生産プロジェクトにおけるリスケジューリング手法の提案及び適用

○青松祐亮　Chang Shuang　出口弘　(東京工業大学)

Proposing a Rescheduling Method in Production Project and its Application

∗Y. Aomatsu, S. Chang and H. Deguchi (Tokyo Institute of Technology)

概要– 工場の現場において，機械故障や新たな特急品の発生などにより，当初予定していた生産プロジェクトのスケジュールが変化してしまうことが頻繁に起こる．こうしたトラブルが発生した時、山積み・山崩しといった手法を用いて再度スケジュールを組むケースがあるが，経験と勘に頼っており，並列したプロジェクトを再構築するには非常に手間がかかっている．本研究では，新たなリスケジューリングモデルを提案し，より実時間に近いかたちで現場改善を行うシステムを分析する．

キーワード: 生産現場, リスケジューリング, 特急品, ディスパッチングルール

1　はじめに

近年，工場における工程のスケジューリングは市場の著しい変化に対応するため，複雑な意思決定を必要としており，どの工程においても高品質な製品を生産するために，多品種少量生産のための工程が組まれている．従来スケジューリング問題は OR およびコンピュータサイエンスの分野における基礎的な研究課題として位置づけられており，今後社会インフラを支える技術として発展していくものと予想されている [1)]．

最も一般的なスケジューリング手法としてジョブショップ・スケジューリング問題 (以下 JSP とする) が挙げられる．JSP とは，各々のタスクが 2 台以上の機械にあらかじめ指定された順序で処理される場合に，機械の全体稼働時間の最適処理順序を決定する問題となっている．JSP は主に最適解を求める立場から研究され，遺伝的アルゴリズムを含めた局所探索法 [2)] や分岐限界法 [3)] などの手法が提案されている．しかし，JSP は 1 つの機械が複数のタスクを割り当てることができず，半順序関係のある製品工程のプロジェクトの処理が行えないことから，1 製品の生産工程が煩雑化すると，最適解を導き出すことが困難となる．

この問題に対し，出口らはタスクの集合から構成される複数のプロジェクトを並列に処理し，数理的に定式化された資源寄り付き型の動的スケジューリングを提案した [4)]．資源が，プロジェクトを構成するタスクに動的に割付くアルゴリズムのため，JSP とは違った新たなタイプのスケジューリング問題となっている．この手法を用いることにより半順序構成のプロジェクトや 1 つの機械に複数のタスクを割当てることが可能となる．

実際に橋本の研究 [5)] で本手法が用いられ，住戸数が 333 戸，39 職種の作業員 150 名の集合住宅の内装工事の工期が 202 日だったのに対し，スケジューリングを行った結果 166 日となり，大幅な改善が見込まれた．ここでのプロジェクト構成は半順序関係となっているため，JSP では行えないことから資源寄り付き型の動的スケジューリング手法に優位性があることがわかる．

しかし，生産現場においてはスケジューリング結果からガントチャートを作成しても，生産実施段階で遅延が発生することや，機械故障や特急品の投入により工数が伸びるような事態が生じることが頻繁にある [6)]．現場でこうした問題があった時は経験と勘により山積み・山崩しなどの手作業により再度スケジュールを組む現状となっている．したがって，ある時点で問題が発生した時に着手可能なタスクを動的に計算し，割当可能な資源を逐次割り当てるスケジューリング手法が求められている．本研究では，従来研究として扱われているリスケジューリング手法とは異なる新たなモデルを提案し，より実時間に近いかたちで現場改善を行うシステムを分析する．実際の工場現場の工程表をもとにリスケジューリングの発生事案として特急品の発生に注目し，リスケジューリング後の割り付けタスクの変化およびディスパッチングルールを変更した時の挙動を考察する．また，特急品とは，生産計画実施段階において発生する，他の製造工程よりも優先して生産しなければいけない製品を表す．

2　関連研究との位置づけ

2.1　リアクティブスケジューリング

リアクティブスケジューリングとは，最初の準備段階でリスケジュールの作成を前提とし，生産状況に応じて修正のタイミングやスケジュールの決定を行う仕組み [7)8)] であることを表す．リアクティブスケジューリングにおいて考えられる問題として，リスケジューリングのタイミング決定問題と，どのようなスケジュールに更新するかの問題が挙げられる．この最適な実施時期や更新スケジュールを理論的に決定するための数理モデルの構築に関して様々な研究が行われており，多くが遺伝的アルゴリズムを用いた最適手法となっている．

坂口らの研究 [9)] では，1 週間程度のスケジュールにおけるジョブの追加を考慮したリアクティブスケジューリングシステムのプロトタイプを開発している．具体的には，スケジューリング区間を，機械の状況により動かすことのできない固定区間，遺伝的アルゴリズムにより区間の長さを実験的に決定するリアクティブスケジューリング区間，定期的にディスパッチングルールに基いてスケジュールの変更を行うローリングスケジューリング区間から成り立つ．評価基準として納期遅れの最小化を取り扱っており，リアクティブスケジューリングの区間で適切なスケジュールの変更を行い，効果的にスケジュールの修正ができるスケジューリング手法を提案している．この研究では，ジョブの追加は定期的に投入されることを想定しており，ローリングス

ケジューリング段階であらかじめ決められた時刻 t の時点でジョブが追加された場合に実行されるため，中長期的なメイクスパンにおける突発的な事象に即座に対応することが困難だと考えられる．

2.2 プロダクションルールによるリスケジューリング

植野らは工程作業の途中で着手不可能となった場合に，リアルタイム性を考慮したリスケジューリングのアルゴリズムを提案した [10]．この研究ではリスケジューリングにプロダクションルールを用いており，生じる可能性のある支障を数値化したものを前件部とし，後件部に解決策となり得る事柄を数値化し，マッチングを行って処理をしている．このプロダクションルールを用いたリスケジューリングの結果，突発的な支障が生じた場合の本来の基本スケジューリングへの復帰が可能となっている．ここでの基本スケジューリングのディスパッチングルールとして，各工程の製造リードタイムの長い工程を優先しているのは本研究と同様であるが，人的資源の作業負荷の平準化を評価している点で異なり，本研究では物的資源の配置問題を軸としている．

3 方法

本章では，生産プロジェクトにおけるリスケジューリングの手法について述べる．まずベースとして扱う資源寄り付き型動的スケジューリングのアルゴリズムに関して説明し，その後に提案手法となるリスケジューリングに関して論述する．

3.1 資源寄り付き型動的スケジューリング

Table 1: スケジューリングフロー

Step1	着手可能タスク集合の更新
Step2	着手可能タスク集合への割り当て
Step3	現在時間の更新
Step4	終了タスク集合の更新
Step5	終了判定

生産計画段階では，出口らにより提案された資源寄り付き型動的スケジューリングのアルゴリズムに基づき作成する．スケジューリングに関する大まかなフローを Table 1 に示す．

Table 1 について，Step1 では未終了タスク集合から着手可能タスク集合を洗い出し，Step2 でディスパッチングルールを適用して，未割当資源集合から着手可能タスク集合に資源を割当てる．ここで割当てられた資源は割当済みとなり，割当てられたタスクは仕掛りタスク集合へ遷移する．Step3 では仕掛りタスク集合の中で最も工数の小さいタスクの工数を，現在時間から足し合わせる．Step4 で仕掛りタスク集合から終了したタスクは終了タスク集合へ遷移し，割当てられていた資源を解放する．最後に Step5 で未完了タスク集合が空になっていた場合に，スケジューリングを終了し，空でなければ再度 Step1 を行う順序となっている．

3.2 詳細設計

3.2.1 事例の定式化

本研究の方法論に関して説明する際に扱う事例として，Fig.1 のような半順序構成の生産プロセスを考える．本モデルは出口らの研究の構想を参考に構築されている [4]．スケジューリング期間を [0,T] とし，この事例をもとにスケジューリング問題を定式化する．

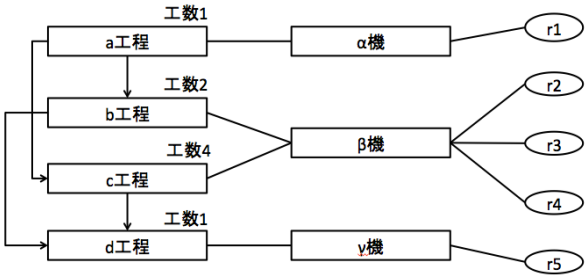

Fig. 1: モデルケース

本モデルのプロジェクト数を $m \in N$ とした時，プロジェクト集合を $ProjS = \{p1, p2, \cdots, pm\}$ と定義する．1 プロジェクトはロット単位の個数分を生産するものとしており，本モデルでは 5 つのプロジェクトと想定する．また，タスク集合を $TaskS = \{a$ 工程$, b$ 工程$, c$ 工程$, d$ 工程 $\}$ とし，TaskS[x] をプロジェクト数が x 個あるタスクの集合とすると，プロジェクトごとのタスク集合は $TaskS = \cup_{x \in ProjS} TaskS[x]$ と与えられる．ここでタスクを遂行するのに必要となる機械のタイプを職能と呼び，職能集合を $ProfS = \{\ \alpha$ 機 $, \beta$ 機$, \gamma$ 機 $\}$ と定義した時，タスクと職能の関係を，$Task\_Profession \subseteq TaskS \times ProfS$ と表す．

次に機械の資源集合を $ResS = \{r1, r2, r3, r4, r5\}$ と定義する．このとき，資源と職能の関係を，$Profession\_Resource \subseteq ProfSet \times ResSet$ と表し，ある資源 $x \in ResS$ に対する職種 $rt(x) \in ProfS$ を割り当てる関数は，$rt : ResS \rightarrow ProfS$ となる．また，ある職種 $y \in ProfS$ に対する資源の数は $rn(y) = |\{x \in ResS | rt(x) = y\}|$ となる．

3.2.2 実行段階

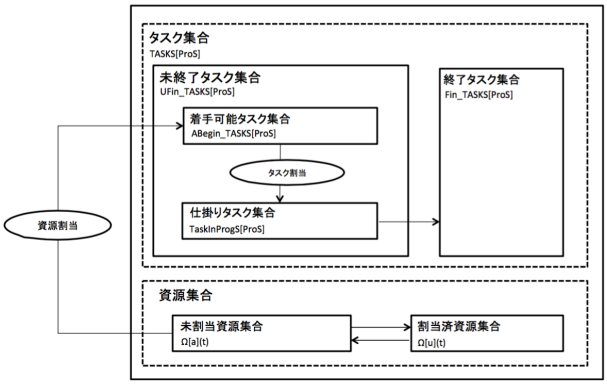

Fig. 2: スケジューリングの全体像

Fig.2 のように，タスク集合は未終了タスク集合，着手可能タスク集合，仕掛りタスク集合，終了タスク集合に分類し，資源集合は未割当資源集合と割当済資源集合に分類する．未終了タスク集合から着手可能なタスク集合を洗い出し，着手可能タスク集合の中から実行可能な職能を持つ資源を，割当可能資源集合の中から割り当てる．また，割当可能な資源の少ない順から割り当てられ，割り当てるタスクが複数ある場合はタスク工数の長いものを優先し，複数のタスクの中で同

Table 2: 資源割付詳細表

PT#	プロジェクト名	タスクパス名	開始許可時間	タスク名	工数	職能制約	資源コード	資源名	職能	開始時間
1	p1	path	0	a 工程	2		10001	r1	α 機	0
2	p1	path	0	b 工程	2		10003	r3	β 機	2
3	p1	path	0	c 工程	4		10002	r2	β 機	2
4	p1	path	0	d 工程	1		10005	r5	γ 機	6
5	p2	path	0	a 工程	2		10001	r1	α 機	2
⋮	⋮	⋮	⋮	⋮	⋮		⋮	⋮	⋮	⋮
18	p5	path	0	b 工程	2		10004	r4	β 機	10
19	p5	path	0	c 工程	4		10002	r2	β 機	10
20	p5	path	0	d 工程	1		10005	r5	γ 機	14

Table 3: プロジェクトの割付表

PT#	プロジェクト名	タスク名	0	1	2	3	⋯	11	12	13	14
1	p1	a 工程	a 工程 [p1]/r1	a 工程 [p1]/r1							
2	p1	b 工程			b 工程 [p1]/r3	b 工程 [p1]/r3					
3	p1	c 工程			c 工程 [p1]/r2	c 工程 [p1]/r2	⋯				
4	p1	d 工程					⋯				
5	p2	a 工程			a 工程 [p2]/r1	a 工程 [p2]/r1					
⋮	⋮	⋮					⋮	⋮	⋮		
18	p5	b 工程					⋯	b 工程 [p5]/r4			
19	p5	c 工程					⋯	c 工程 [p5]/r3	c 工程 [p5]/r3	c 工程 [p5]/r3	
20	p5	d 工程									d 工程 [p5]/r5

じ資源があった場合は単能工を優先する．ここで単能工は，一対一対応となる職能を示し，多能工は一対一対応とならないものと定義する．本モデルの場合，α 機と γ 機が単能工となり，β 機は多能工として扱われる．

ここで，本モデルを用いて具体的に着手可能タスク集合を計算する．まず未終了タスク集合から着手可能なタスク集合の洗い出しを行う．

$$a \text{ 工程} = \{r1\},\ b \text{ 工程} = \{r2, r3, r4\}$$
$$c \text{ 工程} = \{r2, r3, r4\},\ d \text{ 工程} = \{r5\}$$
$$\Omega(a)(0) = \{r1, r2, r3, r4, r5\}$$
$$ABegin\_TASKS[ProjS] = \{a \text{ 工程 } [ProjS]\}$$
$$AssignableResource[ProjS](a \text{ 工程 } [ProjS]) = \{r1\}$$

$\Omega(a)(t)$ は時間 t における割当可能な資源集合を表している．この時，割当可能な資源が r1 のみであるため，p1 の a 工程 [p1] に r1 が割当てられる．a 工程は 2step かかるので，2step 経過した時点で a 工程 r1 の資源は解放され，a 工程 [p1] が終了タスク集合に遷移する．次に割当可能となるのは b 工程 [p1], c 工程 [p3], a 工程 [p2], ⋯, a 工程 [p5] となり同様に割り付けを行う．このフローを繰り返し，最終的に 15step でスケジューリングが終了する．

実際のスケジューリングの計算は，研究室で開発されたデータ加工ツールである Falconseed[11] に組み込んだ動的スケジューリングのモジュールを用いている．Falconseed で実行する際には，実行に必要となるプロジェクトに関する情報を入力ファイルとして指定し，結果は csv 形式で出力される．スケジューリング結果には，各タスクへの資源割付詳細表，プロジェクトの時間ごとの割付表，資源の時間ごとの割付表が出力される．この出力結果を Table 2, Table 3, Table 4 に示す．生産スケジューリング内で資源の割り付けがなかったタスクでは，資源コード，資源名，職能，開始時間は空欄となっている．

3.3 リスケジューリング - 特急品発生時

先程作成したケースをもとに，7step 経過途中にある特急品のプロジェクトが 3 つ発生した場合のリスケジューリングを考える．この時のスケジューリング区間をリスケジューリング前の [0, 8) と，リスケジューリング後の [8,T] に分ける．8step の時点で終了していない集合を仕掛りプロジェクト集合とし，タスク集合から 8step の時点での終了タスク集合と仕掛りタスク集合を引いたものとする．まず，特急品プロジェクトに関して必要な情報を定義する．

$$CutInProS = \{CIp1, CIp2, CIp3\}$$
$$CutInTASKS = \{CIa \text{ 工程}, CIb \text{ 工程}, CIc \text{ 工程 }\}$$
$$CutInPath = \{V, E\}$$
$$E \in V \times V$$
$$V = CutInTASKS$$
$$E = \{(CIa \text{ 工程}, CIc \text{ 工程}), (CIb \text{ 工程}, CIc \text{ 工程})\}$$

また，それぞれのタスクに対する職能や工程数を以下に定義する．

$$ProfAssignF : TASKS \to Profession$$
$$ProfAssignF(CIa \text{ 工程}) = \alpha \text{ 機}$$
$$ProfAssignF(CIb \text{ 工程}) = \beta \text{ 機}$$
$$ProfAssignF(CIc \text{ 工程}) = \gamma \text{ 機}$$
$$EstTimeAccomodationF : TASKS \to Time$$
$$EstTimeAccomF(CIa \text{ 工程}) = 1$$
$$EstTimeAccomF(CIb \text{ 工程}) = 2$$
$$EstTimeAccomF(CIc \text{ 工程}) = 1$$

タスクと職能の関係と，職能と資源の関係は 4.2.1 節で与えられた数式と同様になる．仕掛りプロジェクト集合と特急品プロジェクト集合は同時進行で進め，割付可能タスク集合がどちらの集合にも含まれていた場合は，特急品プロジェクトを優先する．つまり，未割当資源の中で特急品プロジェクト集合内で割付可能な資源がなければ，次に仕掛りプロジェクト集合から探し出す状態となる．資源の割り付けに関するフローチャートを Fig.3 に示す．また，特急品が発生した時点で仕掛りタスク集合に存在しているタスクは，作業が完了するまでは資源の解放ができないので，仕掛り中のタスク終了後に資源の割り付けを特急品優先に切り替える．

Table 4: 資源の割付表

資源コード	資源名	PT#	プロジェクト名	0	1	2	3	···	11	12	13	14
10001	r1	1	p1	a 工程 [p1]	a 工程 [p1]							
10001	r1	5	p2			a 工程 [p2]	a 工程 [p2]					
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
10002	r2	3	p1			c 工程 [p1]						
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
10005	r5	16	p4							d 工程 [p4]		
10005	r5	20	p5									d 工程 [p5]

Table 5: リスケジューリング後のプロジェクトごとの割付表

PT#	プロジェクト名	タスク名	0	1	2	···	8	9	10	···	17
1	p1	a 工程	a 工程 [p1]/r1	a 工程 [p1]/r1							
2	p1	b 工程			b 工程 [p1]/r3	···					
3	p1	c 工程			c 工程 [p1]/r2	···					
⋮	⋮	⋮									
8	p2	d 工程					d 工程 [p2]/r5				
⋮	⋮	⋮									
11	p3	c 工程				···	c 工程 [p3]/r2	c 工程 [p3]/r2			
⋮	⋮	⋮									
20	p5	d 工程									d 工程 [p5]/r5
21	CIp1	CIa 工程					CIa 工程 [CIp1]/r1				
22	CIp1	CIc 工程							CIc 工程 [CIp1]/r5		
23	CIp1	CIb 工程					CIb 工程 [CIp1]/r3	CIb 工程 [CIp1]/r3			
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

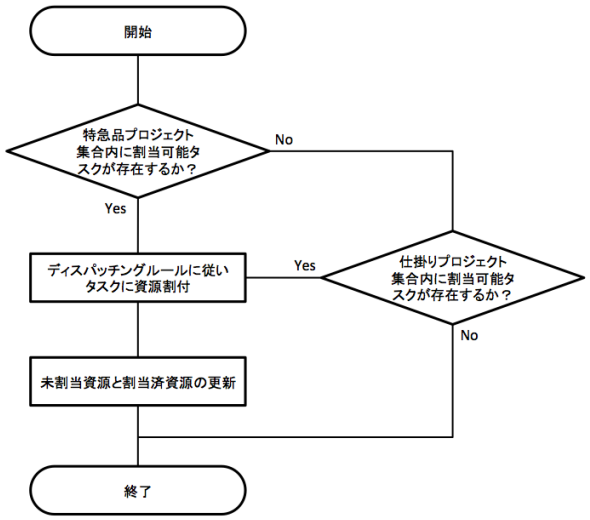

Fig. 3: 資源割当のフローチャート

リスケジューリングを計算した時のプロジェクトごとの割付結果を Table 5 に示す．8step 以降に特急品プロジェクトが開始されている．また，8step の時点で特急品の割付可能なタスク集合の中に r5 の資源はないので，次に仕掛りプロジェクト集合の中から p2 の d 工程が割付可能であることがわかり，割り付けられている．さらに p3 の c 工程は 6step の時点から割当てられているタスクのため，工数分進んで終了タスク集合に遷移するまでは r2 が割り付けられている．

3.4 リスケジューリング - 機械故障時

機械故障時においては，資源割付可能時間を定義したテーブルを作成し，このテーブルに基づいて資源の割付時間を制御する．これにより通常とは異なった資源の制約を加えることができ，機械故障時の対応を容易に行えるようになる．例えば，Table 6 のようにテーブルを作成する．

Table 6 では，r2 は 7 時間目以降使うことができなくなり，r3 は 5 時間目以降利用可能となったことを示

Table 6: 資源割付可能時間定義テーブル

資源コード	割付可能時間	工数判定
10002	-7	true
10003	5-	

している．また，工数判定の項目では 7step の時点で資源が割り付けられている場合は，そのタスクが終了するまで割り付けるかどうかの判定を行っている．工数判定はデフォルトでは false としており，この例では r2 を true にしているので，7step を超えるタスクは割当てられないようになっている．したがって，この定義テーブルを用いて機械故障時の対処は容易に行うことが可能となる．

3.5 ディスパッチングルール

ディスパッチングルールとは，スケジューリングにおいて競合したジョブがあった場合に適応させる基準を示したものであり，本研究で扱うディスパッチングルールに関しては，タスクが資源に割り当てるものとは異なるため本来の定義とは異なるが，着手可能タスク集合の中で必要な職能が競合した場合にルール付けしている．

デフォルトでは工数の長いタスクを優先しており，その他のディスパッチングルールとして工数の短いタスクの優先，プロジェクト単位でのワークフロー優先の 2 つを実装している．さらに割付可能タスクからランダムに割当処理させるプログラムを実装しており，これらのディスパッチングルールに関しては 6 章で取り扱う．

3.6 機械の稼働率

評価指標として機械の稼働率を扱う．ここでの稼働率は全体のリードタイムに対して稼動している割合と定義し，計画段階でのスケジューリングとリスケジューリング実行時の稼働率を求め，実際にどの職能がどれだけ働いているのかを計る．その稼働率の指標を OR(operating rate per profession) とした時，以下のように表せる．

ALL_{time} : スケジューリング全体のリードタイム

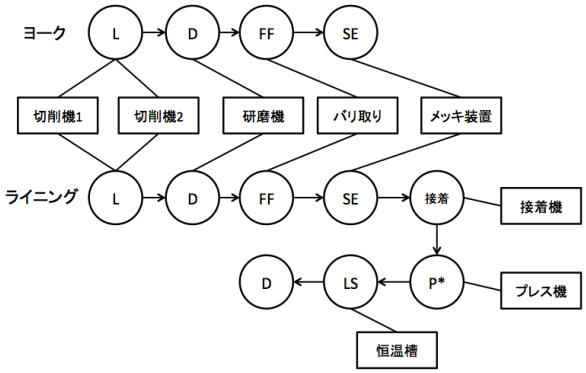

Fig. 4: ヨークとライニングのタスク職能関係

RO_{time} : 機械が実際に作業している時間
OR : 稼働率の値
と置くと，
$OR = \frac{RO_{time}}{ALL_{time}}$
となる．OR が高いほどスケジューリング期間内での稼働時間の長い資源であることがわかり，資源数を増やすことで全体工数を短縮できる可能性があるものとして期待される．本研究ではこの指標を用いてスケジューリング前と後での変化を考察する．

4 モデルの適用

ある工場における現場でのヨークとライニングの作業工程表を参考にし，作成したスケジュール計画から生産計画実施段階で特急品が発生した時のリスケジューリング手法を適用した．適用するまでの過程を本章で述べる．

4.1 作業データ

スケジューリングを実行するために必要なデータを，タスク，パス，職能，資源の 5 つに分けて整理する．ここで，各々のタスクを処理するための資源は，複数のタスクを処理できる多能工的能力を持つと仮定し，この能力を職能と定義している．ヨークとライニングに関するタスクと職能の関係を Fig.4 に示す．

4.1.1 タスク情報

工程に関する情報を Table 7 に示す．それぞれのタスクに対しての必要な工数と職能を示しており，L_y と L_l は複数の職能を用いなければならないことを表している．また，工数は 1 に対して 1 時間に換算した値としている．

Table 7: タスクテーブル

タスク名	工数	職能
L_y	5	切削機 1, 切削機 2
D_y	1	研磨機
FF_y	1	バリ取り
SE_y	3	メッキ装置
L_l	5	切削機 1, 切削機 2
⋮	⋮	⋮
LS	2	恒温槽
$D2_l$	1	研磨機

4.1.2 パス情報

ヨークとライニングのそれぞれの工程順序を Table 8 に示す．今回必要としているパスはヨークとライニング以外にないので，yoke と lining のみ設定している．タスク 1 とタスク 2 は順序関係を示しており，ヨークの場合は $L_y \rightarrow D_y \rightarrow FF_y \rightarrow SE_y$ のパスとなり，全順序関係となっている．

Table 8: パステーブル

タスクパス名	タスク 1	タスク 2
yoke	L_y	D_y
	D_y	FF_y
	FF_y	SE_y
lining	L_l	D_l
	⋮	⋮
	P	LS
	LS	D2

4.1.3 職能情報

タスクで必要となっている職能名と存在している資源数を Table 9 に示す．割付可能数は 1 つの職能でいくつのタスクを割り付けできるかを示している．今回のケースでは 1 つの職能に対するタスクの割付可能数は 1 つなので空欄としている．

Table 9: 職能テーブル

職能名	資源数	割付可能数
切削機 1	2	
切削機 2	2	
研磨機	1	
バリ取り	1	
メッキ装置	2	
接着機	1	
プレス機	1	
恒温槽	1	

4.1.4 資源情報

資源と職能の関係を Table 10 に示す．存在している資源とそれの役割に該当する職能を記入し，資源名は職能名_番号で定義している．

Table 10: 資源テーブル

資源コード	資源名	職能名
101	切削機 1_1	切削機 1
102	切削機 1_2	切削機 1
201	切削機 2_1	切削機 2
⋮	⋮	⋮
701	プレス機_1	プレス機
801	恒温槽_1	恒温槽

4.1.5 プロジェクト情報

プロジェクトの情報では，パスを実行するためのプロジェクトを定義しており，Table 11 に示す．1 つのプロジェクトあたり 30 個の生産が行われるものとし，このケースではそれぞれ 60 個生産する計画となっている．もしプロジェクトの開始時間に指定があれば記入するが，このケースでは設定する必要がないので空欄としている．

Table 11: プロジェクトテーブル

プロジェクト名	タスクパス名	開始時間
project01	yoke	
project02	lining	
project03	yoke	
project04	lining	

4.2 スケジューリングの実行および結果

まとめた作業データをもとに，Falconseed で作成したスケジューリングのモジュールから入力ファイルを指定して実行する．

実行した結果，最初の作業を開始してから最後の作業を終えるまでに 23 時間かかった．出力される 3 つのファイルのうち，資源ごとに割り付けられているタスクのファイルを Table 12 に示す．Table 12 から，今回のケースでは接着_2 のレコードが空欄となっているため，接着工程は接着_1 のみで割り付けが満たされており，接着機_2 の資源を活用せずにディスパッチングルールに基づいたスケジューリング結果が出力された．また，ヨークとライニングのプロジェクトのそれぞれのクリティカルパスは 10 時間, 17 時間となっており，プロジェクトをそれぞれ 2 つ実行しているため，資源が効率よく割り付いて並列処理されていることがわかる．

4.3 リスケジューリング

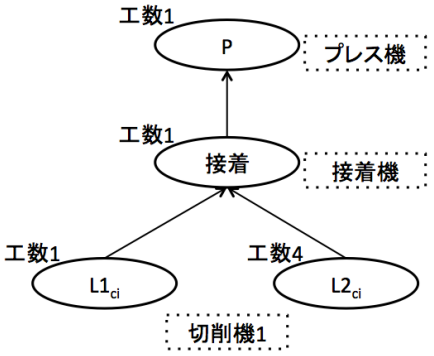

Fig. 5: 特急品プロジェクト

前節のスケジューリング作成後の実行段階で，3 時間目に 120 個の特急品が発生した場合を考える．5.1 節で作成したスケジューリングの入力データから新たに特急品を追加する．追加するデータに必要なのが，タスクと割り付けられる職能，パス情報となる．特急品に関するモデルをを Fig.5 に示す．プロジェクト数は 4 つとし，1 つのプロジェクトあたり 30 個と想定している．

データの追加後，Falconseed を用いて実装したリスケジューリングのモジュールから入力ファイルを指定し，実行する．特急品のプロジェクト名は CIproject01 から 04 とした．

5 考察

現場でのデータからリスケジューリング後のプロジェクトごとの割付結果を Table 13 に示す．切削機 1 の資源が 5 時間分すでに割当てられているため，特急品は 5 時間目からスタートし，14 時間の作業で終了となる．スケジュール全体のリードタイムは 33 時間となった．特急品発生前のスケジューリング計画と比較するとリードタイムが 10 時間伸びているが，特急品プロジェクトのクリティカルパスが 6 時間で 4 つ実行されているので比較的短い時間でスケジューリングが組まれていることがわかる．また，前章のスケジューリングで接着機_2 の資源が用いられていないことを述べたが，リスケジューリング後には接着機_2 を含め全ての資源が割り付けられている．

ここで，リスケジューリング前後のタスク間の関係およびディスパッチングルールを変更した時の影響について考察する．

5.1 特急品プロジェクトのタスクの関係

プロジェクトごとの割付表をみると，特急品が発生した後，5 時間目で切削機 1 の職能をもつ割当可能なタスクが特急品のプロジェクト以外に 2 つ存在しているが，CIproject01 のタスク $L2_{ci}$ と CIproject02 のタスク $L2_{ci}$ が優先して割当てられている．今回の特急品が発生した事例では，project01 と project02 の最初のタスクは進行しているため，この 2 つのプロジェクトは生産計画段階でのスケジューリング計画とのリードタイムに変化がなかった．project03 と project04 のプロジェクトに関しては，生産計画段階では 5 時間目の時点で $L2_{ci}$ が割当てられていたが，リスケジューリング後は 15 時間目以降となっている．これは特急品のプロジェクトの中で切削機 1 を扱う職能のタスクが全て終了するのに 15 時間目までの時間を要しているからである．

特急品プロジェクトのタスクで $L1_{ci}$ より先に $L2_{ci}$ が先行しているのは，同じ職能に対して工数の長いタスクを優先するディスパッチングルールが適用されているためであり，$L2_{ci}$ のタスクが全て終了するまでは $L1_{ci}$ のタスクが開始されないようになっている．

5.2 ディスパッチングルールの変更

リスケジューリング後に職能が競合した時のディスパッチングルールを，タスクの工数の短い順，プロジェクトのワークフロー順およびランダムな割付のそれぞれに適用した時の特急品のプロジェクトの変化について考察する．また，スケジューリング結果の表は，プロジェクトごとの割付結果を省略して表記し，特急品のプロジェクトの開始タスクと終了タスクの時間のフィールドを取り出した．

5.2.1 タスクの工数の短い順

実行した結果を Table 14 に示す．全体のリードタイムは工程が長い順の場合と同じ 33 時間となったが，特急品のプロジェクトの作業が終了する時間は 21 時間となり，3 時間短縮された．これは特急品プロジェクトのスタートで，タスク工数が 1 時間の $L1_{ci}$ と工数 4 時間の $L2_{ci}$ があり，$L1_{ci}$ を優先しているため工数の長い順よりも隙間時間をできるだけ短くなっているためである．

また，13 時間目ではタスクの工数が長い方を優先した場合は project02 の P 工程の作業が開始されているが，ディスパッチングルールを短いタスク優先にした場合は特急品のプロジェクトでプレス機_1 が用いられているため 21 時間目に入るまで project2 の作業が開始されない．これにより特急品の作業時間は短縮され

Table 12: 工場のケースでの資源ごとの割付結果

資源コード	資源名	PT#	プロジェクト名	0	1	⋯	11	12	⋯	16	17	⋯
101	切削機 1_1	4	project01	Ly[Project01]	Ly[project01]	⋯						
⋮	⋮	⋮	⋮									
201	切削機 2_1	16	project01	Ly[project01]	Ly[project01]	⋯						
⋮	⋮	⋮	⋮									
601	接着機_1	12	project02				接着 [project02]	接着 [project02]				
601	接着機_1	24	project04							接着 [project04]	接着 [project04]	
602	接着機_2											
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Table 13: リスケジューリング後のプロジェクトごとの割付表

PT#	プロジェクト名	タスク名	0	1	⋯	4	5	6	⋯	22	23	⋯
⋮	⋮	⋮										
4	project01	L_y	L_y[project01]/切削機 2_1	L_y[project01]/切削機 2_1	⋯	L_y[project01]/切削機 2_1						
4	project01	L_y	L_y[project01]/切削機 1_1	L_y[project01]/切削機 1_1	⋯	L_y[project01]/切削機 1_1						
⋮	⋮	⋮										
18	project04	FF_l					FF_l[project04]/バリ取り_1					
19	project04	SE_l						SE_l[project04]/メッキ装置_2				
⋮	⋮	⋮										
27	CIproject01	$L2_{ci}$					$L2_{ci}$[CIproject01]/切削機 1_1	$L2_{ci}$[CIproject01]/切削機 1_1	⋯			
⋮	⋮	⋮										
31	CIproject02	$L2_{ci}$					$L2_{ci}$[CIproject02]/切削機 1_1	$L2_{ci}$[CIproject02]/切削機 1_1	⋯			
⋮	⋮	⋮										
40	CIproject04	P								P[CIproject04]/プレス機_1	P[CIproject04]/プレス機_1	

たが，project02 の終了時間が 8 時間遅れてしまっている．project02 での各タスクの開始時間を最大工数タスク優先時と比較したグラフを Fig.6 に示す．

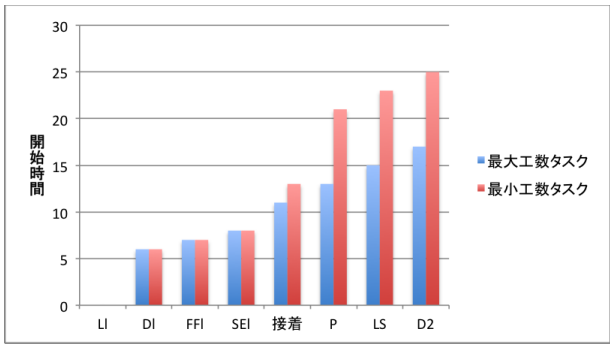

Fig. 6: project02 の最小工数タスク優先時の比較

Table 14: タスクの工数の短い順

PT#	プロジェクト名	タスク名	5	6	⋯	19	20
25	CIproject01	$L1_{ci}$	切削機 1_1				
29	CIproject02	$L1_{ci}$	切削機 1_2				
33	CIproject03	$L1_{ci}$		切削機 1_1			
37	CIproject04	$L1_{ci}$		切削機 1_2			
40	CIproject04	P				プレス機_1	プレス機_1

5.2.2 プロジェクトのワークフロー順

実行した結果を Table 15 に示す．全体のリードタイムは同様だったが，プロジェクト単位で優先して処理されるため，$L1_{ci}$ と $L2_{ci}$ は同時に進行し，特急品の 1 つのプロジェクトあたりの開始から完了までの時間が短くなっている．今回ではワークフローを優先した中で，かつプロジェクト内で職能が競合していた場合，工数の短いタスクを優先している．よって，CIproject01 のタスク L1_ci が終了したら，次に CIproject02 のタスク L1_ci が割り付けられている．また，タスクの工数の短い順の時と同様に，project02 の P 工程の作業が遅れてしまっているが，project02 における終了時間の遅れは 2 時間程度であった．このことから今回のケースではタスクの工数の短い順よりも良いディスパッチングルールとして処理されていることがわかる．project02 での各タスクの開始時間を最大工数タスク優先時と比較したグラフを Fig.7 に示す．

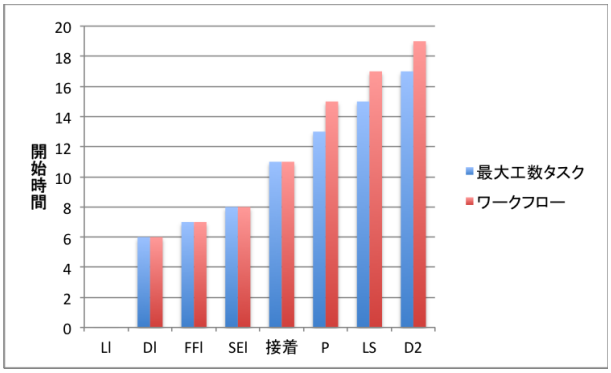

Fig. 7: project02 のワークフロー優先時の比較

Table 15: ワークフロー順

PT#	プロジェクト名	タスク名	5	6	⋯	19	20
25	CIproject01	$L1_{ci}$	切削機 1_1				
27	CIproject01	$L2_{ci}$	切削機 1_2	切削機 1_2			
29	CIproject02	$L1_{ci}$		切削機 1_1			
40	CIproject04	P				プレス機_1	プレス機_1

5.2.3 ランダム

割付可能資源集合からランダムに割り付けるルールを実行した結果を Table 16 に示す．全体のリードタイムが 3 つのディスパッチングルールよりも 6 時間短縮されたが，ランダムに割り付けられているため，特急品の CIproject01 が全体のプロジェクトの中での最後の工程となっており，タスクの割り付けルールがないのでばらついている．しかし，全体のリードタイムに対する project03 のヨーク作業は 15 時間目に終了し，他のディスパッチングルールよりも 9,10 時間短縮されている．したがって，特急品の納期に余裕があり，他の仕掛りプロジェクトにおいてワークフローの順序立てする必要がなければ，他のディスパッチングルールよりも適していると考えられる．

5.2.4 リードタイム

それぞれのディスパッチングルール適用時のリードタイムの関係について Fig.8 に示す．最小工数タスク優

Table 16: ランダム時の資源割付

PT#	プロジェクト名	タスク名	10	11	···	25	26
28	CIproject01	P				プレス機_1	プレス機_1
35	CIproject03	$L2_{ci}$	切削機 1_2	切削機 1_2			
37	CIproject03	$L1_{ci}$	切削機 1_1				
39	CIproject04	$L2_{ci}$		切削機 1_1			

先時とワークフロー優先時での全体のリードタイムと特急品の終了時間は同じだが，最小工数タスク優先時の project02 の終了時間はどのディスパッチングルールよりも 6 時間以上のロスが発生しており，更に CIproject01，CIproject02 の終了時間はワークフロー優先時よりも 2 時間長くなっている．また，ランダムで実行した場合，全体のリードタイムは 6 時間短縮されたが，特急品の終了時間が 6 時間長くなった．これらのグラフから全体のリードタイムが同じであっても各プロジェクトの終了時間にはばらつきがあることがわかる．

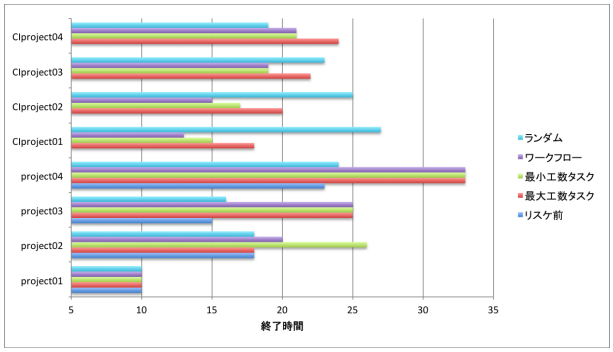

Fig. 8: 各プロジェクトの終了時間

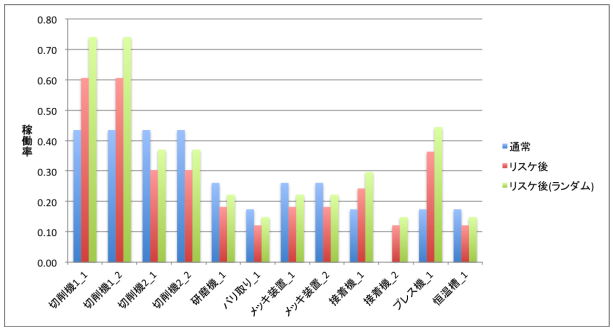

Fig. 9: 各資源の稼働率

5.2.5 資源の稼働率

計画段階でのスケジューリングとリスケジューリング後の資源ごとの稼働率について Fig.9 に示す．ディスパッチングルールについては，ランダム以外は全体のリードタイムに変化がないため 1 つに統合した．特急品の生産に必要な職能が切削機 1，接着機，プレス機であるため，それぞれに該当する資源の稼働率はリスケジューリング前よりも増加している．その他の資源は活用されていないが，全体のリードタイムに大きな差がないため稼働率の変化もわずかとなっている．今回のリスケジューリングでは，ランダムに割当てる方が稼働率が高くなっているので，その他のディスパッチングルールよりも効率良く資源が使用されていることがわかる．

6 まとめ

本研究では，リアルタイム性を考慮した新たなリスケジューリングの手法を提案し，実際の工場現場の事例から特急品が発生した場合のリスケジューリングを適用し，タスクの関係性やディスパッチングルールに着目して分析を行った．その結果，計画段階でのスケジューリングで利用されていなかった資源が特急品の発生後は活用され，全資源が満遍なく割り付けられた．また，リスケジューリング後のディスパッチングルールの変更により，特急品のプロジェクトの終了時間に変化があったが，プロジェクト全体の終了時間に変化はなかった．しかし，特急品を優先せずランダムに割り付け可能としたディスパッチングルールにおいては全体のリードタイムが 6 時間短縮されることがわかった．このことから，現場の状況に合わせてディスパッチングルールを設定することが求められる．

現状の問題点として，特急品が一度発生した場合の対処は可能だが，連続して特急品が発生した場合は考慮されていない．機械故障であればその都度資源割付可能時間のテーブルに追加すれば対処可能だが，特急品の場合はプロジェクト全体に変化が生じてしまう．したがって，今後の課題では特急品が複数回発生した場合の解決案として特急品開始時間テーブルを作成し，特急品の発生時間を設定した分だけリスケジューリングが適用されるようなプログラムを実装する必要がある．

参考文献

1) 名嘉村盛和：ペトリネットに基づくスケジューリング問題へのアプローチ，Fundamentals Review，8-4，314/321(2015)

2) 中野，田，三宮：改善された局所探索法によるジョブショップスケジューリング問題の解法，電学論 C，121-10，1627/1633(2001)

3) 森川，中村：分岐限界法によるジョブショップのメイクスパン最小スケジュールの探索に対するラグランジュ緩和法の活用，53-6，466/473(2003)

4) 出口，市川，石塚，志手，染谷，湯浅：並列プロジェクト・タスク処理への多能工割付けの動的スケジューリング，Journal of the Inter-national Association of P2M，6-1，179/189(2011)

5) 橋本：動的スケジューリング手法を用いた集合住宅内装工事における工程分析 (2016)

6) 藤原，諏訪，森田:資源制約付きプロジェクト・スケジューリング問題に関する基礎的研究，数理解析研究所講究録，1629，125/130(2009)

7) Smith, S.F.: Reactive Scheduling Systems, Kluwer Academic Publishers, 3, 155-192(1995)

8) 杉川，井本，玉置：リアクティブ・スケジューリング問題の形式モデル，第 50 回システム制御情報学会研究発表講演会 (2006)

9) 坂口，神村，白瀬：大規模スケジュールのためのリアクティブスケジューリングに関する研究，日本機械学会，27/28(2008)

10) 植野，北村，西尾，中村：リアルタイム性を考慮した製造リードタイムおよび作業負荷平準の多目的最適化，第 52 回自動制御連合講演会 (2009)

11) SOARS Project, http://www.soars.jp/