情報処理学会論文誌　数理モデル化と応用　Vol.11　No.3　42–57 (Dec. 2018)

多目的遺伝的アルゴリズムによるITプロジェクトスケジューリング

小林 敬明[1,a)]　森口 聡子[2]

受付日 2018年3月19日，再受付日 2018年6月4日，
採録日 2018年7月23日

概要：本研究の目的は，IT プロジェクトマネジメントにおけるスケジューリング作業の効率向上によるプロジェクトマネージャの負荷軽減である．一般にプロジェクトの納期とコストはトレードオフの関係にあり，優先度はプロジェクトがおかれた環境や状況次第で変化する．そこで本研究では，(1) 納期，(2) 要員の重複タスク日数，(3) 要員数の 3 式の目的関数を最小化するための多目的遺伝的アルゴリズムを用いた自動スケジュール生成ソフトウェアを提案する．メタヒューリスティクスを用いたプロジェクトスケジューリング手法に関する多くの先行研究があるが，本研究では IT プロジェクトスケジューリング特有の目的関数と制約条件に着目する．本研究のモデルでは，パレートフロントの一部を厳密解として少ない計算量で得ることができ，それらを初期集団に組み込むことで探索効率を向上できることを示す．プロジェクトマネージャは，提案するソフトウェアで生成された複数の準最適解のなかから，プロジェクトの状況にあったスケジュールを選択することができる．提案するソフトウェアは，一般的な PC を用いて現実的な時間内でスケジュールを生成する．数値実験およびインタビューにより，提案するソフトウェアがプロジェクトマネージャの負荷軽減に有効であることを示す．

キーワード：プロジェクトスケジューリング，遺伝的アルゴリズム，多目的最適化

IT Project Scheduling Based on a Multi-objective Genetic Algorithm

Takaaki Kobayashi[1,a)]　Satoko Moriguchi[2]

Received: March 19, 2018, Revised: June 4, 2018,
Accepted: July 23, 2018

Abstract: The purpose of this research is to reduce the workload on the project manager by improving the efficiency of scheduling work in IT project management. In general, the delivery time and cost of the project are in a trade-off relationship, and the priority varies depending on the environment and circumstances. Therefore, in this research, we propose the software which generates schedules automatically using multi-objective genetic algorithm to minimize the objective functions of (1) the delivery time, (2) the number of days in duplicate task for all members and (3) the number of members. While there are many prior studies on project scheduling method using metaheuristics, in this research, we focus on objective functions and constraints specific to IT project scheduling. In the model of this study, we show that part of the Pareto front can be obtained as an exact solution with a small computational complexity, and search efficiency can be improved by incorporating them into the initial population. The project manager can select a schedule suitable for the situation of the project from semi-optimal solutions generated by the proposed software. The proposed software generates the schedule in realistic time using a general PC. The results of numerical experiments and interviews show that the proposed software is effective for reducing the workload of the project manager.

Keywords: project scheduling, genetic algorithm, multi-objective optimization

[1] 株式会社 TransRecog
TransRecog Co., LTD., Minato, Tokyo 105–0004, Japan
[2] 首都大学東京大学院経営学研究科経営学専攻
Department of Management, Graduate School of Management, Tokyo Metropolitan University, Hachioji, Tokyo 192–0397, Japan
[a)] takobaya@transrecog.com

1. はじめに

1.1 研究の背景

日経コンピュータの調査 [1] によると，IT 業界におけるプロジェクトの平均成功率は 31.1%とのことである．ここ

での成功の定義は納期超過，コスト超過や，プロジェクト中止を経ずに当初計画どおりに完遂できたものを示す．納期超過の原因の内訳を見ると，要件定義が長くなった（43.6%），設計作業が長くなった（33.0%），開発作業が長くなった（33.0%）と，プロジェクト実行段階において当初計画より長い期間がかかったことが上位を占めている．また，コスト超過の原因の内訳を見ると，追加の開発作業が発生（58.9%），追加の設計作業が発生（47.5%），追加の企画作業が発生（32.6%）と，当初の計画になかった追加作業がプロジェクト実行段階に発生したことが上位を占めている．これは，昨今，IT と事業の関係が密接になってきているため，自社のビジネスはどうあるべきかという検討に引きずられて要件定義の遅れや仕様変更が頻発することに起因する [1]．加えて，IT プロジェクトは，有形の成果物だけを持つプロジェクトを管理することよりも難しい．特に IT を構成する要素の 1 つであるソフトウェアは実体も形もなく，触ることも直接計測することもできない．成果物の開発が進展していく過程を見ることも，リスクを認識したり予測したりすることも有形の成果物に比べ困難であり，プロジェクトの状況や開発の方向性をつねに容易に把握できるとは限らない [2]. このように IT プロジェクトにおいては，計画どおりの遂行を阻害する事象が多発する．

一方，プロジェクトマネージャがプロジェクトマネジメントにおいて重要と考えているのがスケジュールである [3]．プロジェクトマネージャはスケジュールに責任を持つため，スケジュールを変更するときは自らがメンテナンスすべきとされている [4]．しかしプロジェクトマネージャはきわめて厳しい立場におり，プロジェクトにかかわる誰からも引っ張りだこであり多忙であるため [5]，多くの時間を割くことは困難である．時間のないなかで，プロジェクトマネージャは経験と勘と度胸に頼る手動での再スケジューリングを行う [6]．このため，プロジェクトマネージャは使える時間がさらに減少し，プロジェクトマネージャのストレスを高め，その結果チームの生産性，士気，成長，さらにはプロジェクトの品質，コスト，スケジュールに悪影響を及ぼす [5]．

加えて，世界の IT 投資額は増加傾向にあり 2017 年から 2020 年にかけて毎年 2.9%～4.3%の成長となる見通しである [7]．日本においても，2017 年度 IT 予算の対前年度伸び率は，直近 10 年中で第 3 位であり，伸び率は依然として高水準である [8]．これにともない IT 人材の需要も高まっているが，2017 年時点でも米国においても日本においても IT 人材不足が深刻な状況であるため [9], [10]，今後 IT 人材の確保がより難しくなる可能性がある．プロジェクトマネージャは数少ない IT 人材を用いてプロジェクトを遂行する必要があるだけでなく，プロジェクトマネージャ自身も人材不足でなり手が少ないため，プロジェクトマネージャの負荷は今後高まるばかりである．

以上，IT プロジェクトにおいては計画どおりの遂行を阻害する事象が多発する点，プロジェクトマネージャはスケジューリングを最優先する点，プロジェクトマネージャはつねに多忙でストレスフルである実情に加え IT 人材不足がプロジェクトマネージャの負荷を増大させる点から，IT プロジェクトマネジメントにおいてスケジューリング作業の効率向上によるプロジェクトマネージャの負荷軽減が喫緊の課題であることが分かる．本研究は，この課題の解消を目的とする．

1.2　先行技術と提案するソフトウェア

スケジューリング作業の効率向上策として，プロジェクトマネジメントツールの活用があげられる．**表 1** は，2015 年プロジェクトマネジメントツールの世界市場シェアから，トップ 3 を抜粋したものである [11]．いずれのツールも，スケジュール作成を支援する機能は持っているが，プロジェクトの納期とコストの最小化に対するトレードオフを考慮した複数の最適スケジュールを提案する機能はない．

そこで，プロジェクトの納期とコストのトレードオフを考慮した IT プロジェクト向けの自動スケジューリングソフトウェアを提案する．提案するソフトウェアは，あらかじめ制約条件を入力すると，その制約条件を満たす複数のパレート準最適解をデータとして出力するものとし，またそのデータをユーザがガントチャートとして閲覧することにより準最適解の集合のなかからプロジェクトの状況にあったスケジュールを選ぶことができるものとする．ユーザは，必要に応じて，選択した準最適解を Excel やプロジェクトマネジメントツールにインポートして，プロジェクトマネジメントに使用できるものとする．この提案するソフトウェアにより，IT プロジェクトマネジメントにおけるスケジューリング作業の効率向上によるプロジェクトマネージャの負荷軽減を実現する．

提案するソフトウェアの対象 IT プロジェクトのタスク数を示す．本研究におけるタスクの定義は以下とする [12]．

- 入力に価値を与えて出力を生み出す作業である．
- 要員 1 名が生成した出力を他の要員に転送するポイントを境界とする．
- 最大で 2 週間である．

日本情報システム・ユーザー協会は，IT プロジェクトを

表 1　2015 年プロジェクトマネジメントツール世界市場シェア
Table 1　2015 Project management tool world market share.

順位	ベンダ	製品名	2015 年 市場シェア(%)
1	Microsoft	Microsoft Project	35%
2	Oracle	Primavera P6 EPPM	19%
3	ServiceNow, Inc.	Project Portfolio Management	7%

開発工数 100 人月未満，100 人～500 人月，500 人月以上の 3 つに大別している [13]．このうち全体の 500 人月未満の IT プロジェクトは 85%を占めるため，提案するソフトウェアの対象 IT プロジェクトの工数は 500 人月とする．一般的に 1 タスクに割り当てる要員数は 1 名であり [14]，1 タスクあたり 2 週間が目途であるため [12]，500 人月をタスク数に換算すると，500 人月 × 20 日（1 カ月の稼働日数）÷ 10 日（2 週間稼働日数）= 1,000 タスクとなる．よって，提案するソフトウェアの対象タスク数は 1,000 タスクとする．

計算時間は，スケジュール変更が発生してから 1 時間以内とする．

1.3　先行研究

プロジェクトスケジューリング問題のなかで，優先順位制約やリソース制約など資源の制限を課した問題を「資源制約付きプロジェクトスケジューリング問題」という．プロジェクトスケジューリングを行う際の代表的な問題として，時間とコストのトレードオフがある．これは「離散時間コストトレードオフ問題」（Discrete time-cost tradeoff problem；DTCTP）と呼ばれ，近年研究がさかんに行われている．DTCTP の解法は，以下の 3 種類に大別される [15]．

(1) **厳密解アルゴリズム**

線形計画法，整数計画法，動的計画法，分枝限定法などを用いる．Szmerekovsky ら [16] は，時間コストのトレードオフをともなう不規則なコストのプロジェクトスケジューリング問題に対して 4 つの整数計画式を提案した．標準的な代入型変数を使用する 3 つの定式化を，新しい整数計画式に対して実験したところ，多くのケースで妥当な時間内に最大 90 のタスクの問題を解決することができた．

(2) **ヒューリスティクスアルゴリズム**

シーメンス近似法（SAM）などを用いる．Vanhoucke ら [17] は，伝統的なフォワードパスクリティカルパス計算と近傍検索法，タブーサーチ，動的計画法を組み合わせて近似解を求めている．数値実験では，50 のタスクで時間/コストトレードオフ，現在正味価値最大化計算を行い，42 秒で近似解を生成した．

(3) **メタヒューリスティクスアルゴリズム**

遺伝的アルゴリズム，焼きなまし法，散布探索法，多目的粒子最適化法，蟻コロニー最適化などを用いる．メタヒューリスティクスは，多目的トレードオフの問題を解決するためによく使用されている [15]．なかでも遺伝的アルゴリズムは多くの実践的なプロジェクトスケジューリング問題の解決に非常に有用であることが分かっているので，多くの研究者によって採用されてきた [18]．Tavana ら [15] は，マルチモードかつタスク中断ありの時間/コスト/品質トレードオフ多目的最適化問題に対して，NSGA-II（Fast Elitist Non-dominated Sorting Genetic Algorithm-II）[19] を改良したアルゴリズムを提案している．特徴としては遺伝子が多段構成である点，突然変異が 3 種類用意されている点，ペナルティ関数を適合度に加えている点などである．これにより，100 タスクの問題に対し，妥当なパレートフロントを CPU time 214.6609 秒で得ることに成功した．

ただし，上記 (1)，(2)，(3) のいずれも業種を特定しない汎用的なプロジェクトスケジューリングに関する研究である．たとえば Tavana ら [15] については，一般的なプロジェクトの指標である品質，コスト，納期を目的関数として採用しているが，品質はコストや納期よりも複雑な概念である．納期には年月日時分，コストには円やドルといった単位が想起されるが，品質には想起される単一の単位が存在しない．特にソフトウェアは実体も形もなく，触ることも直接計測することもできないため，品質を測ることは容易ではない．コスト，納期がそれぞれ目標値（予定値）と実績値の乖離度で定量的に把握し評価ができるのに対し，IT プロジェクトの品質はそれ自身の状況を一意的に表し評価する標準的な尺度がない．最終的な品質は，本番稼働後の障害の発生件数や影響度で評価されることが多いが，プロジェクト途中の品質は，その組織において経験的にはかり得た何種類かの品質関連指標を測定して評価されることがつねである [20]．品質は，プロジェクトと企業に応じて個別に評価され，またスケジュールとは別に管理されるため，本研究において，品質はスコープ外とする．

一方，一般的なプロジェクトと比べて IT プロジェクトで考慮しなければならない指標もあると考える．IT プロジェクトをターゲットとしたプロジェクトスケジューリングの研究はきわめて少ない．以下に IT プロジェクトの特徴を示す．

(a) IT プロジェクトは知識労働集約型である

IT プロジェクトは，コストのほとんどを人件費が占め，また成果物も無形物が中心である知識労働集約型である．よって，スケジューリングの中心的な課題は適正な要員配置となる．参考までに，IT プロジェクトと建設プロジェクトとの比較を**表 2** に示す [21]．

表 2　IT プロジェクトと建設プロジェクトとの比較
Table 2　Comparison between IT project and construction project.

項目	IT プロジェクト	建設プロジェクト
コスト構造	知識労働集約型，人件費が主体	ハードウェアが中心
管理・設計情報	完全に文書化できない	完全文書主義
成果物	ドキュメント，プログラム中心	ハードウェアが中心
成果物のスコープ	明確に表現しにくい	明確
成果物の可視性	不明確	可視的

(b) IT プロジェクトの要員は能力差が大きい

Grant, Sackman (1967) の実証実験結果を再検証した Prechelt [22] によると，最も作業の早いソフトウェアエンジニアと最も作業の遅いソフトウェアエンジニアの生産性の比は最大 14 倍である．IT プロジェクトにおいては，能力のある要員に複数のタスクを並行で遂行させる場合がある．

(c) IT プロジェクトはリソースの大部分を人間が占める

IT と同じ知識労働集約型業務のなかで，IT プロジェクトよりスケジューリング問題について先行研究が豊富にあるナース・スケジューリング問題を参考に制約条件を検討する．池上 [23] によると，ナース・スケジューリング問題の制約条件は以下の 2 つに分けられる．

- シフト拘束条件

各シフトに適した人数とスキルレベルのナースを割り当てることにより看護の質を守ろうというものである．具体的には，各シフトの合計勤務人数や各グループからの人数に下限値と上限値を設定するものである．

- ナース拘束条件

以下の 3 つのタイプの拘束条件からなり，各ナースの労働負荷を考慮するものである．

Type I： 各シフト（日勤/夜勤/休み，または，日勤/準夜勤/深夜勤/休み）が適切な回数であること

Type II： 休みやシフトの希望日やセミナー参加を達成すること

Type III：ナースの健康に悪影響をもたらすシフトの並びを避けること

池上は，すべての拘束条件を満たす勤務表を作ることはできず，一般的にはナース拘束条件を緩和するが，それができるのはスタッフ・ナースの好み，性格，健康状態などをよく把握している師長や主任だけとしている．

これは，IT プロジェクトにおいて，性格や体調といった要員の属性をプロジェクトマネージャが把握し，場合によっては要員に無理をしてもらう，もしくは休んでもらうなどの調整と類似している．

2. 目的関数と制約条件

2.1 目的関数

以上の IT プロジェクトの特徴から，

(1) プロジェクト完了日数
（単位：日，以降「納期」とする.）

(2) 全要員の重複タスクの日数の合計
（単位：日，以降「重複日数」とする.）

(3) プロジェクトに参画する要員数
（単位：人，以降「要員数」とする.）

の 3 つの目的関数の最小化を目的とする．

納期と要員数はそれぞれ DTCTP で主題となる時間とコストを表す目的関数である．IT プロジェクトのコストは人件費が主体となるため，コストを示す目的関数をプロジェクトに参画する要員数とする．重複日数は IT 人材が不足しているため，同一要員に並行して複数のタスクをかけ持ちしてもらわざるを得ない場面が想定されるが，要員の能力差が大きいため有能な要員はそれが可能という IT プロジェクトの実状を反映した目的関数である．また，要員の性格，体調，人間関係に応じて，負荷が重くてもタスクをかけ持ちできることもあれば，逆に精神的な問題などでタスクをかけ持ちできないこともあるという，リソースの大部分を人間が占める IT プロジェクトの特性を反映した目的関数でもある．

メタヒューリスティクスの様々な手法を用いたプロジェクトスケジューリング手法は，単目的/多目的問わず数多く提案されてきている [15], [18], [24]．しかし，先行研究では，前述の IT プロジェクト特有の目的関数である重複日数は設定されていない．そこで本研究では，重複日数を目的関数に追加し，IT プロジェクトならではの制約条件を設定した場合に，解集合の探索性能を向上させる手法の提案を主眼におくこととする．

2.2 制約条件

本研究では，IT プロジェクトのスケジューリングで一般的に用いられている以下を制約条件として設定する [14], [25], [26].

(1) プロジェクトスケジュールは複数のタスクから構成され，各タスクには先行タスクと期間（日数）が設定されている．

(2) 最初のタスクを除くすべてのタスクには先行タスクが設定されている．

(3) 先行タスクは複数設定できる．

(4) タスクに割り当てる要員は要員リストに登録されている．

(5) 1 タスクに割り当てる要員数は 1 名である．

(6) プロジェクトスケジュールは 1 つのタスクから始まり，1 つのタスクで終わる．

最小化したい関数が 3 つであることから，多目的最適化問題となる．多目的最適化問題におけるパレート最適解集合を獲得する手段として，進化計算が注目されている [27]. 進化計算では，多点探索により，パレートフロントを近似する解集合がアルゴリズムの 1 回の実行で獲得されるため，特に多目的最適化に適した手法である．また，進化計算が取り扱える最適化問題の幅広さや，実問題と多目的最適化問題のモデルの近さから，進化計算による多目的最適化は，産業応用との親和性も高い [27]．そこで本研究では，進化計算による多目的最適化で最も有名なアルゴリズムとして応用分野で頻繁に利用されている [27]，NSGA-II を採用することとする．

なお，(1) で示したタスクの期間（日数）はタスクの作

業量を示す．IT プロジェクトにおけるタスク作業量の指標は，ソフトウェア開発であればプログラムステップ数やファンクションポイントなどがあげられるが [28]，ソフトウェア開発以外のタスク，たとえば調査・企画，移行，インフラ整備，ユーザ教育などには標準的な指標がなく，「人月」「人日」が一般的である [29]．本研究では (5) のとおり 1 タスクに割り当てる要員数は 1 名としているため，タスクの作業量を日数とする．

タスクの作業量である日数は自然数とする．タスクの単位は最小でも 1 日が一般的である [30]．タスクの中には 1 日未満で完了するものも存在するが，これも 1 日としてスケジューリングする．IT プロジェクトにおいて行うべきタスクは漏れなくスケジュールに記載する必要がある [30]．たとえば，顧客先のデータセンタで作業するために 5 営業日前に入館申請を出さなければならないというルールがある場合，入館申請の作成と提出は 1 日もかからないかもしれないが，これを実施しないとデータセンタに入館できずスケジュールに支障をきたす．よって，目的関数の 1 つである重複日数は実態より過大に算出される可能性がある．その結果，ある要員 1 名に対し 1 日に完遂できない作業量を割り当てられた場合，プロジェクトマネージャは増員を検討する必要があるかもしれないが，これは算出された 1 日の作業量だけで単純に判断できるものではない．その理由は，目的関数や制約条件に含まれていないプロジェクトマネージャの制約や好みがあるためである [24]．制約や好みはプロジェクトマネージャの心の中にある [24]．それらをすべて定式化しても，他のプロジェクトマネージャには適用できないため，一般化できないものとなる．先行研究では，意思決定のため，プロジェクトマネージャにできるだけ多くの情報を提供することが重要だとしている [24]．本研究では，IT プロジェクトの要員は能力差が大きい点，要員の「気持ち次第」でパフォーマンスが上下するという人的リソースである点，上述のとおり作業を漏れなくスケジュールするために 1 日未満のタスクも 1 日として入力している点などを考慮して，数値上は実行不可能に見える可能性のある重複日数多数の解もプロジェクトマネージャに提示することとする．

3. 多目的最適化によるモデルと数値実験

3.1 制約条件の設計

2.2 節で述べた制約条件を表すため，以下のデータ構造で表す．

$$(k_1, d_1, \boldsymbol{A}_1, taskname_1),$$
$$(k_2, d_2, \boldsymbol{A}_2, taskname_2)$$
$$,\ldots,$$
$$(k_n, d_n, \boldsymbol{A}_n, taskname_n)$$

ここで

k_i：タスク番号

d_i：タスクの所要日数

$\boldsymbol{A}_i = (a_{i1}, a_{i2}, \ldots, a_{ij_i})$

ただし a_{ij_i} は先行タスク番号．

j_i は該当タスクに対応する先行タスクの数で，先行タスクがない場合は $a_{i1} = 999999$

$taskname_i$：タスク名（文字列）

$i = 1, 2, \ldots, n$（n はプロジェクトの総タスク数）

である．

制約条件のなかに出てくる要員リストは，以下のデータ構造で表す．

$$(e_1, membername_1),$$
$$(e_2, membername_2)$$
$$,\ldots,$$
$$(e_c, membername_c)$$

ここで

e_i：要員番号

$membername_i$：要員名（文字列）

$i = 1, 2, \ldots, c$（c は要員数）

である．

3.2 遺伝子の設計

遺伝子は以下のデータ構造とする．

$$(t_1, m_1), (t_2, m_2), \ldots, (t_n, m_n)$$

ここで

t_i：タスク開始遅延日数

m_i：要員番号

$=$ (要員リスト $e_1, \ldots, e_j$ のうちいずれか 1 つ)

$i = 1, 2, \ldots, n$（n はプロジェクトの総タスク数）

$j = 1, 2, \ldots, c$（c は要員数）

である．

上記のように設計することで，制約条件を満たさない致死遺伝子の生成を回避できる．加えて，3 つの目的関数，(1) 納期，(2) 重複日数，(3) 要員数のうちいずれか 2 つを最小化するパターンを生成できる．制約条件と目的関数 (1)，(2)，(3) のいずれか 2 式を最小化する場合の遺伝子例を，以下 (a)，(b)，(c) に示す．

〈遺伝子例の前提〉

制約条件と要員リストを以下とする．

- 制約条件

  $(1, 2, (999999),$ “企画”$)$,

(2, 5, (2), “宣伝”),

(3, 2, (2), “開発”),

(4, 2, (3), “製造”),

(5, 2, (2, 4), “販売”)

- 要員リスト

(1, “佐藤”)

(2, “鈴木”)

上記をガントチャートで表したものを**図 1** に示す．

(a) 目的関数 (1) 納期，目的関数 (2) 重複日数が最小の場合の遺伝子例

(0, 1), (0, 1), (0, 2), (0, 2), (0, 1)

この遺伝子例を適用した場合のスケジュールを**図 2** に示す．制約条件下において目的関数は，(1) 納期が 9 日で最小，(2) 重複日数が 0 日で最小，(3) 要員数が 2 人となる．

(b) 目的関数 (2) 重複日数，目的関数 (3) 要員数が最小の場合の遺伝子例

(0, 1), (0, 1), (5, 1), (0, 1), (0, 1)

この遺伝子例を適用した場合のスケジュールを**図 3** に示す．制約条件下において目的関数は，(1) 納期が 13 日，(2) 重複日数が 0 日で最小，(3) 要員数が 1 人で最小となる．

(c) 目的関数 (1) 納期，目的関数 (3) 要員数が最小の場合の遺伝子例

(0, 1), (0, 1), (0, 1), (0, 1), (0, 1)

図 1　例の制約条件を満たすガントチャート

Fig. 1　Gantt chart satisfying the constraints in this example.

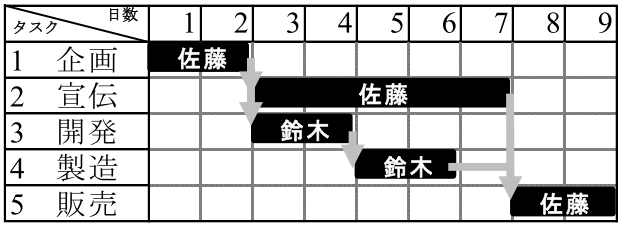

図 2　目的関数 (1)，(2) が最小の場合

Fig. 2　The case of minimization of the objective functions (1) and (2).

この遺伝子例を適用した場合のスケジュールを**図 4** に示す．制約条件下において目的関数は，(1) 納期が 9 日で最小，(2) 重複日数が 4 日，(3) 要員数が 1 人で最小となる．

3.3　評価関数

評価関数は制約条件と遺伝子から目的関数を生成する関数である．処理は以下の順に行う．

(a) トポロジカルソート結果を用いた各タスク開始終了日の算出

制約条件にある各タスクに対応する先行タスクのデータを用いたグラフを内部的に生成し，トポロジカルソートでシリアライズを行う．2.2 節の制約条件より，グラフは有向非巡回グラフとなるので，トポロジカルソートが可能となる．なおトポロジカルソートは，1 回だけ行えばよいので，遺伝的アルゴリズムの本処理を行う前にあらかじめ実行しておく．

シリアライズ後のタスクデータを用いて，先頭から順番にタスクの開始終了日を算出する．タスクの開始日は，終了日算出済みの先行関係のあるタスクの中から，最も遅いタスクの終了日の翌日に，対応する遺伝子のタスク開始遅延日数を加算した値とする．あわせて対応する遺伝子の要員番号に従い要員をタスクに割り当てる．

(b) 納期の算出

生成した各タスク開始終了日のなかから最も遅いタスク終了日を目的関数 (1) 納期とする．

(c) 要員数の算出

割り当てられた要員数を目的関数 (2) 要員数とする．

(d) 重複日数の算出

要員ごとに，1 つの日に複数のタスクが割り当てられて

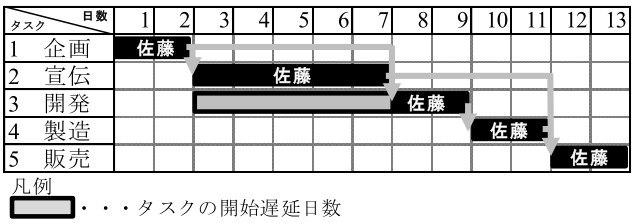

図 3　目的関数 (2)，(3) が最小の場合の例

Fig. 3　The case of minimization of the objective functions (2) and (3).

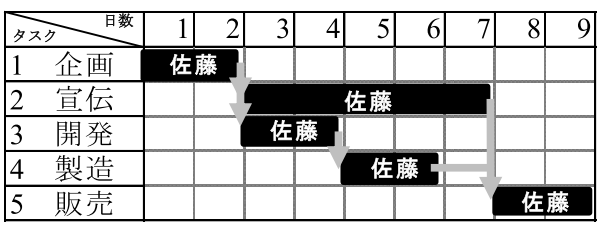

図 4　目的関数 (1)，(3) が最小の場合の例

Fig. 4　The case of minimization of the objective functions (1) and (3).

いる日について，重複しているタスク数 −1 を順に積算してゆく．たとえばある要員がある日に 3 タスク重複している場合は，2 を積算する．この値を目的関数 (3) 重複日数とする．重複日数の算出アルゴリズムを以下に示す．

Input：
c：要員数
n：プロジェクトの総タスク数
$lastDay$：納期
t_k：k 番目のタスク開始遅延日数
m_k：k 番目のタスクの要員番号
$taskStart_k$：k 番目のタスク開始日
$taskEnd_k$：k 番目のタスク終了日
Output：
ovd：重複日数
Step1 ： ovd=0
Step2 ： **For** i=0 **To** c
Step3 ： E[0...$lastDay$]=0
Step4 ： **For** j=0 **To** n
Step5 ： **If** m_j ==i **Then**
Step6 ： **For** l= $taskStart_j + t_j$ **To** $taskEnd_j$+1
Step7 ： E[l]=E[l]+1
Step8 ： **For** j=0 **To** $lastDay$
Step9 ： **If** E[j]>1 **Then**
Step10： ovd=ovd+ E[j]-1
Step11：**Return** ovd;

2.2 節で示した遺伝子例の制約条件と，遺伝子の例として

$$(0,1),(2,1),(0,1),(0,2),(0,1)$$

を用いた本処理のイメージを図 **5**，図 **6**，図 **7**，図 **8** に示す．

3.4　提案するソフトウェアの流れ

提案するソフトウェア全体の流れを図 **9** に示す．入力は制約条件と要員リストであり，出力は 1.2 節に示した要件どおり，パレート準最適解を表す遺伝子とガントチャートのデータである．本処理は多目的遺伝的アルゴリズムである NSGA-II を用いる．交叉は DEAP [31] で提供されている上限・下限を指定できる Simulated Binary Crossover を用いる．変異は同じく DEAP で提供されている上限・下限を指定できる Polynomial Mutation を用いる．DEAP におけるこれらの実装は，Deb ら [19] が作成した C 言語版 NSGA-II ソースの移植である．

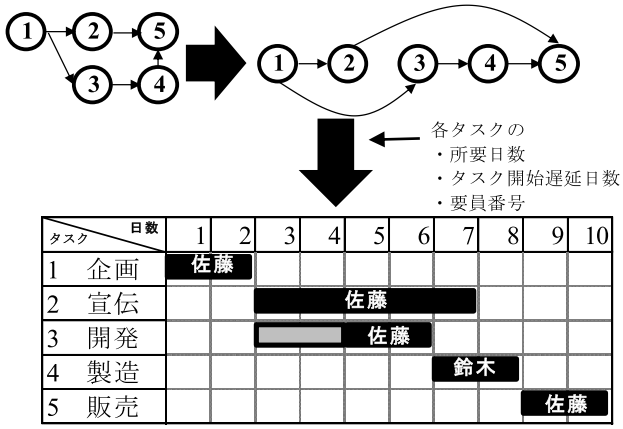

図 **5**　(a) トポロジカルソート結果を用いた各タスク開始終了日の算出

Fig. 5　(a) Calculation of start and end day of each task by using topological sorting results.

3.5　提案するソフトウェアの環境および構成

1.1 節に示したとおり，本研究は，実務的な課題を解消することを目的としているため，提案するソフトウェアは，実務で利用可能な環境で動作しなければならない．そこで提案するソフトウェアは，プログラミング実行環境として

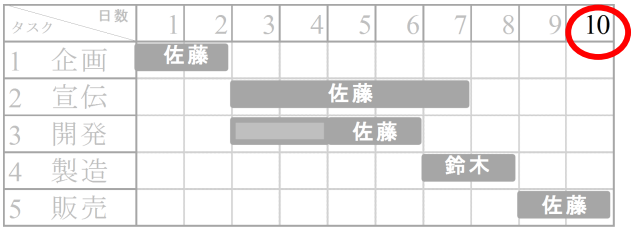

図 **6**　(b) 納期の算出

Fig. 6　(b) Calculation of the delivery time.

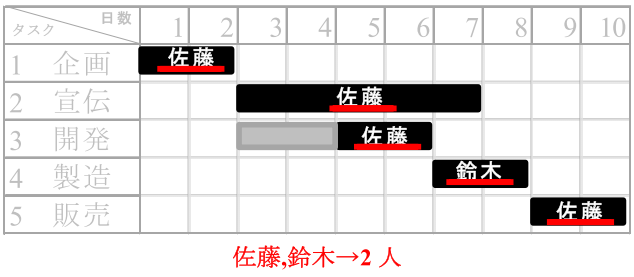

図 **7**　(c) 要員数の算出

Fig. 7　(c) Calculation of the number of members.

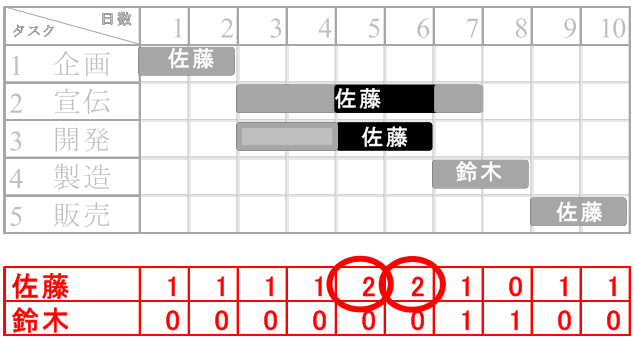

図 **8**　(d) 重複日数の算出

Fig. 8　(d) Calculation of the number of days of duplicate tasks.

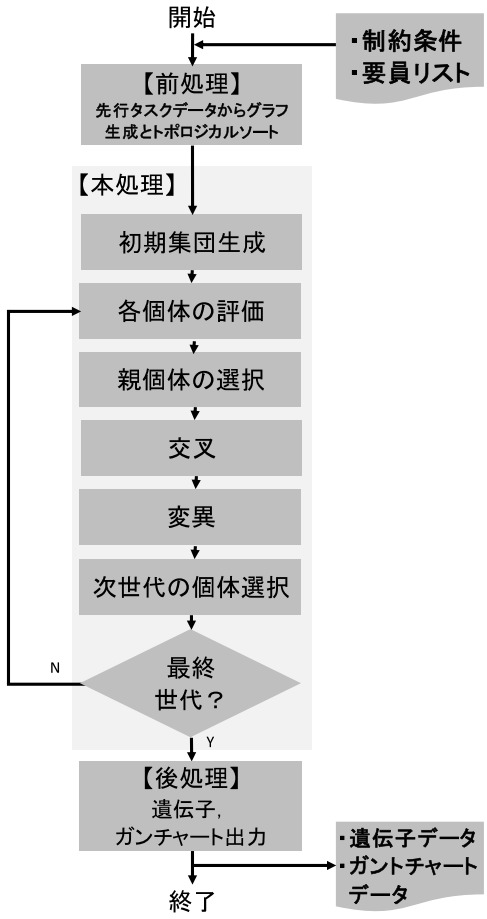

図 9　提案するソフトウェア全体の流れ

Fig. 9　Flowchart of the proposed software.

表 3　ライブラリと利用用途

Table 3　Library and usage.

ライブラリ	利用用途
NetworkX[33]	グラフの生成とトポロジカルソート
Python-Gantt[34]	ガントチャートデータ生成
NumPy[35]	各種数値計算
DEAP[31]	遺伝的アルゴリズム

急速に利用が拡大している Python [32] で動作するように開発する．提案するソフトウェアが使用するライブラリと利用用途を**表 3** に示す．いずれも GNU General Public License もしくは BSD ライセンスであり，無償で利用できる．商用ソルバを使用しないため，OS 以外にソフトウェア購入費用はかからず，容易に実務で利用できる．

3.6　数値実験 1

提案するソフトウェアの性能と精度を図る目的として 100 タスクの制約条件と，50 人の要員リストで数値実験を行う．制約条件は実務で使用されたスケジュールデータから 100 タスク分を抜粋したものを使用する．実験環境を**表 4** に示す．遺伝的アルゴリズムの世代数は 300，個体数は 200，交叉率は 0.8，突然変異率は 0.05，Simulated Binary Crossover と Polynomial Mutation の η は 30.0 とする．

表 4　実験環境

Table 4　Experiment environment.

区分	項目	内容
ハードウェア	CPU	Intel Core i7-7600U@2.80GHz
	メモリ	LPDDR3 SDRAM 16GB
ソフトウェア	OS	Windows 10 Professional (64bit)
	実行環境	Python 3.6.2
	ライブラリ	NetworkX 1.11
		Python-Gantt 0.6.0
		NumPy 1.12.1
		DEAP 1.2.2

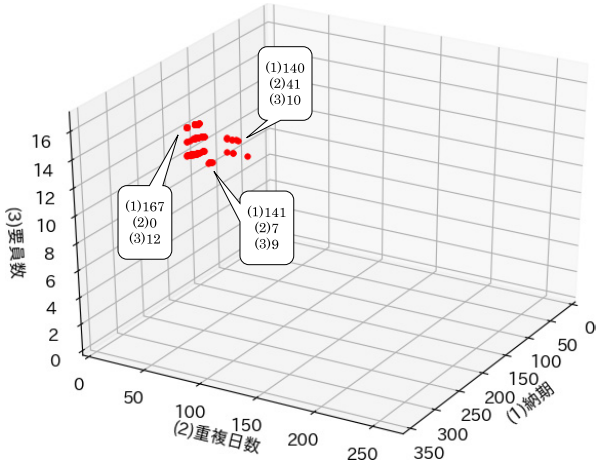

図 10　実験結果（数値実験 1）

Fig. 10　Experimental results (Numerical experiment 1).

実験結果を**図 10** に示す．計算時間は，プログラム開始から 300 世代目計算完了までが 3 分 5 秒であった．

この実験結果には解集合の広がり（多様性）が見られない．遺伝的アルゴリズムはヒューリスティクスアルゴリズムであり，必ずしも真のパレートフロントの求解はできない点を考慮しても，目的関数 (3) 要員数は最小 9，最大 12 と範囲が狭い．

ところで，本研究のモデルは，前述のとおり IT プロジェクトの特徴を反映したうえで設計した．目的関数は IT プロジェクトのスケジュールで重要な 3 指標を反映したものであり，また制約条件はきわめて一般的な IT プロジェクトのスケジュールの枠組みに従っているため，様々な IT プロジェクトに適用可能である．そのうえで，遺伝子は 3 つの目的関数，(1) 納期，(2) 重複日数，(3) 要員数のうちいずれか 2 つを最小化するパターンを生成できるように設計した．これらをふまえ多様性の向上策を検討した結果，IT プロジェクトのスケジューリングに特化した本研究のモデルは，単一目的最適化でパレートフロントの一部を厳密解として少ない計算量で得ることができる特殊なスケジュール問題であることが分かった．具体的には，3 つの目的関数，(1) 納期，(2) 重複日数，(3) 要員数のうち，(2)，(3) を最小にした状態で (1) を最小化する遺伝子，(1)，(3) を最小にした状態で (2) を最小化する遺伝子，(1)，(2) を最小

にした状態で (3) を最小化する遺伝子の計 3 つのパレートフロントの端にある厳密解を表す遺伝子をそれぞれ O（タスク数）の計算量で獲得することができる．そこで，これら 3 つの遺伝子を，遺伝的アルゴリズムを実行する前に算出し，初期集団に含めることとする．これにより，解集合の広がり（多様性）を期待できると考える．このことは先行研究ではみられない，IT プロジェクトスケジューリングを扱う本研究特有の工夫である．

4. 厳密解の初期集団への適用

4.1 厳密解の生成アルゴリズム

3 つの厳密解遺伝子の生成アルゴリズムを以下に示す．

(a) 目的関数 (2) 重複日数，目的関数 (3) 要員数が最小の状態で目的関数 (1) 納期を最小にする遺伝子

これは，トポロジカルソートされたタスク順に重複しないようタスクを左詰めに並べることで容易に算出できる．要員は，要員リストからランダムに選んだ 1 名をすべてのタスクに割り当てる．目的関数 (2) 重複日数は 0 日，目的関数 (3) 要員数は 1 人となる．3.2 節の例で示すと，ランダムで選んだ要員 1 名が 1 番だとすると，スケジュールは図 3 のようになり遺伝子は，

$$(0,1),(0,1),(5,1),(0,1),(0,1)$$

となる．この場合，目的関数 (1) 納期は 13 日となる．

(b) 目的関数 (1) 納期，目的関数 (3) 要員数が最小の状態で目的関数 (2) 重複日数を最小にする遺伝子

これは，トポロジカルソートされたタスク順に重複を許しつつタスクを左詰めに並べることで容易に算出できる．要員は，要員リストからランダムに選んだ 1 名をすべてのタスクに割り当てる．目的関数 (3) プロジェクトに参画する要員数は 1 人となる．3.2 節の例で示すと，ランダムで選んだ要員 1 名が 1 番だとすると，スケジュールは図 4 のようになり遺伝子は，

$$(0,1),(0,1),(0,1),(0,1),(0,1)$$

となる．この場合，目的関数 (1) 納期が 9 日，目的関数 (2) 重複日数が 4 日となる．

(c) 目的関数 (1) 納期，目的関数 (2) 重複日数が最小の状態で目的関数 (3) 要員数を最小にする遺伝子

これは，目的関数 (1) 納期を (b) で得た値に設定し，目的関数 (2) 重複日数を 0 にした場合に，目的関数 (3) 要員数を最小化する 1 目的最適化問題となる．

3.2 節の例で示すと遺伝子は，

$$(0,m_1),(0,m_2),(0,m_3),(0,m_4),(0,m_5)$$

となり，目的関数 (3) 要員数を最小化する $m_1,\ldots,m_5 \in$ 要員リスト中から要員番号 1 つの組合せを求めることになる．この 1 目的最適化問題の求解アルゴリズムを以下に示す．

Step1：初日から納期までの各日で，タスク重複の最大数を算出する．このタスク重複最大数が目的関数 (3) 要員数の最小値となる．

Step2：トポロジカルソートされたタスク順に要員リストから先頭順に要員を割り当てる．1 番目のタスクは要員リストの先頭の要員を割り当てる．

Step3：$n = 2$

Step4：1 番目から $n-1$ 番目までのタスクについて，n 番目のタスクの該当期間と重複している場合は，重複しているタスクに割り当てられている要員を要員リストから除外する．

Step5：n 番目のタスクに要員リストの先頭の要員を割り当てる．

Step6：要員リストの除外をすべて解除する．

Step7：全タスク要員の割り当てが完了したら終了．

Step8：$n = n + 1$

Step9：Step4 に戻る．

4.2 数値実験 2

3.6 節の数値実験 1 と同じ制約条件，実験環境で 3 つの厳密解を初期集団に含めた提案ソフトウェアを実験する．実験結果を**図 11** に示す．

数値実験 1 の結果である図 10 と比較して，解集合の広がり（多様性）が大幅に改善された．実験結果 1 と実験結果 2 を評価するため，それぞれの世代ごとの HyperVolume 値 [36] を比較する（**図 12**）．HyperVolume の参照点は $(2000, 300, 100)$ とした．

数値実験 2 は 0 世代目からパレートフロントの端の解を抑えているため始めから高い HyperVolume 値を示している．対して数値実験 1 は，0 世代目から世代が進むにつれて HyperVolume 値が改善されてはいるものの，300 世代

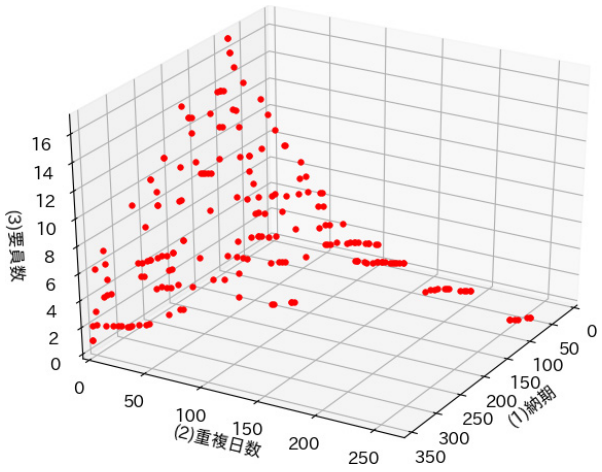

図 11　実験結果（数値実験 2）

Fig. 11　Experimental results (Numerical experiment 2).

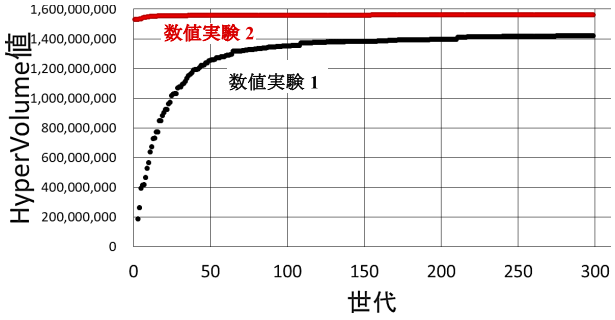

図 12　HyperVolume 値比較（数値実験 1，2）

Fig. 12　Comparison of HyperVolume values (Numerical experiment 1 and 2).

表 5　実務データのプロジェクトプロファイル

Table 5　The project profile of our practical data.

項目	内容
プロジェクト内容	チケット販売代行会社における顧客応対記録保管システムの構築
実施年	2010 年代
国	日本
スコープ	要件定義．基本設計，詳細設計，構築，コーディング，単体テスト，結合テスト
製品構成	Windows サーバ，Windows クライアント，記録保管ソフトウェア，専用端末
拠点	5 か所（青森，吉祥寺，神戸，山口，データセンタ）
プロジェクト参加要員数	最大 100 人
利用ユーザ数	約 3000 人
タスク数	962 タスク

目でも数値実験 2 の HyperVolume 値には届いていない．HyperVolume 値は大きいほどより多くの空間を支配していることを示すので，3 つのパレートフロントの端にある厳密解を初期集団に入れることが解集合の広がり（多様性）に対して有効であったことが分かる．

計算時間は厳密解遺伝子 (a) が 0.45 秒，厳密解遺伝子 (b) が 0.04 秒，厳密解遺伝子 (c) が 0.02 秒，遺伝的アルゴリズムの 0 世代目から 300 世代目計算完了までが 3 分 14 秒であった．

5.　962 タスクでの数値実験とその結果に対するインタビューと考察

5.1　数値実験 3

提案するソフトウェアの目標である 1,000 タスクを 1 時間以内で計算完了することを確認するため，実務で使用された 962 タスクのデータを用いて，数値実験 3 を行う．実務データのプロジェクトプロファイルを**表 5** に示す．なお，実務データであるためプロジェクトを特定できないようにタスク名称などを一部加工しているが，数値には手を加えていない．

要員リストは 100 人分，遺伝的アルゴリズムの世代数は 600，個体数は 600 とする．交叉率，突然変異率，Simulated Binary Crossover と Polynomial Mutation の η，実験環境は数値実験 1 と同じである．実験結果を**図 13** に示す．

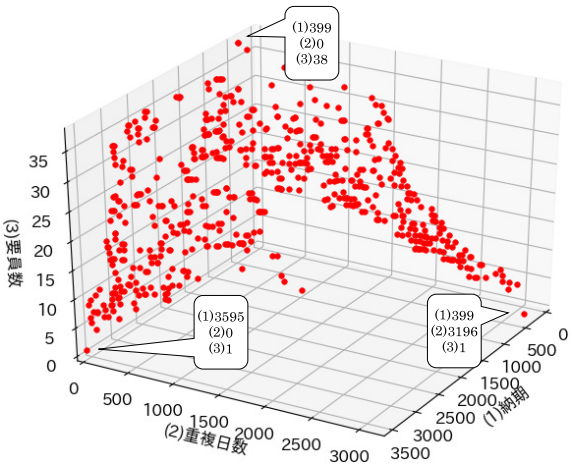

図 13　実験結果（数値実験 3）

Fig. 13　Experimental results (Numerical experiment 3).

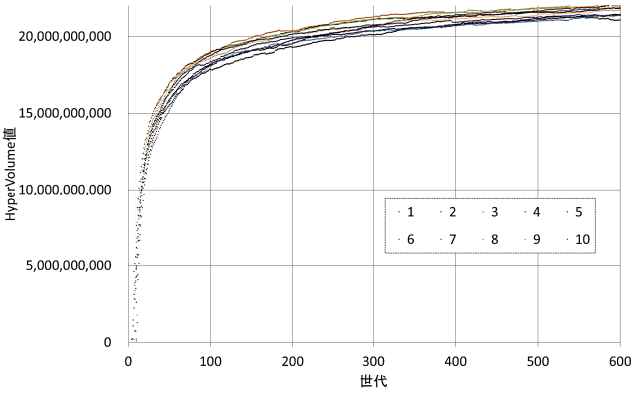

図 14　各回の HyperVolume 値推移の比較（数値実験 3）

Fig. 14　Comparison of transition of HyperVolume values (Numerical experiment 3).

表 6　各回の最終世代の HyperVolume 値および計算時間（数値実験 3）

Table 6　HyperVolume values of final generation and calculation time (Numerical experiment 3).

回	最終世代 HyperVolume 値	計算時間（時:分:秒.ミリ秒）				
		厳密解(a) ※	厳密解(b) ※	厳密解(c) ※	NSGA-II	合計
1	21,429,451,764	0:00:01.110	0:00:00.480	0:00:00.180	0:53:09.960	0:53:11.730
2	21,858,926,262	0:00:01.080	0:00:00.540	0:00:00.190	0:53:06.910	0:53:08.720
3	21,995,036,311	0:00:00.810	0:00:00.470	0:00:00.210	0:49:17.290	0:49:18.780
4	21,361,867,312	0:00:01.140	0:00:00.660	0:00:00.220	1:11:34.110	1:11:36.130
5	21,067,590,025	0:00:01.390	0:00:00.630	0:00:00.240	1:09:30.450	1:09:32.720
6	22,085,705,932	0:00:01.050	0:00:00.650	0:00:00.000	0:52:25.060	0:52:26.760
7	21,373,924,790	0:00:01.420	0:00:00.460	0:00:00.160	0:48:53.600	0:48:55.630
8	22,021,614,057	0:00:01.000	0:00:00.840	0:00:00.160	0:47:46.220	0:47:48.210
9	21,705,267,769	0:00:00.630	0:00:00.490	0:00:00.390	0:48:03.100	0:48:04.600
10	21,831,719,278	0:00:00.640	0:00:00.000	0:00:00.140	0:48:36.860	0:48:37.640
平均	21,673,110,350	0:00:01.026	0:00:00.522	0:00:00.189	0:54:14.355	0:54:16.093
標準偏差	344,553,385	0:00:00.272	0:00:00.219	0:00:00.096	0:08:50.669	0:08:50.891

※：厳密解(a)〜(c)は 4.1 節で示した 3 つの厳密解遺伝子の生成を表す

数値実験 2 と同様に，多様性を持った面を得ることができた．遺伝的アルゴリズムは確率的解探索法であるため，安定的に良い解集合を獲得できることを確認する．同一データ同一環境で 10 回実行した際の各回の HyperVolume 値推移の比較を**図 14** に，各回の最終世代の HyperVolume 値および計算時間を**表 6** に示す．HyperVolume の参照点は $(6000, 1500, 40)$ とした．

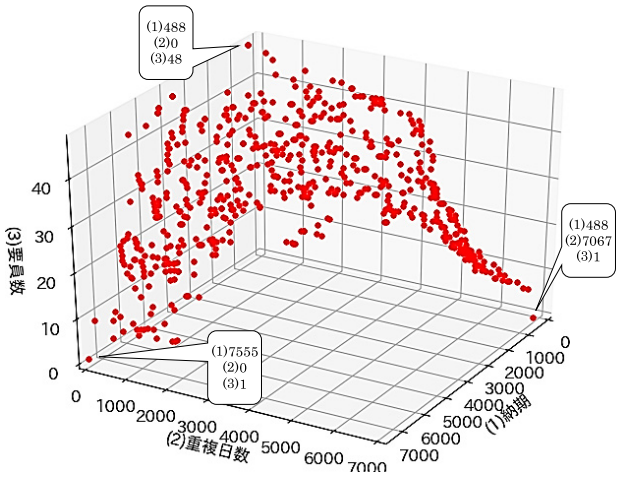

図 15　実験結果（数値実験 4）

Fig. 15　Experimental results (Numerical experiment 4).

図 14 は，各回とも世代が進むに従って HyperVolume 値が増加し，増分が次第に減少する曲線を描いていることを示している．各世代において各回ともに類似した増分率となっているように見受けられる．これは数値実験 2 で示した結果どおり，各回とも解が支配する空間について世代が進むにつれて広がっていることを表している．参考までに，最終世代 HyperVolume 値の分布が正規分布に従うと仮定した場合の 99%信頼区間は 21,319,016,891～22,027,203,810 であり，10 回中 5，6 回目以外の計 8 回が信頼区間内であった．

提案するソフトウェアの目標である 1,000 タスクを 1 時間以内に計算完了する点については，実務データの都合上 1,000 タスクに 4%ほど不足している 962 タスク分ではあるが，平均 54 分 16 秒と 1 時間以内を達成できていることが分かった．

5.2　数値実験 4

提案するソフトウェアがデータによらず安定した結果を出力することを確認するため，本研究のモデルに応じてランダムに生成した人工データを用いて数値実験 4 を行う．人工データは，以下の手順で作成する．

(a) 1,000 タスクを生成する．
(b) 2.2 節の制約条件を満たす先行制約をランダムで生成する．
(c) 各タスクの所要日数に 1～10 の乱数を割り当てる．

そのほかの条件は，数値実験 3 と同じく要員リストが 100 人分，遺伝的アルゴリズムの世代数が 600，個体数が 600 とする．交叉率，突然変異率，Simulated Binary Crossover と Polynomial Mutation の η，実験環境は数値実験 1 と同じである．実験結果を**図 15** に示す．

ランダムに生成したデータでも，あらかじめ算出した厳密解であるパレートフロントの端 3 点を含む多様性を持った面を得ることができた．

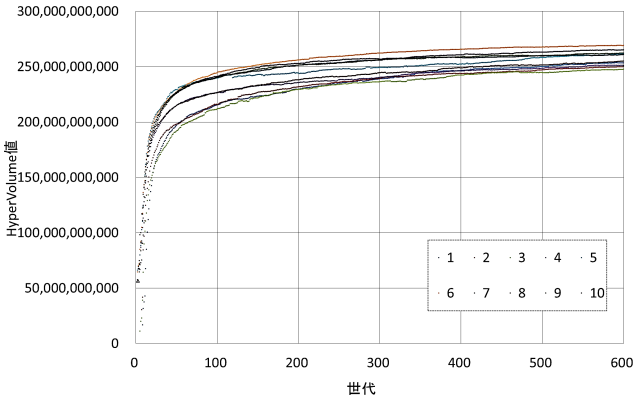

図 16　各回の HyperVolume 値推移の比較（数値実験 4）

Fig. 16　Comparison of transition of HyperVolume values (Numerical experiment 4).

表 7　各回の最終世代の HyperVolume 値および計算時間（数値実験）

Table 7　HyperVolume values of final generation and calculation time (Numerical experiment 4).

回	最終世代 HyperVolume 値	計算時間（時:分:秒.ミリ秒）				
		厳密解(a)※	厳密解(b)※	厳密解(c)※	NSGA-II	合計
1	253,573,185,276	0:00:00.770	0:00:01.000	0:00:00.360	0:53:50.620	0:53:52.740
2	250,505,419,288	0:00:00.800	0:00:00.830	0:00:00.470	0:53:26.180	0:53:28.270
3	247,702,919,827	0:00:00.750	0:00:00.850	0:00:00.470	0:53:46.620	0:53:48.690
4	250,987,148,053	0:00:01.440	0:00:01.420	0:00:00.750	1:13:01.450	1:13:05.060
5	261,596,765,054	0:00:00.770	0:00:00.920	0:00:00.420	0:50:37.850	0:50:39.970
6	269,233,657,096	0:00:00.790	0:00:00.880	0:00:00.440	0:50:43.470	0:50:45.580
7	260,974,471,624	0:00:01.050	0:00:00.900	0:00:00.490	0:50:11.850	0:50:14.280
8	255,068,520,799	0:00:00.840	0:00:00.800	0:00:00.370	0:50:35.900	0:50:37.910
9	262,403,256,164	0:00:00.770	0:00:00.780	0:00:00.430	0:50:19.160	0:50:21.140
10	265,157,580,477	0:00:00.800	0:00:00.800	0:00:00.450	0:50:26.860	0:50:28.910
平均	257,720,292,366	0:00:00.878	0:00:00.917	0:00:00.464	0:53:41.996	0:53:44.255
標準偏差	7,128,857,342	0:00:00.214	0:00:00.190	0:00:00.110	0:06:57.461	0:06:57.922

※：厳密解(a)～(c)は 4.1 節で示した 3 つの厳密解遺伝子の生成を表す

数値実験 3 と同様に，安定的に良い解集合を獲得できることを確認する．同一データ同一環境で 10 回実行した際の各回の HyperVolume 値推移の比較を**図 16** に，各回の最終世代の HyperVolume 値および計算時間を**表 7** に示す．HyperVolume の参照点は $(6000, 3000, 50)$ とした．

図 16 を見ると，各回の HyperVolume 値の推移が数値実験 3 と類似した曲線を描いていることが分かる．また，同様に各世代において各回ともに類似した増分率となっているように見受けられる．参考までに，最終世代 HyperVolume 値の分布が正規分布に従うと仮定した場合の 99%信頼区間は 250,394,049,559～265,046,535,173 であり，10 回中 3，6，10 回目以外の計 7 回が信頼区間内であった．

また，研究の目的である 1,000 タスクを 1 時間以内に計算完了する点については，平均 53 分 44 秒と 1 時間以内を達成できていることが分かった．数値実験 3 のデータは 962 タスクであり数値実験 4 と比較して 4%ほどタスク数が少ないので参考値となるが，数値実験 3 と数値実験 4 の各回の計算時間合計の 2 群について有意水準 5%でウェルチの t 検定を行ったところ，p 値は 0.88 となり 2 群の平均に差がないことは棄却されなかった．

5.3　実務家へのインタビューによる評価

提案するソフトウェアに対する定性的な評価を行う．数値実験 3 は実務データを使用したため，その実験結果を用いて本研究の目的の達成度合いについて実務家の評価を受けることとする．

(1)　評価方法

プロジェクトマネージャとして現在活動している実務家 4 名を対象として，提案するソフトウェアの現場への導入を前提とした本研究の目的の達成度合いについてヒアリングを行う．対象者には，提案するソフトウェアの開発の背景，仕様，962 タスクの出力結果（計 600 ガントチャート）を提示し，あらかじめ設計した定型用紙への回答を依頼する．定型用紙は辻ら [37] を参考にし，自由記入欄およびリッカートスケールで構成する．定型用紙を回収後，より詳細なヒアリングが必要な場合は，追加でメールと電話で個別に確認を行う．4 名のヒアリング対象者のプロフィールを**表 8** に示す．

(2)　ヒアリング日時

2017 年 11 月 21 日～2017 年 12 月 1 日

(3)　ヒアリング項目

ヒアリング項目を**表 9** に示す．

(4) **回答結果**

設問 1：提案するソフトウェアによりプロジェクトマネージャにスケジュール作成ノウハウが集中するという属人化を解消できるか（4 段階），およびその理由

4 段階評価では 4 人中 1 人が「4：たいへん貢献できると思う」を選び，残る 3 名が「3：それなりに貢献できると思う」を選び，総じて評価は高かった．内容としては，たたき台としての利用であれば十分貢献できる，機械的にアウトプットされることで手間が大幅に省けるという回答が得られた．一方で，リスクを考慮したスケジュールを出力してほしいという回答，ガントチャートを選ぶための指標がほしいという回答も得られた．

設問 2：962 タスクのスケジュールを手動で作成した場合の所要時間

4 人の回答はそれぞれ，20 時間，24 時間，40 時間，48 時間であった．1 日の業務時間を 8 時間とすると，これは 2.5 日～6 日かかることになる．プロジェクトマネージャがこれだけの時間，スケジュール作成作業にかかりきりになるということは，1.1 節で示したスケジュール作成に対するプロジェクトマネージャの負荷軽減が喫緊の課題であるということの証左となるといえる．提案するソフトウェアによってその作業が仮に 3 割でも削減できた場合，実質 0.75 日～1.8 日の削減となる．これは多忙なプロジェクトマネージャにとって貴重な時間を確保できることを意味する．

設問 3：提案するソフトウェアによりプロジェクトマネジメント業務の効率が向上するか（4 段階），およびその理由

4 段階評価では 4 人全員が「3：それなりに貢献できると思う」を選び，一定の評価を得た．内容としては網羅的に出力される点が良いという回答などがあったが，あわせて機能追加の要望も多かった．具体的には，特定タスクに特定要員を割り当てたいという要望，スキルマッチングを行いたいという要望であった．

設問 4：提案するソフトウェアは 962 タスクを約 1 時間で作成するが，実務で使用するにあたって十分短いか（4 段階），およびその理由

本設問は回答が割れた．4 段階評価では「4：たいへん短いと思う」が 1 名，「3：それなりに短いと思う」が 2 名，「2：少し長いと思う」が 1 名であった．内容としては，1 時間であれば実務に十分に利用可能という回答に対し，何回か条件を変更しながら出力する運用を想定すると 20 分程度にしてほしいという回答があった．

設問 5：提案するソフトウェアについて評価できる点

最適なスケジュールを選定できるという本研究の目的通りの評価だけでなく，考慮漏れ防止など基本的なミスを防ぐことにも有用との評価を得た．一方で 600 ものガントチャートのなかから選ぶためにはなんらかの基準がほしいとの回答もあった．これは，現状の目的関数である納期，重複日数，要員数以外に，各ガントチャートの特性を示す指標があればより選択しやすくなるという意見である．

設問 6：提案するソフトウェアについて追加・改善すべき点

表 8　ヒアリング対象者のプロフィール
Table 8　The profile of persons who answered the questionnaire.

項目＼対象者	A	B	C	D
年齢	30 代	40 代	50 代	40 代
プロジェクトマネージャ歴	3~5 年	10 年以上	10 年以上	5~10 年
主な経歴	金融系 SE	商社系 SE	メーカ系 SE	公共系 SE

表 9　ヒアリング項目
Table 9　The items of questionnaire.

項番	内容
設問 1	提案するソフトウェアによりプロジェクトマネージャにスケジュール作成ノウハウが集中するという属人化を解消できるか（4 段階評価），及びその理由
設問 2	962 タスクのスケジュールを手動で作成した場合の所要時間
設問 3	提案するソフトウェアによりプロジェクトマネジメント業務の効率が向上するか（4 段階評価），及びその理由
設問 4	提案するソフトウェアは 962 タスクを約 1 時間で作成するが，実務で使用するにあたって十分短いか（4 段階評価），及びその理由
設問 5	提案するソフトウェアについて評価できる点
設問 6	提案するソフトウェアについて追加・改善すべき点

現実にはプロジェクト規模が大きくなるにつれて要員数は指数関数的に増加することを考慮してほしいという回答，タスクの難度の重みづけでバッファを積めるようにしてほしいという回答，タスクの属性情報をより拡充してほしいという回答があった．

5.4　考察

数値実験 3 では 962 タスクを平均 54 分 16 秒，数値実験 4 では 1,000 タスクを平均 53 分 44 秒で計算が完了した．これは目標である 1,000 タスク 1 時間以内を達成できたといえる．計算時間は，当然世代数に依存する．数値実験 3 と数値実験 4 の世代ごとの HyperVolume 値の推移を見る限り，600 世代目付近ではほぼ横ばいになっているため，世代数は 600 で十分あるといえる．

数値実験 3 は実務で使用された現実のデータ，数値実験 4 はランダムで作成された人工データを用いた．タスク数が約 4%異なるため参考値ではあるが，数値実験 3 と数値実験 4 の各回の計算時間の合計の 2 群についてウェルチの t 検定を有意水準 5%で行ったところ，p 値が 0.88 となり 2 群それぞれの平均に差があるとはいえなかった．

実務家へのインタビューでは，リッカートスケールによる点数と自由記入により提案するソフトウェアを評価した．点数は 4 点満点とした．プロジェクトマネージャにスケジュール作成ノウハウが集中するという属人化を解消できるかという設問は平均 3.25 点，962 タスクを約 1 時間で生成は実務で使用するにあたって十分短いかという設問は平均 3 点，プロジェクトマネジメント業務の効率が向上するかという設問は平均 3 点と，一定の評価を得られた．さらなる機能の追加や，計算時間は 1 時間以内ではなく 20 分以内にしてほしいという意見もみられたが，これは提案するソフトウェアについて，実務で利用可能という一定の評価がなされて，そのうえでの期待の表れが示されたといえる．根幹となる機能や性能については総じて問題ないといえる．

以上より，IT プロジェクトマネジメントにおいてスケジューリング作業の効率向上によるプロジェクトマネージャの負荷軽減という課題に対して，提案するソフトウェアは有効であるといえる．

6.　妥当性への脅威と限界

実証的ソフトウェア工学の実証研究の結果には，伝統的に妥当性への 4 つの脅威がある．それは，構成概念妥当性，外部妥当性，内部妥当性，信頼性である [38], [39]．それぞれの妥当性に対する脅威について以下に示す．

6.1　構成概念妥当性

構成概念妥当性は，測定するために使用した尺度や基準が適切であることを表す．ここでは本研究のモデルおよび数値実験の構成概念妥当性の 2 つを示す．

(1)　モデル

本研究の目的関数は，1.3 節で示した時間，コスト，品質を目的関数とする多目的最適化をメタヒューリスティクスアルゴリズムで解くという先行研究に基づいている．本研究では目的関数から品質を外し，重複日数を加えたが，これは 1.3 節および 2.1 節に示したとおり，IT プロジェクトの特性を反映したものである．現時点では妥当性は高いといえるが，景気などの影響で IT 投資が縮小した場合や，中国や東南アジア諸国の IT エンジニアの台頭などにより人材不足の解消や能力の底上げが起こる場合は，目的関数の妥当性に影響を及ぼす可能性がある．

本研究の制約条件は，2.2 節に示したとおり標準的な IT プロジェクトスケジュールの制約条件に従っているため，妥当性は高いといえる．

メタヒューリスティクスアルゴリズムによるプロジェクトスケジュール問題の解法は Tavana ら [15] など近年多くみられ，また 2.2 節に示したとおり NSGA-II は最も多く使用されているため，妥当性は高いといえる．

(2)　数値実験

数値実験で測定した HyperVolume 値は，多目的最適化における進化的手法の性能を測るために最も使用されているものであり [40]，妥当性は高いといえる．また計算時間は数値実験を行った PC の内蔵時計で得たものであり，手動や目視による測定と比較して誤差は小さく妥当性は高いといえる．インタビューで使用したリッカートスケールは各種調査で使用される順序尺度データとして一般的なものであり [37]，妥当性は高いといえる．

6.2　外部妥当性

外部妥当性は，結果が一般化される程度を表す．本研究では，実務で使用されたデータとランダムで作成した 2 つのデータを用いて数値実験を行った．実務で使用されたデータは，現実にある無数の IT プロジェクトの 1 つにすぎないため，ランダムで作成したデータを用いて追加で数値実験を行った．2 つの数値実験では HyperVolume 値の世代ごとの推移，複数回実行時の結果，計算時間のいずれも類似した傾向がみられた．また，実務で使用されたデータによる実験結果を実務家に提示したところ，一定の評価を得た．これらにより，数値実験で使用した 2 つのデータにおいては，提案するソフトウェアはスケジューリング作業の効率向上を実現するといえるが，2 つのデータ以外のデータでは結論が変わる可能性は排除できない．実務上の課題を解消するという提案ソフトウェアの目的からも，さらなる現実のプロジェクトのデータでの追試を実施する必要がある．しかし，プロジェクトのスケジュールデータのほとんどは社外秘であり，実務データを集めて一般化することは当然限界がある．PSPLIB [41] を活用する方法もあ

るが，ジェネレータによって疑似的に作られたデータであるうえ，本研究のモデルに合うようにデータを修正する必要があるため，実務データを採用した実験とはいえず，外部妥当性を補強するものにはなり得ない可能性がある．一般化を主張するための十分なデータセット数での実験はきわめて困難であるが，本研究では現実と人工の異なる出自のデータを用いた数値実験を行っていること，そのうち1つは現実的な実務で使用されたデータであり実験結果について実務家に一定の評価を得たことは，妥当性を補強できるといえる．

6.3　内部妥当性

内部妥当性とは，独立変数が従属変数に与える因果関係について結論が導き出される程度を表す．本研究の場合，数値実験3および数値実験4においてそれぞれ10回の試行を行った結果に対する標準誤差の評価が該当する．サンプルサイズ10は大きいとはいえず，また最終世代HyperVolume値の分布は不明ではあるが，仮に正規分布に従うと仮定した場合，各回の最終世代HyperVolume値は99%信頼区間内に数値実験3は10回中8回，数値実験4は10回中7回が含まれていた．よって，おおむね妥当性はあるといえる．

6.4　信頼性

信頼性は，再現可能性を表す．本研究では，モデル，提案するソフトウェアの仕様，数値実験の実験環境，実験データの特性を提示しているため，再現性は高く，信頼性は高いといえる．

7.　おわりに

ITプロジェクトにおいては計画どおりの遂行を阻害する事象が多発する点，プロジェクトマネージャはスケジューリングを最優先する点，プロジェクトマネージャはつねに多忙でストレスフルである実状に加えIT人材不足がプロジェクトマネージャの負荷を増大させている点から，ITプロジェクトマネジメントにおいてスケジューリング作業の効率向上によるプロジェクトマネージャの負荷軽減が喫緊の課題であるため，本研究はこの課題の解消を目的とした．課題解消のために，プロジェクトの納期とコストのトレードオフを考慮したITプロジェクト向けの自動スケジューリングソフトウェアを提案した．メタヒューリスティクスの様々な手法を用いたプロジェクトスケジューリング手法は，単目的/多目的問わず数多く提案されてきているが，先行研究ではITプロジェクト特有の目的関数である重複日数は設定されていない．そこで本研究では，重複日数を目的関数に追加し，ITプロジェクトならではの制約条件を設定した場合に，解集合の探索性能を向上させる手法の提案に主眼をおいた．

提案するソフトウェアで100タスクのデータを用いて数値実験をしたところ，解集合について多様性が見られなかった．対策を検討したところ，本研究のモデルではパレートフロントの端の厳密解を少ない計算量で算出できる特殊なスケジュール問題であることが分かったため，該当の厳密解を遺伝的アルゴリズムの初期集団に含めることとしたところ，多様性を持った解集合を確保できた．これは先行研究ではみられない，ITプロジェクトスケジューリングを扱う本研究特有の工夫であった．962タスクからなる実務で使用された現実のプロジェクトデータ，および1,000タスクからなるランダムで生成された人工のプロジェクトデータの2つのデータを用いて数値実験を行ったところ，提案するソフトウェアの目標計算時間である1時間以内の達成を確認した．そのうえで，実務で使用された現実のプロジェクトデータの実験結果を複数の実務家に提示しインタビューを行ったところ，一定の評価を得ることができた．加えて実験環境のハードウェアは，2017年時点の標準的なビジネス用ノートPCであるため，提案するソフトウェアは性能的に実務で十分に活用可能である．

以上より，提案するソフトウェアによって，ITプロジェクトマネジメントにおいてスケジューリング作業の効率向上によるプロジェクトマネージャの負荷軽減という本研究の目的は達成できたといえる．

最後に今後の課題2点を示す．1点目はさらなる追加実験の実施である．6.2節で示したとおり，本研究では現実と人工の2つのデータセットによる数値実験を行った．提案するソフトウェアがデータによらず有効である点を示すうえでも，異なる特性のデータを用いて実験をする必要がある．プロジェクトデータはほとんどの場合社外秘であり外部に持ち出すことができないため，提案するソフトウェアを実際のITプロジェクトに提供し，使用させたうえで結果を収集するほうが現実的である．幸い実務家へのインタビューにおいて，提案するソフトウェアを使ってみたいという声があがっていたため，タイミングを見計らって実証実験をしたい．2点目は，インタビューであがった実務家が期待する機能の実装である．具体的には，タスクと要員のスキルマッチング機能の追加，タスク属性情報の拡充，各ガントチャートの特性を示す指標の開発，計算時間を20分以内としたさらなる高速化である．機能追加や拡充は難しくないが，指標の開発はスケジュール特性の表現方法など検討に時間を要する可能性がある．同じスケジュールでも，プロジェクトの状況に応じてとらえ方が変わるため，新たな入力と関数が必要になるだろう．

謝辞　本研究はJSPS科研費JP26350430, JP17K00037の助成を受けたものである．また，原稿を注意深くお読みいただき適切な助言をいただいたことに対して，2名の匿名査読者に感謝する．

参考文献

[1] 日経 BP 社：第 2 回プロジェクト実態調査，日経コンピュータ，2008 年 12 月 1 日号，pp.38–49 (2008).

[2] Rothman, J.: Manage It!: Your Guide to Modern, Pragmatic Project Management, Pragmatic Bookshelf, pp.xvii–xix (2007).

[3] 初田賢司，尾中章行，小川純平：特集 2「本物のプロマネ」七つの条件，日経 SYSTEMS，2016 年 2 月号，pp.52–59 (2016).

[4] 日経 BP 社：特集 1 プロジェクト遂行編：気付いたときにはもう遅い，手戻り・遅延は起こさない，日経 SYSTEMS，2010 年 4 月号，pp.62–69 (2010).

[5] Project Management Institute（編），PMI 東京支部（監訳）：プロジェクトマネジメントプリンシプル：変革の時代を生き抜くための人と組織の挑戦，pp.230–255 (2006).

[6] 布川　薫：重要性増すモダン PM　そのシステム開発での実践体系を知る，日経 IT プロフェッショナル，2002 年 7 月号，pp.152–157 (2002).

[7] Gartner: Gartner Worldwide IT Spending Forecast 3Q17 Update (2017).

[8] 日本情報システム・ユーザー協会：第 23 回企業 IT 動向調査 2017 (2017).

[9] Big tech aims at talent shortage with 'Hour of Code' campaign, Technology News, REUTERS (2014).

[10] 独立行政法人情報処理推進機構 IT 人材育成本部：IT 人材白書 2017，pp.111–113 (2017).

[11] Apps Run The World: Top 10 Project Portfolio Management Software Vendors and Market Forecast 2015-2020 (2016), available from ⟨https://www.appsruntheworld.com/top-10-project-portfolio-management-software-vendors-and-market-forecast-2015-2020⟩.

[12] 初田賢司，神子秀雄：思い込みや楽観的な予測を排除緻密な WBS 定義こそが重要—作業計画とスケジュール作成の実践知識特集 1　進ちょく管理の大本命 EVM を極第 2 部　作業計画とスケジュール作成の実践，日経 IT プロフェッショナル，2004 年 12 月号，pp.42–47 (2004).

[13] 日本情報システム・ユーザー協会：第 20 回企業 IT 動向調査 2014 (2014).

[14] 初田賢司，吉兼　寛：講座　はじめての WBS の作り方［第 5 回］メンバーアサイン　一つの作業に担当 1 人が原則，日経 SYSTEMS，2011 年 2 月号，pp.74–79 (2011).

[15] Tavana, M. et al.: A new multi-objective multi-mode model for solving preemptive time-cost-quality trade-off project scheduling problems, *Expert Systems with Applications*, Vol.41, pp.1830–1846 (2014).

[16] Szmerekovsky, J.G. and Venkateshan, P.: An integer programming formulation for the project scheduling problem with irregular time-cost tradeoffs, *Computers & Operations Research*, Vol.39, pp.1402–1410 (2012).

[17] Vanhoucke, M. and Debels, D.: The discrete time/cost trade-off problem: Extensions and heuristic procedures, *Journal of Scheduling*, Vol.10, pp.311–326 (2007).

[18] Xu, J. et al.: Discrete time-cost-environment trade-off problem for large-scale construction systems with multiple modes under fuzzy uncertainty and its application to Jinping-II Hydroelectric Project, *International Journal of Project Management*, Vol.30, pp.950–966 (2012).

[19] Deb, K. et al.: A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evolutionary Computation, Vol.6, No.1, pp.182–197 (2002).

[20] 定量的品質予測のススメ 1，独立行政法人情報処理推進機構，p.6 (2008).

[21] 大坂　宏：海外大型建設プロジェクトのコミュニケーション管理—コミュニケーション管理手法の IT プロジェクトへの適用の可能性を探る，プロジェクトマネジメント学会誌，Vol.6, No.4, pp.58–61 (2004).

[22] Prechelt, L.: An empirical study of working speed differences between software engineers for various kinds of task, working paper (2000).

[23] 池上敦子：ナース・スケジューリング—調査・モデル化・アルゴリズム，統計数理，Vol.53, No.2, pp.231–259 (2005).

[24] Luna, F. et al.: The software project scheduling problem: A scalability analysis of multi-objective metaheuristics, *Applied Soft Computing*, Vol.15, pp.136–148 (2014).

[25] 初田賢司，尾中章行：講座　はじめての WBS の作り方［第 4 回］工数とスケジュール，日経 SYSTEMS，2011 年 1 月号，pp.74–79 (2011).

[26] 布川　薫：プロジェクトマネジメントの理論と実践 No.5，日経 IT プロフェッショナル，2002 年 11 月号，pp.130–135 (2002).

[27] 佐藤寛之，石渕久生：進化型多数目的最適化の現状と課題，オペレーションズ・リサーチ：経営の科学，Vol.62, No.3, pp.156–163 (2017).

[28] 三毛功子：定量的品質管理，独立行政法人情報処理推進機構 (2011).

[29] 井ノ口伸人ら：システム基盤構築工数見積もりモデルの継続的改善と普及展開，IPSJ/SIGSE Software Engineering Symposium SES2014 (2014).

[30] 初田賢司，吉兼　寛：はじめての WBS の作り方［第 2 回］タスクの洗い出し（その 1），日経 SYSTEMS，2010 年 11 月号，pp.72–77 (2010).

[31] Fortin, F.-A. et al.: DEAP: Evolutionary Algorithms Made Easy, *Journal of Machine Learning Research*, Vol.13, pp.2171–2175 (2012).

[32] 神嶌　敏：Python による科学技術計算の概要，電子情報通信学会・情報処理学会第 15 回科学技術フォーラム，入手先 ⟨http://www.kamishima.net/archive/scipy-overview.pdf⟩ (2016).

[33] Hagberg, A.A. et al.: Exploring network structure, dynamics, and function using NetworkX, *Proc. 7th Python in Science Conference* (*SciPy2008*), pp.11–15 (2008).

[34] Norman, A.: Python-Gantt (2014), available from ⟨http://xael.org/pages/python-gantt-en.html⟩.

[35] van der Walt, S. et al.: The NumPy Array: A Structure for Efficient Numerical Computation, *Computing in Science & Engineering*, Vol.13, pp.22–30 (2011).

[36] Fonseca, C.M. et al.: A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers, TIK-Report, No.214 (2006).

[37] 辻　新六，有馬昌宏：アンケート調査の方法—実践ノウハウとパソコン支援，朝倉書店 (1987).

[38] Runeson, P. and Höst, M.: Guidelines for conducting and reporting case study research in software engineering, *Empirical Software Engineering*, Vol.14, pp.131–164 (2009).

[39] Zhang, Y. et al.: Comparing the performance of metaheuristics for the analysis of multi-stakeholder tradeoffs in requirements optimization, *Information and Software Technology*, Vol.53, No.7, pp.761–773 (2011).

[40] Riquelme, N. et al.: Performance metrics in multi-objective optimization, *Computing Conference* (*CLEI*), IEEE (2015).

[41] Kolisch, R. and Sprecher, A.: PSPLIB — A project scheduling problem library, *European Journal of Operational Research*, Vol.96, No.1, pp.205–216 (1997).

小林 敬明　（正会員）

東京理科大学理学部応用数学科卒業．首都大学東京大学院社会科学研究科経営学専攻博士前期課程修了．株式会社日立製作所，マイクロソフト株式会社，丸紅情報システムズ株式会社でシステムエンジニアリング業務やプロジェクトマネジメント業務に従事．現在，株式会社 TransRecog 代表．

森口 聡子　（正会員）

東京工業大学大学院情報理工学研究科博士後期課程修了，博士（理学）．科学技術振興機構研究員，上智大学助教，産業技術大学院大学助教を経て，2013年より首都大学東京准教授．組合せ最適化問題に対するアルゴリズム，離散凸構造に関する研究に従事．