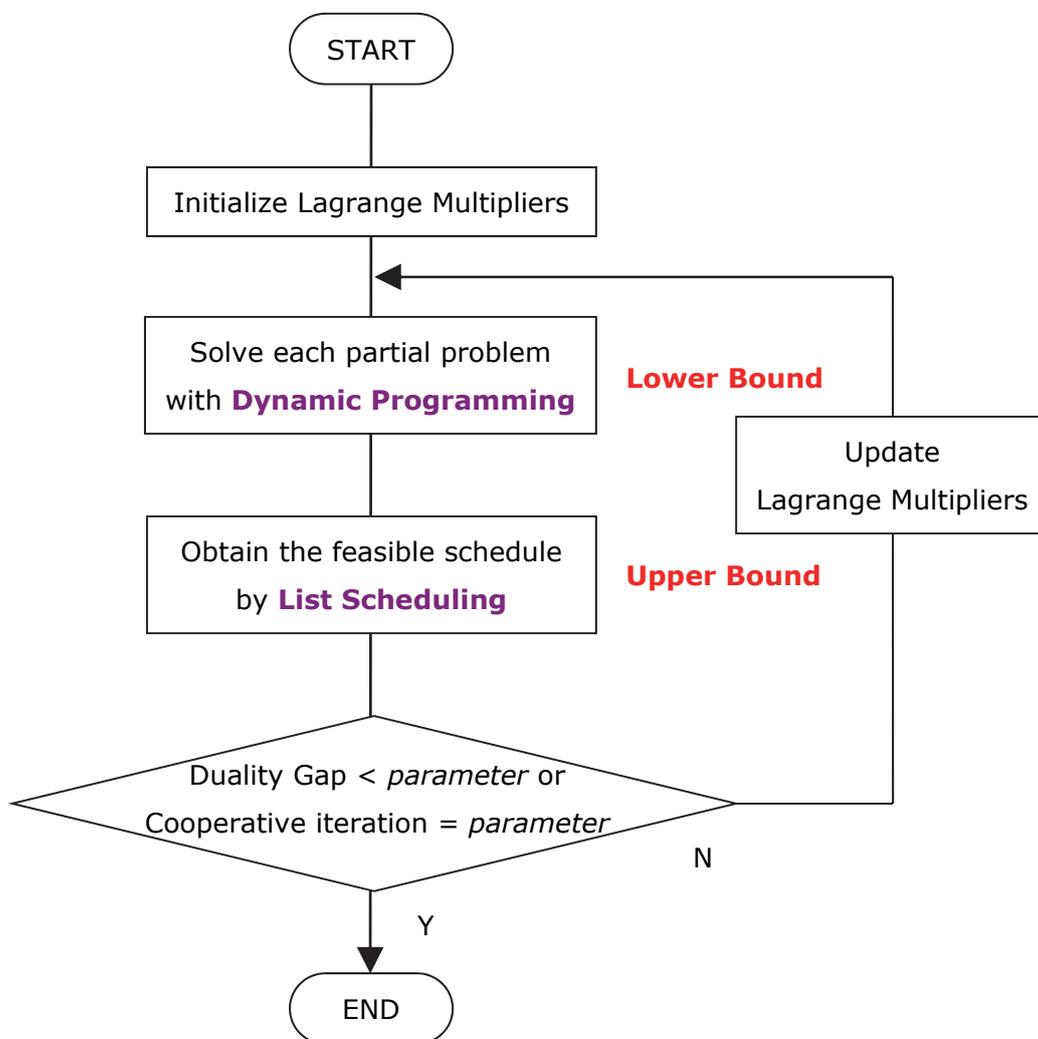


Fig. 3.2 Algorithm with LDC

### 3.4 提案手法（リストスケジューリングのアルゴリズム改善）

#### 3.4.1 最も単純なアルゴリズム

まず、ディスパッチングルールにおける FIFO（先入先出）と同じ考え方による、最も単純なアルゴリズムを Fig. 3.3 に示す. 先行研究においてそのアルゴリズムが詳述されることはないが、おそらく多くの研究においてこのアルゴリズムが用いられていると推察する.

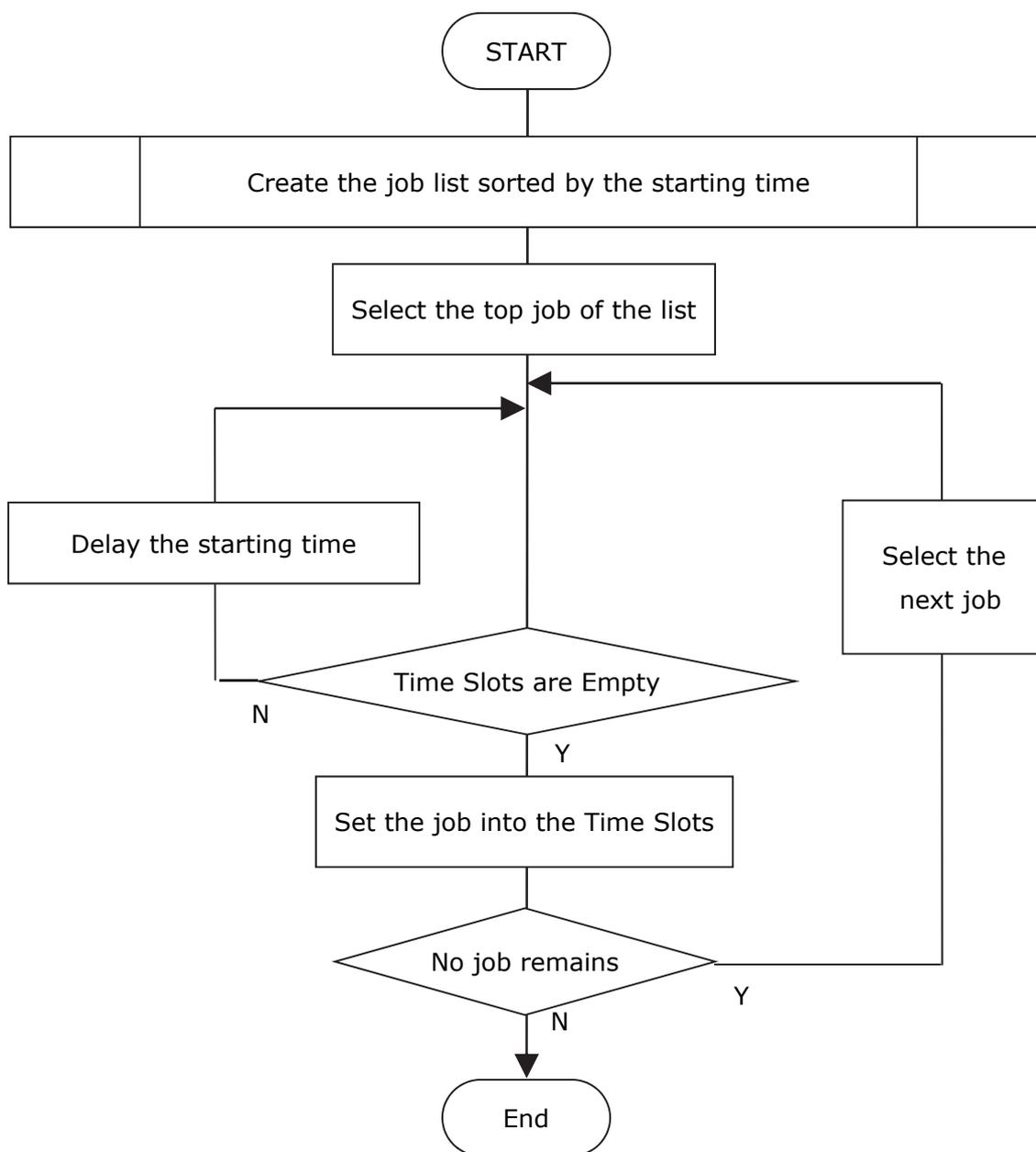


Fig. 3.3 Algorithm of Simple List Scheduling

Fig. 3.4 のガントチャートは、DP により部分問題の解法が完了した時点进行を想定した、治すとスケジューリングのアルゴリズム説明に用いるための例を示す。この時点で装置#2 のタイムスロット  $t=2$  に装置干渉が残っていると仮定した。

	1	2	3	4	5	6	7	8	9	10
#1	C	C								
#2	A	A		D	D					
#3			B							
#4			A	B	C	D				
#5					B	B	B			
#6						C	C	C		
#7				A	A	A	D	D	D	

Fig. 3.4 Original Gantt chart by DP

リストはジョブの各工程開始時刻順に作成する。このケースでは、A の第一工程 (#2) > C の第一工程 (#1) > B の第一工程 (#2) > A の第二工程 (#4) > A の第三工程 (#7) > B の第二工程 (#3) > D の第一工程 (#2) > (以下省略) の順で実行可能スケジュールを作成する。

Fig. 3.3 に示す単純なアルゴリズムにより得られる実行可能スケジュールを Fig. 3.5 に示す。全体的にスケジュールが後倒しになる上に、ボトルネック装置として設定している装置#4 のタイムスロット  $t=4$  に空き (機会損失) が生じている。

	1	2	3	4	5	6	7	8	9	10
#1	C	C								
#2	A	A → B	B	B	D	D				
#3										
#4			A	● → B	C	D				
#5					● → B	B	B	B		
#6						● → C	C	C	C	
#7				A	A	A	● → D	D	D	D

Fig. 3.5 Optimized Gantt chart by Simple Algorithm

### 3.4.2 アルゴリズムの改善

Fig. 3.3 に示す単純なアルゴリズムによる機会損失を無くすため、出来る限り前詰めにスケジューリングするよう改善したアルゴリズムを Fig. 3.6 に示す。

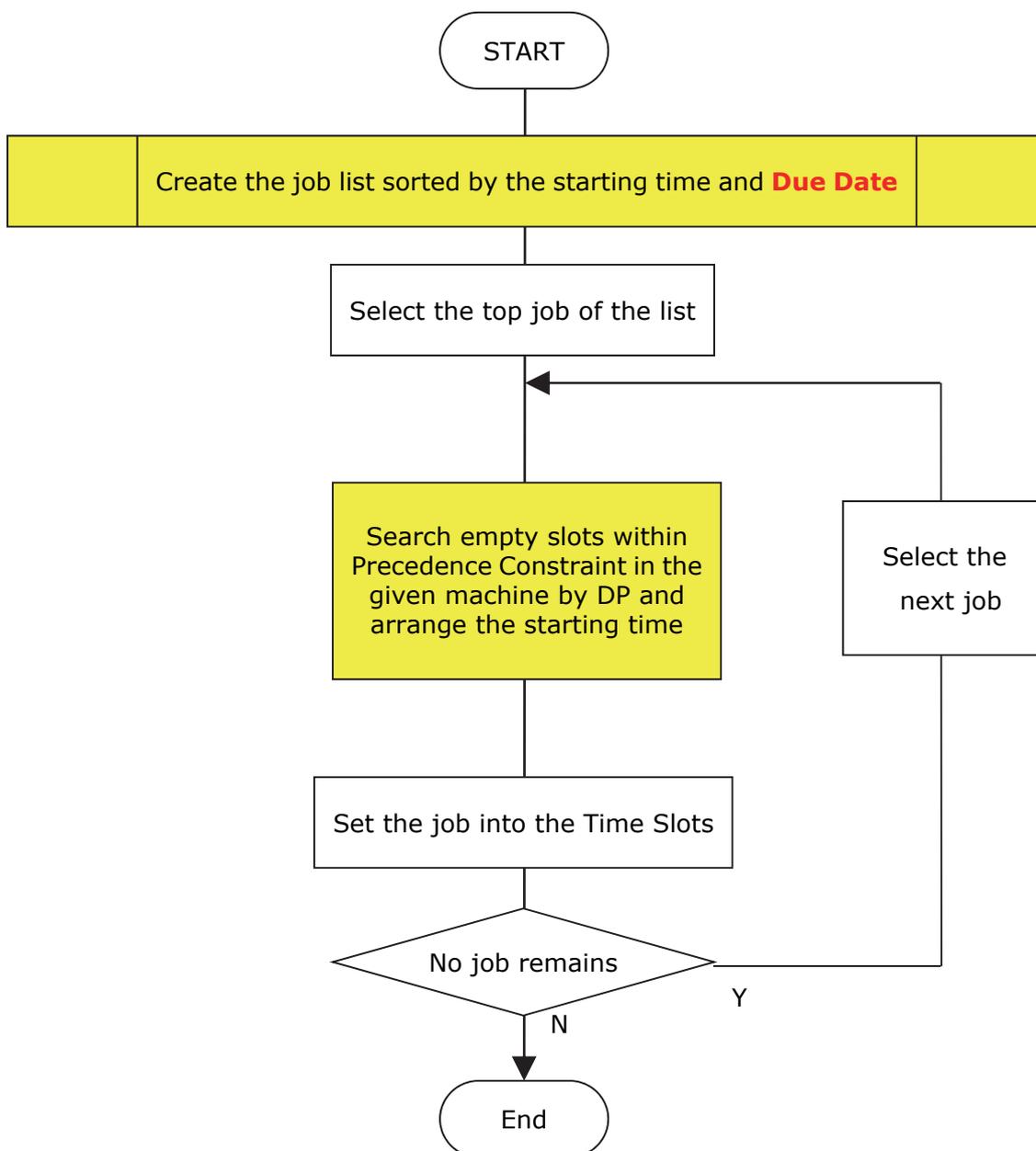


Fig. 3.6 Algorithm of Improved List Scheduling

まずリストを作成する際、ジョブの各工程開始時刻だけではなく、各工程開始時刻が同じ場合は、納期が早いジョブを優先する、すなわち、先に割り付けるように改善した。

次に、装置干渉が発生した場合、該当装置上で、ジョブの先行関係制約下において可能な限り前倒してスケジューリングを実施するよう改善した。すなわち、DPにより解法された部分問題の解のうち、ジョブの各工程開始時刻と割付装置情報を参照し、部分的な入替を可能とした。

Fig. 3.6 に示すアルゴリズムを Fig. 3.4 に示す部分問題の解に適用して得られる実行可能スケジュールを Fig. 3.7 に示す。

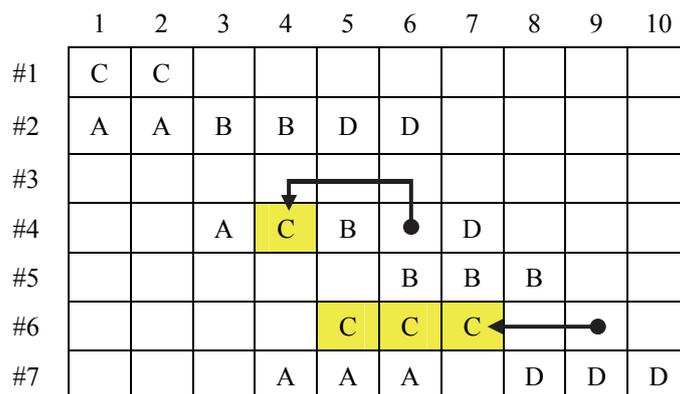


Fig. 3.7 Optimized Gantt chart by Proposal Algorithm

最も単純なアルゴリズムでは空き（機会損失）となった装置#4 のタイムスロット 4 に、ジョブ C の第二工程が前倒し可能となり、ジョブ C の全体スケジュールも前倒しすることが可能となる。

### 3.5 計算機実験

#### 3.5.1 実験条件

##### (1) ジョブタイプ

以下に今回の計算機実験に用いた、Fig. 3.1 に示す半導体前工程を抽象化した基本システムをベースにする 3 つのジョブタイプを示す。太枠は各ジョブが該当工程において使用可能な代替装置を意味し、複数代替装置が存在する場合は、何れか 1 台が使用可能である。

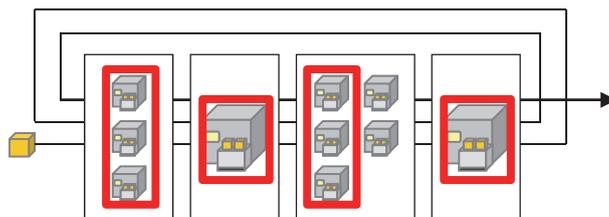
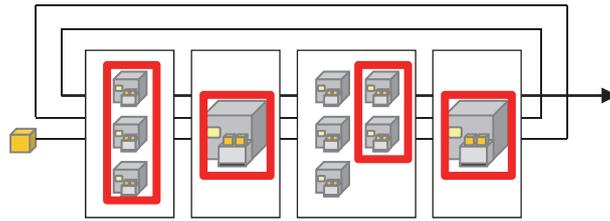
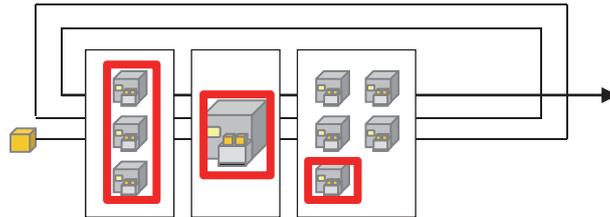


Fig. 3.8 Job Type 1 (Mass Production)



**Fig. 3.9 Job Type 2 (Mass Production)**



**Fig. 3.10 Job Type 3 (Pilot Production)**

Fig. 3.8 に示すジョブタイプ 1 と Fig. 3.9 に示すジョブタイプ 2 は量産品を想定する. 3 番目の工程はかなり高い加工精度が要求される工程を想定し, 各々のジョブタイプにおいて利用可能な装置を限定している. Fig. 3.10 に示すジョブタイプ 3 は, 品質安定化のために実施している 4 番目の工程を省略して他のジョブタイプよりも短 TAT で生産する, 開発が完了したばかりで量産工場における歩留 (良品率) 向上を目的としたとした, パイロット品と呼ばれる量産立上産品を想定した. 通常このような量産立上産品は, 技術的に比較的簡単な工程以外では代替装置が存在しない. 品質すなわち, 歩留が安定し生産量を増やすに  
従い, 装置個別のパラメータを調整する実験を経て, 利用可能装置を増やしていく.

各ジョブタイプの詳細を Table 3.1 に示す.

**Table 3.1 Process Chart of Job Type**

$j$	Job Type 1			Job Type 2			Job Type 3		
	$pt_{i,j}$	$U_{i,j}$		$pt_{i,j}$	$U_{i,j}$		$pt_{i,j}$	$U_{i,j}$	
1	2	1	2 3	2	1 2 3	2	1 2 3		
2	1	4		1	4	1	4		
3	3	5 6 7		3	8 9	3	7		
4	1	10		1	10	2	1 2 3		
5	2	1 2 3		2	1 2 3	1	4		
6	1	4		1	4	3	7		
7	3	5 6 7		3	8 9	2	1 2 3		
8	1	10		1	10	1	4		
9	2	1 2 3		2	1 2 3	3	7		
10	1	4		1	4				
11	3	5 6 7		3	8 9				
12	1	10		1	10				

(2) 納期決定

ジョブを 10 個から 2 個ずつ増やし、それぞれのジョブに対して以下の手順に従い納期をランダムに決定する.

STEP1 与えられたジョブに対し、ジョブタイプを一定の比率で決定する.

STEP2 予備実験の結果により決定したスケジューリング期間に、タイムスロット 24 毎に均等にインプットするよう開始時刻  $st_{i,l,m}$  を決定する. これは,

STEP3 各ジョブの納期  $d_i$  を式 (3.19) を用いてランダムに決定する.

$$d_i = st_{i,l,m} + (1.2 \sim 2.0) \times \sum_{j=1}^{J_i} pt_{i,j} \quad (3.19)$$

これは、現実の生産ラインにおいて 1 日 1 回生産ロットがほぼ同じ比率で生産開始されることを踏まえたものである.

また、式 (3.19) にていて、処理時間の総和、すなわち理論 TAT の 1.2 倍から 2.0 倍の時間を開始時刻に加算して納期を設定しているのは、Hot Lot と呼ばれる特急進捗ジョブは平均的に理論 TAT の 1.2 倍、Normal Lot と呼ばれる一般進捗ジョブは平均的に理論 TAT の 2.0 倍で生産されるという、実際の生産ラインの現状に基づいている.

ジョブ 16 個の場合の一例を Table 3.2 に示す.

Table 3.2 Job Sample ( $I=16$ )

$I$	Type	$st_{i,l,m}$	$d_i$
1	1	1	41
2	1	1	33
3	1	1	34
4	1	1	33
5	1	25	51
6	1	25	51
7	1	25	49
8	1	25	53
9	2	1	26
10	2	1	25
11	2	1	33
12	2	25	55
13	2	25	58
14	2	25	59
15	3	1	30
16	3	25	50

(3) その他の実験パラメータ

予備実験の結果などにより決定した，ラグランジュ乗数更新による協調計算の終了条件などその他の実験パラメータを Table 3.3 に示す．

**Table 3.3 Experimental Parameters**

Parameter	Value
Job Type Ratio	4 : 3 : 1
Scheduling Period (Time Slot)	72
Cooperative iteration number $\nu$	10,000
Duality Gap	0.001

(4) 試行回数

1つの実験条件に対し，ジョブのモデルを 100 パターン作成してそれぞれ実験（最適化計算）する．実験結果は 100 回の平均値を用いる．

### 3.5.2 実験シナリオ

本章における実験シナリオを Table 3.4 に示す．

**Table 3.4 Experimental Scenarios**

No.	Purpose	Improved Method	
		Dynamic Programming	List Scheduling
I	Simple case without any improvements		
II	Verify the DP improvement	√	
III	Verify the List Scheduling improvement		√
IV	Verify details with arranging experimental parameters		√

まず，実験 I において，最も単純なアルゴリズムにより実験する．代替装置が増えることへの対策を施さないため，従来手法のディスパッチングルールより解精度が悪いことが予想される．次に LDC における最適化プロセスの主要部となる下界値計算部の DP の改善（実験 II），上界値計算のリストスケジューリングの改善（実験 III）を試みる．従来手法より有効であることが確認できた手法により，各種実験パラメータを変化させた場合の詳細な検証を実験 IV として最後に実施する．

### 3.5.3 予備実験

本モデルにおける生産ライン負荷の定常状態を確認するため、従来手法であるディスパッチングルールを用いて予備実験を実施した。ジョブ数に対する生産ライン負荷、すなわち、ボトルネック装置と設定した4号機の稼働率は、Fig. 3.11に示す通り、ジョブが16個を超えた時点で100%に達し、それ以降は負荷がオーバすることを確認した。

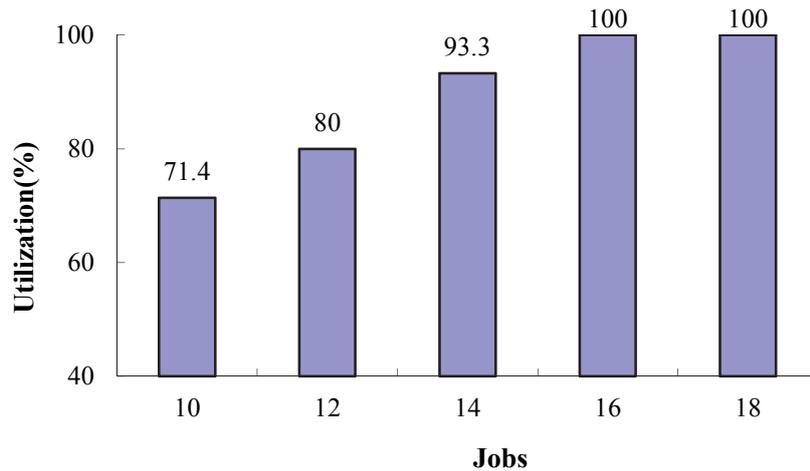


Fig. 3.11 Utilization of Bottle-Neck Machine (#4)

また Fig. 3.12 に示す通り、本問題の目的関数である総納期遅れ量も FIFO (First-In First-Out), EDD (Earliest Due Date) の何れのディスパッチングルールにおいてもジョブが16個を超えた時点で顕著になっており、対象モデルがジョブ16個程度で定常状態になることを確認した。

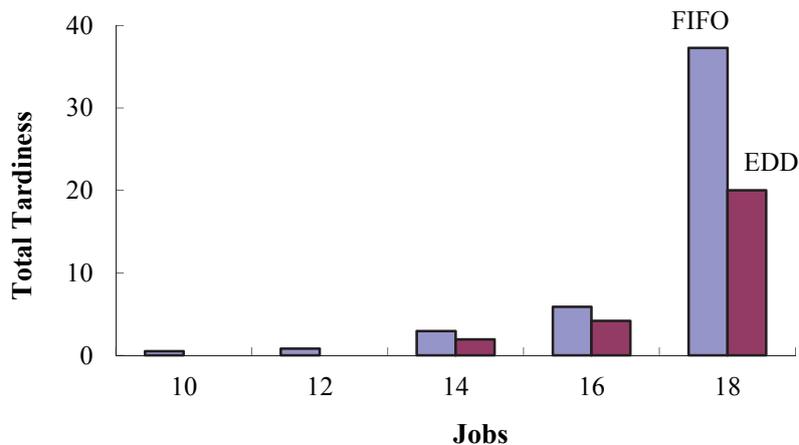


Fig. 3.12 Total Tardiness By Dispatching rule

### 3.5.4 実験 I (単純アルゴリズム)

まず、予備実験で確認したジョブ 16 個の定常状態において、実験 II 以降の改善手法により比較検証するための、Fig. 3.3 に示す単純なアルゴリズムによる実験を行った。

#### (1) 総納期遅れ量 (目的関数)

目的関数である総納期遅れ量を Table 3.5 に示す。

**Table 3.5 Experimental Results I (Total Tardiness)**

Proposed Method I ( $\gamma=0.01$ )		Dispatching rule (FIFO)		Dispatching rule (EDD)	
Average	S.D.	Average	S.D.	Average	S.D.
99.89	17.27	5.90	3.06	<b>4.19</b>	<b>2.42</b>

事前に想定したとおり、従来手法であるディスパッチングルールの FIFO および EDD と比較して、目的関数である総納期遅れ量は、15 倍以上と非常に悪い結果となった。

#### (2) 計算時間

実用化のための重要な指標の一つである計算時間を Table 3.6 に示す。従来方式と比較すれば約 1,000 倍の時間を要してはいるが、4.1372 秒は、Table 2.1 に示したほぼ全ての業務において十分実用的であると言える。

**Table 3.6 Experimental Results I (CPU Time(sec))**

Proposed Method I ( $\gamma=0.01$ )		Dispatching rule (FIFO)		Dispatching rule (EDD)	
Average	S.D.	Average	S.D.	Average	S.D.
4.1372	0.4634	<b>0.0004</b>	<b>0.0008</b>	0.0005	0.0010

次に、提案手法における計算時間を、Fig. 3.2 に示す最適化のアルゴリズムの主要部分別に分析したものを Table 3.7 に示す。計算時間の実に 97.74%が部分問題解法の DP に費やされていることを確認した。

**Table 3.7 Experimental Results I (CPU Detail Time(sec))**

Dynamic Programming	List Scheduling	Update Lagrange Multipliers	Others	Total
<b>4.0437</b> <b>(97.74%)</b>	0.0562 (1.36%)	0.0045 (0.11%)	0.0328 (0.79%)	4.1372

(3) 100回の試行結果の詳細分析

さらに分析を進めるため、100回の試行中の特徴的な Table 3.8 に示す 3つのケースについて分析した。

**Table 3.8 Experimentl Results I in each trial**

#	Feature	Total Tardiness	Iteration to obtain the best schedule	Iteration in completing
33	Best Tardiness	41	574	1,493
73	Largest Iteration to obtain the best schedule	94	1,895	2,007
92	Worst Tardiness	158	50	1,884

それぞれの試行におけるジョブモデル Table 3.9 に示す。違いは納期  $d_i$ のみである。

**Table 3.9 Job Model I in each trial**

$I$	Type	$st_{i,l,m}$	$d_i$		
			#33	#73	#92
1	1	1	34	35	41
2	1	1	40	36	33
3	1	1	33	33	34
4	1	1	34	40	33
5	1	25	64	53	51
6	1	25	62	60	51
7	1	25	62	61	49
8	1	25	59	57	53
9	2	1	36	38	26
10	2	1	41	28	25
11	2	1	40	30	33
12	2	25	54	54	55
13	2	25	63	59	58
14	2	25	59	52	59
15	3	1	30	25	30
16	3	25	56	45	50
Max			64	61	59
Average			47.9	44.1	42.6
Min			30	25	25

i) ベストケース (33 回目の試行) の分析

先ずベストケースにおける双対ギャップの推移は Fig. 3.13 に示す通り、協調回数を重ねる毎に小さくなっており、最適化計算は正しく機能していると考えられる。総納期遅れ量は Fig. 3.14 に示す通り、協調回数 574 回目の時に最小の 33 となっている。

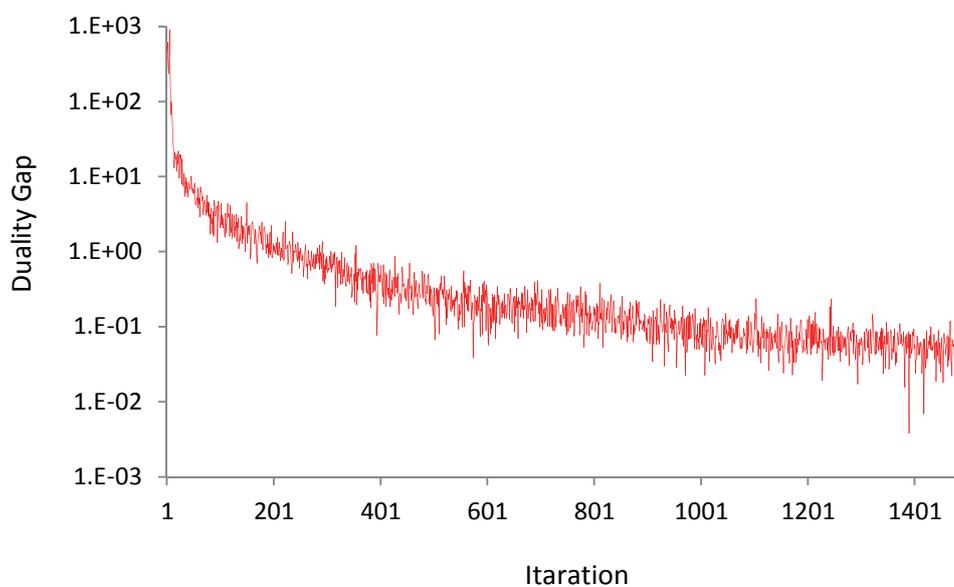


Fig. 3.13 Duality Gap I by Iteration in the 33<sup>rd</sup> trial

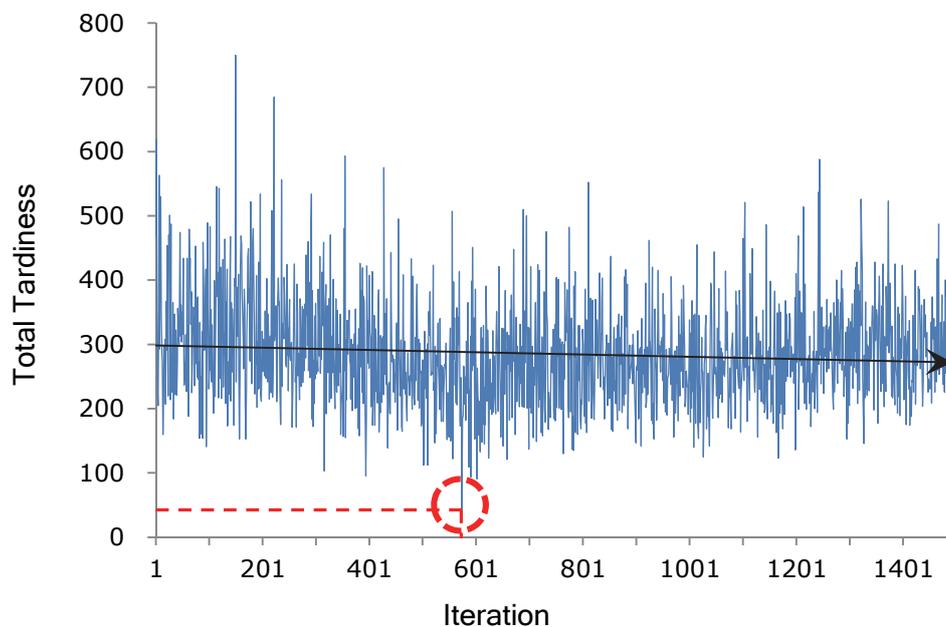


Fig. 3.14 Total Tardiness I by Iteration in the 33<sup>rd</sup> trial

ii) 最適解算出時の協調回数が最大ケース（73回目の試行）の分析

次に最適スケジュール算出時の協調回数が最も多いケースにおける双対ギャップの推移は Fig. 3.15 に示す通り、これも協調回数を重ねる毎に小さくなっており、最適化計算は機能していると考えられる。総納期遅れ量は Fig. 3.16 に示す通り、協調回数 1,895 回目の時に最小の 94 となった。ベストケースと比較して、最適解算出時の協調回数の大きさ以外に際立った特徴は見られない。

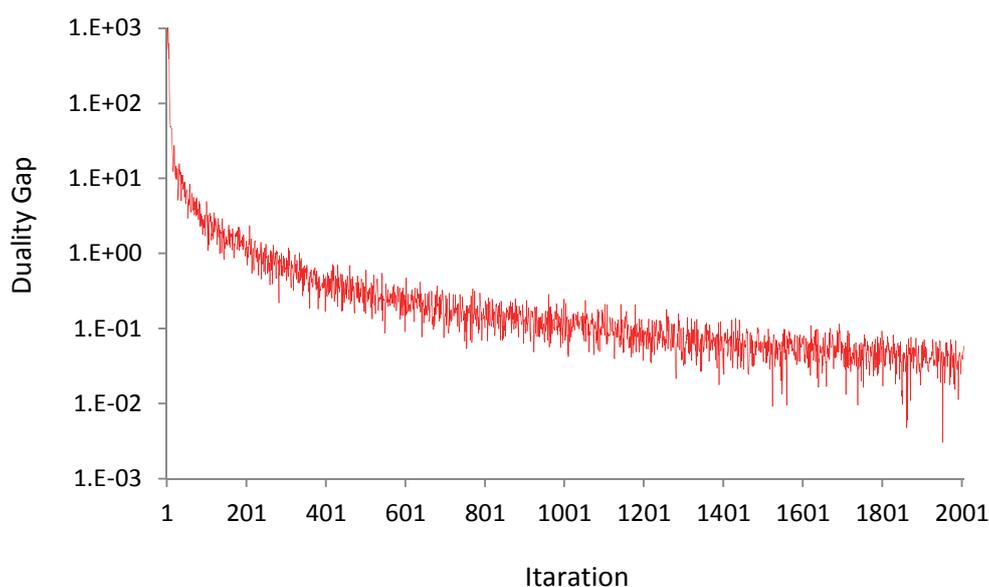


Fig. 3.15 Duality Gap I by Iteration in the 73<sup>rd</sup> trial

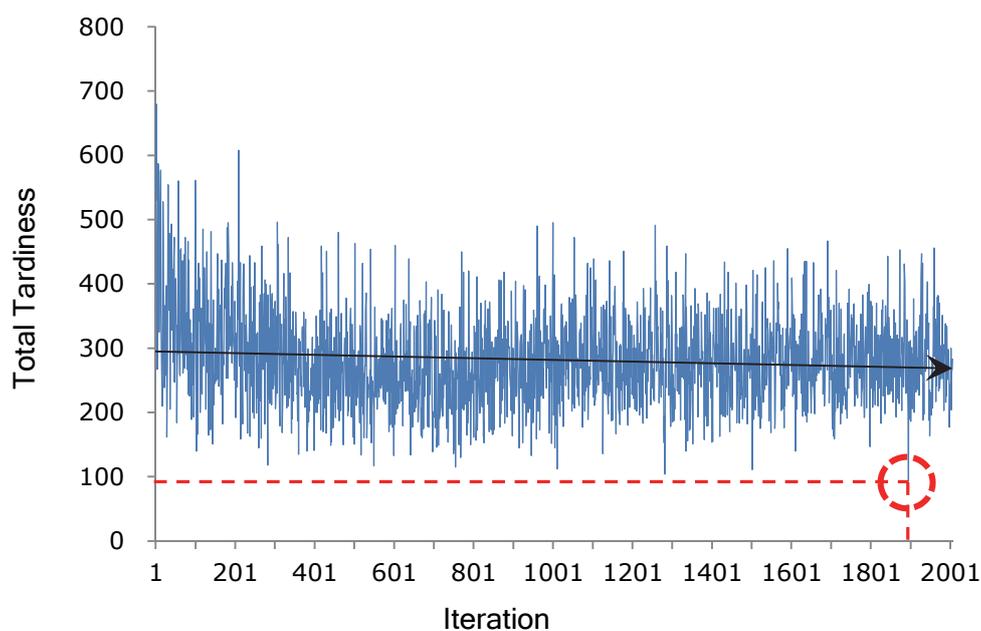


Fig. 3.16 Total Tardiness I by Iteration in the 73<sup>rd</sup> trial

iii) ワーストケース (92 回目の試行) の詳細分析

ワーストケースにおける双対ギャップの推移は Fig. 3.17 に示す通り、最適化計算は機能しているが、総納期遅れ量は Fig. 3.18 に示す通り、協調回数 50 回目の時に最小の 158 となり、双対ギャップが終了条件の 0.001 を下回るまで、上昇傾向となっている。他のケースにおいても、最適解算出の後は上昇傾向が見られており、最適化のアルゴリズムの問題では無く、最適化の終了条件の問題であると考えられる。

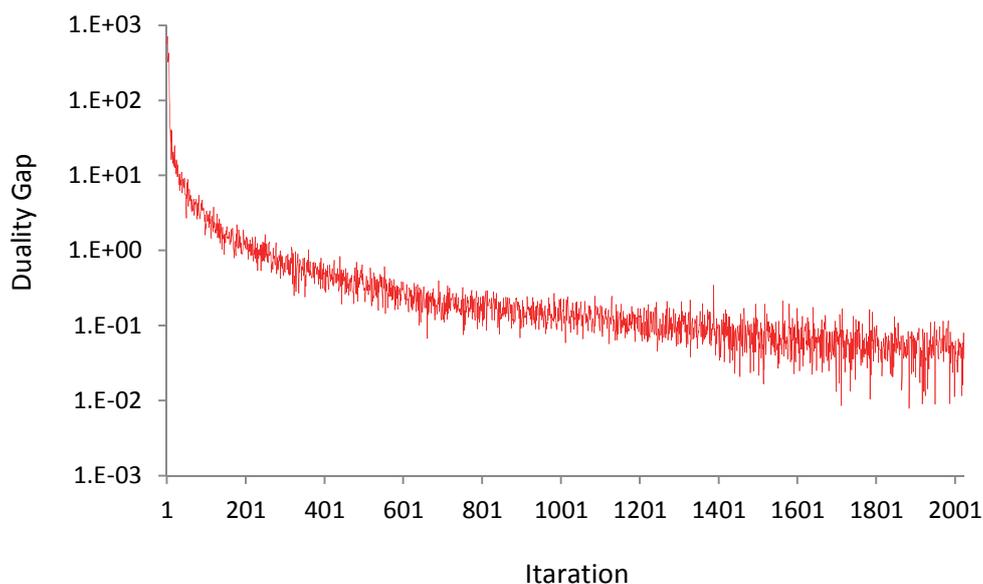


Fig. 3.17 Duality Gap I by Iteration in the 92<sup>nd</sup> trial

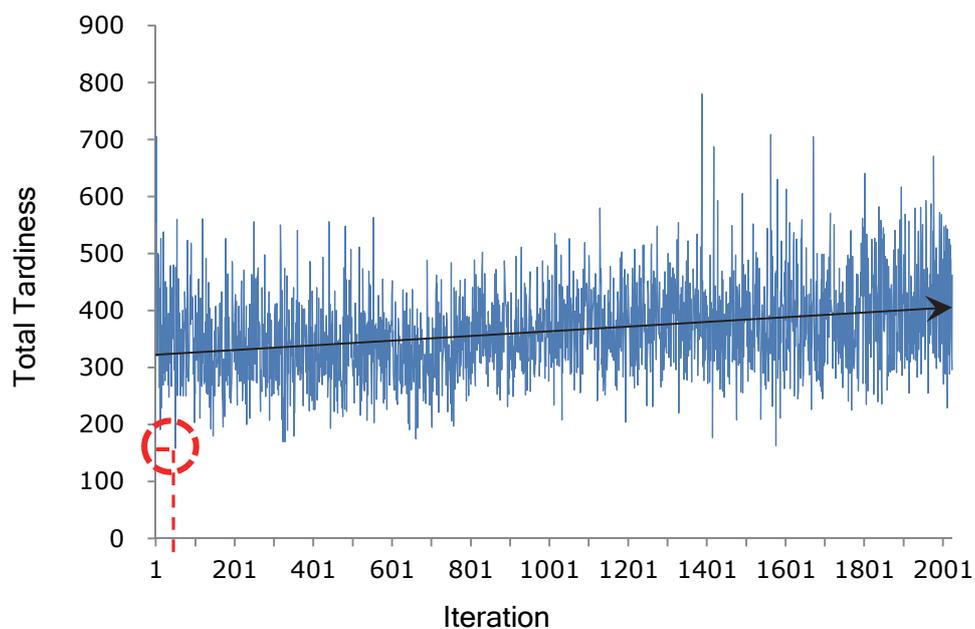


Fig. 3.18 Total Tardiness I by Iteration in the 92<sup>nd</sup> trial



ことが一目で判断できる。

ここで、DPによる最適化のプロセスについても分析した。Fig. 3.20に、横軸が $t(1, \dots, T)$ 、縦軸に各ジョブの開始工程含む3工程全てにおいて利用可能な装置1号機(#1)～3号機(#3)に着目、各協調回数における部分開放終了直後のタイムスロットの様子を示す。セル内の数字は、各タイムスロットに割り付いているジョブ数を示す。協調回数2回目以降のハッチングは、前回の計算において装置干渉が発生し、ペナルティが発生、すなわちラグランジュ乗数の更新(ステップサイズ分加算)されていることを示す。太枠で囲っているのは、開始時刻1に投入された各ジョブの先頭工程(処理時間2)の8個のジョブが、協調回数を重ねる毎にどのように割付が推移しているかを示したものである。

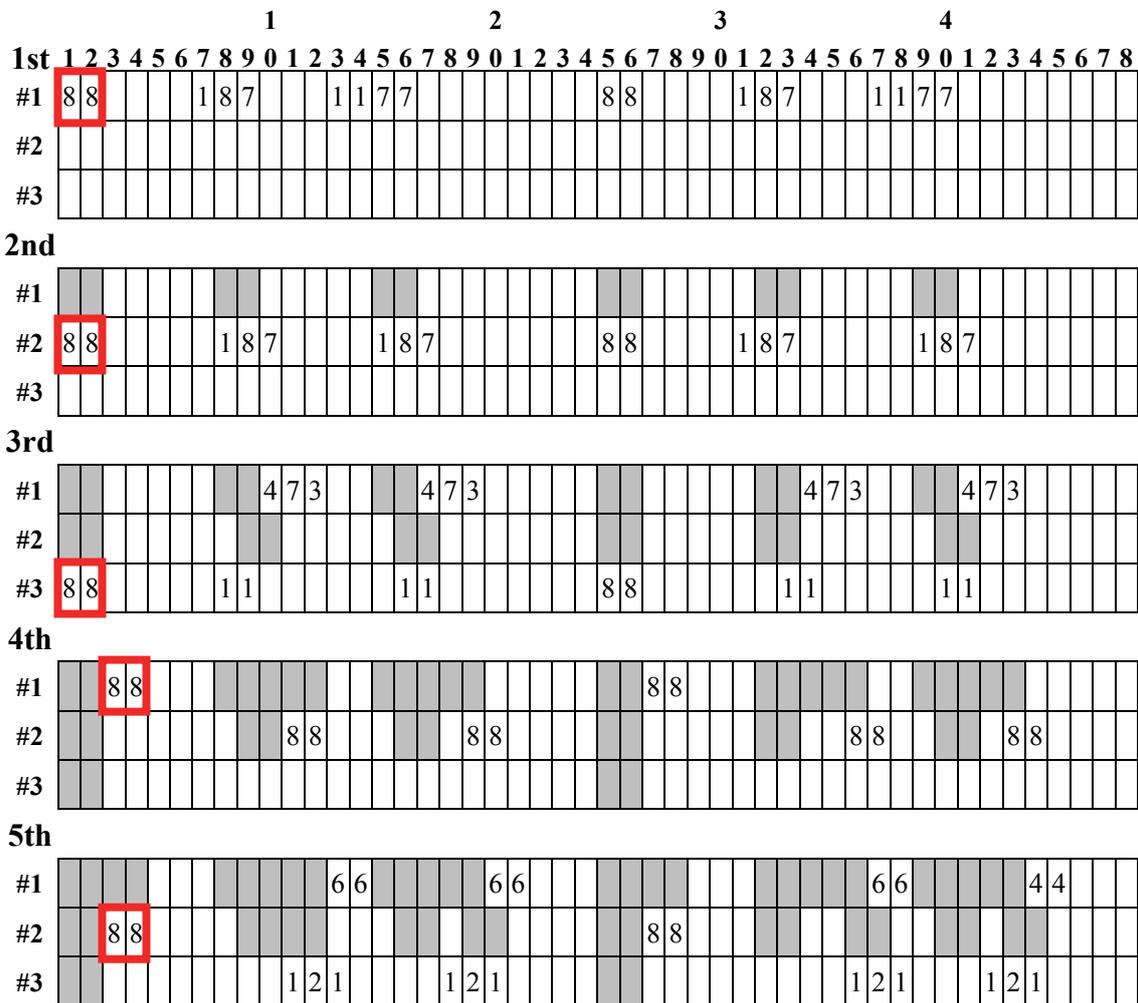


Fig. 3.20 Optimizing process I with DP in the 92<sup>nd</sup> trial

ペナルティであるラグランジュ乗数の小さい(同じ場合は、装置番号の小さい)装置に割り付いているが、代替装置を有効に活用できていないことが確認できる。

### 3.5.5 実験 II (下界値計算のアルゴリズム改善)

ここで解の改善を図るため、DP による部分問題解法時により効率よく代替装置を利用するよう、装置割付けの際、ペナルティであるラグランジュ乗数が同じ値であれば、既に割り付いているジョブが少ない装置を選択するようアルゴリズムを工夫した実験を行った。

#### (1) 総納期遅れ量 (目的関数)

総納期遅れ量は Table 3.10 に示す通り、先の実験 I の結果と比較して 99.89 から 73.27 と約 27%の改善が見られたが、従来手法である FIFO の 5.90 と比較すると 12 倍も悪い結果であり、大きな成果は得られなかった。

**Table 3.10 Experimental Results II (Total Tardiness)**

Proposed Method I ( $\gamma=0.01$ )		Proposed Method II ( $\gamma=0.01$ )		Dispatching rule (FIFO)	
Average	S.D.	Average	S.D.	Average	S.D.
99.89	17.27	73.27	15.55	<b>5.90</b>	<b>3.06</b>

#### (2) 計算時間

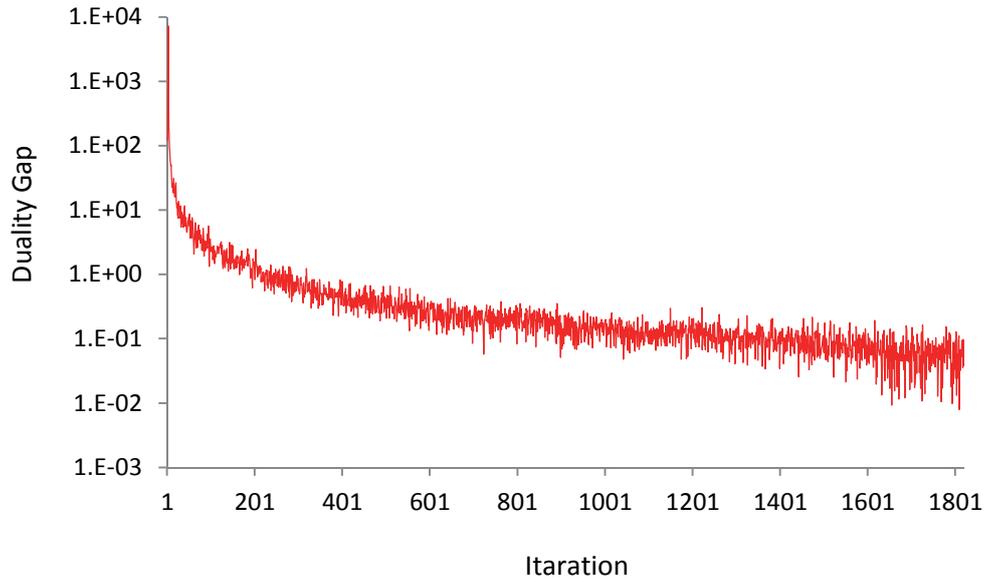
一方計算時間は、Table 3.11 に示す通り、4.137 秒から 5.3325 秒と約 1.29 倍となった。これは、アルゴリズムが複雑化したためだと考えられる。解が 27%改善されたとは言え、従来手法より 12 倍も悪い結果である上、計算時間が 1.29 倍になることを考えれば、本アプローチによる解の改善は相応しくないと考えられる。

**Table 3.11 Experimental Results II (CPU Time(sec))**

Proposed Method I ( $\gamma=0.01$ )		Proposed Method II ( $\gamma=0.01$ )		Dispatching rule (FIFO)	
Average	S.D.	Average	S.D.	Average	S.D.
4.1372	0.4634	5.3325	0.7577	<b>0.0004</b>	<b>0.0008</b>

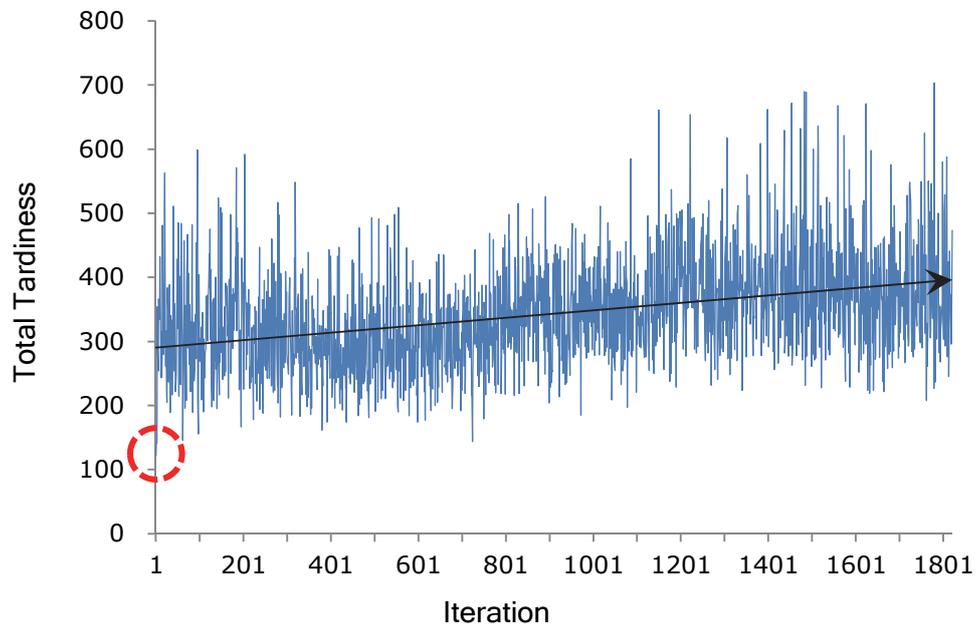
#### (3) ワーストケース (92 回目の試行) の詳細分析

実験 I と同様に、100 回の試行中のワーストケースである総納期遅れ量が 122 の 92 回目に試行したモデルにおける実験結果についてさらに分析した。双対ギャップは Fig. 3.21 に示す通り、協調回数を重ねる毎に小さくなっており、最適化計算は機能していると考えられるが、Fig. 3.17 と比較してバラツキすなわち、解の振動が大きい。



**Fig. 3.21 Duality Gap II by Iteration in the 92<sup>nd</sup> trial**

次に、総納期遅れ量を Fig. 3.22 に示す。本実験においては、協調回数 1 回目の時に納期遅れ量は最小の 122 となり、協調回数を重ねる毎に上昇傾向となる。



**Fig. 3.22 Total Tardiness II by Iteration in the 92<sup>nd</sup> trial**



次に、第 92 回目の試行において得られた最も総納期遅れ量の小さい実行可能スケジュールを Fig. 3.24 に示す。

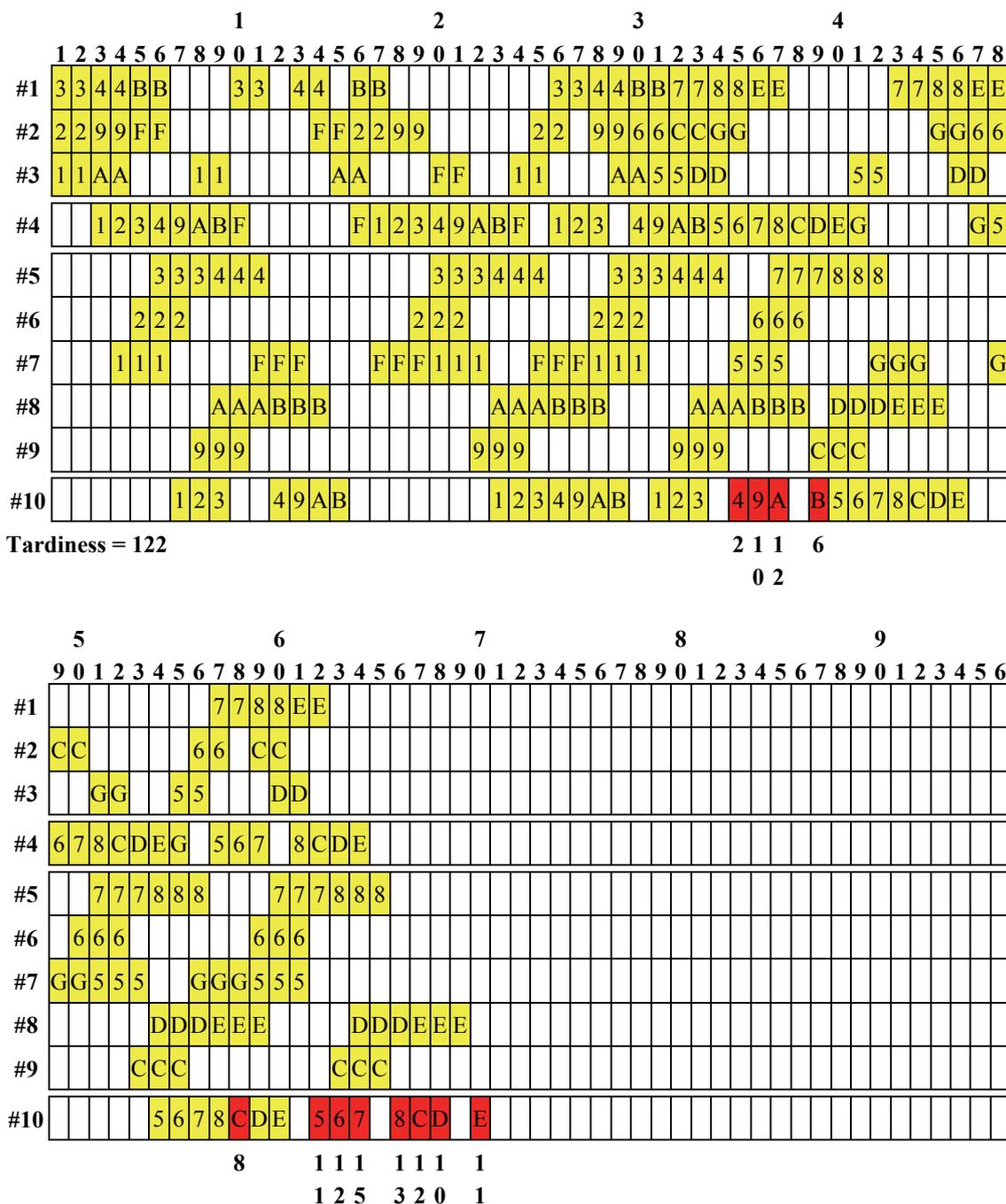


Fig. 3.24 Gantt Chart of the feasible schedule II in the 92<sup>nd</sup> trial

Fig. 3.19 と比較して全体的に前倒しにはなっているが、ボトルネック装置として設定している装置#4 においてタイムスロットの空きが 14/62 個、稼働率 77.4%，すなわち機会損失もまだまだ多く、まだまだ最適化されているとは言えない状態である。

### 3.5.6 実験 III（上界値計算のアルゴリズム改善）

実験 I の単純なアルゴリズムでは、Fig. 3.4 に示すような装置干渉が発生した場合、Fig. 3.5 に示す通り該当装置上で後倒して最も早い空きスロットを探索していたが、同じく該当装置上で、Fig. 3.6 に示す通りジョブの先行関係制約の下に可能な限り前方の空きスロットも探索するよう改善した。

#### (1) 総納期遅れ量（目的関数）

総納期遅れ量は Table 3.12 に示す通り、実験 II の結果と比較して、99.89 から 0.79 へ約 99% の大幅な改善ができ、従来手法の FIFO よりも良い解を得ることができるに至った。

**Table 3.12 Experimental Results III (Total Tardiness)**

Proposed Method I ( $\gamma=0.01$ )		Proposed Method III ( $\gamma=0.01$ )		Dispatching rule (FIFO)	
Average	S.D.	Average	S.D.	Average	S.D.
99.89	17.27	<b>0.79</b>	<b>1.23</b>	5.90	3.06

#### (2) 計算時間

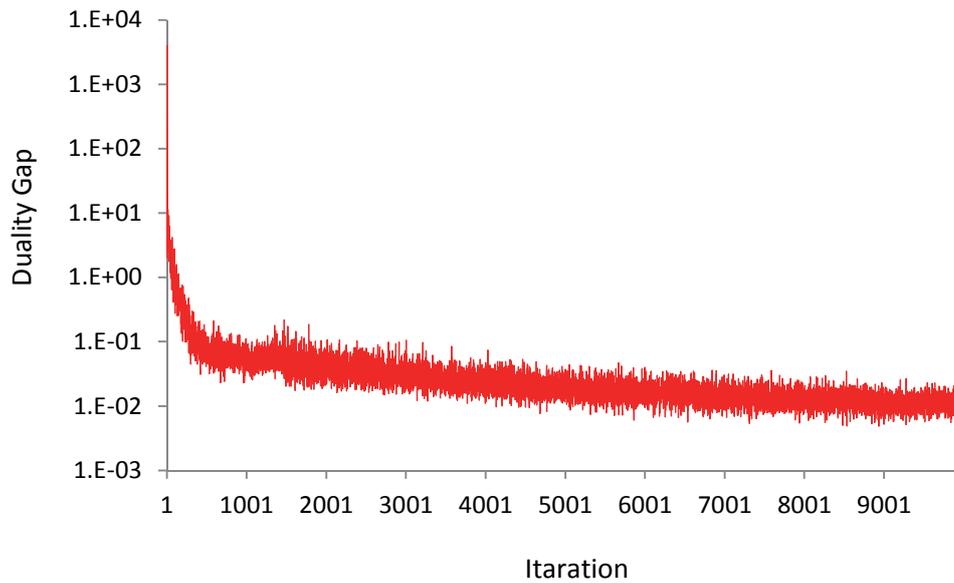
計算時間も Table 3.13 に示す通り、実験 II の結果と比較して、4.1372 秒から 4.9387 秒へと 1.19 倍となったが、実験 II における 1.29 倍と比較すれば約 7.4% 改善できている。バラツキが大きいのは、総納期遅れ量 0 の解が増えたこと、すなわち、その時点で最適化計算を打ち切ったケースが増えたことに起因する。

**Table 3.13 Experimental Results III (CPU Time(sec))**

Proposed Method I ( $\gamma=0.01$ )		Proposed Method III ( $\gamma=0.01$ )		Dispatching rule (FIFO)	
Average	S.D.	Average	S.D.	Average	S.D.
4.1372	0.4634	4.9387	7.2227	<b>0.0004</b>	<b>0.0008</b>

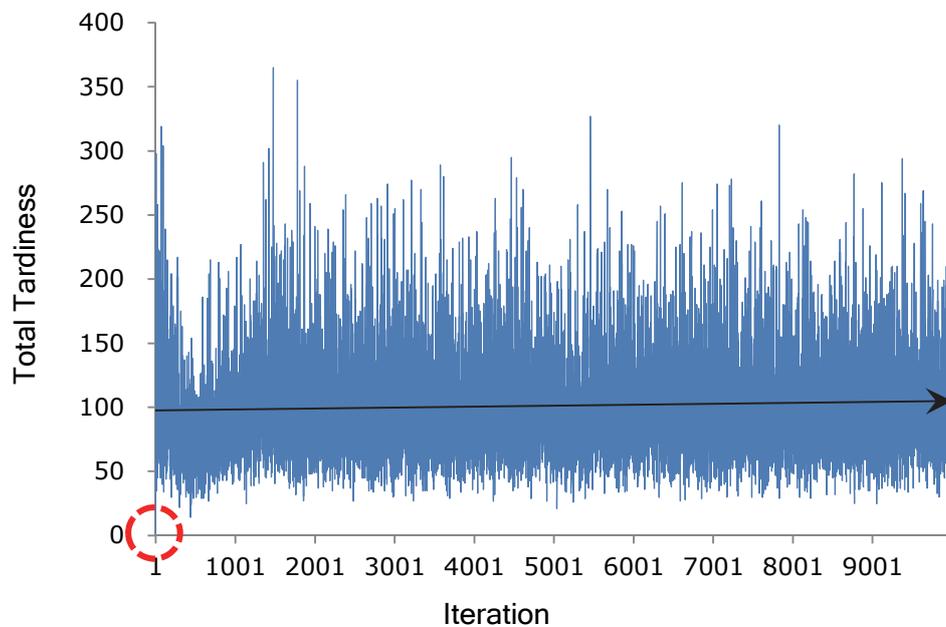
#### (3) 92 回目の試行結果の詳細分析

本実験におけるワーストケースは総納期遅れ量 14 の 42 回目の試行であったが、実験 I および実験 II の結果と比較するため、総納期遅れ量 2 であった 92 回目に試行したモデルにおける実験結果についてさらに分析した。Fig. 3.25 に双対ギャップの推移を示す。本モデルにおいては協調回数の規定数である 10,000 回に達する前に双対ギャップが規定の 0.01 以下になる、すなわち収束することはなかったが、バラツキながらも小さくなる傾向は示しており、最適化計算は問題なく行われていたと考えられる。



**Fig. 3.25 Duality Gap III by Iteration in the 92<sup>nd</sup> trial**

次に、Fig. 3.26 に総納期遅れ量の推移を示す。本実験においても協調回数 1 回目の時に総納期遅れ量は最小の 2 となった後は、以降協調回数を重ねる毎に大きくばらつきながら、若干の上昇傾向を示している。これはやはり、3.5.5 で行った部分問題解法のアルゴリズム見直しにおいて、最初から装置を効率よく利用するよう改善した影響であると考えられる。



**Fig. 3.26 Total Tardiness III by Iteration in the 92<sup>nd</sup> trial**



### 3.5.7 実験 IV (LDC の詳細特性確認)

これまでの3つの実験により、提案手法であるLDCが従来手法のディスパッチングルールと比較して、その最適化のアルゴリズム、特に、実行可能スケジュールを導出するリストスケジューリングのアルゴリズムを改善することでより良い解を算出することが可能であることが確認できた。ここでは更に、一般的にLDCが持つ特性を、半導体前工程のモデルにおいて、定常状態以外の様々な生産負荷の下でも確認できるかどうかを検証する。予備実験の結果により今回の終了条件とした協調回数10,000回以内で収束する範囲で実験パラメータ $\beta$ および $\gamma$ を決定した。また、ジョブが18個以上のケースにおいては、明らかに過負荷の状態となりほとんど収束しなかったため、考察の対象外とした。

#### (1) 総納期遅れ量 (目的関数)

各生産ライン負荷において目的関数である総納期遅れ量を Table 3.14 に示す。

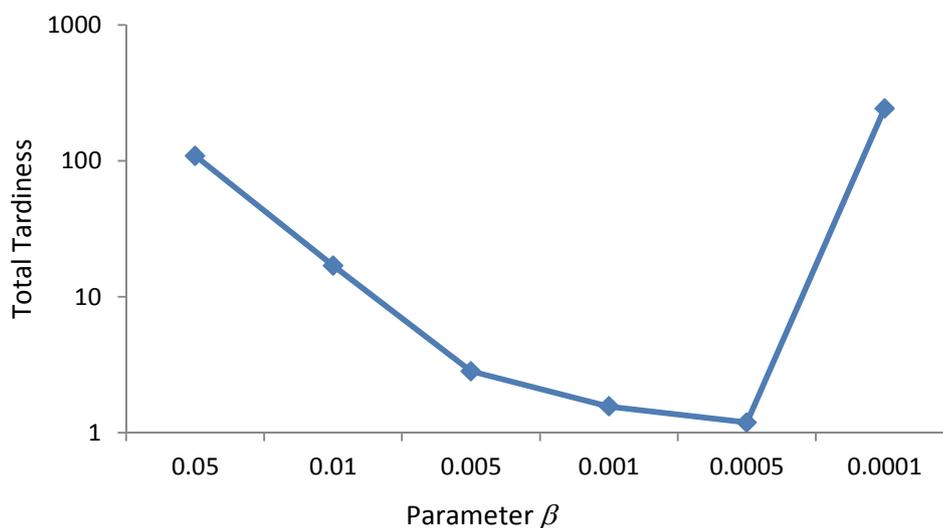
**Table 3.14 Experiment Results IV (Total Tardiness and CPU Time(sec))**

Jobs		Subgradient			Monotonic Non-Decreasing			Dispatching rule	
		$\beta=$ 0.005	$\beta=$ 0.001	$\beta=$ 0.0005	$\gamma=$ 0.001	$\gamma=$ 0.0005	$\gamma=$ 0.0001	FIFO	EDD
10	Average	0	0	0	0	0	0	0.52	0
	S.D.	0	0	0	0	0	0	0.73	0
	CPU(sec)	0.0036	0.0039	0.0041	0.0033	0.0039	0.0039	<b>0.0002</b>	0.0003
12	Average.	0	0	0	0	0	0	0.81	0
	S.D.	0	0	0	0	0	0	0.91	0
	CPU(sec)	0.0033	0.0029	0.0045	0.0044	0.0041	0.0045	<b>0.0004</b>	0.0013
14	Average.	0.20	0.15	<b>0.12</b>	0.41	0.36	0.21	2.94	1.95
	S.D.	0.38	0.29	<b>0.23</b>	0.72	0.65	0.39	2.26	1.79
	CPU(sec)	0.0821	0.2455	0.5458	0.0403	0.0556	0.2201	<b>0.0003</b>	0.0005
16	Average.	2.83	1.56	<b>1.19</b>	3.44	3.00	1.77	5.90	4.19
	S.D.	3.51	2.20	<b>1.76</b>	3.97	3.62	2.35	3.06	2.43
	CPU(sec)	0.4301	1.3959	2.3472	0.2029	0.3122	1.0562	<b>0.0004</b>	0.0005

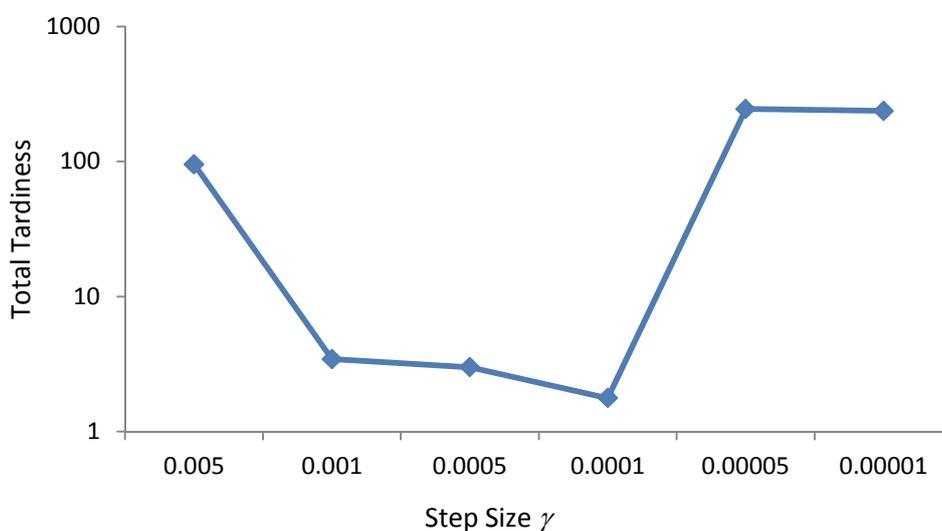
生産ライン負荷、すなわち、ジョブ数によらずほぼ全てのケースにおいて、従来手法と比較して、ほぼ同等の計算時間でより精度の高い最適解が導出できた。

#### (2) 初期パラメータ

次に定常状態であるジョブ 16 個のモデルにおいて初期パラメータと解精度について検証する。Fig. 3.28 に列勾配法の初期パラメータ  $\beta$  の値を，Fig. 3.29 に単調非減少法のステップサイズ  $\gamma$  の値を変化させた場合の総納期遅れ量を示す。



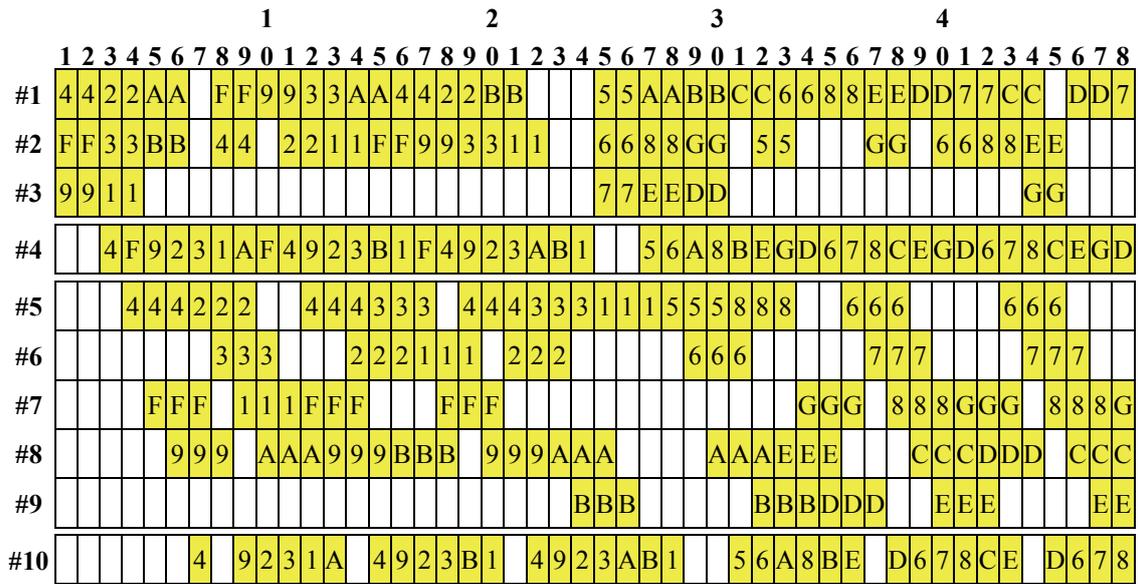
**Fig. 3.28 Total Tardiness IV in 16 jobs with Subgradient Method**



**Fig. 3.29 Total Tardiness IV in 16 jobs with Monotonic Non-Decreasing Method**

何れのラグランジュ乗数更新方法においても，更新幅を小さくすることで解精度を向上できた。しかしながら，値を小さくし過ぎると解精度は急激に悪くなる。これは，最適解の探索空間が小さくなることが原因と考えられる。初期パラメータが解精度に及ぼす影響が大きいことを確認した。本モデルにおいては解精度について全般的に列勾配法の方が良いものの特に大きな優位性は見られない。





Tardiness = 0

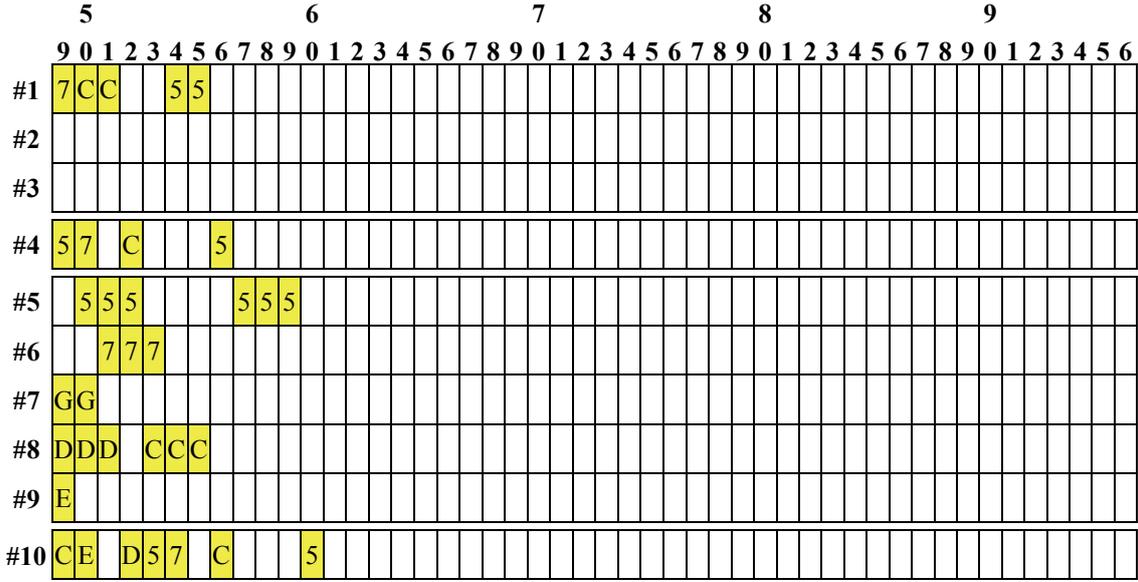


Fig. 3.31 Gantt chart IV of the best case in 16 jobs by Dispatching rule (EDD)

Fig. 3.30 に示す提案手法によるベストケース（総納期遅れ量=0）は、ボトルネック装置として設定した 4 号機（#4）の稼働率は 96.0%となり、Fig. 3.31 に示した従来手法によるベストケースの 88.9%と比較してもわかる通り、全体的にジョブ完了の早い前倒しの良いスケジュールとなっている。





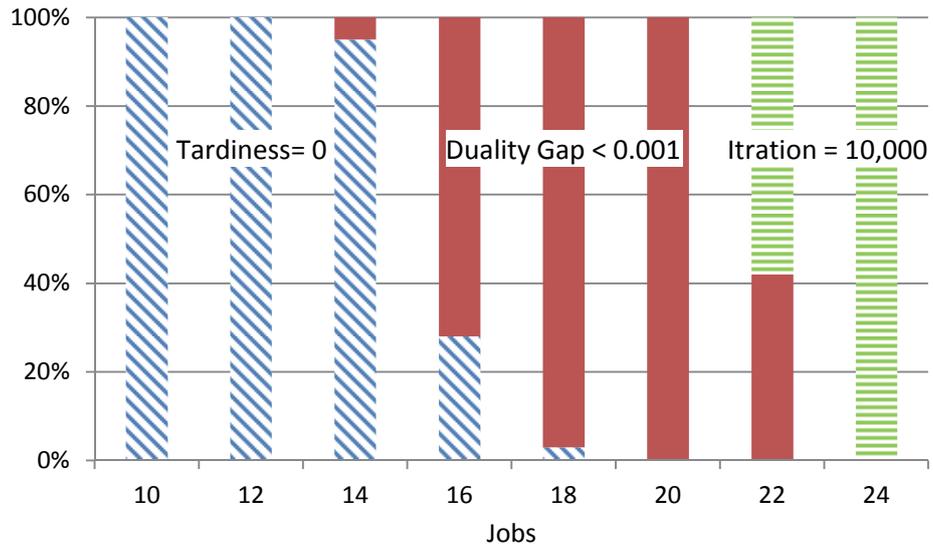


Fig. 3.34 Completed condition ratio with Subgradient Method ( $\beta=0.0005$ )

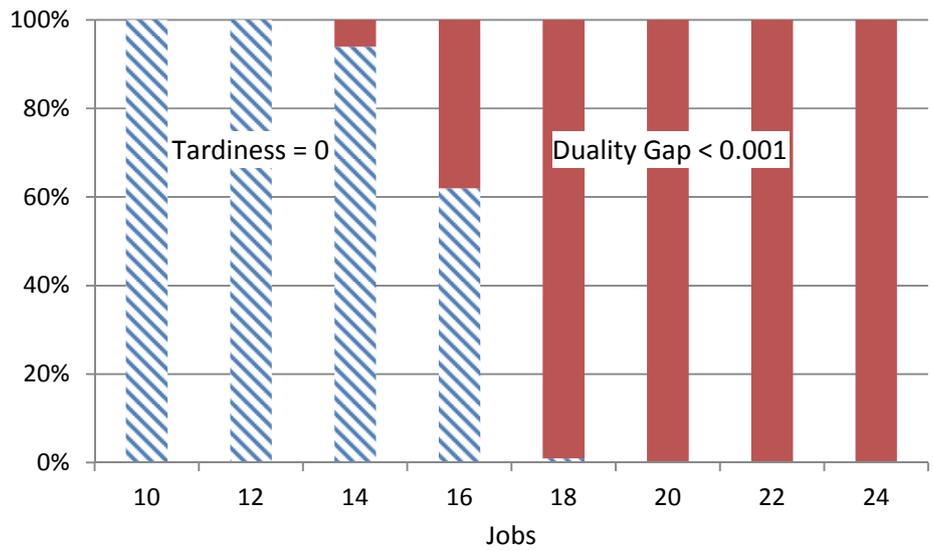


Fig. 3.35 Completed condition ratio with Monotonic Non-Decreasing Method ( $\gamma=0.0005$ )

単調非減少法が劣勾配法と比較して、ジョブ数が 16 個の定常状態より少ない場合に総納期遅れ量 0 の最適解を導出している割合が高く、ジョブ数が 16 個より大きい過負荷状態であっても、最適化計算は協調回数が規定回数の 10,000 回に到達する前に収束していることがわかる。

次に、Fig. 3.36 に列勾配法の初期パラメータ  $\beta$  の値を、Fig. 3.37 に単調非減少法のステップサイズ  $\gamma$  の値を変化させた場合の計算時間を示す。

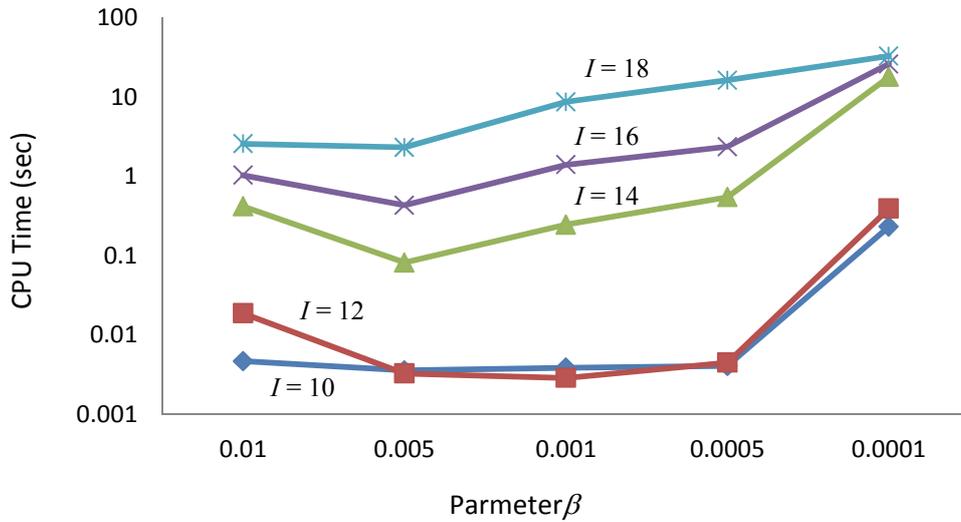


Fig. 3.36 CPU Time VI with Subgradient Method

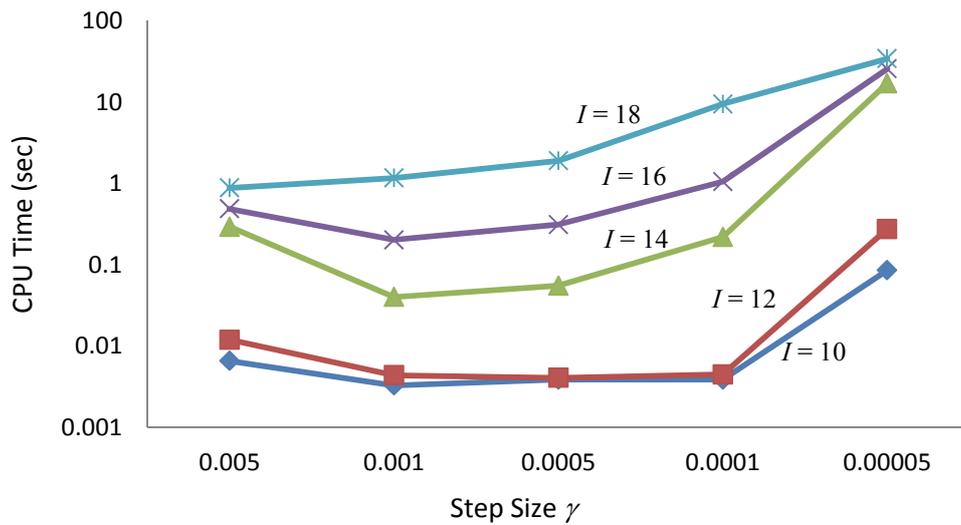


Fig. 3.37 CPU Time VI with Monotonic Non-Decreasing Method

何れのラグランジュ乗数更新方法においても、ジョブ数が増えること、更新幅を小さくすることで計算時間が指数関数的に増加することが確認できる。ジョブ数が増えるに従い、若干ながら単調非減少法の方が高速に解くことがきている。これは、解の振動が少なく、収束しやすくなっていることが原因であると考えられる。

Fig. 3.38 に、劣勾配法および単調非減少法における、最適化計算終了時の平均協調回数を示す。何れの手法においても、ジョブ数が稼動に多くなると規定回数以内では収束していないが、全てのモデルにおいて、単調非減少法の方が、劣勾配法より早い回数で最適化計算を終了していることがわかる。これが、計算時間の差の原因となっている。

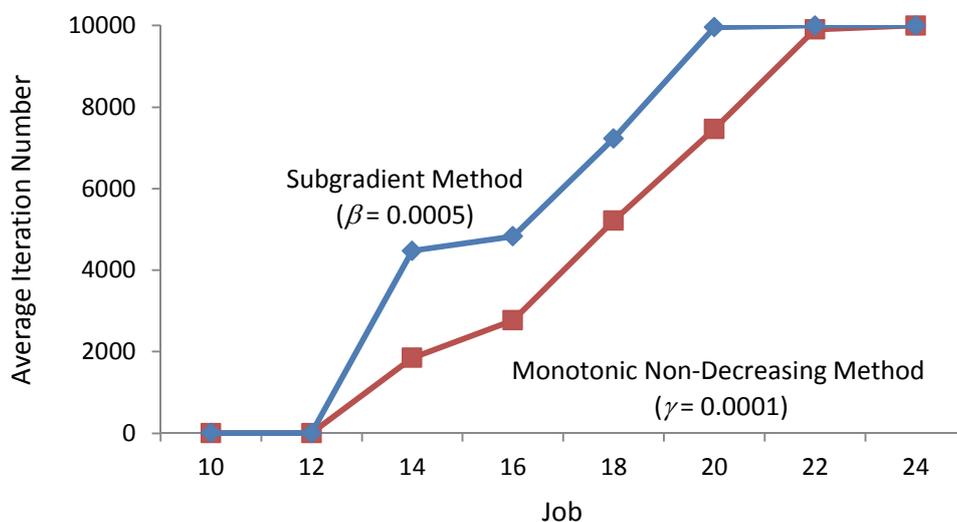


Fig. 3.38 Average iteration number VI

### 3.6 結言

本章では、半導体生産ラインを抽象化したモデルを対象としたスケジューリング問題にラグランジュ分解・調整法を適用し、現実的な生産ライン負荷の状況で、従来手法と比較して計算時間は増加するが実時間内での計算が可能であり、本提案手法が充分実用性のあることを確認した。

また、計算時間の増加を抑制しつつ解精度を向上させるためには、実行可能スケジュールを生成するリストスケジューリングのアルゴリズムを改善することが有効であることを示した。

さらに、初期パラメータが解精度ならびに計算時間に大きく影響していること、ラグランジュ乗数更新手法としては、単調非減少法の方が、劣勾配法よりも比較的解精度が良く、且つ高速に解けることも確認、終了条件を工夫することで計算時間の短縮が可能であることも示唆した。



## 4 分散協調型スケジューリング手法の半導体前工程実規模モデルへの適用

### 4.1 緒言

第3章では、実際の半導体前工程生産ラインを抽象化した、現実規模の約100分の1の小規模モデルにおいて、分散協調型スケジューリング手法の一つであるラグランジュ分解・調整法を適用し最適スケジュールが作成できること、従来手法であるディスパッチングルールと比較して有効であることを確認すると共に、解の改善のためにリストスケジューリングのアルゴリズム改善が効果的であることを示した。

本章では、現実生産ラインでのスケジューリング業務における実用化を目指すため、第3章と同じく半導体前工程生産ラインを対象に、そのモデルの規模を100分の1から少しずつ、1分の1まで拡大する上での課題を明確にし、本論文での提案手法であるリストスケジューリングのアルゴリズムを改善することでの解決策を探る。

### 4.2 対象システムおよび定式化

第3章と同じものを用いる。

### 4.3 計算機実験

#### 4.3.1 実験条件

##### (1) 規模モデル

第3章において現実の100分の1に相当する装置10台の小規模モデルはFig. 3.11に示す通り、ジョブ16個で定常状態になることを確認した。この100分の1モデルを基本とし、装置台数 $M$ およびジョブ数 $I$ を1分の1になるまで単純に整数倍して作成した7つの規模モデルをTable 4.1に示す。

Table 4.1 Scale Models

Scale	Machine Number $M$	Job Number $I$
1/100	10	16
1/50	20	32
1/20	50	80
1/10	100	160
1/5	200	320
1/2	500	800
1/1	1000	1600

(2) ジョブタイプ

第 3 章と同じく Fig. 3.8~Fig. 3.10 に示す量産品 2 タイプ, 量産立上品 1 タイプの合計 3 つのジョブタイプを用いる.

(3) 納期決定

第 3 章と同じ手順により納期をランダムに決定する.

(4) その他の実験パラメータ

ラグランジュ乗数更新には第 3 章にて優位性を確認した式 (3.18) に示す単調非減少ルールを用い, ステップサイズの  $\gamma$  値は解精度より計算時間を優先するため比較的大きな値に設定した. その他協調計算の終了条件に用いた, 実験パラメータを Table 4.2 に示す.

**Table 4.2 Experimental Parameters**

Parameter	Value
Job Type Ratio	4 : 3 : 1
Scheduling Period (Time Slot)	72
Step Size $\gamma$	0.01
Cooperative iteration number $\nu$	10,000
Duality Gap	0.001

(5) 試行回数

1 つの実験条件に対し, ジョブを 100 パターン作成してそれぞれ実験(最適化計算)する. 実験結果は 100 回の平均値を用いる.

#### 4.3.2 実験 I (第 3 章における提案手法の適用)

まず, 第 3 章の実験において従来手法であるディスパッチングより有効であることを確認した, 上界値計算に用いるリストスケジューリングのアルゴリズムを改善した提案手法による実験を行った. 100 分の 1 モデルの実験は第 3 章の実験 III と同じである.

Table 4.3 に示す通り, 目的関数である総納期遅れ量は, 従来方式のディスパッチングルールのうち最も単純な先入先出 (FIFO) ルールとの比較においては優位性を示したものの, しても, 規模の大きなモデルにおいては, 多くのケースで EDD より悪い結果となった.

Table 4.3 Experimental Results I (Total Tardiness)

Scale	Proposed Method		Dispatching rule (FIFO)		Dispatching rule (EDD)	
	Average	S.D.	Average	S.D.	Average	S.D.
1/100	<b>0.79</b>	<b>1.23</b>	5.90	3.06	4.19	2.42
1/50	<b>7.92</b>	4.86	19.29	5.95	8.03	<b>3.32</b>
1/20	17.53	7.83	27.50	6.87	<b>17.29</b>	<b>5.60</b>
1/10	38.90	11.87	56.84	10.36	<b>31.88</b>	<b>8.30</b>
1/5	78.55	16.29	113.97	14.29	<b>63.18</b>	<b>13.06</b>
1/2	410.58	42.16	419.01	25.81	<b>238.40</b>	<b>25.57</b>
1/1	<b>1027.56</b>	69.97	1922.63	<b>60.59</b>	1859.37	70.96

次に、それぞれの規模モデルにおける最適化過程を分析した。Fig. 4.1 に示す通り、何れの規模モデルにおいても、目的関数の総納期遅れ量は、協調回数の少ない段階で最適解が導出され、協調回数を重ねても改善されていないことを確認した。

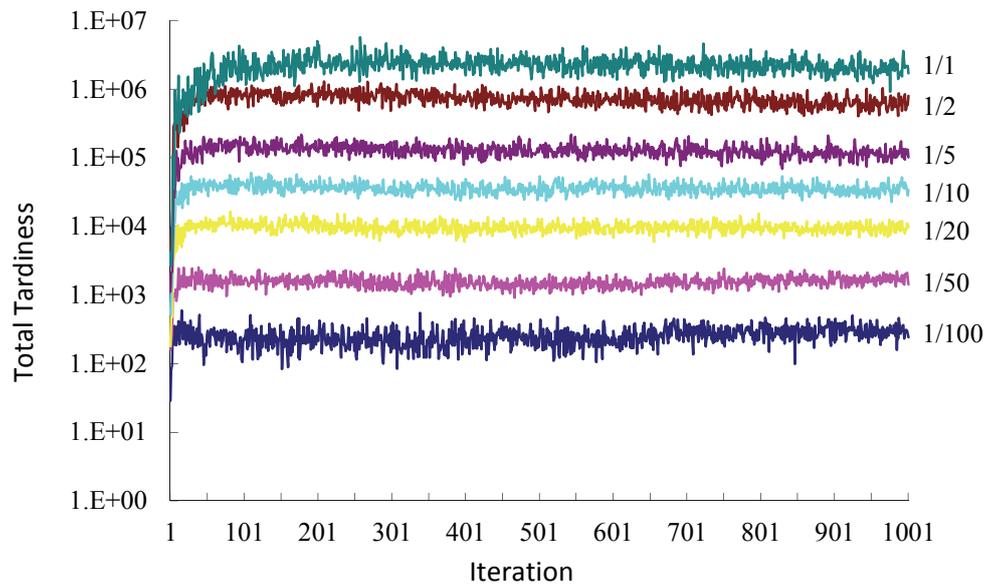


Fig. 4.1 Total Tardiness I by Iteration

次に、ラグランジュ乗数更新のステップサイズ $\gamma$ を変更して解の改善を試みた。しかしながら、Fig. 4.2 に示す通り、ステップサイズ $\gamma$ の大きさに関係無く、1回目もしくは少ない協調回数で得られた結果が最適解となり協調回数を重ねても解は改善されていない。

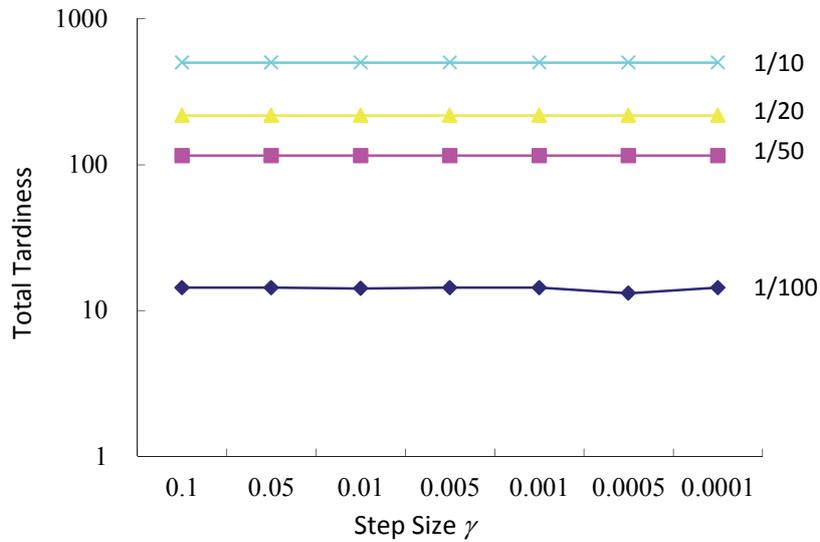


Fig. 4.2 Total Tardiness I by Step Size

これは、第 3 章の小さな規模では解決できた、代替装置が増えるに従い多くの装置のラグランジュ乗数の値が同じになることでより早いタイムスロットへ割当てることが困難になるという課題が未だ残っていることが原因と考えられる。

#### 4.3.3 実験 II (リストスケジューリングの更なるアルゴリズム改善)

第 3 章で提案した Fig. 3.6 のアルゴリズムでは、Fig. 3.7 に示す通り、装置#2 の代替装置である装置#3 を有効に活用できないという課題が残っていた。さらに代替装置選択における無駄省くため、DP により解法された部分問題の解のうち、ジョブの各工程開始時刻のみを参照し、割付装置情報を利用しないよう変更を試みた。このアルゴリズムを Fig. 3.4 に示す部分問題の解に適用して得られる実行可能スケジュールを Fig. 4.3 に示す。

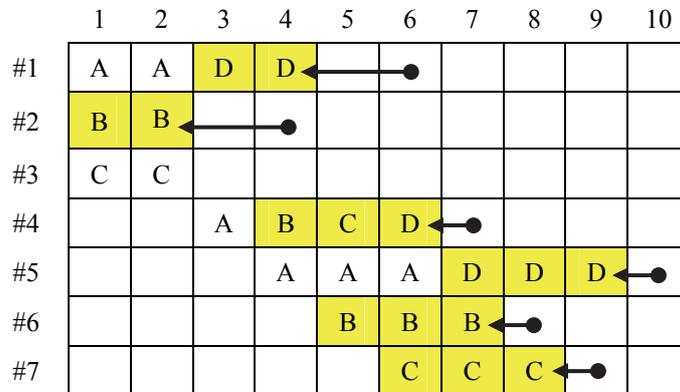


Fig. 4.3 Optimized Gantt chart by Proposal Algorithm Type2

本アルゴリズムでは先行関係制約の下で全ての代替装置の中から最も早い開始時刻のスロットに割付けていくため、全体的に前倒したスケジューリングが可能となる。この例では、ボトルネック装置と位置づけている装置#4のスロットの空き（機会損失）を無くすことが可能になっている。割付け順序を決めるリスト作成時に納期情報も考慮しているため、目的関数改善への寄与も大きい。改善したアルゴリズムを Fig. 4.4 に示す。区別するために第3章での提案手法を Type1, 本章における新しい提案手法を Type2 と呼ぶ。

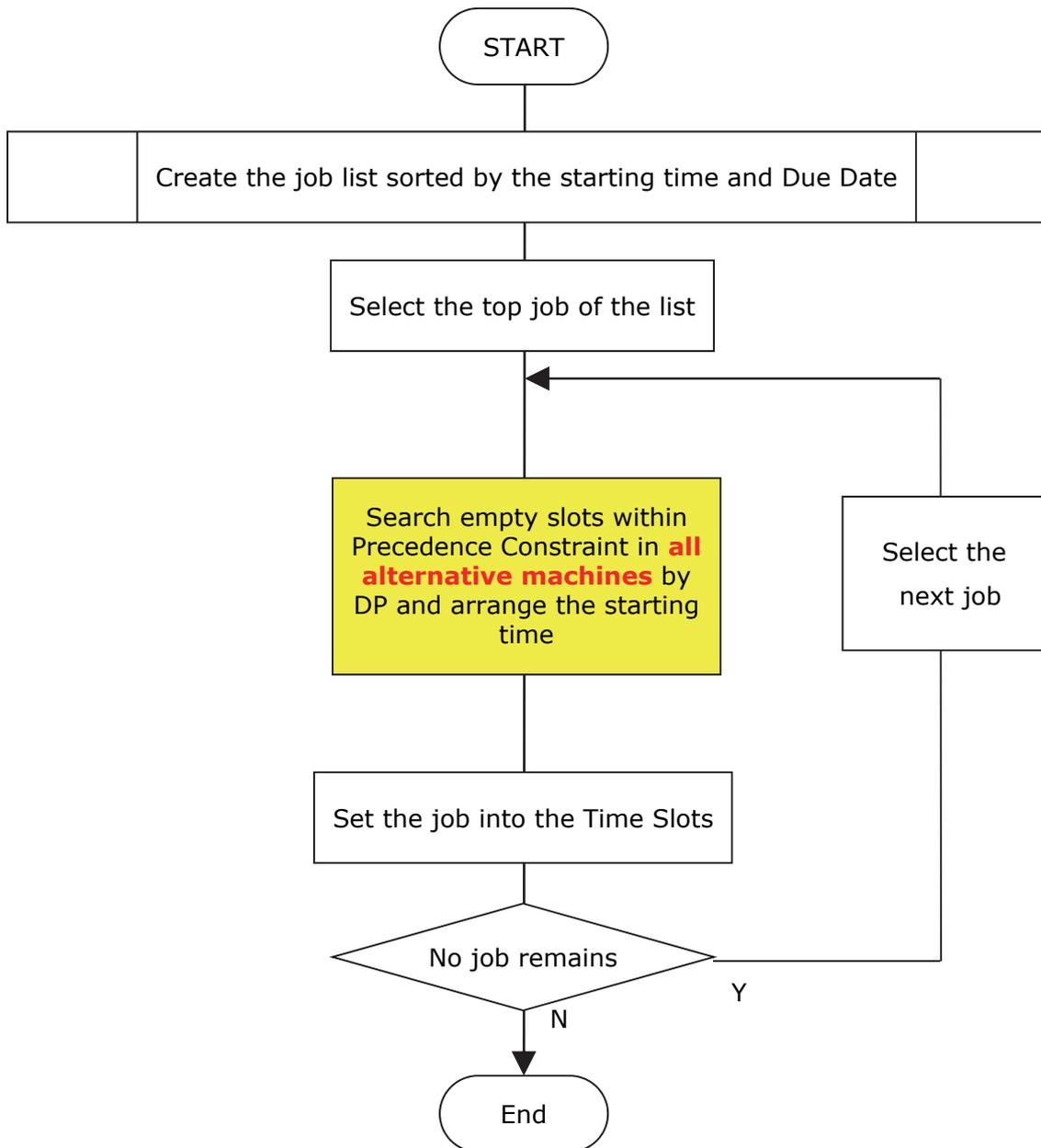


Fig. 4.4 Algorithm of Improved Type2 List Scheduling

(1) 総納期遅れ量 (目的関数)

Table 4.4 にリストスケジューリングのアルゴリズム改善による実験結果を示す。アルゴリズム以外の実験条件は実験 I と全く同じである。

**Table 4.4 Experimental Results II (Total Tardiness)**

Scale	Proposed Method (LDC)		Traditional Method (Dispatching rule)		
	Type1	Type2	FIFO	EDD	Slack
1/100	0.79	<b>0.00</b>	5.90	4.19	0.06
1/50	7.92	<b>0.00</b>	19.29	5.05	0.01
1/20	17.53	<b>0.00</b>	27.50	17.29	<b>0.00</b>
1/10	38.90	<b>0.00</b>	56.84	31.88	<b>0.00</b>
1/5	78.55	<b>0.00</b>	113.97	63.18	<b>0.00</b>
1/2	410.58	<b>0.00</b>	419.01	238.40	<b>0.00</b>
1/1	1016.40	<b>0.00</b>	1922.63	1859.37	<b>0.00</b>

Type2 のアルゴリズム改善により、Type1 と比較しても格段に良い解を得ることができた。更なる比較検証のため追加したディスパッチングルールの一つである納期迄の余裕度最小優先 (Slack)、すなわち、納期までの残日数を残工期 (完了までに必要な工程処理時間の総和) で除算した値が小さいジョブを優先するルールを用いた解と比較しても、全ての規模モデルにおいて納期遅れを全く発生させない最良の解を得ることができた。

(2) 計算時間

**Table 4.5 Experimental Results II (CPU Time (sec))**

Scale	Proposed Method (LDC)		Traditional Method (Dispatching rule)		
	Type1	Type2	FIFO	EDD	Slack
1/100	6.11	0.01	<b>0.0001</b>	0.0002	0.0002
1/50	10.57	0.01	<b>0.0005</b>	0.0006	0.0006
1/20	33.44	0.06	0.0032	<b>0.0025</b>	0.0032
1/10	98.07	0.22	<b>0.0083</b>	0.0104	0.0089
1/5	447.56	0.99	0.0638	<b>0.0571</b>	0.0927
1/2	5651.02	7.50	3.8246	<b>1.8396</b>	3.9231
1/1	57485.38	86.20	<b>45.4256</b>	62.2185	73.9597

Table 4.5 に示す通り、提案手法および従来手法ともに、モデルの規模が大きくなるに従い、計算時間が指数関数的に増加していることがわかる。Type2 のアルゴリズムでも全ての規模モデルにおいて従来手法より計算時間は多少増加するが、1 分の 1 モデルにおいても 86.2 秒と、十分実用化できるレベルであることが確認できた。

Fig. 4.5 に単純アルゴリズムの、Fig. 4.6 に提案アルゴリズム Type1 の、Fig. 4.7 に提案アルゴリズム Type2 の 100 回の試行における、最適計算終了条件の割合を示す。Fig. 4.5 に示す通り、単純アルゴリズムでは装置干渉のない最適解は導出できず、2 分の 1 モデル以上では、収束できず最適化計算を規定回数の 10,000 回で打ち切っている。

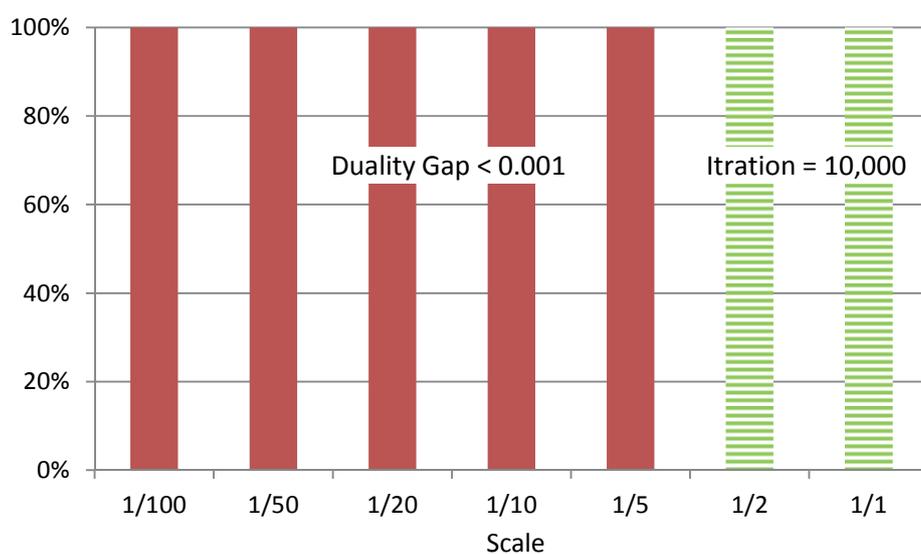


Fig. 4.5 Calculation completed condition with Simple algorithm

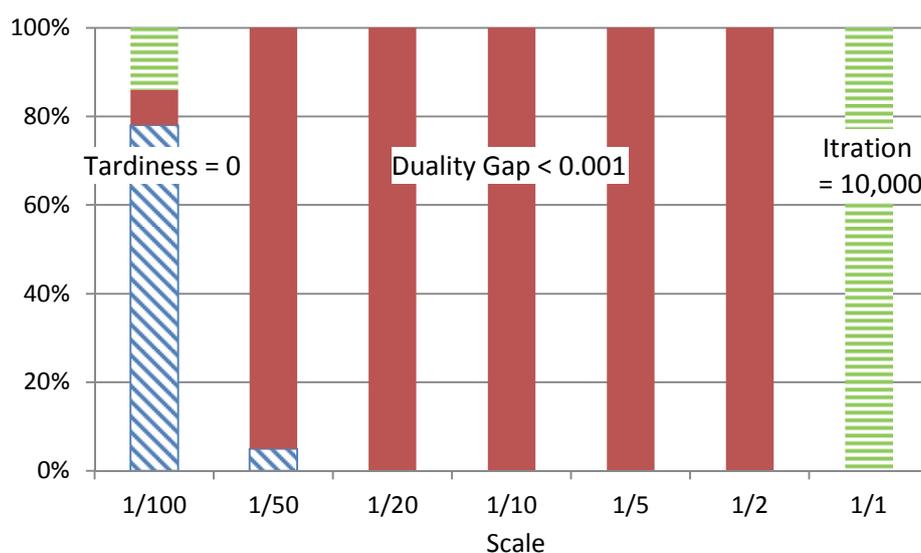


Fig. 4.6 Calculation completed condition with Proposal algorithm Type1

Fig. 4.6 に示す通り，Type1 では1分の1および100分の1モデルにおいて最適化計算が収束していないが，Fig. 4.7 に示す通り Type2 では全てのモデルにおいて，総納期遅れ量0の最適解を規定回数以内に導出している。

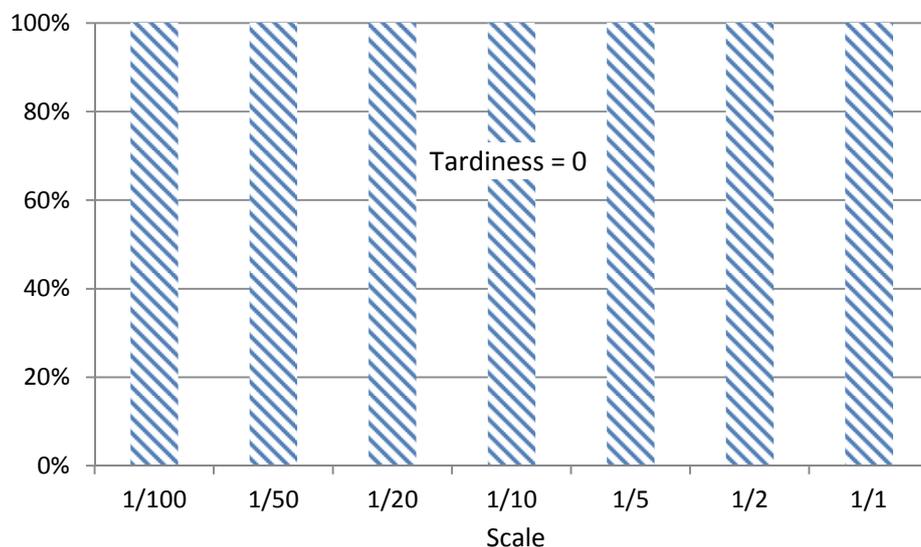


Fig. 4.7 Calculation completed condition with Proposal algorithm Type2

Fig. 4.8 にそれぞれのアルゴリズムにおける最適化計算終了時の平均協調回数を示す。モデルが大きくなるに従い，協調回数も増加しているが，Type2 ではかなり早い段階で総納期遅れ量なしの最適解を導出しているため，Table 4.5 に示す通り，計算時間が短くなっている。少ない協調回数で収束させることが実用化に向けて重要な課題であると言える。

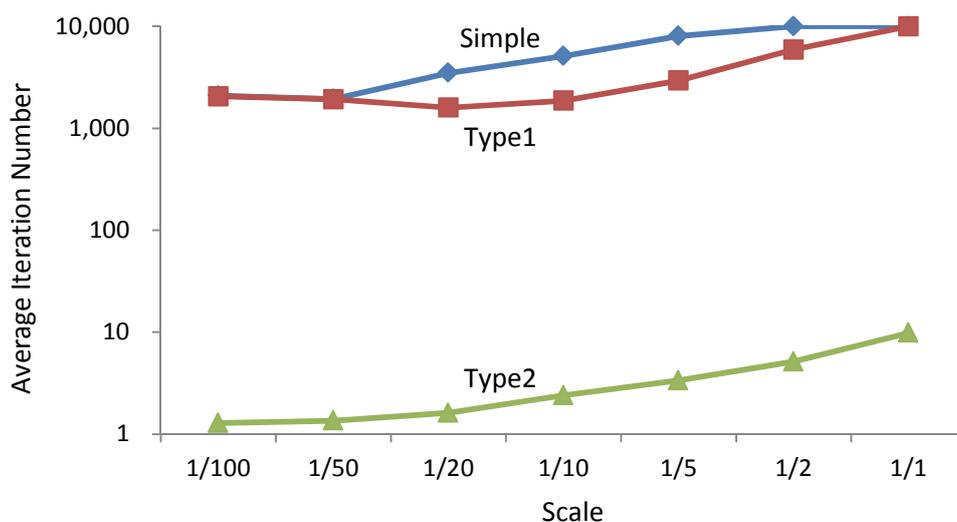


Fig. 4.8 Average Iteration Number

#### 4.3.4 実験 III (高負荷状態における追加検証)

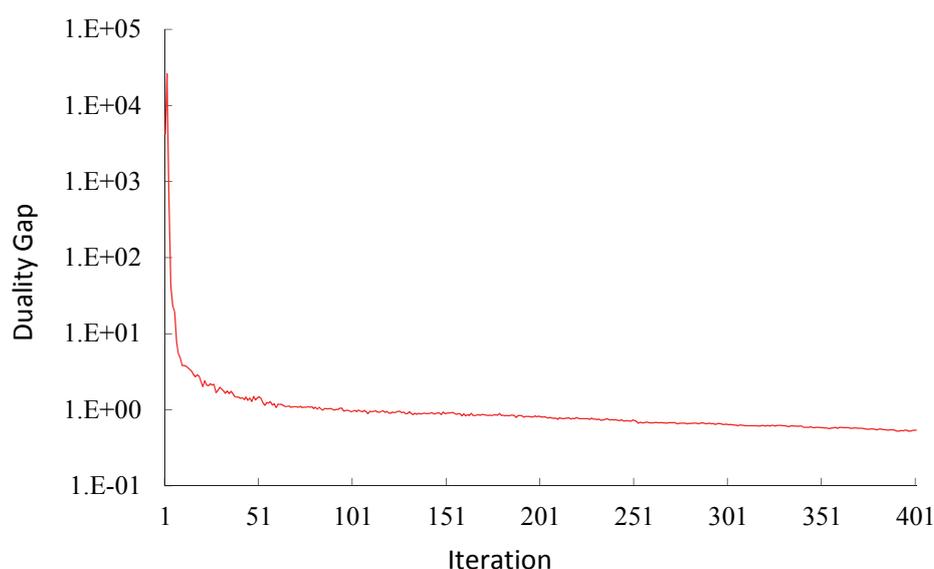
##### (1) 総納期遅れ量 (目的関数)

Type2 の有効性をさらに確認するため、各規模モデルにおいて、ジョブ数を 10%程度追加して過負荷状態にしての実験を行った。ただし、提案手法の 1 分の 1 モデルの計算時間が 1 回の試行に約 22hr 要したため、協調回数の規定回数を 1,000 回と 10 分の 1 短縮した。実験結果に影響ないことは後述する。

**Table 4.6 Experimental Results III (Total Tardiness)**

Scale	Job <i>I</i>	Proposal (Type2)		Dispatching rule (Slack)	
		Average	S.D.	Average	S.D.
1/100	18	<b>1.22</b>	<b>1.81</b>	2.07	2.73
1/50	36	<b>1.64</b>	<b>2.52</b>	4.58	6.01
1/20	90	<b>0.49</b>	<b>0.84</b>	0.50	0.85
1/10	180	<b>0.12</b>	<b>0.23</b>	0.15	0.29
1/5	360	0.04	0.08	<b>0.00</b>	<b>0.00</b>
1/2	900	5.14	5.21	<b>0.19</b>	<b>0.36</b>
1/1	1800	<b>1791.13</b>	<b>69.38</b>	2114.02	111.64

Table 4.6 に示す通り、ほとんどの規模モデルにおいて、従来手法 (Slack) より提案手法 (Type2) の方が良い解を得られた。1 分の 1 モデルにおいてその差は 323hr (15%) と明らかな優位性が示された。



**Fig. 4.9 Duality Gap by Iteration in 1/1 scale**

1分の1モデルにおける最適化過程を確認するため、Fig. 4.9に協調回数毎の双対ギャップの推移を、Fig. 4.10に協調回数毎の総納期遅れ量の推移を示す。それぞれ協調回数を重ねるごとに改善されており、最適化計算が正しく行われていることが確認できる。

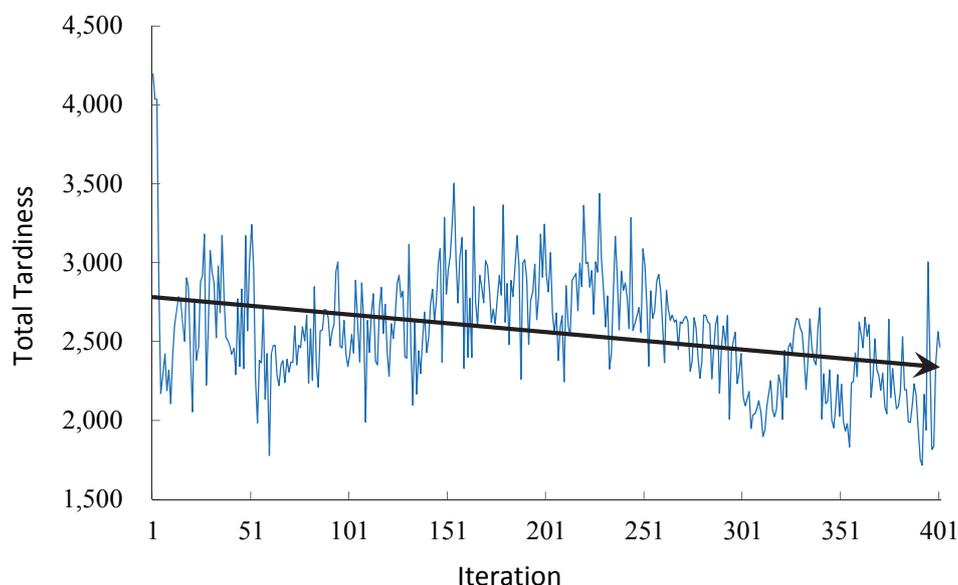


Fig. 4.10 Total Tardiness by Iteration in 1/1 scale

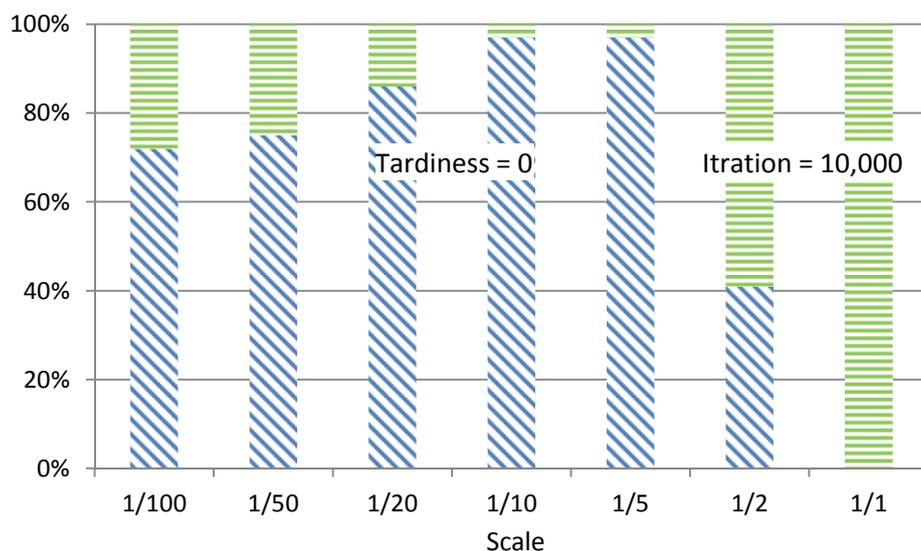
(2) 計算時間

次に実際のスケジューリング業務における実用化に向けてのハードルとなる計算時間を考察する。ただし、提案手法の1分の1モデルの計算時間が1回の試行に約22hr要したため、協調回数の規定回数を10,000回に戻し、試行回数を8回として実験した。

Table 4.7 Experimental Results III (CPU Time (sec))

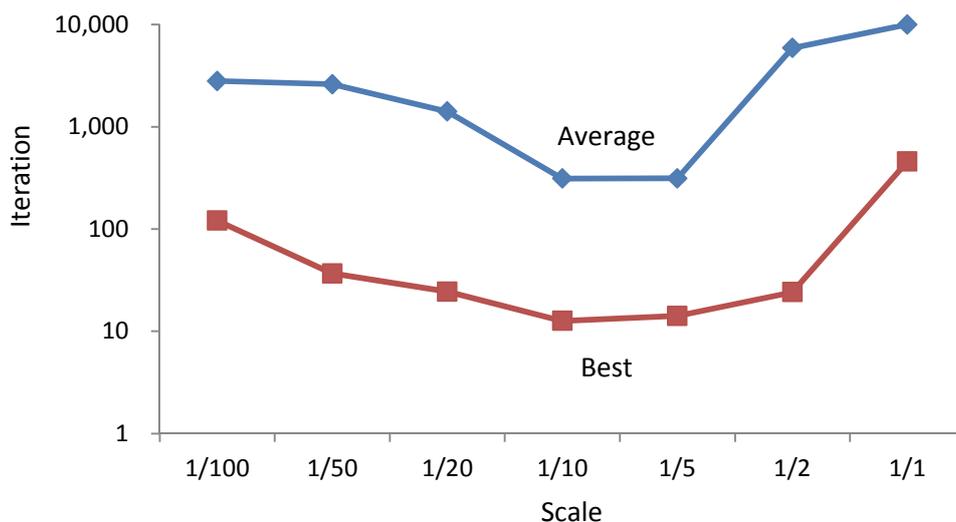
Scale	Job <i>l</i>	Proposal (Type2)		Dispatching rule (Slack)	
		Average	S.D.	Average	S.D.
1/100	18	7.28	10.44	<b>0.0006</b>	<b>0.0011</b>
1/50	36	15.66	23.07	<b>0.0014</b>	<b>0.0024</b>
1/20	90	27.53	46.79	<b>0.0038</b>	<b>0.0047</b>
1/10	180	18.73	33.82	<b>0.0144</b>	<b>0.0080</b>
1/5	360	57.88	106.26	<b>0.0754</b>	<b>0.0081</b>
1/2	900	6930.08	5659.01	<b>2.7580</b>	<b>0.0762</b>
1/1	1800	79089.10	305.95	<b>57.3663</b>	<b>2.1271</b>

Table 4.7 に示す通り, 全ての規模モデルにおいて, 従来手法 (Slack) より提案手法 (Type2) の方が数千倍以上時間が掛かっている上に, 1 分の 1 モデルにおいては前述の通り 1 回の試行に約 22hr 要しており実用化を検討できるレベルにない.



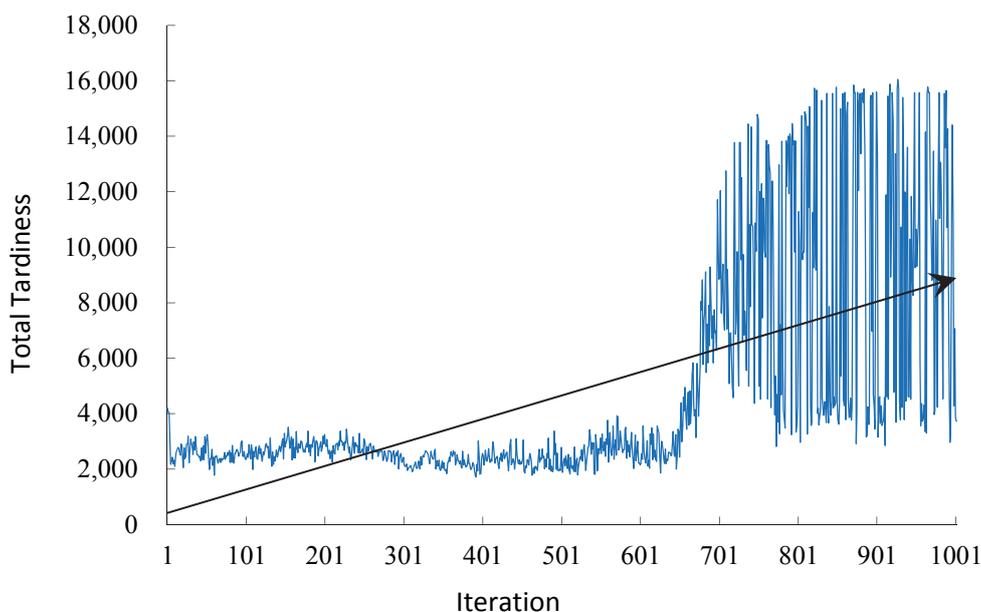
**Fig. 4.11 Calculation completed condition in the overload condition with Proposal algorithm Type2**

Fig. 4.11 に示す通り, この過負荷の状態における数値実験においては, 総納期遅れ量 0 の最適解が導出された以外は, 10,000 回の規定回数内で収束しておらず, 1 分の 1 モデルが極端に計算時間が遅い原因もこれである.



**Fig. 4.12 Average Iteration and the Iteration in solving best result**

Fig. 4.12 に示す通り、モデルの規模の違いはそれぞれの規模において与えたジョブ数との関連性が強いいため考察はしない。それぞれの規模において十数回から数百回と違いはあるものの、1 回目ではない比較的少ない協調回数の段階で最適解が導出されており、双対ギャップによる終了条件と最適解との間には強い関連性が無いことがわかる。



**Fig. 4.13 Total Tardiness by Iteration in 1/1 scale**

ここで最も計算時間の長い 1 分の 1 モデルにおける最適化過程の詳細を確認してみる。Fig. 4.13 に示す通り、目的関数である総納期遅れ量は、協調回数 392 回目に最小となっているが、その後、協調回数を重ねるにつれ大きく振動している。双対ギャップが終了条件に達しないため、もう一つの終了条件である 10,000 回に達するまで協調回数を重ねた結果、計算時間が長くなっていることが確認できる。

従って、最適化計算は終了しないものの、解の改善が見られなくなった時点で早めに最適化計算を打ち切る判断ができれば、十分実用化できる範囲内に最適解を導出することができると考えられる。

更に計算時間を分析するため、その内訳を Table 4.8 に示す。計算時間のうち、モデルの規模に関係なくほとんど 90%以上を占めるのが、DP による部分問題解法であることが確認できる。

**Table 4.8 Experimental Results III (CPU Detail Time (sec))**

Scale	CPU Time (sec)	Shared Ratio		
		DP	List Scheduling	Update Lagrange Multipliers
1/100	7.13	<b>97.91%</b>	1.41%	0.08%
1/50	15.28	<b>97.59%</b>	1.81%	0.07%
1/20	26.68	<b>96.88%</b>	2.59%	0.06%
1/10	17.95	<b>95.80%</b>	3.82%	0.03%
1/5	53.93	<b>93.18%</b>	6.53%	0.02%
1/2	6339.90	<b>91.48%</b>	8.38%	0.01%
1/1	74374.47	<b>94.04%</b>	5.89%	0.01%

#### 4.4 結言

本章では、第 3 章の小規模モデルにおいて有効性を確認したラグランジュ分解・調整法を、半導体前工程を抽象化した大規模モデルへ適用、その規模を 100 分の 1 から 1 分の 1 まで拡大するにあたり、実行可能スケジュールを算出するリストスケジューリングのアルゴリズムを工夫することで、従来手法のディスパッチングルールと比較して精度の高い解を得ることができることを確認した。これは、動的計画法 (DP) による部分問題解法において、代替装置が増えるにつれ最適な装置を 1 台選択すること困難になる欠点を補うことができるためと考える。

また実用化に向けての大きな課題となる計算時間は、モデルの規模が大きくなるに従い指数関数的に増大しているが、最適化計算の終了条件を改善することで解決できる可能性があること、またモデルの規模に依らず計算時間の 90%以上を占める動的計画法 (DP) による部分問題の解法を高速化することがポイントであることを確認した。



## 5 分散協調型スケジューリング手法の半導体後工程生産ラインへの適用

### 5.1 緒言

第 3 章では、実際の前工程生産ラインを抽象化したモデルにおいて、多品種生産における生産ライン負荷に対する特性を評価し、従来手法であるディスパッチングルールと比較して有効であることを確認した。さらに第 4 章では、同じく半導体前工程生産ラインを対象に、より現実に近い大規模モデルへの適用するにあたっては、代替装置の数が増加するに伴い指数的に計算時間が長くなるため、その約 90%を占める部分問題解法をできる限り簡素化し、実行可能スケジュール取得アルゴリズムに様々なヒューリスティックを組み合わせることにより、実時間内で精度の高い解が得られることを確認した。

本章においては、将来テスト工程含めた半導体全工程への適用も視野に入れ、まずは実際の組立生産ラインを対象に、前工程生産ラインのスケジューリング研究で得られた知見の有効性を検証する。その上で、品種切り替え回数の最小化による納期遅れ量最小スケジュールの策定、将来の装置増強（治具手配による品種切替可能化）の必要性検証など、実際のスケジューリング業務において実用化していく上での課題を明確にする。

### 5.2 対象システム

現実の半導体後工程生産ラインは、Fig. 2.7 に示すような製造工程が、類似製品グループ毎に割り当てられた Fig. 5.1 のようなフローショップ複数から構成されるが、スケジューリングは個別のフローショップ単位で実施する。金型の寿命や、定期的あるいは使用回数に応じた定期メンテナンスについても実際のスケジューリングの際には考慮が必要であるが、本章においては、第一ステップとして、常時利用可能なものと単純化して取り扱う。

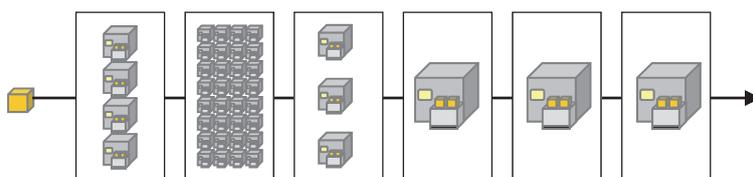


Fig. 5.1 Flow Shop

本章で対象とするのは、 $M$  台の装置で構成されるフローショップにおける、セットアップ最適化を考慮した、 $I$  個のジョブごとの納期遅れ時間を最小化するスケジューリング問題である。

### 5.3 定式化

半導体全工程への適用を目指すため、第 3 章で述べた定式化と同じものを利用する。但し、先行関係制約式 (3.7) を、ジョブの開始時刻と終了時刻との間に装置  $m$  における品種切り替え時間  $ct_m$  を考慮して、式 (5.1) へと拡張する。

$$ed_{i,j,m} = \begin{cases} st_{i,j,m} + pt_{i,j} - 1 & (m \in U_{i,j}, P_{m,k-1} = P_{m,k}) \\ ct_m + st_{i,j,m} + pt_{i,j} - 1 & (m \in U_{i,j}, P_{m,k-1} \neq P_{m,k}) \end{cases} \quad (5.1)$$

このとき、セットアップ時間  $ct_m$  を 0 に設置することで式 (5.1) は式 (3.7) と同じになり、前工程含む全工程へ適用するための汎用性は保たれる。

### 5.4 計算機実験

#### 5.4.1 実験条件

##### (1) ジョブタイプ

Fig. 5.2 から Fig. 5.4 に今回の計算機実験に用いた 3 種類のジョブタイプを示す。

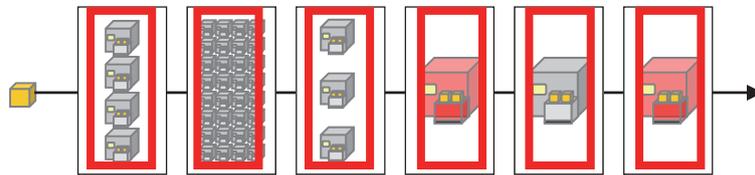


Fig. 5.2 Job Type 1 (Mass Production)

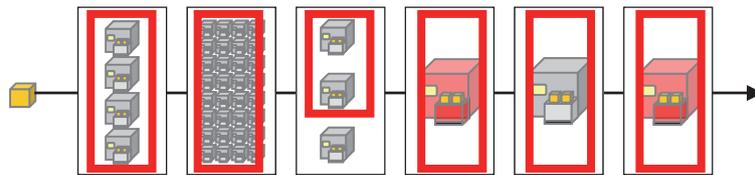


Fig. 5.3 Job Type 2 (Mass Production)

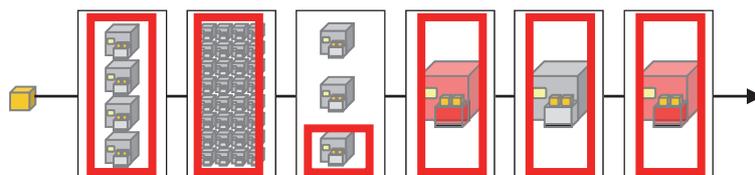


Fig. 5.4 Job Type 3 (Pilot Production)

太枠は各ジョブが該当工程において使用可能な代替装置を意味し、複数代替装置が存在する場合は、何れか 1 台が使用可能である。赤色の装置は品種切り替えが発生した場合に治具や金型の交換が必要な装置を示す。Table 5.1 に各ジョブタイプの詳細を示す。j は工程番号であり、 $pt_{i,j}$  はジョブ i 工程 j の処理時間、 $U_{i,j}$  はジョブ i 工程 j で使用可能な装置の集合を示す。

**Table 5.1 Process Chart of Job Type**

j	Job Type 1				Job Type 2				Job Type 3						
	$pt_{i,j}$	$U_{i,j}$			$pt_{i,j}$	$U_{i,j}$			$pt_{i,j}$	$U_{i,j}$					
1	4	1	2	3	4	3	1	2	3	4	3	1	2	3	4
2	28	5	~	36		34	5	~	36		39	5	~	36	
3	4	37	38	39		2	37	38			4	39			
4	1	40				1	40				1	40			
5	1	41				1	41				1	41			
6	1	42				1	42				1	42			

(2) 納期決定

ジョブはモデルとした実際の生産ラインの通り、毎日の開始の時刻に一定量 (50 個/日) を、ジョブタイプ毎に一定の比率に配分し、式 (3.19) を用いてそれぞれのジョブに対して納期をランダムに決定する。

(3) その他の実験パラメータ

ラグランジュ乗数更新には式 (3.18) に示す単調非減少ルールを用い、その他協調計算の終了条件に用いた、その他の実験パラメータを Table 5.2 に示す。

**Table 5.2 Experimental Parameters I**

Parameter	Value
Job Number $I$	100
Job Type Ratio	29 : 20 : 1
Scheduling Period (Time Slot)	288
Step Size $\gamma$	0.01
Cooperative iteration number $\nu$	1,000
Given Duality Gap	0.001

(4) 試行回数

1 つの実験条件に対し、ジョブのモデルを 100 パターン作成してそれぞれ実験 (最適化計算) する。実験結果は 100 回の平均値を用いる。

#### 5.4.2 実験 I (後工程単純モデルにおける LDC の有効性確認)

まず、品種切り替えを考慮しない、つまり、第 4 章で提案した手法により、従来手法のディスパッチングルールである FIFO (First-In, First-Out: 先入先出) および、EDD (Earliest Due Date: 最早納期のジョブを優先) と比較することで提案手法の有効性を検証した。

##### (1) 総納期遅れ量 (目的関数)

Table 5.3 に示す通り総納期遅れ時間の平均値が 405.38 と最も小さくなり、後工程生産ラインにおいても従来手法と比較して本提案手法が有効であることを確認した。

**Table 5.3 Experimental Results I (Total Tardiness)**

Proposed Method ( $\gamma=0.01$ )		Dispatching rule (FIFO)		Dispatching rule (EDD)	
Average	S.D.	Average	S.D.	Average	S.D.
<b>405.38</b>	54.08	1061.46	<b>53.80</b>	417.60	60.10

##### (2) 計算時間

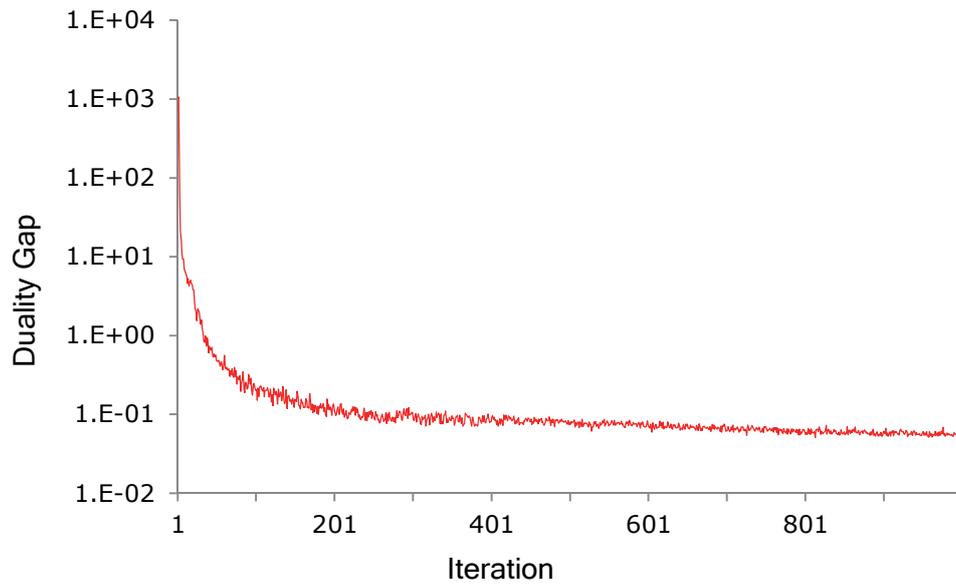
Table 5.4 に示す通り、実用化のための重要な指標である計算時間は、従来方式と比較すれば約 30,000 倍の時間を要してはいるが、117.86 秒は十分実用的であると言える

**Table 5.4 Experimental Results I (CPU Time(sec))**

Proposed Method ( $\gamma=0.01$ )		Dispatching rule (FIFO)		Dispatching rule (EDD)	
Average	S.D.	Average	S.D.	Average	S.D.
117.86	0.5861	0.004	0.0048	<b>0.0037</b>	<b>0.0047</b>

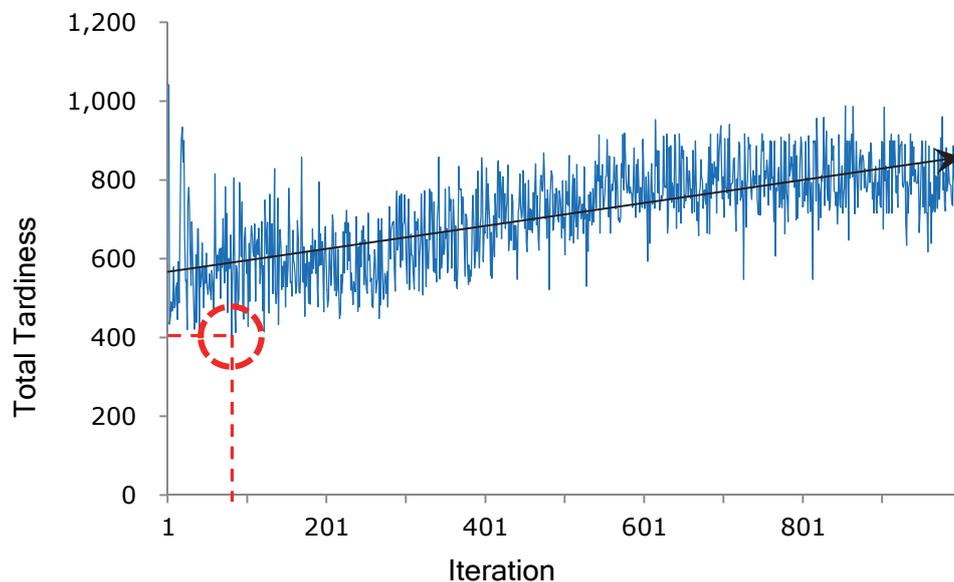
##### (3) 7 回目の試行結果の詳細分析

総納期遅れ量が平均値とほぼ同じ 405 となった 7 回目の試行結果を更に詳細に分析した。



**Fig. 5.5 Duality Gap I by iteration in the 7<sup>th</sup> trial**

双対ギャップは Fig. 5.5 に示す通り，協調回数を重ねるごとに最適化が進んでいることが確認できる。



**Fig. 5.6 Total Tardiness I by iteration in the 7<sup>th</sup> trial**

Fig. 5.6 に，協調回数毎の目的関数である総納期遅れ量の推移を示す．本ケースにおいては 81 回目に総納期遅れ量が 405 と最小になった以降は，総納期遅れ量は増加傾向となっている．これも，第 3 章および第 4 章の半導体前工程のモデルにおける数値実験の結果と同様，

代替装置が多く、協調回数を重ねるごとにラグランジュ乗数の値が同じ装置が増える結果、比較的后倒しにスケジューリングされる結果となるが、半導体前工程と違いリエントラントではないため比較的その傾向が小さいと考えられる。これらの結果より、双対ギャップが規定値以下になるか規定繰り返し回数に達すれば終了としている今回の終了条件を工夫し、目的関数が増加傾向となる点をいち早く検知することで、より短時間での最適解探索が可能となると言える。

#### 5.4.3 実験 II（後工程単純モデルにおける LDC の特性確認）

更に、ラグランジュ乗数更新におけるステップサイズの変化が解精度に及ぼす影響について検証した。

**Table 5.5 Experimental Results II (Total Tardiness and CPU Time(sec))**

Step Size $\gamma$	Total Tardiness		CPU Time (sec)	
	Average	S.D.	Average	S.D.
0.1	432.18	55.33	<b>110.91</b>	<b>0.57</b>
0.05	424.21	53.63	111.32	0.60
0.01	405.38	54.08	117.85	0.69
0.005	404.88	52.82	123.46	0.61
0.001	<b>402.04</b>	52.12	130.35	0.74
0.0005	407.16	50.77	130.53	0.72
0.0001	409.48	<b>49.38</b>	130.52	0.71

Table 5.5 に示す通り、ステップサイズの増加分を調整し、最適解の探索を細かく行うことで解精度を改善できることが確認できた。本モデルにおいては、ステップサイズが 0.001 のときに総納期遅れ量は 402.04 と最小となった。しかしながらステップサイズを更に小さくすると解精度が逆に悪くなった。これは、規定協調回数内で最適解に到達しないためであると考えられる。

実験 I および実験 II の結果により、第 4 章において半導体前工程の現実規模モデルでその特性を発揮できると確認した提案手法ラグランジュ分解・調整法 (LDC) が、半導体後工程の実際の生産ラインにおける生産スケジュール最適化にも有効であることが確認できた。

#### 5.4.4 実験 III（後工程複雑モデルにおける LDC の有効性確認）

次に同じモデルにおいて、2 台のマシンにそれぞれ Table 5.6 に示す品種切り替え時間を

設定して実験を行った.

**Table 5.6 Experimental Parameters III**

Machine Number $m$	Setup Time $ct_m$
40	3
42	1

(1) 総納期遅れ量 (目的関数)

Table 5.7 に示す通り, 総納期遅れ量 840.32 と, 従来手法より有効であることが確認できた. なお, ジョブは品種毎に順番に作成するため, ディスパッチングルールにおいてはジョブの開始時刻が品種毎に並び, ジョブ選択の際に FIFO での方が EDD より品種が連続しやすい結果がそのまま現れている.

**Table 5.7 Experimental Results III (Total Tardiness)**

Proposed Method ( $\gamma=0.01$ )		Dispatching rule (FIFO)		Dispatching rule (EDD)	
Average	S.D.	Average	S.D.	Average	S.D.
<b>840.32</b>	62.35	1214.74	<b>54.11</b>	5089.68	639.05

(2) 計算時間

Table 5.8 に示す通り, 品種切り替え時間を考慮しない実験 I とほぼ同じで 118.19 秒であり, 従来方式と比較すれば時間が掛かるが, 十分実用的であると言える.

**Table 5.8 Experimental Results III (CPU Time(sec))**

Proposed Method ( $\gamma=0.01$ )		Dispatching rule (FIFO)		Dispatching rule (EDD)	
Average	S.D.	Average	S.D.	Average	S.D.
118.19	0.6379	0.0043	0.0048	<b>0.0040</b>	<b>0.0047</b>

(3) 詳細スケジュール (実行可能スケジュール)

Fig. 5.7 から Fig. 5.9 に本実験で得られたガントチャートの一部を示す. 横軸が  $t(1, \dots, T)$ , 縦軸に装置の 37 号機 (#37) ~ 42 号機 (#42) で, 各セルがタイムスロットを示す. セル内の記号はジョブタイプ番号で, 水色のハッチングは品種切り替えを示す.



#### 5.4.5 実験Ⅳ（更なる解の改善のための方案検証）

更に解精度を向上させるためのアルゴリズム改善を実施した。従来手法のディスパッチングルールにおいても解精度の向上が可能であること、およびその結果と提案手法の差を更に検証するため、SST（Shortest Setup Time）による実験を追加した。提案手法においては、これまで施してきた改善の通り、リストスケジューリングにおける改善を試みるため、部分問題の解法が終了した時点でリスト作成のときに、開始時刻が同一であれば、まず同一品種で並べた上で納期が早い順にジョブを並べるように改善した。

##### (1) 総納期遅れ量（目的関数）

Table 5.9 に示す通り、SST においては総納期遅れ量が 973.98 と、FIFO の 1214.74 と比較して解が改善された。これは、品種切り替えが必要な装置以外の装置がボトルネックになっていることで、品種切り替えの影響を無くすことが出来たためと考えられる。

**Table 5.9 Experimental Results IV (Total Tardiness)**

Proposed Method 2 ( $\gamma=0.01$ )		Dispatching rule (SST)	
Average	S.D.	Average	S.D.
<b>832.98</b>	64.66	973.98	<b>75.78</b>

ラグランジュ分解・調整法におけるリストスケジューリングのアルゴリズム改善では、従来手法との比較において良い解を得られたが、約 9.9%しか解を改善（840.32→832.98）することができなかった。これは従来方式において FIFO が EDD よりも良い結果が得られたことと同様に、式 (3.19) を用いてジョブを作成する際にジョブタイプ毎にジョブを作成するためジョブ番号で並べると同じジョブタイプが連続しており、ジョブ毎に部分問題を解法した直後に追加していくリストにも同じジョブタイプが連続し易い構造であることに起因すると考えられる。

##### (2) 計算時間

Table 5.10 に示す通り、実験 III とほぼ同じで 117.58 秒であり、従来方式と比較すれば時間が掛かるが、十分実用的であると言える。

**Table 5.10 Experimental Results IV (CPU Time(sec))**

Proposed Method 2 ( $\gamma=0.01$ )		Dispatching rule (SST)	
Average	S.D.	Average	S.D.
117.58	0.5125	<b>0.0042</b>	<b>0.0049</b>

(3) 1回目の試行結果の詳細分析

Fig. 5.10 に1回目の試行における双対ギャップを示す. 双対ギャップは協調回数を重ねるごとに小さくなっており最適化計算が正常に機能していることがわかる.

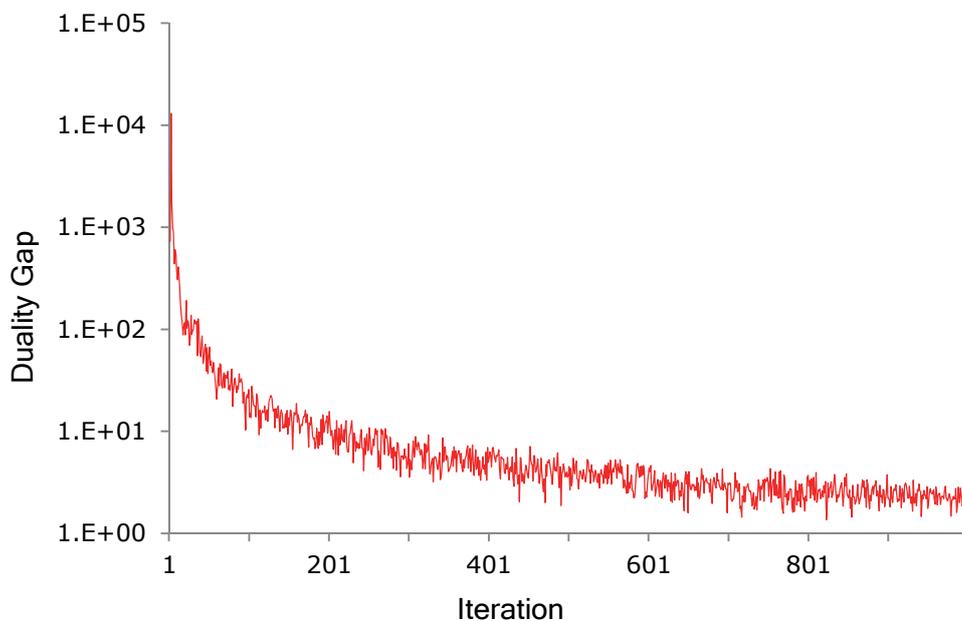


Fig. 5.10 Duality Gap IV by iteration in the 1<sup>st</sup> trial

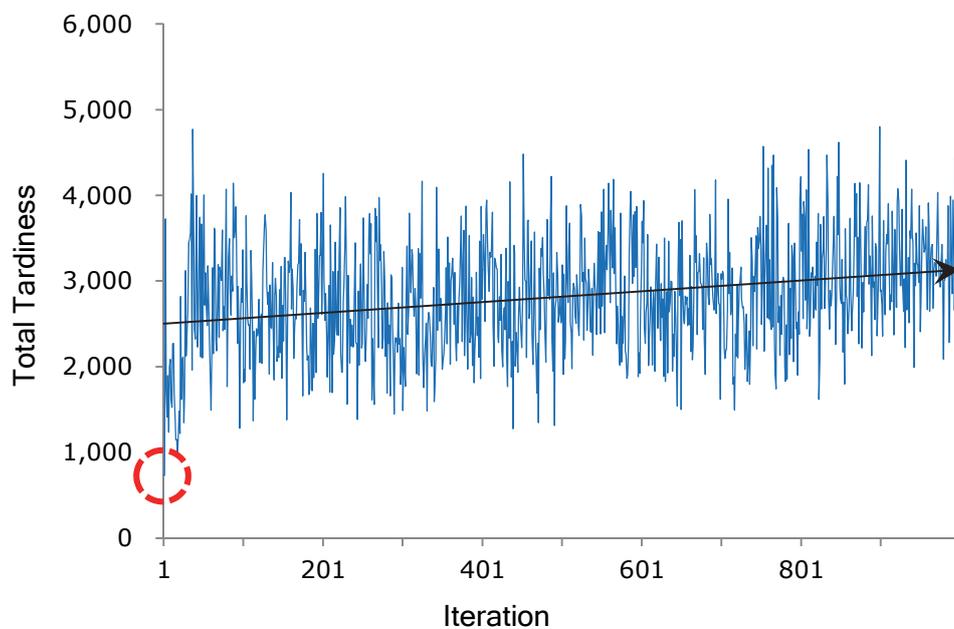


Fig. 5.11 Total Tardiness IV by iteration in the 1<sup>st</sup> trial



(4) ワーストケース (15 回目の試行) の詳細分析

更にワーストケースである総納期遅れ量 1,013 であった 13 回目に試行したモデルにおける実験結果についても分析した。Fig. 5.14 に示す通り、双対ギャップは協調回数を重ねるごとに小さくなっており最適化計算が正常に機能していることがわかる

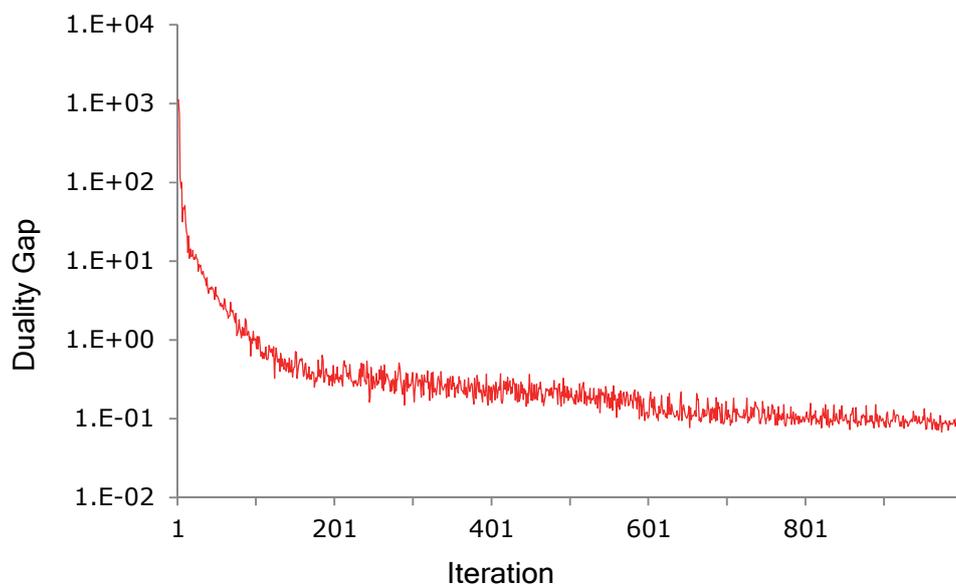


Fig. 5.14 Duality Gap IV by iteration in the 15<sup>th</sup> trial

Fig. 5.15 に総納期遅れ量の各協調回数における推移を示す。

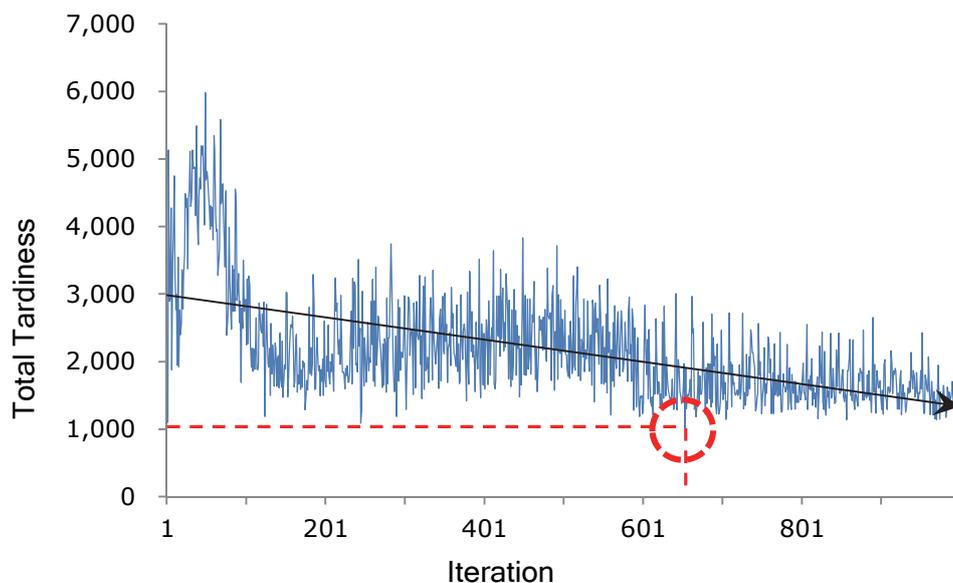


Fig. 5.15 Total Tardiness IV by iteration in the 15<sup>th</sup> trial

本ケースにおいては、協調回数 653 回目において総納期遅れ量が最小の 1,013 となった後も減少傾向が見えた。

念のため協調回数の規定回数を 10,000 にして実験したところ、協調回数 3,719 回目に総納期遅れ量が最小の 815 と、更に 19.5%改善された。Fig. 5.16 にこのときの総納期遅れ量の各協調回数における推移を示す。協調回数が 1,000 回を超えたあたりからほとんど減少傾向は見られず、解が大きく振動している中で下に大きく振れたときに最適解が導出されていることがわかる。

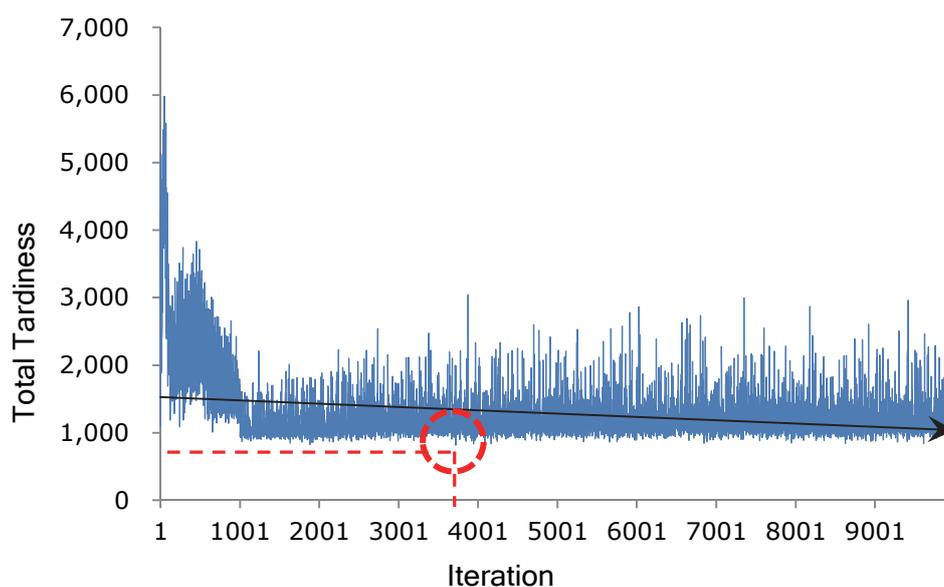


Fig. 5.16 Total Tardiness IV-2 by iteration in the 15<sup>th</sup> trial

(5) 協調回数の規定回数を 10,000 にした再実験

協調回数の規定回数を 10,000 に変更して再実験した結果を Table 5.11 に示す。

Table 5.11 Experimental Results IV -2 (Total Tardnes and CPU Time(sec))

Total Tardiness		CPU Time(sec)	
Average	S.D.	Average	S.D.
680.65	64.55	1093.37	1.8088

総納期遅れ量は 15 回目の試行における結果とほぼ同じ 19.3% (832.98→680.65) と改善されたが、そのトレードオフとして計算時間は、117.58→1093.37 と約 9.3 倍も悪化した。協調回数の規定回数を実験前に最適に決定することは困難であると考えられる。

## 5.5 結言

本章では、現実の半導体組立生産ラインを対象としたスケジューリング問題にラグランジュ分解・調整法を適用し、現実的な生産ライン負荷の状況下にて計算機実験を行った。その結果、従来手法と比較して計算時間は多少増加するが実時間内での計算が可能であることを確認、Table 2.1 に示す将来数ヶ月の生産計画に対して装置や金型の組合せによる過不足を検証する業務や、日々の作業日程計画作成業務における実用化に大きく近付いた。

しかしながら、数秒、できれば 1 秒以内での判断が必要となるディスパッチング業務への適用には、更なる計算時間短縮が必要となる。この計算時間は、解精度に影響する協調回数の規定回数に依存しているが、この規定回数を様々なケースにおいて一律的に最適に決定することは困難であり、最適化計算の進捗状況により動的に決定することが望ましいことも確認した。

## 6 結論

### 6.1 結論

本論文では、半導体製造の生産スケジューリング業務において、これまでに唯一実用化されたと言えるディスパッチングルールを用いたボトルネック装置など一部装置における最適化だけでは解決できない、生産ライン全体の効率化を目的として、現実の生産ラインで実用化可能な生産スケジューリング手法についての研究を行った。

第2章では、半導体製造を構成する3つに大別される製造工程（前工程、組立工程、テスト工程）それぞれの特徴を整理し生産スケジューリングにおける課題を明確にした。それらの解決策の中で未だ解決困難で先行研究の少ない大規模モデルにおける実用化に向けての解決策として、製造に限らず様々な分野の最適化問題での研究が盛ん、且つ、現実生産ライン規模の大規模モデルでの最適化に対して可能性を見出した分散協調型スケジューリング手法の一つであるラグランジュ分解・調整法の適用を提案した。

第3章では、第2章で提案したスケジューリング手法につき、リエントラントを特徴とする半導体前工程生産ラインを抽象化した小規模モデルにおいて、提案したスケジューリング手法を用いて効率の良い生産スケジュールが作成できること、および、従来手法のディスパッチングルールと比較して有効であることを確認した。その中で、半導体製造の特徴でもある代替装置が多いことが最適化計算において解決すべき大きな課題になること、課題解決のためにはアルゴリズムの改善が有効であるが、DPによる部分問題解法のアルゴリズム改善はメリットよりも計算時間悪化のデメリットの方が大きいこと、一般には最適とされないリストスケジューリングによる実行可能スケジュール作成のアルゴリズム改善により、高速且つ精度の高い解を算出できることを確認した。また、初期パラメータが解精度ならびに計算時間に大きく影響していること、ラグランジュ乗数の更新には、単調非減少法の方が、劣勾配法よりも比較的高速に解法できることも確認した。

第4章では、第3章の小規模モデルにおいて有効性を確認した提案手法を、半導体前工程を抽象化した大規模モデルへ適用、第3章で提案した解精度の改善に効果的なリストスケジューリングのアルゴリズムに更に改善を加えることで、現実生産ライン規模の大規模モデルにおいても提案手法が有効であることを数値実験により確認した。また実用化に向けての大きな課題となる計算時間の内訳を分析、計算時間短縮のための方向性を示した。

第5章では、第4章までに確立した大規模モデルにおける生産スケジューリングの最適化手法を、生産能力の大きな装置を複数品種が共用し、段取り時間を必要とする品種切り替えが頻繁に行われることを特徴とする現実の半導体後工程生産ラインに適用し従来手法と比較して有効であること、計算時間は多少増加するが実時間内での計算が可能であることを数値実験により確認した。

以上により本論文は、分散協調型スケジューリング手法の一つであるラグランジュ分解・調整法を用いて、現実の半導体生産ラインを対象とした大規模モデルにおいて、実用

的な計算時間で生産ライン全体を効率化する生産スケジュールが作成できることを示した。

## 6.2 今後の展望

今後の展望としては以下の点が挙げられる。

### (1) 半導体テスト工程への適用および企業レベルスケジューリングへの拡大

本研究では、3つに大別される製造工程のうち、リエントラントを特徴とする前工程、品種切り替えが頻繁に行われるがリエントラントではない組立工程での有効性は確認したが、その両方の特徴を有するテスト工程の確認には及ばなかった。装置と治具の組合せの制約に加え、ある工程のテスト結果により後続の製造工程フローが異なるなどスケジューリングの複雑性が更に増すため、新たなアプローチを実施する。テスト工程への適用まで確認できれば、この3つの製造工程全てを連携した企業レベルのサプライチェーンマネジメントとの連携など更なる展開が期待できる。この実現を目指すには、大規模システムの課題解決に有効とされる社会適応型システムアプローチの一つである組合せオークションや市場指向プログラミングなどの適用が考えられる[30][31]。

### (2) 解精度の更なる向上

後工程生産ラインにおけるスケジューリングの大きな課題である品種切り替えについては、最適化手法にヒューリスティクスを組み合わせるだけでは解精度の大きな改善は望めず、セミアクティブスケジュールなど別のアプローチを実施する。また更に解精度を向上するためには、ラグランジュ乗数更新などにおける各種実験パラメータの動的な調整などを検討する必要がある。

### (3) 計算時間の更なる短縮

数秒程度での計算は可能になったが、1秒以内での応答が必要な、例えば、処理を完了したばかりの装置において次にどのジョブの処理を開始するかというディスパッチング業務において実用化するためには、更に計算時間の短縮が必要である。そのためには計算時間の90%以上を占める部分問題解法の改善が不可欠で、GPUを用い並列計算により高速化を図ることが考えられる[32]。動的計画法を用いて厳密に解法するのではなく、何らかの簡便法を用いて解精度を高める工夫を追加するようなアプローチも興味深い。また、早い段階での協調の見切りなど終了条件の見直しは解の改善の進捗度により残りの協調回数を調整するなどのアルゴリズムを工夫することで比較的容易に実現できると考える。

## 参考文献

- [1] 黒田充, 斤海洪. 半導体製造工程のスケジューリングに関する二三の考察. 生産スケジューリング・シンポジウム'95 講演論文集, pp. 261-266, 1995.
- [2] 貝原俊也, 藤井信忠, 指尾健太郎, 長谷川弘幸, 黒瀬伸二. 組合せオークションを用いた最適化手法に関する研究-リエントラントフローショップにおける資源割当問題への適用-. 第 52 回システム制御情報学会研究発表講演論文集, pp. 497-498, 2008.
- [3] 三木秀敏, 黒瀬伸二, 乾口雅弘. 半導体製造工程における待ち時間制御に関する研究. 第 49 回自動制御連合講演論文集, pp.SU8-2-5 1-2, 2006.
- [4] Shari L. Murray de Diaz, John W. Fowler, Michele E. Pfund, Gerald T. Mackulak, and Matt Hickie. Evaluating the Impacts of Reticle Requirements in Semiconductor Wafer Fabrication. *IEEE Transactions on Semiconductor Manufacturing*, Vol. 18, No.4, pp.622-632, 2005.
- [5] 大鷹宏, 乾口雅弘, 鶴見雅代, 黒瀬伸二, 田地 宏一. 半導体加工におけるネック工程の効率化に関する研究. 第 48 回システム制御情報学会研究発表講演論文集, pp.293-294, 2004.
- [6] 村上嘉浩, 黒瀬伸二, 乾口雅弘. 半導体生産における汎用装置の長期故障発生時の効率的運用法. 第 51 回システム制御情報学会研究発表講演論文集, pp.209-210, 2007.
- [7] Y. Shiraki, Y. Murakami, M. Inuiguchi, K. Masunaga, S. Kurose. On Exclusive Usage of General-Purpose Machines for Semiconductor Manufacturing. *Proceedings of the 6th International Conference on Modeling Decisions for Artificial Intelligence*, pp.117-128, 2009.
- [8] Qi Kang, Hui Xiao, Lei Wang, Qidi Wu. A Swarm-Dynamic Scheduling Method for Semiconductor Assembly Production Line. *IEEE International Conference on Automation Science and Engineering*, pp.91-96, 2008.
- [9] Yeong-Dae Kim, Jae-Hun Kang, Gyeong-Eun Lee, Seung-Kil Lim. Scheduling Algorithms for Minimizing Tardiness of Orders at the Burn-in Workstation in a Semiconductor Manufacturing System. *IEEE Transactions on Semiconductor Manufacturing*, Vol. 24, No. 1, pp.14-26, 2011.
- [10] Chung-Yang Liu and Shi-Chung Chang. Scheduling Flexible Flow Shops with Sequence-Dependent Setup Effects. *IEEE Transactions on Robotics and Automation*, Vol. 16, No. 4, pp.408-419, 2000.
- [11] Chih-Chiang Chen, Yu-Shian Chiang, and Chen-Fu Chien. Real Option Analysis for Capacity Investment Planning for Semiconductor Manufacturing. *Proceedings of the International Symposium on Semiconductor Manufacturing*, pp.305-307, 2007.

- [12] 田地宏一, 小山裕, 岩田義雄. リエントラントフローショップ問題に対する局所探索法によるスケジューリング. システム制御情報学会論文誌, Vol. 17, No. 6, pp. 249–256, 2004.
- [13] 能勢和夫, 平松綾子, 小西正躬. 繰り返し工程を有するジョブショップスケジューリング問題への遺伝的アルゴリズム. 電子情報通信学会技術研究報告, Vol. 99, No. 96, pp. 1–8, 1999.
- [14] J.S.Chen, J.C.H.Pan, C.M.Lin. A hybrid genetic algorithm for the re-entrant flow shop scheduling problem. *Expert Systems with Applications*, Vol. 34, pp. 570–577, 2008.
- [15] A. Ono, S. Kitamura, K. Mori. Optimal load allocation method for keeping target TAT - application of Nonlinear Programming -. *Proceedings of the International Symposium on Semiconductor Manufacturing*, pp.159-162, 2008.
- [16] Duffie NA, Shi L. Maintaining Constant WIP-Regulation Dynamics in Production Networks with Autonomous Work Systems. *CIRP Annals – Manufacturing Technology* Vol. 58, Issue 1, pp.399–402, 2009.
- [17] 今泉淳. ラグランジュ分解・調整法による生産スケジューリング: 近年の研究の流れと将来の展望に関する考察. *経営論集*, Vol. 54, pp.1–19, 2001.
- [18] 米田清. ラグランジュ緩和法によるスケジューリング. *システム/制御/情報*, Vol. 41, No. 4, pp.130–138, 1997.
- [19] P.B. Luh, D.J. Hoiomt, E. Max and K.R. Pattipati. Schedule generation and reconfiguration for parallel machines. *IEEE Transactions on Robotics and Automation*, Vol. 6, No. 6, pp. 687–696, 1990.
- [20] P.B. Luh and D.J. Hoiomt. Scheduling of Manufacturing Systems Using the Lagrangian Relaxation Technique. *IEEE Transactions on Automatic Control*, Vol. 38, No.7, pp.1066-1079, 1993.
- [21] D.J. Hoiomt, P.B. Luh, E. Max and K.R. Pattipati. A practical approach to job-shop scheduling problems. *IEEE Transactions on Robotics and Automation*, Vol. 9, No. 1, pp.1-13, 1993.
- [22] T.Nishi, Y.Hiranaka and M.Inuiguchi. A Successive Lagrangian Relaxation Method for Solving Flowshop Scheduling Problems with Total Weighted Tardiness. *Proceedings of the 3rd Annual IEEE Conference on Automation Science and Engineering*, pp.875-880, 2007.
- [23] S.Jiang and L.Tang. Lagrangian relaxation algorithms for re-entrant hybrid flowshop scheduling. *Proceedings of the International Conference on Information Management, Innovation Management and Industrial Engineering (ICIII 2008)*, Vol. 1, pp.78-81, 2008.

- [24] 長谷川弘幸, 黒瀬伸二, 貝原俊也. ラグランジュ分解・調整法による生産スケジューリング手法-リエントラントフレキシブルフローショップ問題への適用. 第 52 回自動制御連合会講演論文集, pp. A1-3 1-6, 2009.
- [25] 黒田充. ラグランジュ分解・調整法と動的スケジューリング. オペレーションズ・リサーチ: 経営の科学, Vol. 45, No. 6, pp. 263-269, 2000.
- [26] 貝原俊也, 藤井信忠, 西林聡, 藤井進, 梅田豊裕. ロット編成問題を有するフレキシブル・フローショップを対象とした分散協調型スケジューリング手法の一提案. 日本機械学会論文集. C 編, Vol. 76, No. 768, pp. 2127-2134, 2010.
- [27] 八尾佳宏, 貝原俊也, 藤井信忠, 動的環境下における部門間生産スケジュールの最適化についての研究. 日本機械学会論文集. C 編, Vol.76, No.772, pp.3198-3205, 2010
- [28] 貝原俊也, 藤井信忠, 野中洋一, 蔵野嵩子. リエントラントフローショップにおけるメンテナンス計画に関する研究 (第 1 報) : ラグランジュ分解・調整法の適用. 精密工学会誌, Vol. 76, No. 4, pp. 468-473, 2010.
- [29] ワディティア・ワルマン, 松村健児. 段取時間のある多品目多段工程動的ロットサイズスケジューリング: ラグランジュ分解調整法. 日本経営工学会論文誌, Vol. 53, No. 5, pp. 385-396, 2002.
- [30] 貝原俊也. 社会指向型フレキシブル・オートメーション: 人とシステムの調和に基づくモノづくりを目指して([総合特集] フレキシブル・オートメーション). システム/制御/情報: システム制御情報学会誌, Vol. 55, No. 2, pp. 48-53, 2011.
- [31] 藤井信忠, 貝原俊也. 大規模生産を対象とした社会適応型システムによる問題解決アプローチ(特集大規模生産のためのシステム技術). 計測と制御, Vol. 50, No. 7, pp.444-449, 2011.
- [32] 今西啓太. GPU を用いた並列計算処理による分散協調型最適化手法の高速化. 神戸大学大学院修士論文, 2012.



## 本論文の構成

本論文の第3章は以下の論文からなる。

- 黒瀬伸二, 貝原俊也, 藤井信忠. 半導体製造における協調型スケジューリング手法の一提案. 日本機械学会論文集. C 編, Vol. 76, No. 772, pp. 3206–3211, 2010
- Kurose, S., Kaihara, T., Fujii, N., A Study on Cooperative Scheduling Method for Semiconductor Manufacturing, Proceedings of 2010 International Symposium on Flexible Automation, JPS2509(CD-ROM), 2010

本論文の第4章は以下の論文からなる。

- Kurose, S., Kaihara, T., Fujii, N., A Proposal of Optimization Methodology on Manufacturing Scheduling and its Application into Actual Scale Problem in Semiconductor Manufacturing, Proceedings of 8th IEEE International Conference on Mechatronics and Automation, pp. 732-737, 2011
- Kaihara, T., Kurose, S., Fujii, N., A Proposal on Optimized Scheduling Methodology and its Application to an Actual-scale Semiconductor Manufacturing Problem, CIRP Annals - Manufacturing Technology, Vol.61, Issue 1, pp.467-470, 2012

本論文の第5章は以下の論文からなる。

- 黒瀬伸二, 貝原俊也, 藤井信忠. 半導体製造におけるスケジューリングの最適化および組立ラインへの適用. 日本機械学会論文集. C 編, 2012 (掲載予定)



## 謝辞

本研究論文の執筆にあたり、終始多大なるご指導、ご鞭撻をいただきました貝原俊也教授に心より御礼申し上げます。特に研究途中で会社事情により中国（北京）へ転勤になるなど、社会人学生であるがための勝手な都合にも懇切丁寧にご対応いただきましたこと、重ねて御礼申し上げます。また研究の遂行にあたり、数々の有益なご助言とご指導をいただきました藤井信忠准教授に厚く御礼を申し上げます。またお忙しい中、本論文の副査をお引受けいただき、有意義なご助言と細部にわたるご指導を賜りました多田幸生教授、白瀬敬一教授に厚く御礼を申し上げます。

本研究の遂行にあたり、共同研究として本研究の礎となる様々な課題解決に向け共に議論させていただきました指尾健太郎氏、長谷川弘幸氏、今西啓太氏に深く感謝いたします。また研究の目指す方向と手法について共に議論し、研究遂行の支援をしていただきました益永健一郎氏に深く感謝いたします。同じ博士後期課程の学生として、共に研究に勤しみましたOpadiji Jaiola Femi 氏、松本真基也氏、八尾佳宏氏、銭毅氏に心から感謝いたします。研究室に配属されて以来、非常に多くの方々から温かいご支援を賜りました。本研究室の卒業生・在学生の皆様にも心から御礼申し上げます。

入社以来、研究者としての心構えをご指導いただきました岩崎武正氏、石川勝彦氏、前田和彦氏に深く感謝いたします。

最後に本論文の執筆にあたり、家族の理解と協力、そして励ましが大きな支えとなりました。この場を借りて心から感謝の意を表したいと思います。本当にありがとうございました。