Vol.2016-SE-191 No.26
2016/3/15

同時期に複数のプロジェクトに従事する作業者の作業優先順序決定方法の提案

古宮誠一[†1]　比護徹治[†2]　森一樹[†2]　橋浦弘明[†3]

少ない作業要員で，より多くのソフトウェアを開発するために，企業では一人の作業要員が同時期に複数のプロジェクトに携わることが珍しくない。一人の作業要員が同時期に担当する複数の作業がそれぞれ別のプロジェクトに属する場合,この作業要員は，自身が所属するそれぞれ別のプロジェクトのマネージャの作業指示に従わなければならない。このため，状況によってどのマネージャの作業指示に従うべきかを自分で判断しなければならず，この作業要員の判断能力とスケジュール管理能力が問われる。そこで，このような状況にある作業要員が，それぞれ別のプロジェクトに属する複数の作業に対して，どのように優先順序付けをし，どのようなスケジュールで実施すべきかを決定する新たな方法を提案する。

A Proposal of a Method for Prioritizing Several Tasks in Different Projects Which a Worker Works Simultaneously

Seiichi Komiya[†1]　Tetsuharu Higo[†2]　Kazuki Mori[†2]　and　Hiroaki Hashiura[†3]

In companies, a worker often works on several projects simultaneously in order to develop more software products with fewer workers. If several tasks which the worker engages in around the same time belong to different projects, the worker must follow the direction of different managers who belong to different projects. In this case, the worker must decide by oneself whether to follow whose directions according to circumstances. Therefore, his/her ability to make decisions and to manage of a schedule will be tested. This paper proposes a new method for helping the worker in this position to decide what the priority of his/her several tasks which belong to different projects should.

1. はじめに

企業では少ない作業要員で，より多くのソフトウェアを開発するために，一人の作業要員が同時期に複数のプロジェクトに携わることが珍しくない。特に，職能別組織による専門職化が進んでいる企業やその部署では，１つのプロジェクト内で，１人の作業要員が担当する作業の範囲は縮小する傾向が見られる。それは，高度なスキルを持つ一人の作業要員が同時期に複数のプロジェクトを掛け持ちすることにより，多数のプロジェクトに対して高度な技術を提供するという考えによるものである。一人の作業要員が同時期に担当する作業が複数あっても，それらが同一のプロジェクトに属するものばかりであれば，プロジェクト内にあるすべての作業を管理するマネージャがいるので，作業要員はこのマネージャの作業指示の下に行動すればよい。我々は，このようなマネージャの作業を支援するプロジェクト管理システムを長年に亘って研究開発してきた(例えば，1), 2), 3), 4), 5), 6), 7), 8), 9)がある)。しかし，一人の作業要員が同時期に担当する複数の作業がそれぞれ別のプロジェクトに属する場合には，この作業要員が担当するすべてのプロジェクトの状況を把握しているようなマネージャは存在しない。何故なら，プロジェクトというものの特性上，マネジメントはその１つのプロジェクトに閉じて管理すべきものであるからである。上記のような状況では，作業の競合(同一の作業要員に割り当てられた作業のスケジュールが日程的に重なってしまうこと)が発生することが少なくない。このとき作業要員はどの作業を優先すべきかの判断を迫られるが,その判断が困難なことが少なくない。このような場合に備え，ソフトウェア開発の現場では，予め優先度の高いプロジェクトを選び，それらの中での優先順序を決めておくことにより，作業がプロジェクト間で競合したときに，作業要員はより優先度の高いプロジェクトを優先して行動するように決められている。この方法は，上記のような場合に混乱を避ける上では有効な方法であるが，競合するプロジェクトの状況を詳細に調べると，常に最適な判断を提供しているわけではない。そこで本研究では，競合するプロジェクト(とその工程)が持つ工程管理上の特性に着目するとともに，各プロジェクトにおける後続工程への影響の大きさを比較することにより，競合するプロジェクト全体から見て，後続工程への影響が最も少なくなるように優先順序を決定する方法を提案する。

2. 研究開発の契機となった事例

本章では，本研究の契機となった事例（図 1）を採り上げ，それについて考察する．図 1 の上部には 2 つプロジェクト X と Y のスケジュールが表されており，下部には，これらのプロジェクトに従事する 4 人の作業要員（人的リ

†1 国立情報学研究所　先端ソフトウェア工学・国際研究センター
National Institute of Informatics
†2 芝浦工業大学大学院
Graduate School of Science and Engineering, Shibaura Institute of Technology
†3 日本工業大学
Nippon Institute of Technology

ⓒ 2016 Information Processing Society of Japan

ソース）α，β，γ，δの工程への割り当て状況が表されている．図 1 中の矢印は工程間の作業順序を表している。例えば，図 1 の工程 a と工程 b は，工程 a の作業が完了してから工程 b の作業が開始されるという工程間の作業順序を表している．そして，図 1 におけるプロジェクトとリソース(作業要員)のスケジュールを照合すれば，工程 a を担当する作業要員がαで，工程 b を担当する作業要員がβであることが分かる。さらに，図 1 を見れば，プロジェクト X には予め高い優先度が与えられているが，プロジェクト Y には与えられていないことが分かる。プロジェクトに高い優先度が与えられるのは，プロジェクトの開発規模が大規模であったり，特別な顧客であったりすることなどが，その理由である。一般に，高い優先度を与えられたプロジェクトは，プロジェクトの遂行に必要なリソースが，他のプロジェクトよりも優先的に配分され，作業が競合する折には優先的に作業が行えるという特権を持つ。

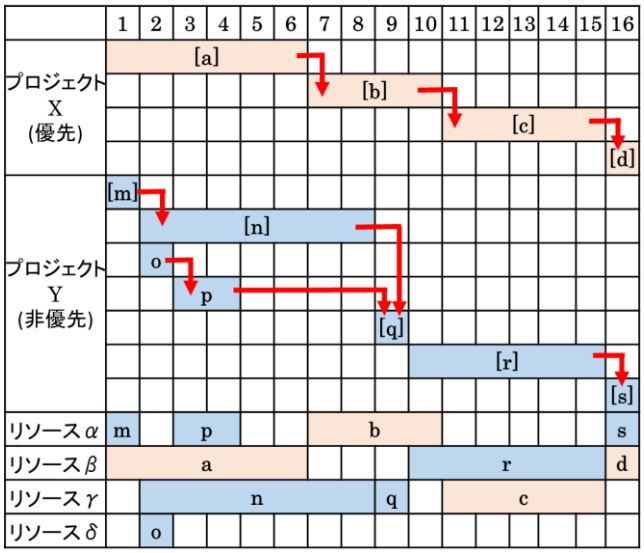

図 1　優先度の異なる 2 つプロジェクトにおける スケジュールの例（工程遅延の発生前）

プロジェクトにおいて工程遅延を防止するためには，プロジェクトにおけるスケジュールの最短経路を構成する工程，即ち，Critical Path 上にある工程のすべて(図 1 では，工程名を[]で囲った工程 a, b, c, d; m, n, q, r, s が該当する)を遅延させないことが必要不可欠である。このため，プロジェクトマネージャはそれらの工程が予定通り進んでいるかどうかについて普段から特段の注意を払っている．これに対し，Critical Path 上にない工程(図 1 では，工程 o と工程 p が該当する)では，多少の遅延は大きな問題とはならない．

ここで，工程 o の工程が 4 日遅れ，工程 p の開始が 7 日になった場合(図 2)を考えてみる．プロジェクト Y のスケジュールのみに注目した場合，作業要員δの工程 o が遅延し，2 日の終了予定がずれ込み，6 日に終了したとしても，工程 p を 7 日と 8 日に行えばよいので，一見すると何の問題もないように見える。しかし，表の下部に注目してみると，工程 p を 7 日と 8 日に担当する作業要員αは，工程 b も 7 日から担当することになっており，工程 b と工程 p が競合状態になる．工程 b は優先プロジェクト X に属するので，非優先プロジェクト Y に属する工程 p よりも優先して遂行されなければならない。もしここで，工程 p が特殊なスキルが要求される工程であり，他人が代行できない工程だったすると，非優先プロジェクト Y に遅延工程が発生することは明らかであろう。これは，1 人の作業要員が複数のプロジェクトを兼務することに起因する問題である．しかし，プロジェクト Y にしか参加していない作業要員δは，他のプロジェクトの状況を知ることができないので，自分が担当する工程 o の遅延が引き起こす問題(後続工程への影響)を予見できない．

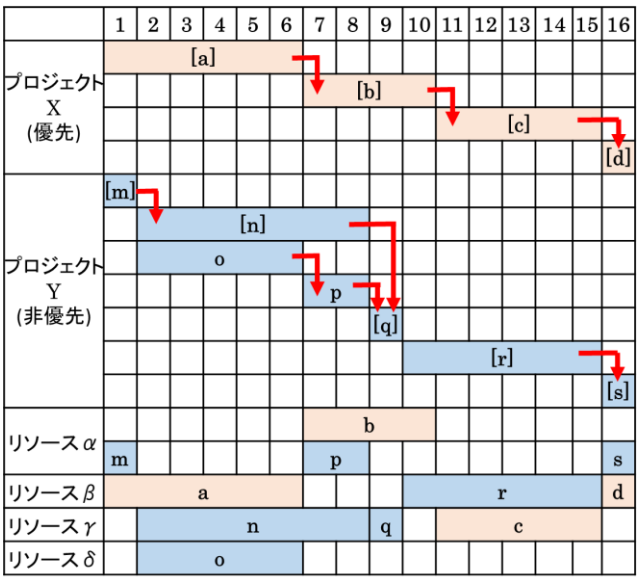

図 2　優先度の異なる 2 つプロジェクトにおける スケジュールの例（工程遅延の発生後）

より優先度の高いプロジェクト X を遅延させないことだけを考え，プロジェクト X の工程を優先すると，工程 p の開始が 11 日になる（図 2）ので，放置されたプロジェクト Y の完了日は 8 日間も伸びてしまう．一方，作業要員αがプロジェクトの優先順位を考慮せず，自分に割り振られていた工程を当初の順番通りに行うと，工程 b の着手が 2 日伸びるため，プロジェクト X の終了日が 2 日延びてしまう．

本研究は，1 人の作業要員が担当する，それぞれ別のプロジェクトに属する複数の工程が競合状態にあるとき，この作業要員が，それぞれ別のプロジェクトに属する複数の作業に対して，どのような優先順序を付け，どのようなスケジュールで実施すべきかを決定する方法を提案するものである．

ⓒ 2016 Information Processing Society of Japan

3　ソフトウェア開発計画が持つ制約

ソフトウェア開発計画を実際に遂行可能なものとするためには，以下に述べる制約を満たす必要がある 2).

3.1　作業順序に関する制約

図 3 の工程 b を例に挙げて説明すると，工程 b を遂行するには，工程 a による成果物（中間成果物）α が工程 b に着手する前に生成されていなければならない．これを工程 b の『事前条件』と呼ぶ．また，工程 b の成果物（中間成果物）β が工程 c に着手する前に生成されていなければならない．これを工程 b の『事後条件』と呼ぶ．中間成果物 α，β によって工程 a，b，c の実施順序が決まる．このような制約を『作業順序に関する制約』と呼ぶ．

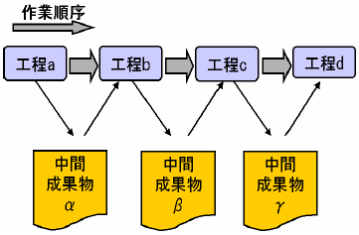

図 3　中間成果物に基づく作業順序に関する制約

3.2　リソースの割り当て条件に関する制約

ソフトウェア開発の各作業は，組織に所属するメンバーであれば誰でも担当できるわけではなく，その作業を実施する上で必要となる能力，資格，経験などのスキルを持つ人的リソース（作業要員）でなければその作業を担当できない．同様のことが開発環境等の非人的リソースにも言える．人的リソースと非人的リソースに関する，このような制約を『リソースの割り当て条件に関する制約』と呼ぶ．

3.3　リソースの割り当て可能期間に関する制約

ソフトウェア開発の各作業には，前述の割り当て条件を満足するリソースでも，そのリソースにとって割り当て可能な期間にしか，割り当てをすることができない．これは通常，稼働率向上のために，リソースは複数のプロジェクトで共用されているからである．このような制約を『リソースの割り当て可能期間に関する制約』と呼ぶ．

3.4 リソースの能力的限界に関する制約

各リソースにはそれぞれ処理能力に限界があるので，処理能力の限界を超えてまで作業を割り当てることはできない．処理能力の限界を表現するために『容量（Capacity）』という概念を導入し，リソースの属性として表現する．

容量は各リソースが 1 日に稼働可能な時間のうち，プロジェクトの作業に割り当て可能な時間の占める割合(占有率)の上限値で表す．作業計画の立案時には残業はないものとして考えるので，一日の稼働可能な時間はどのリソースも 8 時間であると考える．また，容量は 100%にならないことに注意されたい．これは，各リソースが一日に稼働可能な時間に，プロジェクト外作業の時間（バッファ）が含まれるためである．一例を挙げると，人的リソースであれば管理業務や受注対象となる新たな仕事に対する見積作業などの時間が，非人的リソースであればメンテナンスなどの時間がバッファとして含まれる．このバッファの大きさは，例えば，管理職か否かなどのリソースのランクに依存して変化するので，リソースごとに容量の値は異なってくる．複数の工程を並行して遂行する場合を考えると，各工程の遂行に必要な作業時間の総和の占める割合が，そのリソースの容量(能力的限界)を超えない限りにおいて，そのリソースは 1 日の中でそれらの作業を並行して行うことができる．このような制約をリソースの『能力的限界に関する制約』と呼ぶ．

4 作業の競合状態

着手日を迎えている複数の工程に対して，あるリソースがその日に遂行すべき作業時間の合計の占める割合が，そのリソースの容量（能力的限界）を上回るときを『作業の競合状態』という．

作業要員は各々の容量の範囲内であれば，複数の工程を並行して遂行することができるので，この状態は作業の競合状態とは言わない．また，ソフトウェア開発計画の初期計画の立案時には，前述のすべての制約を満たしている計画案を採用するため，プロジェクト開始時には作業の競合は発生してはいない．これはリソースが，単一のプロジェクトに割り当てられている場合でも，複数のプロジェクトを掛け持ちしている場合でも同じである．それにも拘わらず，作業の競合状態が発生する原因は，前工程または現在仕掛かり中の工程の遅延によるものである．

4.1　競合状態の発生の例

作業の競合状態の発生の様子を図 4 と図 5 におけるリソース α を用いて説明する．

▼現在

	日程	1	2	3	4	5	6	7	8	9	10
リソースα（容量上限80%）	プロジェクトV	工程u (80%)							工程w (80%)		
	プロジェクトW				工程v (80%)						
α の占有率		80	80	80	80	80	80	80	80	80	80

図 4 工程遅延による競合の発生（工程遅延の発生前）

▼現在

	日程	1	2	3	4	5	6	7	8	9	10
リソースα（容量上限80%）	プロジェクトV	工程u (80%)							工程w (80%)		
	プロジェクトW				工程v (80%)						
α の占有率		80	80	80	160	80	80	80	80	80	80

▲ 競合状態

図 5 工程遅延による競合の発生（工程遅延の発生時）

このリソース α の容量は 80%である．図 4 の当初計画では，α は 1 日目から 3 日目までプロジェクト G の工程 u(占

有率 80%）を遂行し，その後 4 日目から 7 日目まではプロジェクト H の工程 v（占有率 80%），8 日目からはプロジェクト G の工程 w を遂行することになっている. このようなスケジュールの基で作業を進め，4 日目に至ったときに，工程 v に 1 日の工程遅延が発覚したとすると（図 5），α が遂行すべき工程 u は（占有率 80%）と v（占有率 80%）の２つとなる．このときこれら両方の工程を遂行しようとすると占有率は 160%となり，α の容量（能力的限界）80%を超過してしまう。このため，リソースα は工程 u か工程 v のどちらかを選択しなければならない。即ち，リソースα に作業の競合状態が生じている．図 4 と図 5 の例では，プロジェクトを跨いで遂行する複数の工程の場合を競合状態と言ったが，競合状態というのは，プロジェクトを跨いで遂行する複数の工程の場合のみを言うのでない。同一プロジェクトに属する複数の工程の場合にも，上記と同様の状況が起これば競合状態という。

4.2 競合状態発生時の対応

あるリソースに作業の競合状態が発生したとき，このリソースは着手中のすべての工程を予定の期日までに完了できないことを意味する．よって，このような事態が発生した場合には速やかにマネージャにその旨を報告し，指示を仰がなければならない．また，マネージャは作業要員からの申告に基づいて，遅延に対する対策案の立案を行う．

ここでは，プロジェクトを跨いで競合状態が発生した場合（**図 5**）を考える．作業要員α は，自分が担当している工程が遅延しそうだという見通しが得られたら，速やかにプロジェクト G と H のマネージャに, 現在の状況を報告しなければならない。例えば, プロジェクト H のマネージャには, 自分が担当しているプロジェクト G の工程 u の作業が 1 日遅延し，このためプロジェクト H の工程 v への着手が 1 日遅れる旨を報告する必要がある。そして，プロジェクト G のマネージャには, 自分が担当しているプロジェクト G の工程 u の作業が 1 日遅延し，このためプロジェクト H の工程 v への着手が 1 日遅れる。工程 v の作業は完了までに 4 日掛かるので，このまま進むと，プロジェクト G の工程 w への着手が 1 日遅れる旨を報告する必要がある。このとき，報告を受けたマネージャ同士が話し合うことにより, 問題の解決を図ることになる. 通常はこのようになる。

このとき，競合するプロジェクトの間に優先権の差異があれば, 優先順序に基づいて問題の解決を図ることになる. 競合するプロジェクトの間に優先権の差異がなければ，競合するそれぞれの工程が Critical path 上にある工程であるか否かで工程の優先権が決まる。しかし，このような場合でも，マネージャ同士の力関係に差があれば，強いほうの意見が通り，弱い方は泣きを見る場合も少なくない。力関係に差がない場合には，双方が自プロジェクトの利益だけを考え，なかなか決着が付かないという事態にもなりかねない。そのようなことがないように，競合するプロジェクト全体から見て最適な解を導くような詳細なルールを用意して，それに基づいて対処すべきである。

本研究では，プロジェク間で競合する工程への対処の仕方によって生ずる，各プロジェクトにおける後続工程への影響の大きさを評価して，競合するプロジェクト全体で最適(被害が最小)となるように意思決定する方法を提案する.

5. 後続工程に及ぼす影響の大きさから見た工程の分類とその定義

或る工程の遅延がプロジェクトの後続工程にどのような影響を及ぼすかを議論するために，本章ではクリティカルパスとボトルネック工程という概念を導入する.

5.1 クリティカルパス上にある工程であるか否か

先行工程とその後続工程との間に時間的にまったく余裕のない工程同士を，プロジェクトの開始から完了まで繋いで出来る，単位作業が連なる道筋をクリティカルパス(Critical Path)と呼ぶ．そのため，クリティカルパス上にある工程の遅延は, 開発計画全体の遅延に直結する．以降，クリティカルパスを CP と略記し，クリティカルパス上にある工程を **CP 工程**，クリティカルパス上にない工程を**非 CP 工程**と略記する．なお，図 1～2 において工程名を[]で囲んだ工程（例えば工程 a, b, c; m, n, q, r, s）が **CP 工程**であり，工程 0, p が**非 CP 工程**である.

5.2 ボトルネック工程であるか否か w

5.2.1 ボトルネック工程の定義とその事例

複数のプロジェクトで共通の人的リソース(作業要員)または非人的リソース(マシン環境など)を使用している工程があり，この工程は日程を調整することも，この工程で使用するリソースを取り替えることもできない状況にあったとする. この工程のこのような状況を『**ボトルネック状態**』と呼び，ボトルネック状態にある工程を『**ボトルネック工程**』と呼ぶ[9]．**図 6** にその具体例を示す．ここでは工程 c に割り当て可能なリソースはα のみであると仮定する.

このような仮定の下で，プロジェクト T の工程 b に遅延が発生したとする.『**作業順序に関する制約**』により，工程 b の遅れた日数分だけ，後続工程 c の開始日と終了日が予定より遅くなる．プロジェクト T のスケジュールだけを見ると，工程 c には３日間の余裕日があるため，３日間の遅れまでなら，遅れた日数分だけ工程 c の開始日と終了日を遅らせることで調整できると読み取れてしまう. しかし，そのようにすると『**リソースの割り当て可能期間に関する制約**』により，工程 c，q，r に割り当てられている作業要員α は，遅れた日数分だけ工程 q の作業に着手するのが遅くなり，工程 q のために３日間の作業日数を確保できなくなるので，工程 q にはリソースα を割り当てることができない．そのため，リソースα を工程 c に割り当てるには，

リソースαの空きスケジュールが３日以上連続する 19 日まで待たなければならない．このため，要員の追加，プロジェクト U 工程 q のリソース変更などを行わない限り，プロジェクトの終了日が大幅に遅延する．このため，工程 c はその作業日程を１日も動かすことができない．また，工程 c に割り当て可能な作業要員はαだけなので，工程 c では使用する作業要員を取り替えることもできない．従って，図 6 の例における工程 c は(顕在的)ボトルネック工程である．

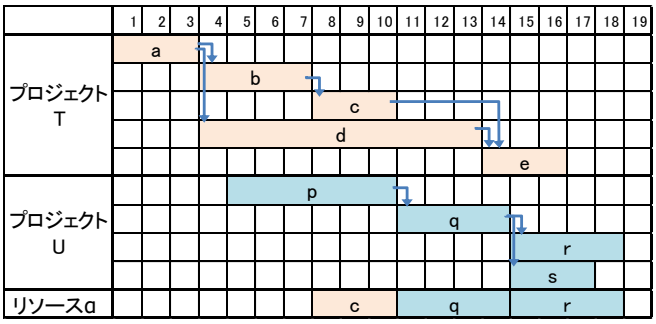

図 6 顕在的ボトルネック工程の例

5.2.2 潜在的ボトルネック工程の定義とその事例

元来はボトルネック状態になかった工程が，先行工程の遅延の影響を受け，その実施日が遅くなることによって，ボトルネック工程に変質する場合がある．そのような状況にある工程を『潜在的ボトルネック工程』と呼ぶ．

図 7 を例に採って説明する．工程 b, c, d に作業要員β，α，γがそれぞれ割り当てられていてβ，α，γ以外に割り当て可能な作業要員は存在しないとする．また，工程 a, e に関しては，割り当てる作業要員に何の制約もないので何日でも遅らせることが可能であるとする．図 7 において，工程 c に割り当て可能な作業要員はαしか存在しない．しかも，作業要員αの制約により，工程 c は最大で２日間までしか遅らせることができない．このため，２日間の工程遅延が工程 c に及ぶと，工程 C はその作業日程を動かすことも，使用する作業要員を取り替えることもできなくなる．つまり，２日間の工程遅延で工程 c はボトルネック工程に変質する．工程 c は**潜在的ボトルネック工程**である．

これに対して，図 6 における工程 c のように，先行工程の遅延の影響を受けなくても既にボトルネック状態にある工程を，『顕在的ボトルネック工程』(**Overt Bottleneck Process**)（略称：**OBN 工程**）と呼び，潜在的ボトルネック工程(**Potential Bottleneck Process**)（略称：**PBN 工程**）と区別する．そして，OBN 工程と PBN 工程を総称してボトルネック工程（略称：**BN 工程**）と呼ぶ．また，ボトルネック工程ではない工程を非ボトルネック工程（略称：非 BN 工程）と呼ぶ．なお，図 1 では工程 o の遅延によってボトルネック工程と化した工程 b と工程 p が，PBN 工程である．

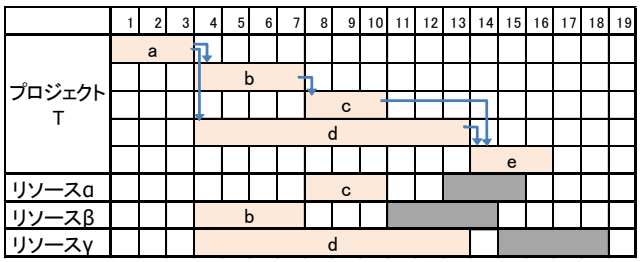

図 7 潜在的ボトルネック工程の例

6. 競合状態が発生した場合の優先順序の決定方法

本章では前章で述べた CP 工程と BN 工程の考え方を基に，競合状態が発生した場合の優先順位の決定方法を述べる．

6.1 CP 工程と BN 工程が競合する場合の考え方

CP 工程と BN 工程は，その工程に遅延が生じると後続工程への影響が大きいという点では変わらないが，これらの工程が競合した場合にどちらを優先すべきかを考えたい．

(1) CP 工程と OBN 工程とが競合状態にある場合

この場合，OBN 工程は既にボトルネック状態にある．そのため，既に１日の余裕もなく直ぐに着手しなければならない状況にある．この点では CP 工程も同じである．この場合には次のように考える．OBN 工程は，その定義からも分かるように，既に日程を調整することも，この工程を遂行するための要員を交代させることもできない状況にある．一方，CP 工程は１日の遅延も許されないが，この工程を遂行するための要員を交代させることは可能である．このため，競合状態にあるこれら２つの工程を担当する作業要員としては，CP 工程よりも，OBN 工程を優先して選んで作業をすべきである．本稿ではこのことを次のように表現する．

$$\{OBN, CP\} \Rightarrow OBN < CP \qquad (式 1)$$

このとき，この作業要員が OBN 工程を優先させることにより，自分が従事できなくなる CP 工程の作業を誰かに代わって貰えるように可能な限り速やかに申告する必要がある．

(2) CP 工程と PBN 工程とが競合状態にある場合

この場合，PBN 工程はボトルネック状態になるまでに未だ何日かの余裕がある．しかし，CP 工程は既に１日の余裕もなく直ぐに着手しなければならない状態にある．このとき，CP 工程の終了日の翌日まで PBN 工程への着手を遅らせても，PBN 工程がボトルネック状態にならない場合には，CP 工程を優先させ，この工程の作業が終了してから PBN 工程の作業に着手すればよい．この場合の作業の優先順序は PBN 工程がボトルネック状態になる日を TB_{PBN}, CP 工程の終了日 TD_{CP} とすると，次のように表現できる．

$$\{PBN, CP \mid TD_{CP} < TB_{PBN}\} \Rightarrow CP < PBN \qquad (式 2)$$

ここで $TD_{CP} < TB_{PBN}$ は CP 工程の終了日が，PBN 工程が

ボトルネックになる日よりも早く到来することを表している．しかし，CP 工程の終了日の翌日まで PBN 工程への着手を遅らせたら，PBN 工程がボトルネック状態になる場合には，この作業要員は PBN 工程の作業を優先的に選ぶ必要がある．この場合の優先順序は次のように表現できる．

$\{PBN, CP \mid TB_{PBN} < TD_{CP}\} \Rightarrow PBN < CP$ (式 3)

このとき，この作業要員が PBN 工程を優先させることにより，自分が従事できなくなる CP 工程の作業を誰か別の要員に代わって貰えるように，速やかに申告する必要がある．

6.2 ｛OBN 工程，PBN 工程，非 BN 工程｝中の２者が競合状態にある場合の考え方

(1) OBN 工程同士が競合状態にある場合

どちらのプロジェクトを優先すべきかということが予め決まっていれば，優先順序が高いほうのプロジェクトに属する工程を選択する．そして，選択しないほうの工程が属するプロジェクトのマネージャに，そのことを出来るだけ速やかに報告し，それによる工程遅延対策を検討して貰う．この場合の作業の優先順序は競合状態にある 2 つの OBN 工程を OBN_a, OBN_b，それぞれ工程が所属するプロジェクトの優先度を P_a，P_b とすると，次のように表現できる．

$\{OBN_a, OBN_b \mid P_a > P_b\} \Rightarrow OBN_a < OBN_b$ (式 4)

(2) OBN 工程と PBN 工程とが競合状態にある場合

この場合，PBN 工程はボトルネック状態になるまでに未だ何日かの余裕がある．しかし，OBN 工程は既に 1 日の余裕もなく直ぐに着手しなければならない状態にある．このとき，OBN 工程の終了日の翌日まで PBN 工程への着手を遅らせても，PBN 工程がボトルネック状態にならない場合には，OBN 工程を優先させ，この工程の作業が終了してから PBN 工程の作業に着手すればよい．しかし，OBN 工程の終了日の翌日まで PBN 工程への着手を遅らせたら，PBN 工程もボトルネック状態になるような場合（$TB_{PBN} < TD_{OBN}$ のとき）には，OBN 工程同士で競合する場合 (式 4) と同一の行動をとる．この場合の作業の優先順序は PBN 工程がボトルネック状態になる日を TB_{PBN}, OBN 工程の終了日 TD_{OBN}，それぞれ工程が所属するプロジェクトの優先度を P_{OBN}，P_{PBN} とすると，次のように表現できる．

$\{OBN, PBN \mid TD_{OBN} < TB_{PBN}\} \Rightarrow OBN < PBN$ (式 5)

$\{OBN, PBN \mid TB_{PBN} < TD_{OBN} \wedge P_{PBN} < P_{OBN}\} \Rightarrow OBN < PBN$ (式 6)

$\{OBN, PBN \mid TB_{PBN} < TD_{OBN} \wedge P_{OBN} < P_{PBN}\} \Rightarrow PBN < OBN$ (式 7)

(3) OBN 工程と非 BN 工程とが競合する場合

この場合には，無条件に OBN 工程の作業を優先する．そして，選択しないことにより工程遅延が生じることを，選択しない工程が属するプロジェクトのマネージャに速やかに報告する．この場合の優先順序は次のように表現できる．

$\{OBN, 非BN\} \Rightarrow OBN < 非BN$ (式 8)

(4) PBN 工程同士が競合状態にある場合

両方の工程がそれぞれいつからボトルネック状態へ変質するのか調べ，一方の工程の終了予定日までに，もう一方がボトルネック状態にならないことが相互に確認された場合にはプロジェクトの優先順序に基づき優先する工程を選択する．この場合の作業の優先順序は２つの PBN 工程のボトルネック状態になる日を TB_a, TB_b ，それぞれの工程の終了日 TD_a , TD_b ,それぞれ工程が所属するプロジェクトの優先度を P_a , P_b とすると，次のように表現できる．

$\{PBN_a, PBN_b \mid TD_b < TB_a \wedge TD_a < TB_b \wedge P_a > P_b\} \Rightarrow OBN_a < OBN_b$ (式 9)

一方の工程の終了予定日までに，どうしても他方の工程がボトルネック状態になってしまう（TB < TD となる）場合，OBN 工程と PBN 工程が競合状態にある場合 (式 5) と同一の行動をとる．両方ともボトルネック状態に変質してしまう場合には OBN 工程同士で競合状態にある場合 (式 4) と同一の行動をとる．これらは次のように表現できる．

$\{PBN_a, PBN_b \mid TB_a < TD_b \wedge TD_a < TB_b\} \Rightarrow PBN_a < PBN_B$ (式 10)

$\{PBN_a, PBN_b \mid TD_b < TB_a \wedge TB_b < TD_a\} \Rightarrow PBN_b < PBN_a$ (式 11)

$\{PBN_a, PBN_b \mid TB_a < TD_b \wedge TB_b < TD_a \wedge P_a > P_b\} \Rightarrow PBN_a < PBN_b$ (式 12)

(5) PBN 工程と非 BN 工程とが競合状態にある場合

この場合には，無条件に PBN 工程の作業を優先する．そして，選択しないことにより工程遅延が生じることを，選択しない工程が属するプロジェクトのマネージャに速やかに報告する．この場合の優先順序は次のように表現できる．

$\{PBN, 非BN\} \Rightarrow PBN < 非BN$ (式 13)

6.3 ｛CP 工程，非 CP 工程｝中の２者が競合状態にある場合の考え方

(6) CP 工程同士が競合状態にある場合

どちらのプロジェクトを優先すべきか予め決まっていれば，優先順序が高いほうのプロジェクトに属する工程を選択する．そして，選択しないほうの工程が属するプロジェクトのマネージャに，そのことを速やかに報告し，それによる工程遅延対策を検討して貰う．この場合の作業工程の優先順序は工程を CP_a,CP_b, それぞれの工程が属するプロジェクトの優先度を P_a, P_b とすると，次のように表現できる．

$\{CP_a, CP_b \mid P_a > P_b\} \Rightarrow CP_a < CP_b$ (式 14)

(7) CP 工程と非 CP 工程とが競合状態にある場合

この場合には，無条件に CP 工程の作業を優先する．そして，選択しないことにより工程遅延が生じることを，選択しない工程が属するプロジェクトのマネージャに速やかに報告する．この場合の優先順序は次のように表現できる．

$\{CP, 非CP\} \Rightarrow CP < 非CP$ (式 15)

6.4 非 BN 工程同士または非 CP 工程同士が競合状態にある場合のスケジュールの考え方

工程遅延のリスクを低減するために，両者のうちで少しでも時間的余裕のない作業を優先して行う必要がある．

6.5 非 BN 工程同士または非 CP 工程同士が競合状態にある場合のスケジュールの考え方のまとめ

工程同士が競合状態にある場合のスケジュールの考え方のまとめを**表 1** に示す。

表 1 工程間の優先順序

			プロジェクトY					
			OBN		PBN		非BN	
			CP	非CP	CP	非CP	CP	非CP
プロジェクトX（優先）	OBN	CP	X（式4）	X（式4）	X（式5）	X（式5）	X（式8）	X（式8）
		非CP	X（式4）	X（式4）	X（式5）	X（式5）	X（式8）	X（式8）
	PBN	CP	Y*（式5）	Y*（式5）	X*（式1）	X*（式9）	X（式13）	X（式13）
		非CP	Y*（式5）	Y*（式5）	Y*（式2）	X*（式9）	X（式13）	X（式13）
	非BN	CP	Y（式8）	Y（式8）	Y（式13）	Y（式13）	X（式14）	X（式15）
		非CP	Y（式8）	Y（式8）	Y（式13）	Y（式13）	Y（式15）	余裕時間小（式17）

注: *は競合状態にある工程の終了を待っても
PBN には変質しないことを表す.

表の各マスには，該当するルールを導いた式を記載している．作業要員はこの表に基づいて作業の優先順位の決定を行うことで提案手法をプロジェクトに対して適用できる．

7. 研究の契機となった事例への提案手法の適用

開発の契機となった事例に提案手法を適用して，本手法の有効性を確認する．

遅延発生が発覚した時点（**図 2**）では，リソースαの工程 b と工程 p が競合状態にある．工程 b は優先プロジェクトの CP 工程，工程 p は非優先プロジェクトの PBN 工程であるため，**6.1** で述べた「CP 工程と BN 工程が競合する場合の考え方」に沿って対処を行う．ここで，BN 工程が OBN に変化する日（TB_{PBN}）は 5 日目であり，競合する CP 工程の終了日（TD_{CP}）は 10 日目である．よって，TB_{PBN} と TD_{CP} の関係は $TB_{PBN} < TD_{CP}$ であり，工程 b の終了日の翌日まで工程 p への着手を遅らせると，工程 p はボトルネック状態になってしまう．このため，この競合状態は（**式 3**）に該当するということが分かる．この場合の対応としては，**6.1(2)**で述べた通り，リソースαはボトルネック工程である工程 p を実施し，CP 工程である工程 b をリソースδに交代してもらう．このようにすることで，どちらのプロジェクトも当初の終了予定日までに完了する。つまり，遅延を起こさないようにすることができる（**図 8**）．

また，上記のリソースの交代ができない場合であってもプロジェクト X の完了が 1 日遅延してしまうだけであり，プロジェクトの状況によっては十分検討に値するスケジュール案を提示して，プロジェクト間の調整をすることが可能となる．

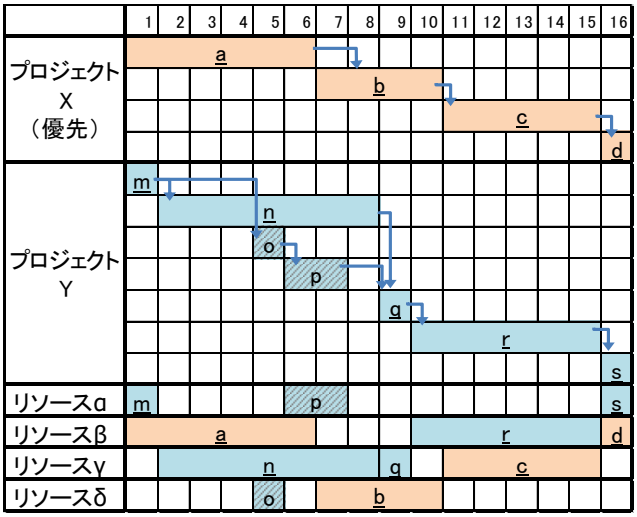

図 8 提案手法適用後の事例

8. おわりに

本稿では同一時期に複数のプロジェクトを兼務する作業要員に対し，どの作業を優先して遂行するべきかの判断を支援するために，後続工程に及ぼす影響の大きさを考慮したタスクの優先順位付けの方法を提案した．また，その方法を開発の契機となった事例に適用した結果を確認し，本手法の有効性を明らかにした．

[参考文献]

1) Project Management Institute Inc.,“プロジェクトマネジメント知識体系ガイド第 4 版（PMBOK ガイド），” 2008.
2) 古宮誠一，澤部直太，櫨山淳雄，“制約に基づくソフトウェア開発計画の立案，” 電子情報通信学会論文誌 D-1，Vol.J79-D-1，No.9，pp.544-557，Sep. 1996.
3) S. Komiya, and A. Hazeyama，” A Meta-Model Framework for Software Project Management System,” IEICE Trans. Inf. Syst., Vol.E81-D, No.12, pp.1415-1428，Aug. 1998.
4) A. Hazeyama, S. Komiya, ” Workload Management Facilities for Software Project Management.” IEICE Transaction Information Systems, Vol.E81-D, No12, pp.1404-1414， Dec 1998.
5) D. Kinoshita, R. Yaegashi, H. Hashiura, K. Uenosono, and S. Komiya, ”An Automatic Schedule Planning System,” Strategies and Evaluation for Implementing the System, Joint Conference on Knowledge-Based Software Engineering 2004(JCKBSE’04), Protvino, Russia, pp.37-48，Aug. 2004.
6) 八重樫理人，木下大輔，橋浦弘明，上之薗和宏，林雄一郎，古宮誠一，“工程遅延発生時におけるファーストトラッキングによる対策案の自動立案，” 電子情報通信学会論文誌 D-1，Vol.J88-D-1，

No.2，pp.215-227，Feb. 2005.
7) 高須賀公紀，嶋村彰吾，内川裕貴，木下大輔，林雄一郎，古宮誠一，“余裕日をも考慮に入れたソフトウェア開発計画の自動立案，”情報処理学会論文誌，Vol.48，No.8，pp.2713-2724，Aug. 2007.
8) D. Kinoshita, R. Yaegashi, K. Uenosono, H. Hashiura, H. Uchikawa, and S. Komiya: "Automatic Creation of a Crashing-Based Schedule Plan as Countermeasures against Process Delay," International Jouranal of Systems Applications, Engineering and Development, Issue 4, Vo. 2, pp. 170-177, 2008.
9) 木下大輔，内川裕貴，小坂祐也，古宮誠一: “摂動を用いた影響波及解析によるボトルネック工程の自動検出,” 情報処理学会論文誌, Vol.50, No.12, pp. 3084-3094, Dec. 2009.
10) Leach, Lawrence P., “Critical Chain Project Management”, Artech House Professional Development Library, London 313, 2000.
11) 稲垣公夫, TOC クリティカル･チェーン革命，日本能率協会マネジメントセンター, 1998.
12) エリヤフ・ゴールドラット著，クリティカルチェーン，ダイヤモンド社, 2003.
13) 中嶋秀隆，津曲公二, PM プロジェクト･マネジメント クリティカルチェーン，日本能率協会マネジメントセンター, 2003.

ⓒ 2016 Information Processing Society of Japan