# 卒業論文

# Web内容マイニングによる個別知識の共通単語を介した相関ルールからの発想支援

Parallel Distributed Solution
of Fuzzy Random Multiobjective Optimization
Utilizing Work History in Schedule Planning

富山県立大学　電子・情報工学科
1615052　山元 悠貴
指導教員　奥原 浩之 教授
令和2年2月19日

# 目 次

# 図一覧

# 表一覧

# 記号一覧

以下に本論文において用いられる用語と記号の対応表を示す.

| 用語 | 記号 |
|---|---|
| 先行作業 | $i$ |
| 後続作業 | $j$ |
| プロジェクトの総作業数 | $n$ |
| 従事者グループ | $k$ |
| 依頼候補の従事者グループ数 | $w$ |
| 作業の所要時間 | $t_{ij}$ |
| 作業を従事者グループに依頼したときの費用 (ファジィ・ランダム変数) | $\tilde{\tilde{c}}_{ik}$ |
| クリティカルパスを選択する 0-1 変数 | $x_{ij}$ |
| 依頼する従事者グループを選択する 0-1 変数 | $y_{ik}$ |
| ファジィ・ランダム変数のメンバシップ関数 | $\mu_{\tilde{\tilde{c}}_{ik}}$ |
| 中心値 (平均値) | $\bar{d}_{ik}$ |
| 左右の広がりパラメータ | $\beta_{ik}$, $\gamma_{ik}$ |
| ファイジィ目標 | $\tilde{G}$ |
| ファジィ目標の最良値 | $g^0$ |
| ファジィ目標の最悪値 | $g^1$ |
| ファイジィ目標のメンバシップ関数 | $\mu_{\tilde{G}}$ |
| ファジィ目標は満たされる可能性の度合い (可能性測度) | $\Pi_{\tilde{\tilde{c}}_{ik}}$ |
| 満足基準値 | $h$ |
| 擬逆関数 | $L^*(h)$, $\mu^*_{\tilde{G}}(h)$ |
| 確率変数の分布関数 | $F$ |

# 第1章

# はじめに

## § 1.1　本研究の背景

現在，情報処理技術の発達に伴って，コンピュータが人間の創造的な問題解決・思考活動を支援する発想支援システムの研究が進んでいる．これからの時代はよりアイデア発想が重要になってくると考える．創造活動をする際，人間は言葉で表現することが多く，その言葉を用い，自分の発想を整理し，修正を行う．実際，認知心理学では，人間にとって思考と言語は深い関連があると言われている [1].

そのため，コンピュータが発想支援を行うためには，人間の言葉を知る必要がある．しかし，自然言語を機械に理解させるのは非常に困難であり，機械独特の自然言語の分析方法が用いられている [2].

発想を支援することは，人間からアイデアが出やすくすることと考えられる．そのため，機械が発想支援を行うには，人間が考えるために使う手段である自然言語の分析方法を利用する必要がある．そこで現状の発想支援について，複数のキーワードからの発想支援に着目した．

典型的には発想支援システムはコンピュータを中心にしたシステムとして実現される．文章を書いたり，図を使ったり，記憶を呼び起こしたりすることは創造的に考えるための条件だといってよいだろうが，これらの行為を実行するうえでの人間の認知的な制限をコンピュータの記憶装置や出力画像などを用いて軽減することができる．

発想支援システムの構築は計算機の最も高度な知的利用技術のひとつで，計算機 が出現して以来の計算機科学者、人工知能研究者の究極の夢のひとつであり，これ について心理学，社会学，工学などの各方面から様々な試みが行われています．情 報技術の進歩により従来は実現が困難であった計算機によるアイデア生成支援が具 体化できるようになってきています．また産業界においても個人個人の独創性を引 き出せるような知的環境の構築が望まれており，様々な発想支援システムの開発が 試みられ，一部は既に実用化されているものもあります．

## § 1.2　本研究の目的

現在，情報処理技術の発達に伴って，コンピュータが人間の創造的な問題解決・思考活動を支援する発想支援システムの研究が進んでいる．これからの時代はよりアイデア発想が重要になってくると考える．創造活動をする際，人間は言葉で表現することが多く，その言葉を用い，自分の発想を整理し，修正を行う．実際，認知心理学では，人間にとって思考と言語は深い関連があると言われている [1].

そのため，人工知能が発想支援を行うためには，人間の言葉を知る必要がある．しかし，自然言語を機械に理解させるのは非常に困難であり，機械独特の自然言語の分析方法が用いられている [2].

発想を支援することは，人間からアイデアが出やすくすることと考えられる．そのため，機械が発想支援を行うには，人間が考えるために使う手段である自然言語の分析方法を利用する必要がある．そこで現状の発想支援について，複数のキーワードからの発想支援に着目した．

本研究では，複数のキーワードについてのネットワークの作成とキーワードの関連語をランキング形式で表示する事を考える．

本研究では，Google 検索を用いた複数のキーワードからの発想支援について考える．既存の発想支援システムでは 1 つのキーワードから発想を広げていくものが多かった．そこで，複数のキーワードから関連する単語を用いてランク付けすることで，より効率の良い発想支援につながると考えた．

## § 1.3　本論文の概要

本論文は次のように構成される．

**第 1 章** 本研究の概要と目的について説明した．

**第 2 章** 既存の発想支援システムと課題について述べる．

**第 3 章** 本研究の発想支援システム構築に用いる技術について述べる．

**第 4 章** ベイジアンネットワークを用いた提案手法についてまとめる．

**第 5 章** まとめと今後の課題を述べる．

# 第2章

# 発想支援システムとは

## § 2.1　発想支援システムとは

発想支援システム

発想支援システムは文字通り人間の創造的問題解決における発想のプロセスを支援するコンピュータシステムである．この発想のプロセスでは, あたまのなかでモヤモヤしている問題に対して
(1) 関係のあると思われることがらを全て洗い出す (発散的思考)
(2) それらの関係の整理・本質の追求 (収束的思考)
(3) 評価・決断 (アイデア結晶化)
が行われる．これまでにこれらのプロセスを支援するための さまざまな発想支援システムが研究されている．本研究のシステムでは Google 検索から収束的思考でキーワード間の共通単語を出したい．

発想支援システムの構築は計算機の最も高度な知的利用技術のひとつで，計算機が出現して以来の計算機科学者，人工知能研究者の究極の夢のひとつであり，これについて心理学，社会学，工学などの各方面から様々な試みが行われている．情報技術の進歩により従来は実現が困難であった計算機によるアイデア生成支援が具体化できるようになってきている．また産業界においても個人個人の独創性を引き出せるような知的環境の構築が望まれており，様々な発想支援システムの開発が試みられ，一部は既に実用化されているものもある．

既存の発想支援システムには AI ブレストスパークのひらめきマップや，ブレストアイデアがある [4]．ひらめきマップはキーワードを入力するとその単語に共起する関連語を表示して発想支援に役立てるというものである．ブレストアイデアはキーワードを用いたワード，フレーズの生成をするシステムである．また，KH coder を用いた階層的クラスタ分析を用いて単語の関連性を分析した研究がある [5].

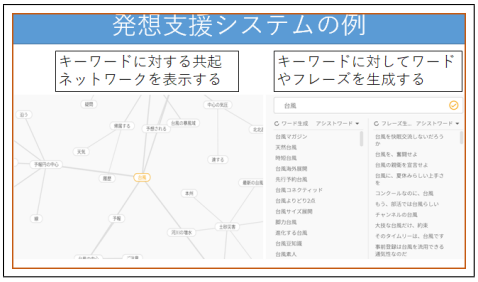


図 2.1: 既存の発想支援システム AI ブレストパーク


図 2.2: AI ブレストパークのブレストアイデア

アイデアは，複数の知識の有意義な組合せである場合が多い．コンピュータで有意義な組合せを自動生成出来れば，それは「発想システム」となる．

しかし，組合せの数は膨大となる．平仮名の数が約 70 として，100 文字から成る文章（平仮名の順列）の総数は 70 の 100 乗（宇宙の原子の推定数より多い）となり，コンピュータでも現実的な時間内で生成するのは不可能である．

優れた文章を書く人間の脳では，何らかの方法で言葉の結び付きを選択している．しかし，その方法は明らかになっていない（従って，プログラム化出来ない）．

そこで，コンピュータが複雑なアイデアを出力する「発想システム」はとりあえず無理と考え，せめてアイデアのヒントを提示するシステムとして「発想支援システム」の開発が試みられた．

## § 2.2　既存の発想支援と課題

**既存の発想支援システム**

既存の発想支援システムには AI ブレストスパークのひらめきマップや，ブレストアイデアがある [4]．ひらめきマップはキーワードを入力するとその単語に共起する関連語を表示して発想支援に役立てるというものである．ブレストアイデアはキーワードを用いたワード，フレーズの生成をするシステムである．AI ブレストパークは 2.1 の発想支援システムにおける発散的思考であるといえる．また，KH coder を用いた階層的クラスタ分析を用いて単語の関連性を分析した研究がある [5].

上記の発想支援システム AI ブレストパークは発散的思考なのに対して，本研究では収束的思考であるので一概にどちらがいいかは評価できないと考える．発想支援システムにおいて目的によって最適なツールが変わってくる．

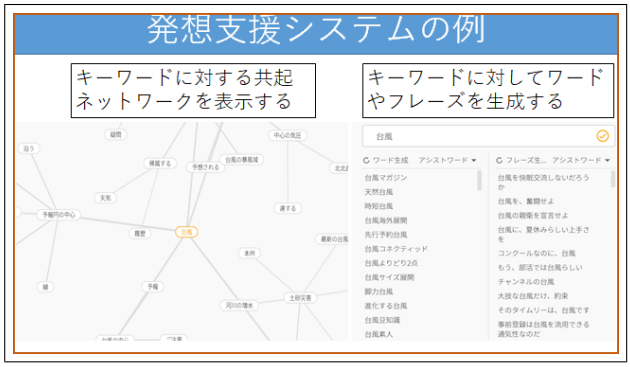


図 2.3: 既存の発想支援システム AI ブレストパーク


図 2.4: AI ブレストパークのブレストアイデア

今の現場に必要な仕組み

1. 職人さんの工事スケジュールの把握と最適な割り振り
2. 天候などの不確定性・不確実性を考慮した日数・費用の見積もり
3. システム処理時間の高速化

## § 2.3

今，建築現場における作業を IT の技術を用いることで，少しでも作業員の負担を減らす工夫がされている．

建設 BALENA[1]

建設業界における，お金と工事，人の流れを Web 上で管理するサービスである．工事の現場よりも，建設会社やリフォーム会社の管 [9].

現場コミュニケーションアプリ Kizuku[2]

このアプリでは，建築の現場における進歩報告，確認作業，指示出しすべてを物件専用トークで管理し，自社社員や協力会社の就労者とリアルタイムでやり取りができる（図 2.3 参照）．また，このアプリでは工程スケジュール情報も管理しており，各作業工程の開始と完了の情報が分かる [10].

今，建築業界では，現場での作業情報を管理・共有する為のアプリが導入されている．そのアプリでは，作業員の入退場や作業時間などが管理されている．したがって，このようなアプリから得られる作業履歴のビックデータから日程計画問題を考えるために必要となるパラメータを得ることができると考えられる．特に，現場コミュニケーションアプリ Kizuku は現場における管理に役立つ機能が充実しているため，このアプリを活用することで，現場で得られたリアルタイムなデータを用いた職人さんの日程計画を管理することも可能であると考えられる．

このような現場アプリの登場により，以前よりも現場での作業情報の管理がしっかりと行われている．よって，過去の作業履歴の情報を蓄積し，分析することが可能であると考えられる．得られる情報としては，職人さんごとの作業にかかる所要時間と費用などが挙げられる．更に，その情報と天気の情報を合わせて考えると，属性（晴れ・雨・雪など）ごとの所要時間と費用の情報を得ることも可能である．

本研究では，このようなデータがアプリから得られると想定し，過去の作業情報から職人さんの作業所要時間や費用の見積もりを検討する．

データの設定例

入力データには，作業工程数と従事者グループ数 (職人さんの数)，属性数，作業にかかる所要時間の最大日数と最小日数，作業にかかる最大費用と最小費用を用いる．
このように，作業履歴から得られるデータを入力データとしたとき，属性における各従事者グループごとの費用の最大値と最小値，平均値などを出力することができる．同時に，その条件のときにかかる費用の最大値と最小値，平均値も出力できる．

このようにして，過去の作業履歴から収集できるデータを用いて，日程計画に必要なパラメータを設定することができる（図 2.5 参照）．

[1]https://mag.branu.jp /archives/2442
[2]https://www.ctx.co.jp/kizuku2pr/

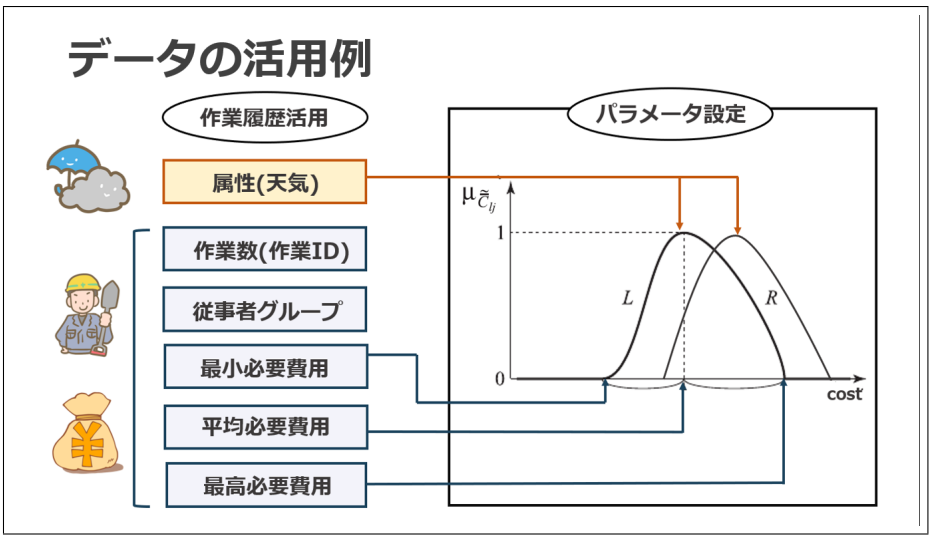


図 2.5: 作業履歴を活用したパラメータの設定

# 第3章

# サイバー空間からのテキストデータの収集

## § 3.1　スクレイピングによるテキストマイニング

現代社会においてインターネット上の情報量は莫大になっており，今後も増え続けることが予想される．このインターネット上の情報を収集して分析することで発想支援に生かせると考える．発想支援において重要なことはキーワードからより関連度の高い単語をより多く表示させることである．そこで, より良いデータを多く収集するためにサイバー空間からテキストデータを収集することとする．

今回，キーワードごとに Google 検索から上から何件分かの URL を取得し，その URL からテキストを抽出してそのテキストに対して自然言語処理を行い，必要な単語を取り出す．その単語群から共起ネットワークを作成する．

図 2 にデータ収集に用いるテキストマイニングの説明を示す．

テキストマイニングとは，大量かつ多量なデータを様々な観点から分析し，役に立つ情報を取り出そうとする技術である．

インターネット上のテキストを用いることで大量のデータを活用することができる．まず，収集した文章に対して HTML タグや JavaScript のコードを取り除くクリーニング処理をする．その次に形態素解析を行う．形態素解析とは文を形態素ごとに分解する技術である [3]．発想支援において必要な名詞や動詞に分解することである．また, 助詞の「は」や「が」助動詞の「です」は不要なので取り除く．そして単語群に対して正規化を行う．正規化することで文字種を統一できる．半角と全角や数字を統一することで同じ単語として扱うことができる．

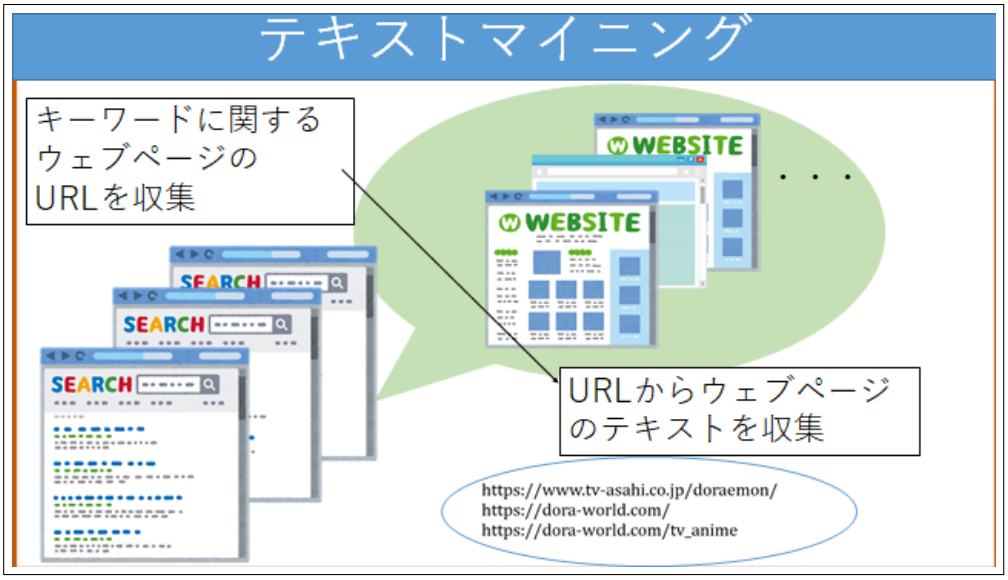


図 3.1: テキストマイニングのイメージ

## § 3.2　複数のキーワード間のネットワーク

自然言語処理の分野において, 単語をベクトル表現する Word2Vec という技術がある．複数のキーワードの関連語がなかった場合にも発想支援できるように単語をベクトル表現することで近いベクトルの単語同士を関連語として見れるのではないかと考える．

Word2Vec について Skip-gram でモデル化する．Skip-gram では，ある単語を入力した時，その周辺にどのような単語が現れやすいか予測することをモデル化することができる．

**厳密解法**

最適化問題に対して，最適性の保証された解を求める解法のことで，主に NP 困難な問題に対して用いられる．NP 困難である問題とは，問題の規模が大きくなると爆発的に計算時間が増加するような問題である. 解法としては，分枝限定法などがある [13].

**近似解法**

厳密解ではなく，ある程度良いとされる解，近似解を求める解法のことで，厳密解法では解を求めることが困難である場合に用いられる．厳密解法に比べて，近似解を得るまでにかかる時間は短くなるが，厳密解ではないため，解に対する信憑性についての評価が必要になる．

前述した厳密解法を用いて，実問題の複雑な日程計画問題を解くと，実行可能な解が求まらなかったり，膨大な計算時間がかかるため，直接用いることは困難とされている．よって，実問題のような複雑で大規模な問題を解く場合には，近似解法である遺伝的アルゴリズムやタブ探索などのメタヒューリスティックスに基づいたシステムが用いられている．更に，実問題における日程計画問題の多くは，単目的な問題ではなく，多目的な問題がほとんどである．

多目的最適化問題

最適化問題とは，目的関数の引数を調整して目的関数を最小化，もしくは最大化する問題のことを指す．その中でも，目的関数が複数のものを多目的最適化問題という．実問題において，単目的な問題は稀であり，多くの問題が多目的最適化問題であるとされる．一般的に，多目的最適化問題においては，目的関数の間に一方が良くなれば他方が悪くなるトレードオフの関係がある．したがって，多目的最適化問題において，最適解は 1 つではない．つまり，多目的最適化問題とは，このいくつかある最適解の集合 (パレート最適フロント) から少なくとも 1 つの解を探索する問題である（図 3.2 参照）．

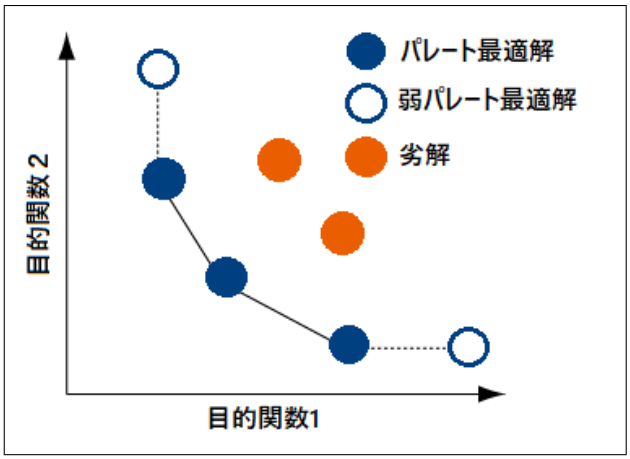


図 3.2: パレート最適解の例

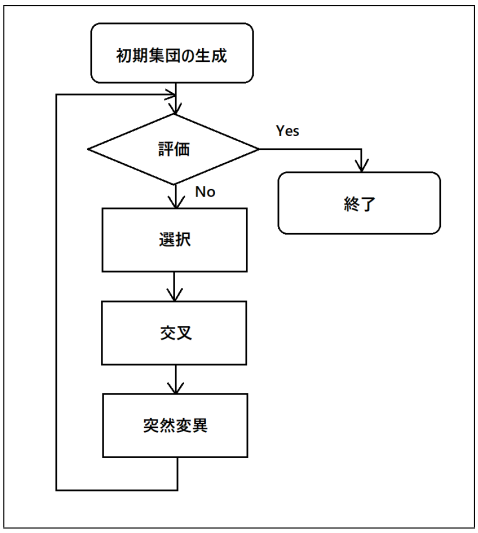


図 3.3: GA のフローチャート

ある単語とある単語が同時に出現することを共起するといい，文章において関係深い単語は共起することが多い．共起分析では単語同士の Jaccard 係数を比較したり，共起関係を持つ単語と単語を線で結んで描かれる共起ネットワークが利用される．文章また単語群に対して共起する単語をネットワークで表した共起ネットワークという．今回，キーワードごとに集めたテキストに対してそれぞれの共起ネットワークを作成する．

### 遺伝的アルゴリズム

遺伝的アルゴリズム (Genetic Algorithm ：GA) とは自然界における生物の進化モデル，すなわち世代を形成している個体の集合（個体群）の中で，環境への適応度の高い個体が次世代により多く生き残り，交叉や突然変異を起こしながら次の世代を形成していく過程を模した最適化手法である（図 3.3 参照）．これまでにも，多くのスケジューリング問題に対して GA の適用が試みられてきた [20]．GA は，遺伝的操作・適応度計算に対する膨大な計算コストが要求される．よって，並列計算機によるアルゴリズムの並列化について様々な検討がなされている [21] [22] [23]．

### 並列分散 GA

並列分散 GA とは，全体の個体群をいくつかの島に分散させ，その中でのみ遺伝的操

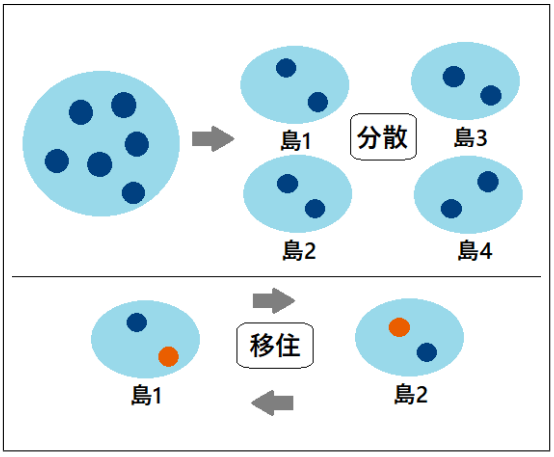


図 3.4: 並列分散 GA のイメージ

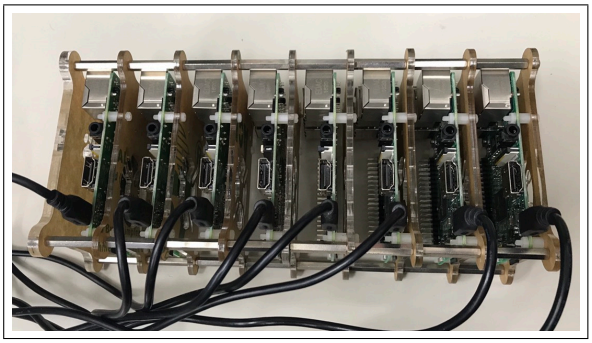

図 3.5: 実験に用いるラズベリーパイ 8 台

作を行うモデルを指す．島ごとに異なる交叉率や突然変異率等のパラメータを設定出来るため，パラメータ設定の困難さを緩和する効果がある．基本的に島の中でのみ遺伝的操作が行われるため，各島ごとに異なる局所優良解に到達する事が期待出来る．様々な実験により， 並列分散 GA では，全体を一つにして行なう単純 GA と比較して，最適解発見確率が格段に高まる事が確認されている．更に，島と島の個体同士を交換（移住）することで局所解に陥るのを防ぎ，島ごとの多様性を維持することができるため，解の探索能力が高まる（図 3.4 参照）．また，並列分散処理では，異なるコンピュータを島ごとに割り当てる事が可能となり，計算時間の大幅な短縮も可能である．

## § 3.3　ベイジアンネットワークによる共通単語のランク付け

ベイジアンネットワークにより単語のつながりを可視化する. 単語群から，単語の遷移をにカウントして隣接行列を作成する．Softmax 関数から単語の遷移確率を求めてベイジアンネットワークを作成する．隣接行列を Pandas で作成し, 隣接行列から NetworkX という Python のライブラリを使用してベイジアンネットワークを作成する．

それぞれのネットワークのある共通単語について，単語の推移確率の和をパスの長さで割った値を双方向から求め，その和をある共通単語のスコアとする．このスコアの高い単語をランク付けして共通単語とスコアの値を表示する．

ベイジアンネットワークにより単語のつながりを可視化する. 単語群から，単語の遷移をにカウントして隣接行列を作成する．Softmax 関数から単語の遷移確率を求めてベイジアンネットワークを作成する．隣接行列を Pandas で作成し, 隣接行列から NetworkX という Python のライブラリを使用してベイジアンネットワークを作成する．

それぞれのネットワークのある共通単語について，単語の推移確率の和をパスの長さで割った値を双方向から求め，その和をある共通単語のスコアとする．このスコアの高い単語をランク付けして共通単語とスコアの値を表示する．

[3]https://jp.rs-online.com/web/p/products/1239470/

Raspberry Pi 3 (Model B）を用いた並列分散環境の構築に必要なもの

1. Raspberry Pi3（ModelB）× 8
2. SD カード 16G 以上× 8
3. 8 ポート以上のハブ
4. AC アダプタ
5. USB 接続ができるキーボード
6. USB 接続ができるマウス
7. HDMI 接続ができるモニタ

**Raspberry Pi 3（Model B）とは**

Raspberry Pi とは，テレビ，キーボード，マウス，電源を繋ぐだけでで使用することができるコンパクトなコンピュータボードである．多種多様な製品があり，カメラや LCD ディスプレイモジュールなどの用途を実現することも可能である．

本研究で使用する Raspberry Pi 3（Model B）は，以前までのモデルからのアップグレードで，接続性が向上し，WLAN，Bluetooth，及び Bluetooth Low Energy (BLE) のすべてがオンボードで利用可能である．また，接続性の向上により，他のサポートと容易に通信ができ，IoT の設計にも最適とされている [24].

**MPI の概要**

MPI とは Message Passing Interface の略で，分散メモリ間のメッセージ通信の実行基盤の 1 つである．ある 1 つの目的を複数のコンピュータを用いて，分散・並列処理する際に用いられる．MPI には，mpich や Open MPI などのいくつかの実装系があり，本研究では，現状フリーで広く使用されている mpich を用いる．

**Raspberry Pi 3（Model B）の初期設定〜並列環境の構築**

本研究では，複数マシンにまたがって複数のプロセスを起動し，それぞれの間でお互いに通信し，並列処理を行う．このとき，外部ネットワークを利用してプロセス間通信を行うため，セキュアシェル（SecureShell： SSH）とネットファイルシステム（Network File System： NFS）の設定，hosts ファイルを準備する必要がある [25]．このとき，担当分の処理を行うと同時に並列処理や結果の集約を担当する「マスターノード」とマスターノードと連携して担当分の処理を行う「スレーブノード」の二種類に分けられる．

手順 1 ：Raspbian のダウンロード

OS(Raspbian) を配布元からダウンロード・解凍し，Raspbian のディスクイメージファイルを入手する．入手したディスクイメージファイルを SD カードに焼く．問題なく SD カードに Raspbian を入れることができたら，各マシンに SD カードをセットし Raspberry Pi 3 を起動する．

手順 2：IP アドレスを固定するためのファイル編集

まず，ノード間並列を行う前に，パスワードなしの ssh 接続ができる状態にしておく必要がある．そのため，各マシンの IP アドレスを固定する．設定ファイルを編集する際，必要であればテキストエディタをインストールする．

```
$ sudo apt-get install emacs
```

次に，各 Raspberry Pi のホスト名を変更する．

```
$ sudo emacs /etc/hostname
```

そして，固定 IP の設定のために 「/etc/dhcpcd.conf」を編集する．

```
$ sudo emacs /etc/dhcpcd.conf
```

ファイル内の「static ip __address=初期設定の IP アドレス」を，それぞれの Raspberry Pi 3 に固定する IP アドレスに書き換える．更に，ホスト名と固定した IP アドレスを一致させるために, 次のように編集する．

```
$ sudo emacs /etc/hosts
```

・ 192.168.0.60 master

・ 192.168.0.61 slave2

⋮

・ 192.168.0.67 slave8

各ファイルの編集が終わったら，全ての Raspberry Pi を再起動する．

手順 3：SSH 接続を有効にする

ネットワークを再起動し終わったら

$ ping ホスト名, ping IP アドレス

がそれぞれのノードに正常に通るか確認を行う．

$ sudo raspi-config

で ssh を enable に設定する．

$ ssh-keygen -t rsa

を実行し，SSH 鍵を作成する．更に，作成した公開鍵を全てのマシンに送信する．

$ ssh-copy-id master

$ ssh-copy-id slave2

⋮

$ ssh-copy-id slave8

上記の作業をすべてのマシン上で行う．ここまでの作業が完了したら，以降はマスターからリモートで各マシンに SSH 接続して設定を行う．

手順 4：mpich のインストール

次のコマンドで全てのマシンに mpich をインストールする．

$ sudo apt-get install mpich

mpich を用いたコンパイルと実行のコマンドは次のようになる．

$ sudo mpicc （ファイル名.c）

$ sudo mpirun ./（ファイル名）

複数台を同時に実行するときは

$ mpirun -np (並列数)(実行コマンド)

となる．

手順 5：NFS を用いた作業ディレクトリの共有

次に，NFS の設定を行う．NFS とは，分散ファイルシステムの一種で，あるマシンが所有する 1 つのディレクトリを，他の複数のノードで共有して使うための仕組みである．MPI を使う際には，実行ファイルが全てのマシンで同じ位置に置かれている必要がある．よって，NFS を用いて，マスターノードの作業ディレクトリをスレーブノードに共有する．まず，マスターで NFS のインストールを行う．

$ apt-get install nfs-kernel-server

その後，/etc/exports ファイルを編集し以下の文を追記する．

・(公開したいディレクトリ) (どのマシンに公開するか)(公開モード)

・例/home/server/foo 192.168.1.200(rw, sync, no __ subtree __ check)

exports を編集したら以下のコマンドで再起動

$ /etc/init.d/nfs-kernel-server restart

マスターでの設定が完了したら，他のスレーブ 7 台の設定も行う．nfs-common のインストール，共有先のディレクトリ（/mpi1/test）の作成とマウント設定を以下のコマンドで行う．

$ sudo apt-get install nfs-common

$ mkdir /home/mpi1/test

$ sudo mount –types nfs 192.168.0.60:/home/mpi1/test /home/mpi1/test

更に，/etc/fstab を編集して (共有したいディレクトリ) (共有先) を追記する．

例 192.168.0.60:/home/mpi1/test /home/mpi1/test nfs rw 0 0

以上の設定で，マスターノードの作業ディレクトリをスレーブノードに共有することができる．

手順 6：host ファイルの設置

最後に，各マシンの IP アドレスまたはホスト名を書いた hosts ファイルを「マスターノード」の実行プログラムと一緒に置く．この hosts ファイルとは，ドメインネームシステム（Domain Name System : DNS）よりも先に参照される IP アドレスとドメイン名の一覧である．DNS とは，インターネットの重要な基盤技術の 1 つで，ドメイン名と IP アドレスの対応付けなどを行うシステムである．

**Raspberry Pi 3（Model B）と mpich を用いた並列分散処理の動作確認**

更に，この構築した環境でプログラムが問題なく動くかを確認するために，Raspberry Pi 8 台を使った「hello world の並列処理」「円周率計算の並列分散処理」「並列分散 GA を用いた巡回セールスマン問題の処理時間比較」を行う [26] [27].

```
pi@master:/home/mpi1/test $ mpirun --hostfile host -np 8 ./hellow
Hello world from process 0 of 8
Hello world from process 1 of 8
Hello world from process 2 of 8
Hello world from process 6 of 8
Hello world from process 4 of 8
Hello world from process 3 of 8
Hello world from process 5 of 8
Hello world from process 7 of 8
```

図 3.6: mpich と Raspberry Pi 8 台を用いた Hello World の並列実行結果

表 3.1: 円周率計算の処理時間

| 台数 | 処理時間 |
|---|---|
| 1 台 | 51 秒 |
| 2 台 | 25 秒 |
| 4 台 | 21 秒 |
| 8 台 | 17 秒 |

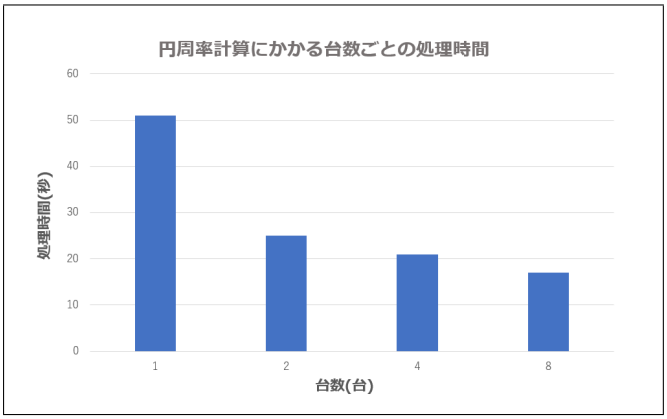


図 3.7: 円周率計算の処理時間グラフ

**動作テスト 1：Hello World の並列処理（付録 A.1 参照）**

動作確認のため，ラズベリーパイ 8 台で「Hello World」を表示させるプログラムを動かした (図 3.6 参照). MPI によって起動された各プロセスには,「ランク」と呼ばれる 0 から始まる識別番号が付与される．図 3.8 の左側の数字が「ランク」を表しており，右側の数字が動いているプロセスの総数である．また，このランクの値によって各プロセスの処理を制御することができる [17].

図 3.8 を見ると，問題なく 8 台分の「Hello World」が表示されており，0 から 7 のランクの値も表示されている．よって，並列処理を正常に行える環境であることが確認できた．

**動作テスト 2 ：円周率計算の並列分散処理（付録 A.2 参照）**

動作テスト 1 で Hello World を 8 台分問題なく表示させることができた．動作テスト 2 では，円周率計算の並列分散処理を行い，計算の処理時間が問題なく計測できるかを確認する．更に，1 台，2 台，4 台，8 台での処理時間を比較し，計算の高速化ができているかの確認も行う（表 3.1，図 3.7 参照）.

表 3.1 のように，問題なく 8 台での並列分散処理を行ったときの処理時間を計測できた．また，図 3.9 をみると，分散する台数を増やすにつれ，計算にかかる処理時間を短縮できていることが確認できる.

表 3.2: TSP 問題の処理時間

| 台数 | 処理時間 |
|---|---|
| 1 台 | 320 秒 |
| 2 台 | 165 秒 |
| 4 台 | 71 秒 |
| 8 台 | 107 秒 |

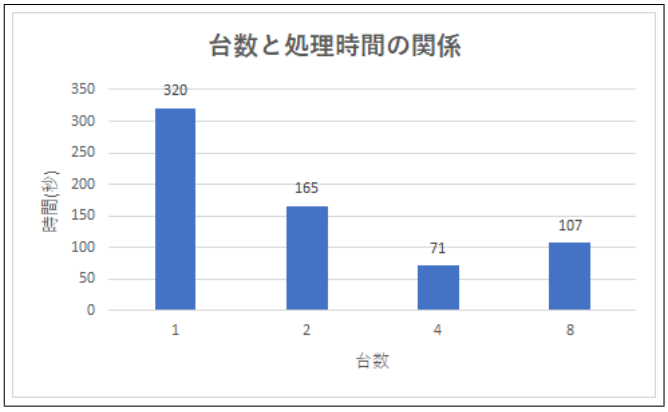


図 3.8: TSP 問題の処理時間のグラフ

**動作テスト 3：並列分散 GA による巡回セールスマン問題（TSP 問題）**

本研究では，日程計画の解法として，MPI を用いた並列分散 GA の活用を検討する．そのため，動作テストとして巡回セールスマン問題に対して，台数ごとに処理時間の変化を比較する（表 3.2, 図 3.8 参照）[18]．MPI を用いた並列分散 GA のフローは次のようになる（図 3.9 参照）．

巡回セールスマン問題 (TSP 問題)

多くの都市と各都市間の移動コストがあたえられたとき，全ての都市を一度だけ回り戻ってくるルートのうち，最小のコストになる回り方を求める問題のことである．

計算に用いたパラメータ

1. マップ：30 × 30
2. 都市数：50 都市
3. 遺伝子数：800 個
4. 世代数：50000 世代
5. 移住数：120 遺伝子 (15 %)
6. 端末数：1 台、2 台、4 台、8 台

動作テスト 1，動作テスト 2 と動作テスト 3 より，並列分散処理による高速化が可能であることが確認できた．最後の動作テスト 3 を行ったときのみ 4 台から 8 台へ分散台数を増やしても処理時間が短縮されなかった．これは，実験中のマシンの様子から，サーバーやネットワーク機器などのハードウェアの処理能力不足や通信のオーバーヘッドによるものであると考えられる．

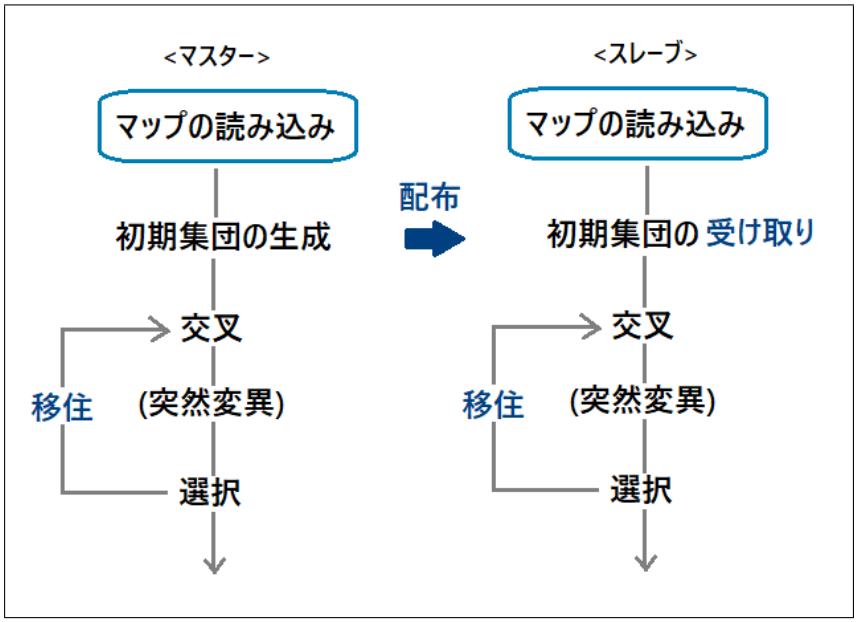


図 3.9: MPI による並列分散 GA のフロー

# 第4章

# 提案手法

## § 4.1

提案するモデルについて説明する．任意の 2 つのキーワードから Google 検索を使ってクローリングしてそれぞれのキーワードのウェブサイトの URL を集めてくる．節 2.1 で述べた従来の日程計画問題における課題と節 2.2 で述べた建築現場における課題を解決し，現場における資源の最適な配分による生産性の向上を目指すための日程計画を考えた．

特に，提案モデルでは，建築現場の悩みを解消するために必要とされる「職人さんの最適な割り振り」と従来の日程計画問題の課題とされる「天候・日数・費用などの不確定で不確実な要素の見積もり」を考慮したファジィ・ランダム多目的日程計画問題を考える（図 4.1 参照）．

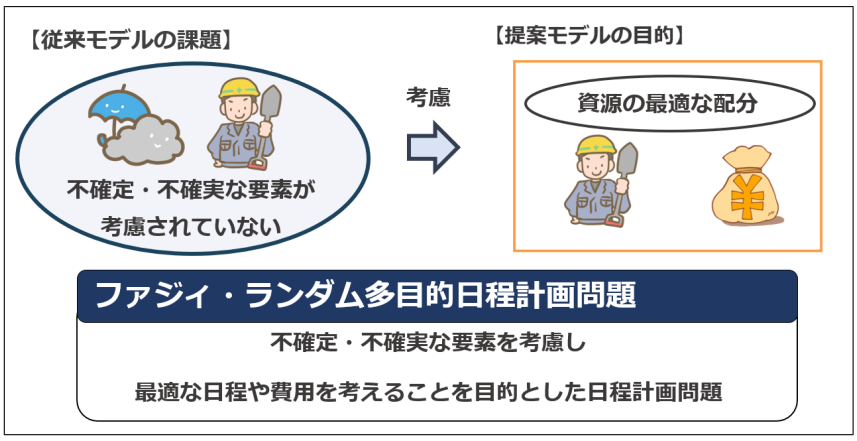


図 4.1: 提案モデルの特徴

図 4.1 のように，住宅建築における日程計画は，天候という「確率変動要素」と費用という「ファジィ要素」が両方同時に存在すると考え，これらの要素を考慮するファジィ・ランダム多目的日程計画を考えた．具体的な作業の順序関係を可視化した住宅建築におけるプ

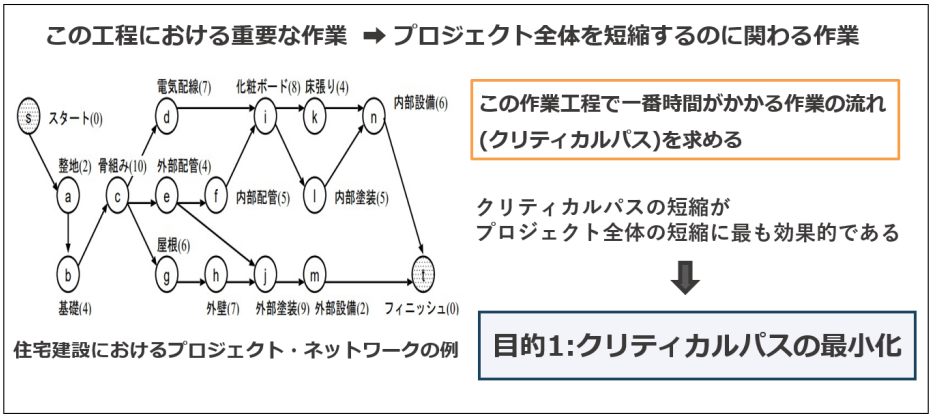


図 4.2: クリティカルパスの所要時間を最小化

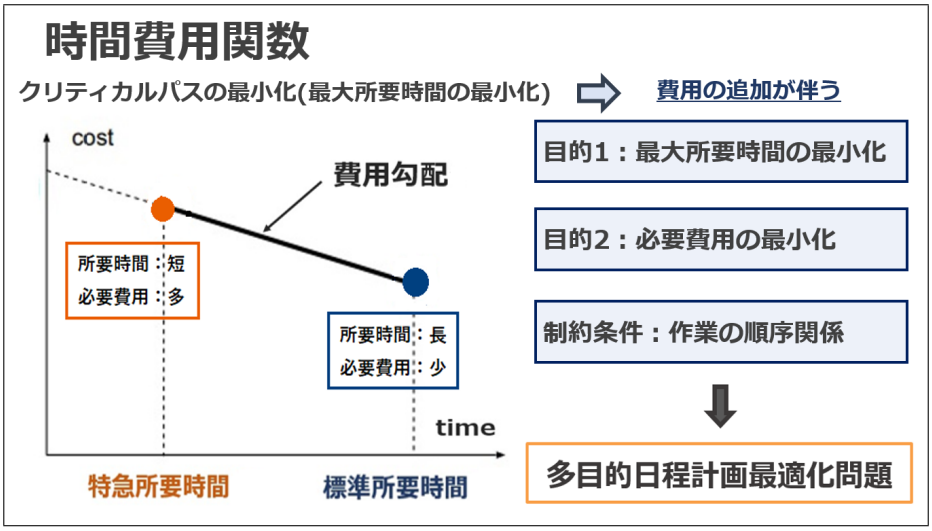


図 4.3: 時間費用関数と多目的最適化問題

ロジェクトネットワークの例を次に示す（図 4.2 参照）.

**目的関数 1 ：最大所要時間の最小化**

このような，プロジェクトネットワークも活用し，全作業の順序関係を視覚的に把握する．そこから，工程における重要な作業，つまり，工程全体を短縮するのに関わる作業を求める．提案モデルでは，作業の所要時間の積算で所要時間が最大となる工程をクリティカルパスとし，このクリティカルパス上の作業を重点的に管理することで，工程全体の作業効率の向上を図る．したがって，提案モデルにおける目的関数の一つは，クリティカルパス上の作業所要時間の最小化（最大所要時間の最小化）とした．

**目的関数 2：必要費用の最小化**

更に，所要時間と費用の両方の最小化を目的とした多目的日程計画問題を考える（図 4.3 参照）．一般的に，ある作業の所要時間を短縮したいと考えたとき，多くの職人さんに依頼するなど，それに伴った費用がかかる．よって，所要時間と費用の関係は，

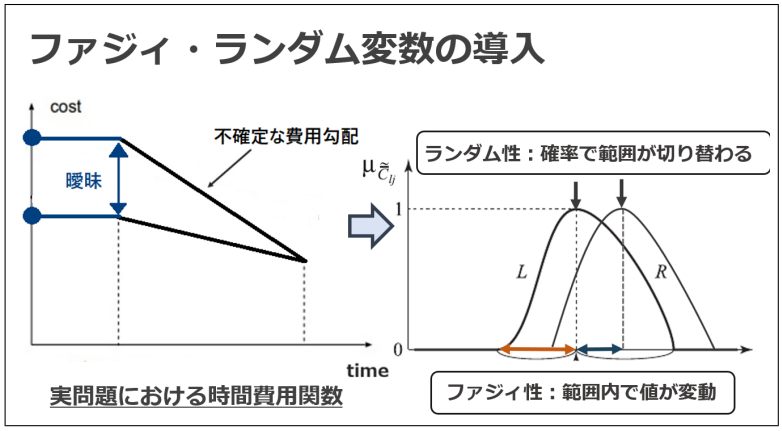


図 4.4: 実問題における時間費用関数

図 4.3 のような時間費用関数で表される．この時間費用関数は，費用をかけない代わり所要時間が長くなる標準所要時間と，その標準所要時間に対して費用をかけることで所要時間をその分短縮することができる特急所要時間の関係を示している．このような，トレードオフの関係にある目的関数を扱う場合，節 3.2 で述べたように，最適解は一つではなくパレート最適フロントとして複数存在する．

### ファジィ・ランダム変数の導入

図 4.3 で説明した時間費用関数は，静的で確定的な値であることを前提として考えられている．しかし，実問題を考える場合，所要時間や必要費用は不確定で不確実な要素であるため，図 4.3 の線形な時間費用関数では対応できない．そこで，実問題における時間費用関数は不確定な費用勾配を持つ関数であると考え，それを表現するためにファジィ・ランダム変数を導入する（図 4.4 参照）．

### 提案手法の定式化

本研究では，考えたファジィ・ランダム多目的日程計画問題を次のように定式化した．

$$
\left.
\begin{array}{ll}
\text{minimize} & \max\left\{\sum_{i=1}^{n}\sum_{j\in N_i}\sum_{k=1}^{w} t_{ijk}x_{ij}y_{ik}\right\} \\
\text{minimize} & \sum_{i=1}^{n}\sum_{j\in N_i}\sum_{k=1}^{w} \tilde{\tilde{c}}_{ik}x_{ij}y_{ik} \\
\text{subject to} & \\
 & \sum_{i=1}^{n}\sum_{j\in N_i} x_{ij} - \sum_{i=1}^{n}\sum_{j\in N_i} x_{ji} = \left\{\begin{array}{ll} 1\,, & (i=s)\,, \\ 0\,, & (i\neq s,l)\,, \\ -1\,, & (i=l)\,. \end{array}\right. \\
 & x_{ij}\in\{0,1\},\ i=1,\ldots,n, j=1,\ldots,n; \\
 & y_{ik}\in\{0,1\},\ i=1,\ldots,n, k=1,\ldots,w;
\end{array}
\right\} \qquad (4.1)
$$

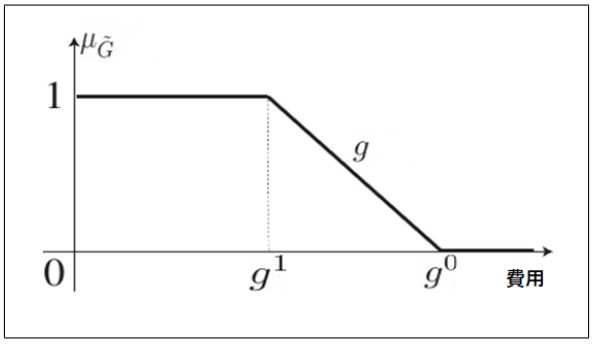


図 4.5: ファジィ目標 $\tilde{G}$ のメンバシップ関数 $\mu_{\tilde{G}}$

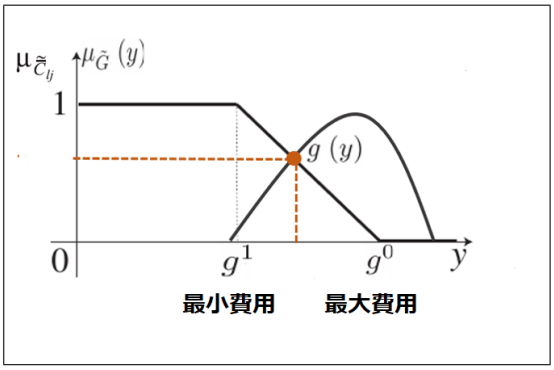


図 4.6: ファジィ目標の設定

式 (4.1) は，クリティカルパス上の作業の総所要時間と総費用の最小化するための解を求める定式化である．ここでの解とは，クリティカルパス上の作業とその職人さん（従事者グループ）の組合せである．ここで，$i$ は先行作業，$j$ は後続作業，$n$ は プロジェクトの総作業数である．また，$k$ は作業を受け持つ従事者グループ，$w$ は依頼候補の従事者グループ数を表している．

よって，$t_{ijk}$ は作業 $i$ から作業 $j$ 開始前までを従事者グループ $k$ が受け持ったときの所要時間，$\tilde{c}_{ik}$ は作業 $i$ を従事者グループ $k$ に依頼したときの費用を表している．$x_{ij}$ は作業 $i$ から作業 $j$ を選択する 0-1 変数，$y_{ik}$ は作業 $i$ を従事者グループ $k$ に依頼するかを選択する 0-1 変数である．また，$\tilde{c}_{ik}$ の各要素は，節 3.1 で述べたメンバシップ関数により特徴づけられるファジィ・ランダム変数を要素とする係数ベクトルである．

## § 4.2　等価確定問題への変換

ここで，ファジィ・ランダム変数を含む式をそのまま取り扱うことは困難であるため，確率計画問題から多目的日程計画問題へ等価確定変換する必要がある．

そこで，式 (4.1) のファジィ・ランダム変数を係数に含む目的関数に対して，可能性測度と確率最大化モデルに基づき式変形を行う [28] [29] [30].

### ファジィ目標 $\tilde{G}$ の導入

ファジィランダム変数を含む目的関数に対して，ファジィ目標を導入する. ファジィ目標 $\tilde{G}$ のメンバシップ関数 $\mu_{\tilde{G}}$ は次の式 (4.2) によって特徴づけられる．式 (4.2) によって特徴づけられるファジィ目標のメンバシップ関数の例を図示する (図 4.5 参照).

$$\mu_{\tilde{G}} = \begin{cases} 1, \text{ if } y < g^1, \\ g(y), \text{ if } g^1 \leq y \leq g^0, \\ 0, \text{ if } y > g^0. \end{cases} \tag{4.2}$$

ここで，$g^0$ は最悪値 (最高費用), $g^1$ は最良値 (最小費用) である．$g$ は狭義単調減少連続関数である（図 4.6 参照）.

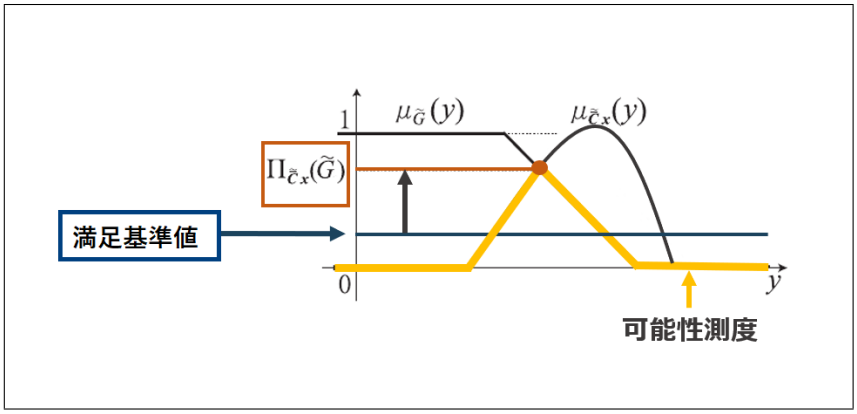


図 4.7: 満足基準値を用いた確率最大化モデル

目的関数のメンバシップ関数 $\mu_{\tilde{c}_{ik}x_{ij}y_{ik}}$ を可能性分布とみなし，このとき，その可能性分布の下でファジィ目標 $\tilde{G}$ が満たされる可能性の度合いを $\Pi_{\tilde{c}_{ik}x_{ij}y_{ik}(\tilde{G})}$ とし，可能性測度を用いて次の式 (4.3) ように与えられる．可能性の度合い $\Pi_{\tilde{c}_{ik}x_{ij}y_{ik}(\tilde{G})}$ を図示した（図 4.7 参照）．

$$\Pi_{\tilde{c}_{ik}x_{ij}y_{ik}}(\tilde{G}) = \sup \min\{\mu_{\tilde{c}_{ik}x_{ij}y_{ik}},\ \mu_{\tilde{G}}\} \tag{4.3}$$

原問題に対して，ファジィ目標が満たされる可能性の度合い (可能性測度) を最大化する問題へ変換して考えると，式 (4.4) のようになる．

$$\left.\begin{array}{rl} \text{minimize} & \max\left\{\sum_{i=1}^{n}\sum_{j\in N_i}\sum_{k=1}^{w} t_{ijk}x_{ij}y_{ik}\right\} \\ \text{maximize} & \Pi_{\tilde{c}_{ik}x_{ij}y_{ik}}(\tilde{G}) \\ \text{subject to} & \\ & \sum_{i=1}^{n}\sum_{j\in N_i} x_{ij} - \sum_{i=1}^{n}\sum_{j\in N_i} x_{ji} = \left\{\begin{array}{ll} 1\ , & (i=s)\ , \\ 0\ , & (i\neq s,l)\ , \\ -1\ , & (i=l)\ . \end{array}\right. \\ & x_{ij}\in\{0,1\}, i=1,\ldots,n, j=1,\ldots,n; \\ & y_{ik}\in\{0,1\}, i=1,\ldots,n, k=1,\ldots,w; \end{array}\right\} \tag{4.4}$$

ここで，可能性測度の導入により，費用のファジィ性を確定的に取り扱うことができるため，問題は確率計画問題となる．

### 確率最大化モデルによる変換

更に，問題における可能性の度合い $\Pi_{\tilde{c}_{ik}x_{ij}y_{ik}(\tilde{G})}$ の最大化を $\Pi_{\tilde{c}_{ik}x_{ij}y_{ik}(\tilde{G})}$ が満足基準値 $h$ 以上となる確率を最大化するという確率最大化モデルに基づき，$\Pr[\Pi_{\tilde{c}_{ik}x_{ij}y_{ik}}(\tilde{G}) \geq h]$ の最大化に置き換える（図 4.6 参照）．問題は式 (4.5) のように定式化される．

$$
\left.
\begin{array}{rl}
\text{minimize} & \max\left\{\sum_{i=1}^{n}\sum_{j\in N_i}\sum_{k=1}^{w} t_{ijk}x_{ij}y_{ik}\right\} \\
\text{maximize} & \Pr[\Pi_{\tilde{\bar{c}}_{ik}x_{ij}y_{ik}}(\tilde{G}) \geq h] \\
\text{subject to} & \\
& \sum_{i=1}^{n}\sum_{j\in N_i} x_{ij} - \sum_{i=1}^{n}\sum_{j\in N_i} x_{ji} = \left\{ \begin{array}{l} 1\ , \quad (i=s)\ , \\ 0\ , \quad (i\neq s,l)\ , \\ -1\ , \quad (i=l)\ . \end{array} \right. \\
& x_{ij}\in\{0,1\}, i=1,\ldots,n, j=1,\ldots,n; \\
& y_{ik}\in\{0,1\}, i=1,\ldots,n, k=1,\ldots,w;
\end{array}
\right\} \qquad (4.5)
$$

ここで，任意の根元事象に対して $\Pi_{\tilde{\bar{c}}_{ik}x_{ij}y_{ik}}(\tilde{G}) \geq h$ は

$$
\begin{aligned}
& \Pi_{\tilde{\bar{c}}_{ij}x_{ij}}(\tilde{G}) \geq h \\
\Leftrightarrow\ & \sup\min\left\{\mu_{\tilde{\bar{c}}_{ik}x_{ij}y_{ik}}, \mu_{\tilde{G}}\right\} \geq h \\
\Leftrightarrow\ & \exists y : \mu_{\tilde{\bar{c}}_{ik}x_{ij}} y_{ik} \geq h,\ \ \mu_{\tilde{G}} \geq h \\
\Leftrightarrow\ & \exists y : L\left(\frac{\bar{d}x_{ij}y_{ik} - y}{\bar{\beta}x_{ij}y_{ik}}\right) \geq h,\ \ R\left(\frac{y - \bar{d}x_{ij}y_{ik}}{\bar{\gamma}x_{ij}y_{ik}}\right) \geq h,\ \ \mu_{\tilde{G}}(y) \geq h \\
\Leftrightarrow\ & \exists y : \{\bar{d} - L^*(h)\bar{\beta}\}x_{ij}y_{ik} \leq y \leq \{\bar{d} + R^*(h)\bar{\gamma}\}x_{ij}y_{ik},\ \ y \leq \mu^*_{\tilde{G}}(h) \\
\Leftrightarrow\ & \{\bar{d} - L^*(h)\bar{\beta}\}x_{ij}y_{ik} \leq \mu^*_{\tilde{G}}(h)\ .
\end{aligned}
$$

のように変形することができる．ここで，$L^*(h)$ 及び $\mu^*_{\tilde{G}}(h)$ は擬逆関数であり，

$$
\begin{aligned}
L^*(h) &= \sup\{r | L(r) \geq h, r \geq 0\} \quad (0 < h \leq 1) \\
\mu^*_{\tilde{G}}(h) &= \sup\{r | \mu_{\tilde{G}}(r) \geq h\} \quad (0 < h \leq 1)
\end{aligned}
$$

というように表わされる．さらに，確率変数 $\bar{t}$ の分布関数を $F(\cdot)$ とすると，問題は

$$
\begin{aligned}
\Pr[\Pi_{\tilde{\bar{c}}_{ij}x_{ij}}(\tilde{G}) \geq h] &= \Pr[\{\bar{d} - L^*(h)\bar{\beta}\}x_{ij}y_{ik} \leq \mu^*_{\tilde{G}}(h)] \\
&= \Pr[\{(d^1 + \bar{t}d^2) - L^*(h)(\beta^1 + \bar{t}\beta^2)\}x_{ij}y_{ik} \leq \mu^*_{\tilde{G}}(h)] \\
&= \Pr\left[\bar{t} \leq \frac{\{L^*(h)\beta^1 - d^1\}x_{ij}y_{ik}] + \mu^*_{\tilde{G}}(h)}{\{d^2 - L^*(h)\beta^2\}x_{ij}}\right] \\
&= F\left(\frac{\{L^*(h)\beta^1 - d^1\}x_{ij}y_{ik} + \mu^*_{\tilde{G}}(h)}{\{d^2 - L^*(h)\beta^2\}x_{ij}y_{ik}}\right)
\end{aligned}
$$

となる．最終的に，定式化した式 (4.1) のファジィ・ランダム多目的日程計画問題は式 (4.6) のような等価な多目的計画問題へ置き換えることができる．

$$
\left.
\begin{array}{rl}
\text{minimize} & \max\left\{\sum_{i=1}^{n}\sum_{j\in N_i}\sum_{k=1}^{w} t_{ijk}x_{ij}y_{ik}\right\} \\
\text{maximize} & F\left(\frac{-\{d^1-L^*(h)\beta^1\}x_{ij}y_{ik}+\mu^*_{\tilde{G}}(h)}{\{d^2-L^*(h)\beta^2\}x_{ij}y_{ik}}\right) \\
\text{subject to} & \\
 & \sum_{i=1}^{n}\sum_{j\in N_i} x_{ij} - \sum_{i=1}^{n}\sum_{j\in N_i} x_{ji} = \begin{cases} 1\ , & (i=s)\ , \\ 0\ , & (i\neq s,l)\ , \\ -1\ , & (i=l)\ . \end{cases} \\
 & x_{ij}\in\{0,1\}, i=1,\ldots,n, j=1,\ldots,n; \\
 & y_{ik}\in\{0,1\}, i=1,\ldots,n, k=1,\ldots,w;
\end{array}
\right\} \tag{4.6}
$$

## § 4.3　提案手法のアルゴリズム

最後に，本研究で提案したモデルのパラメータ設定から解法までのアルゴリズムについてまとめる（図 4.8 参照）．まず，ファジィ・ランダム多目的日程計画問題を解くために必要となるデータの収集を行い，そのデータを用いたパラメータの設定方法を以下に示す．

**Step1：過去の作業情報の蓄積**

建築現場における作業情報を現場コミュニケーションアプリなどを活用して蓄積する．収集するデータは,「属性数 (天候)」「作業数 (作業 ID)」「従事者数」「従事者グループごとの各作業にかかった日数」「従事者グループごとの各作業にかかった費用」である．

**Step2：問題設定**

作業情報からパラメータ設定に必要なデータを収集できたら，次に問題設定を行う．目的関数に最大所要時間の最小化と必要費用の最小化を設定し，制約条件として作業の順序関係を設定する．クリティカルパスの導出は節 2.1 で説明した方法を用いる．

**Step3：作業履歴を活用したパラメータ設定**

蓄積された過去の作業履歴のデータから費用のファジィ・ランダム変数を特徴づけるためのメンバシップ関数に必要な左右の広がりのパラメータ $\bar{\beta}_{ik}$,$\bar{\gamma}_{ik}$ と中心値 $\bar{d}_{ik}$ を設定する．左の広がりパラメータ $\bar{\beta}_{ik}$ は作業履歴の中の最小費用を，右の広がりパラメータ $\bar{\gamma}_{ik}$ には最大費用を，そして，中心値 $\bar{d}_{ik}$ には費用の平均値を設定する．

Step4：モデルの定式化と等価確定変換

ファジィ・ランダム変数にパラメータを設定したら，提案モデルを解くための定式化と式の等価確定変換を行う．作業履歴から収集したデータにより，意思決定者の判断でファジィ目標の最良値 $g_0$ と最悪値 $g_1$ を設定し，ファジィ目標 $G$ のメンバシップ関数 $\mu_{\tilde{G}}$ を設定する．更に，設定したファジィ目標を満たす可能性が満足基準値 $h$ より大きくなる確率を最大化する目的関数へ変換する．

提案モデルを解ける形に等価確定変換したら，遺伝的アルゴリズムなどの近似解法を用いた解の導出が可能になる．次に，MPI と RaspberryPi を用いた並列分散処理の流れを説明する．解法は節 3.3 と節 3.4 で説明したものを用いる．

Step5：並列分散 GA を用いた解法

等価変換した目的関数を並列分散 GA を用いて解く．まず，RaspberryPi 3 （ModelB）のマスター (1 台) で初期集団を生成する．マスターで生成した初期集団を各スレーブ (7 台) へ MPISemd() を使い配布する．配布した初期集団を MPIRecv() を用いて各スレーブにて受け取る．各スレーブで遺伝子操作（交叉・突然変異・選択・移住）を行う．移住は一定の世代間隔でランダムに選んだ遺伝子を他のスレーブへ送信する．この操作を繰り返し解の探索を行う．

**結果:** 所要時間とそのときにかかる費用を見積もることができ，各工程に対する職人さんと費用の最適な配分を補助することができる．

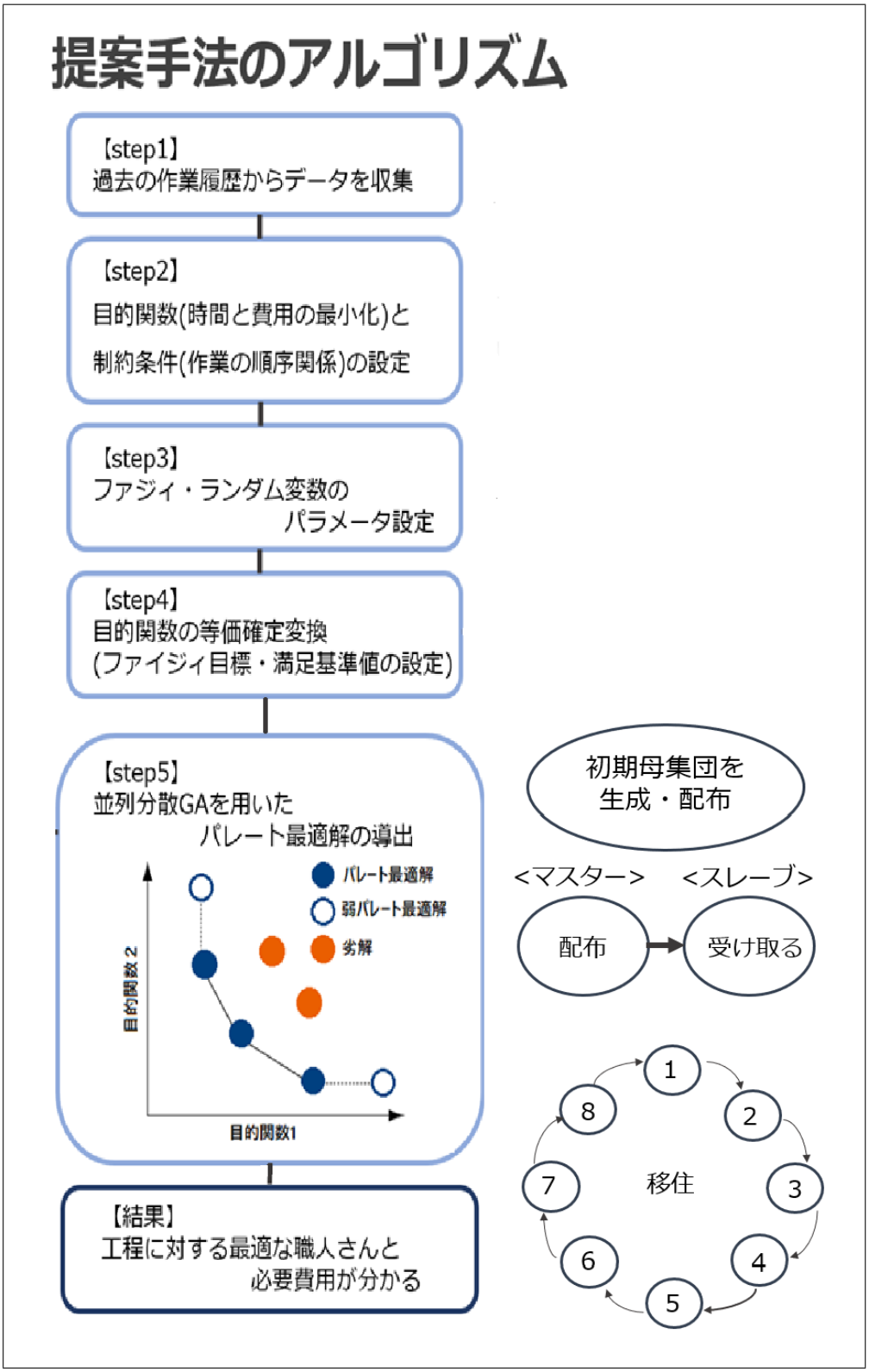


図 4.8: 提案手法の全体的なフロー

# 第5章

# おわりに

本研究では，建築業界における，資源の最適な配分による生産性の向上を目的とした日程計画に着目した．そのなかで，不確定性・不確実性を含む要素に着目し，過去の作業履歴を，実際の現場で活用されている現場コミュニケーションアプリなどを利用して蓄積・分析できると考え，取得したデータからパラメータ設定ができるような日程計画問題の提案を行った．更に，提案する問題に対しての定式化と等価確定変換，データの設定方法から解法までのアルゴリズムを示し，高速化の環境構築までを行った．

本研究で提案したファジィ・ランダム日程計画問題の五つの特徴をまとめる．一つ目の特徴は，既存研究の多くは静的で確定的なものを前提としており，現実問題への活用が困難であったが，現場の作業情報の不確実性と不確定性の両方を表現するために，問題の目的関数の係数にファジィ・ランダム変数を導入した点．二つ目は，建築現場の悩みから，今の現場に求められる仕組みとされる，職人さんの最適な割り振りを補助できる点．三つ目は，時間費用関数にって表現される所要時間と費用のトレードオフの関係を，所要時間と費用の最小化の多目的最適化問題にすることで，両方のバランスを考えた日程計画を考えることができる点．四つ目は，プロジェクトにおけるクリティカルパス上の費用と所要時間の最小化を目的とすることで，全工程のうちどの作業と職人の組合せを優先して管理すべきかが分かる点．五つ目は，処理時間の高速化を目的とした並列分散環境の構築を行った点．結論として，建築現場に必要とされる,「天候などの不確定・不確実な要素を考慮した所要時間と費用の見積もり」と「職人さんの最適な選択・割り振りの補助」を行いつつ，今の現場環境からでも得られるデータの範囲内で，取得したデータからパラメータを設定が行えるファジィ・ランダム多目的日程計画を提案することができた．

本研究の研究成果は，過去の作業履歴から取得できるデータを不確定で不確実な要素，本研究では作業にかかる費用を見積もるために，活用できるモデルの提案と定式化を行ったことである．提案したモデルは，建築現場に限らず，様々な業界における不確定で不確実な要素の見積もりや，最適な人員選択の補助が可能であると考えられる．

今後の課題として，提案したファジィ・ランダム多目的日程計画を解き，従来のファジィ・ランダム変数を用いていないモデルとの比較・検証を行い提案モデルの有効性を示す必要がある．更に，本研究で提案したファジィ・ランダム多目的日程計画問題では，クリティカ

ルパスを求め，そのクリティカルパスにかかる所要時間と費用が最小になるような職人さんの選択を行うことを考えた．しかし，クリティカルパス上の工程を管理することが，最も全体の作業効率の向上に効果的であるとされてはいるが，実問題で管理する工程がクリティカルパス上の作業だけでは実用するには不十分である．よって，ファジィ・ランダム変数を導入したクリティカルパス上への作業へ優先的に最適な人員の割り当てを行いつつ，他の作業の人員も最低限確保できるような問題への改良も検討すべきである．

更に，職人さんの最適な割り当てという部分に注視して考えると，今後の展望として，本研究で提案したような一つの大きなプロジェクトの中の作業のみではなく，いくつかの大きなプロジェクトの作業に対する職人さんのファジィ・ランダム日程計画問題も考えられる．なぜならば，建築現場における職人さん達の多くは，深刻な人手不足により，いくつも工事現場を掛け持ちしているためである．本研究では，より実問題に対応できるモデルにするため「作業費用の不確定性・不確実性の考慮」を考えたが，現実問題に対する日程計画問題には，前述したようなプロジェクトの掛け持ちのような複雑な制約を持つものも多く，まだ様々な方面からアプローチし，改善する余地があると考えられる．

# 謝辞

本研究を遂行するにあたり，多大なご指導と終始懇切丁寧なご鞭撻を賜った富山県立大学電子・情報工学科の奥原浩之教授に深甚な謝意を表します．最後になりましたが，多大な協力をして頂いた，群馬大学社会情報学部社会情報学科の松井猛准教授と奥原研究室の同輩諸氏に感謝致します．

2019 年 2 月

杉山 桃香

# 参考文献


[1] 國藤 進, “発想支援システムの研究開発動向とその課題”, 人工知能学会誌, pp. 552-559, 1993.

[2] イ スンジュ, “発想支援のためのテキストマイニング”, 人工知能学会 第 25 回 セッション ID: 1P2-10in, 2011.

[3] M.L. Puri, and D.A. Ralescu, “Fuzzy random variables,” Journal of Mathematical Analysis and Applications, vol. 114, pp. 409-422, 1986.

[4] 飯田耕司, “不確実性への挑戦 意思決定分析の理論”, 三恵社，2006.

[5] “PERT と CPM” , http://d-engineer.com/industrialeng/pert.html. 閲覧日 2018, 12, 20.

[6] “クリティカルパスでプロジェクト遅延要因を以前に把握せよ” ,https://pmkuma.com/critical-path-method/. 閲覧日 2018, 12, 20.

[7] 木瀬 洋, “スケジューリング研究の過去・現在・未来”, スケジューリングシンポジウム 2000 講演論文集, pp. 5-10, 2000.

[8] 嘉納成男, 石田航星, “繰り返し工程に関する工程計画手法の開発”, 日本建築学会計画系論文集, Vol. 81, No. 723, pp. 1195-1205, 2016.

[9] 吉川和広, 春名 攻, “ネットワーク手法による施工計画のシステムアプローチに関する研究-CPM 計算の簡便化-” 土木学会論文報告集, Vol. 182, pp. 41-58, 1970.

[10] “建築現場を IT 化する「ダンドリ」ツール 6 選”, https://mag.branu.jp /archives/2442. 閲覧日 2018, 6, 5.

[11] “現場コミュニケーションアプリ Kizuku”, https://www.ctx.co.jp/kizuku2pr/. 閲覧日 2018, 6, 5.

[12] David L. Applegate, Robert E. Bixby, Vasek Chvatal, William J. Cook, The Traveling Salesman Problem: A Computational Study, Princeton Univ Pr, 2007.

[13] M. K. Luhan （ljula. and M. M , Gupta； On fuzzy stochastic optimization. Fuzzy Sets and Systems, 81, pp. 47- 55, 1996.

[14] M.Gen, R. Cheng, Genetic Algorithms, Engineering Design, John Wiley, Sons, 1997.

[15] 矢野均, “多目的ファジィランダム単純リコース問題に対する対話型意思決定とその応用”, 日本経営工学会論文誌, Vol. 67, No. 4, 2017.

[16] 矢野均, “ファイジィ・ランダム単純リコース問題の定式化”, 日本知能情報ファジィ学会誌, Vol.27, No. 4, pp. 695-699, 2015.

[17] 蓮池隆, “CPM を用いた不確実・不確定状況下におけるクリティカルパスの求解”, 数理解析研究所講究録, Vol. 1829, pp. 72-79, 2013.

[18] 片桐英樹, 坂和正敏, 加藤浩介, 大崎修嗣, “ファジーランダム多目的線形計画問題に対する可能性測度と必然性測度を用いた満足水準最適化モデルに基づく対話型ファジー満足化手法”, 電子情報通信学会論文誌 A Vol. J87–A, No. 5, pp. 634–641, 2004

[19] 片桐英樹, 坂和正敏, 石井博昭, “ファジィランダム変数を含む線形計画問題に対する可能性計画と確率計画に基づく意思決定 (あいまいさと不確実性を含む状況の数理的意思決定)”, 数理解析研究所講究録, Vol.1252, pp. 240-245, 2002.

[20] 玉置 久, 森 正勝, 荒木　光彦, “遺伝アルゴリズムを用いたパレート最適解集合の生成法”, 計測自動制御学会論文集, Vol. 31, pp. 1185-1192, 1995.

[21] 堀井宏祐, 國藤進, 松澤照男, “非同期型島モデル並列 GA の評価”, 計測自動制御学会論文集, Vol.35, NO.11, 1394-1399, 1999.

[22] 末次峻也, 片山尋貴, 徳丸正孝, “並列 GA による感性データ解析”, 日本知能情報ファジィ学会誌, Vol. 25, NO.6, pp. 924-934, 2013.

[23] 山中亮典, 吉見真聡, 廣安知之, 三木光範, 横内久猛, “遺伝的アルゴリズムの並列計算システム向けフレームワークの提案”, 情報処理学会研究報告, Vol. 2010, No. 8, pp. 1-7, 2010.

[24] “Raspberry Pi とは”, https://furien.jp/columns/58/. 閲覧日 2018, 11, 6.

[25] “MPI の環境構築や基本コマンドのまとめ”, https://qiita.com/kkk627/items/49c9c35301465f6780fa. 閲覧日 2018, 11, 6.

[26] “MPI を用いた並列プログラミングの概要”, http://www.cenav.org/raspi4b/. 閲覧日 2018, 11, 8.

[27] “MPI 並列プログラミング”, http://www.matlab.nitech.ac.jp/ kazu-k/mpi.html. 閲覧日 2018, 11, 8.

[28] Hideki Katagiri , “Interactive multiobjective fuzzy random linear programming: Maximization of possibility and probability”

[29] M. Sakawa, I. Nishizaki, H. Katagiri, Fuzzy Stochastic Multiobjective Programming, Springer, 2011.

[30] 椎名孝之, “確率計画法”, 朝倉書店，2015.

# 付録

## A. 1　Hello World を並列実行するソースコード

Raspberry Pi 3 を 8 台で Hello World を並列実行するソースコード A.1 を示す.

ソースコード A. 1: hellow.c

```
1  #include <stdio.h>
2  #include "mpi.h"
3  int main( int argc, char *argv[] )
4  {
5      int rank;
6      int size;
7      double t0,t1,t2,t_w;
8      MPI_Status stat;
9      MPI_Init(&argc,&argv );
10     MPI_Comm_rank(MPI_COMM_WORLD, &rank);
11     MPI_Comm_size(MPI_COMM_WORLD, &size);
12     MPI_Barrier(MPI_COMM_WORLD);
13
14     t1=MPI_Wtime();
15     printf( "Hello world from process %d of %d\n", rank, size );
16     MPI_Barrier(MPI_COMM_WORLD);
17     t2=MPI_Wtime();
18     t0=t2-t1;
19     MPI_Reduce(&t0, &t_w, 1, MPI_DOUBLE, MPI_MAX, 0, MPI_COMM_WORLD);
20     MPI_Finalize();
21     printf("%f\n",t_w);
22     return 0;
23 }
```

## A. 2　円周率計算を並列分散処理するソースコード

Raspberry Pi 3 を 8 台で円周率計算を並列分散処理するソースコード A.2 を示す.

ソースコード A. 2: pi.c

```
1  #include <stdio.h>
2  #include <math.h>
3  #include <mpi.h>
4  void main(int argc, char* argv[]){
5   double PI25DT = 3.14159265358979323846264 3;
6   double mypi, pi, h, sum, x, f, a;
7   double t1, t2, t0, t_w;
8   int n, myid, numprocs, i, rc;
9   int ierr;
10
11 MPI_Status stat;
```

```
12 MPI_Init(&argc, &argv);
13 MPI_Comm_rank(MPI_COMM_WORLD, &myid);
14 MPI_Comm_size(MPI_COMM_WORLD, &numprocs);
15  if(myid==0){
16    printf("Enter the number of intervals \n");
17    scanf("%d", &n);
18    printf("n=%d \n", n);
19              }
20 MPI_Barrier(MPI_COMM_WORLD);
21 t1=MPI_Wtime();
22
23 MPI_Bcast(&n, 1, MPI_INT, 0, MPI_COMM_WORLD);
24    h = 1.0 / n;
25    sum = 0.0;
26  for(i=myid+1; i<=n; i+=numprocs){
27       x = h * (i - 0.5);
28       sum = sum + 4.0 / (1.0 + x*x);
29                                    }
30  mypi = h * sum;
31 MPI_Reduce(&mypi, &pi, 1, MPI_DOUBLE, MPI_SUM, 0, MPI_COMM_WORLD);
32  if(myid == 0){
33    printf("pi is approximately: %18.16lf Error is: %18.16lf \n", pi,fabs(pi-
          PI25DT));
34                 }
35 MPI_Barrier(MPI_COMM_WORLD);
36 t2 = MPI_Wtime();
37 t0= t2-t1;
38 MPI_Reduce(&t0, &t_w, 1, MPI_DOUBLE, MPI_MAX, 0, MPI_COMM_WORLD);
39
40  if(myid==0){
41    printf("execution time = : %8.4lf [sec.] \n", t_w);
42              }
43   rc= MPI_Finalize();
44
45 }
```