意味情報に基づくエラーリカバリー機能を有する FA ロボットの資源制約下での合理的意思決定に関する研究

松岡 諒

2022 年

摘要

本研究は，産業用ロボットの自律的なエラーリカバリーシステムの構築，ならびに資源制約が存在する FA の現場でのリカバリー行動計画の生成と実行に関する意思決定方式の体系化を目的としたものである．ここでいう FA の資源とは，不確実性下の行動判断に許容される時間資源や実際のシステムが備えるセンサで収集可能な情報資源のことである．従来のような人の介入を必要とせず，リカバリー作業を遂行するロボットの行動は，現場で入手可能な情報とそれらを処理する時間の制約の中で合理性が担保されたものでなければならない．本研究では，時々刻々と変動する生産状況下で熟考を要するエラー対応が機械側に委ねられるべき領域を判断可能な，メタレベル意思決定能力を有する FA ロボットシステムをデザインする．本研究の成果によって持続可能な生産の実現に不可欠である，機械が自身の能力と置かれている状況を考慮した行動を取ることが可能となる．

近年，生産年齢人口の低下による労働力不足が顕在化する中で，産業用ロボットの導入による生産の自動化が進んでいる．大量生産から多品種少量生産，さらに消費者の嗜好に応じた変種変量生産が望まれるようになってきたことで，ロボット化が期待される作業も多様化し，個々のロボットアームが複数工程をこなすセル生産方式の採用が主流となりつつある．ロボットセルでは熟練作業者の代替となる高速高精度かつ柔軟なマニピュレーションを継続することが求められている．

産業用ロボットシステムの安定稼働を実現する上で，課題の 1 つとされているのがチョコ停への対処である．ロボットによる自動化生産を開始する前には，対象作業を成功裏に導く一連の動作プログラムが作り上げられるが，実際に生産を開始すると意図したとおりにロボットの動作が完了せず，エラー停止してしまうことがしばしば発生する．特に作業空間でロボットが取り得る動作とワークの自由度が多いセル生産では，エラー事象を事前に網羅してリカバリー用の動作プログラムを作り上げることは不可能に近い．現状ではロボットが自律的に対処できているのはワークの再把持などの単純なコマンドレベルにとどまり，予期せぬエラー信号に対しては人が駆け付けて原因を調査し，リセットをかけるまで作業を中断せざるを得ないシステムが大半である．

そこで本研究では，エラー要因によって正常系の計画からの乖離が生じた場合に，それを解消するために必要な行動計画レベルでの熟考的リカバリーによって，作業を継続できるロボットシステムの実現を目指す．熟考的なエラーリカバリーを対象作業によらず適用可能とするために，特に以下の 3 つの課題に取り組んだ．第一に，ロボットはエラーリカバリーのための推論に際して，作業空間にあるどこまでの要素を考慮すればよいかというフレーム問題に直面する．失敗した動作の情報が与えられたとき，エラー要因を解釈してプランニング空間でのリカバリーの目標を定めるための汎用的な修正戦略の体系化が必要である．第二に，実環境で検知されるエラー情報はセンシングの不確実性を含むものであり，選択される修正戦略が真のエラー状態に適合しなかった場合のリスクが存在する．正常系とは別の，リカバリー行動計画の実行が現場で受け入れられるためには，抽出されたエラー要因の尤もらしさを評価した上で実行判断がなされるべきである．第三に，リカバ

リー行動計画は，生産現場の目標稼働率により変動するデッドラインまでに完了することをもって価値のあるものといえる．確率的なエラー要因の評価値に対して，今どれだけの時間コストを費やして追加のセンシングによる不確実性低減策を取り入れた計画を求めるべきかという，時間資源や情報資源に限界があるもとでの合理的意思決定モデル (限定合理性モデル) の構築が要請される．本論では．収集した情報量やその処理能力・時間に制約を受ける中で合理性を担保しなければならない FA ロボットの資源制約下での合理性の問題として，限定合理性を取り扱う．

初めに，第 1 章では，本研究の背景ならびに関連研究の動向について概説するとともに，本論文の目的，以下の章の構成について述べる．

第 2 章では，階層的アーキテクチャを持つ提案システムの構成要素と，行動計画手法ならびにリカバリー処理フローについて説明する．本研究で提案する複数台ロボットのリカバリー行動計画システムは，対象作業の行動計画の構造を管理・修正するプランナと，プランナで導出された順序制約の下でロボットの最適な動作実行手順を決定するスケジューラを介して，下位のコントローラから各ロボットへの制御指令を出力する．プランナでは，リカバリーのための副目標が与えられた際に元の計画の部分的な変更により，必要な動作と制約を導出可能な，半順序プランニングを活用する．スケジューラでは，作業再開時点から最小時間で目標状態を達成できるように，A^* アルゴリズムを用いた最適スケジューリングを実行する．これらのことを踏まえて，ロボット作業中のエラーが検知されてから，リカバリーに必要な行動手順を埋め込んだスケジュールを導出して作業を再開できるまでの処理の進み方を示した．

第 3 章では，格文法に則って構成される意味ネットワークである Conceptual Graph を用いたロボットの汎用動作の抽出と，それらに含まれる深層格の意味情報に基づくエラー要因の類型化を行った．人工知能の知識表現において，数ある述語概念は，それらに係る格の構造に注目することで少数の意味素の集合に閉じて定義できるとされている．産業用ロボットの動作コマンドはメーカーごとに専用のプログラミング言語で記述されているが，物体操作を行うために不可欠な共通の述語が存在すると考え，ロボット自身が移動する“Move”，ロボットが物体をつかむ“Grasp”，ロボットが物体に力を加える“Apply”を汎用動作として抽出した．行動計画のレベルでこれらの述語の前提条件と事後条件を整理し，Conceptual Graph の格フレームを定義した．このフレームに含まれるノードと失敗した動作の情報を照合することで，エラー要因の格関係が抽出され，再び前提条件を充足してリカバリーに至るまでに何をしなければならないかが示される．それぞれの汎用動作でエラーが生じる状況と対応する格関係を分析して，修復すべき格関係という形で修正戦略を体系化することができた．

第 4 章では，Conceptual Graph で抽出されたエラー要因の候補となる格関係と，その格関係の状態を示すセンサ情報を結びつけるベイジアンネットワークを構成し，事後確率によって不確実性を評価する枠組みを導入した．ロボットハンドに備えられる力覚センサやセル全体を俯瞰するビジョンセンサなど，入手可能なセンサ情報はシステムによって異なる．提案するフレームではエラー要因の候補に対するセンシング手段を選択して，事後確率を更新するベイジアンネットワークを動的に構成し，信念の伝播によってエラー要因の尤もらしさを評価する体系を得ることができた．

第 5 章では，FA における限定合理性が，時間圧を伴うプラント上流の生産計画に依存することから，不確実性を低減する行動を取り続けることが資源制約下では必ずしも合理的でないことを説明可能な時間依存効用理論を導入し，時間選好とリスク選好の二重性を持つ意思決定を制御するモデルを構築した．Conceptual Graph とベイジアンネットワークによって評価された第 1 候補のエラー要因を最短工数で解消する行動計画の代替案として，可能性を排除できない第 2 候補以下のエラー要因にも対処できる冗長性を持たせた行動計画を求めておくことが考えられ，追加のセンサ情報を取得してエラー要因の評価値が更新されるまで判断を保留し

行動計画の質を高めることも考えられる．一方で，このような熟考を行っている間にも，早期にリカバリーを完了することで得られるはずの利得は減少していく．このような状況下でリカバリー動作の実行に失敗するリスクに対する意思決定主体の態度は，時間依存効用関数で表現される．提案システムは，所定の生産性を果たすために要請されるデッドラインという時間圧が変動したとき，現場で同定可能な効用関数にしたがって意思決定を行う能力を有する．さらに，安価な製品を短い周期で生産する状況では推論時間の短いモデルが有利であり，高価な製品を長い周期で生産する状況になると慎重に考えるモデルを使うべきであるというように，現場の因子に紐づけて意思決定モデルの制御方法を示した．

第 6 章では，提案システムの有効性を検証するため，意図的にロボットの動作中のエラーを起こす実機実験に相当するシミュレーション評価を行った．題材としては，2 台のロボットアームによるギヤユニットの組立作業を取り上げた．ロボットセルの空間的制約の下で，半順序プランに基づき実行可能なスケジュールが導出される．失敗するロボットコマンドは同じでもエラー要因が異なる事例を作成し，与えられた効用関数でリカバリーに至るまでの熟考的対応の結果を比較した．修正戦略の不適合によるリカバリー失敗のリスクが小さい事例では不確実性低減のための追加センシングは実行されず，同じセンシングコストでも失敗のリスクの大きい事例では時間圧が大きくなってくるまで追加センシングを行うという結果が得られた．さらに，リスク回避的な効用関数で評価する場合とリスク志向的な効用関数で評価する場合で，時間圧の変化に対して採用される行動計画に差異が見られた．最後に，提案システムの汎用性の考察として，ロボットの台数が増えた場合や，より複雑な組立作業でエラーが発生した場合にも対処できることを示した．

第 7 章では，本論文で得られた知見をまとめるとともに，今後の課題と展望について述べる．

目次

第 1 章

序論

1.1 背景

世界的な生産年齢人口の低下による人手不足が顕在化する中，産業界ではロボットの導入による生産作業の自動化が進んできている．産業用ロボットは，日本工業規格（JIS）において「自動制御され，再プログラム可能で，多目的なマニピュレータであり，3 軸以上でプログラム可能で，1 か所に固定して又は移動機能をもって，作業自動化の用途に用いられるロボット」と定義されている．工場内で人に代わって組立・搬送・溶接・塗装・検査などを実施するロボットアームが産業用ロボットと呼ばれる．日本では 1990 年代頃から消費者のニーズ・嗜好の多様化に伴い，大量生産から多品種少量生産，そして変種変量生産へと，注文に応じた生産が求められるようになってきた．これに対応するロボットが導入される生産方式も，単工程が直列接続される従来のライン生産方式から，作業者が与えられた区画で複数工程をこなすセル生産方式へとシフトしてきた．自動化ラインの生産性と人セルの柔軟性を併せ持つことを特長とするロボットセルの例を Fig. 1.1 に示す．ロボットセルには，作業を行う少数台のロボットアームとともに，製品のパーツフィーダや工具，回転ステージ，組み上がった製品を掃き出すコンベア，セルの状態を俯瞰的に監視するビジョンセンサ等が設置される．人と同じように柔軟なマニピュレーションをロボットで実現するため，センサデバイス，情報処理，行動計画，行動実行，ハンド制御等に関する知能化要素技術の開発が進められている [1][2]．さらに，ここ数年では，安全柵による隔たりなく人と作業空間を共有する人協働ロボットが普及し始めており，多種多様なマニピュレーションを人と同等以上のタクトタイムでこなすことがロボットに期待されている．

産業用ロボットで生産を始めるためには，対象の作業環境で人がロボットに一連の動作を教示し，その動作を再生して正常に作業が遂行されることを確認する，ティーチング・プレイバックの過程を経ることが一般である．ティーチング方法としては，ティーチングペンダントと呼ばれる端末を用いて目の前の実機に動作を組み込むオンラインティーチング，PC 等で実機に転送可能な動作をプログラミングするオフラインティーチング，ロボットアームをつかんで直接動かしながら教示点を記録するダイレクトティーチング等があり，いずれにしてもロボットは成功裏に作業を完了する事が想定された動作を実行する．しかしながら，実際に生産を始めると，前工程から供給されるワークとハンドの位置合わせの誤差，ワークの形状や剛性による把持力不足，ジグの経年劣化といった様々な要因が組み合わさって，意図したとおりに動作が完了せず，ロボットがエラー停止してしまうことがしばしば発生する．こうしたエラーは機械の致命的な故障ではなく，上記のような要因を取り除いた上でリセットをかければ作業再開できるものであり，産業界では「チョコ停」と呼ばれる．1 日当たり数十回のチョコ停が発生することは珍しくなく [3]，生産設備の稼働時間 = 実際に作業を行っている時間 + チョコ停時間 + 空転等の時間のうち，1 日のチョコ停の合計時間が 1 時間以上にのぼることもある．こ

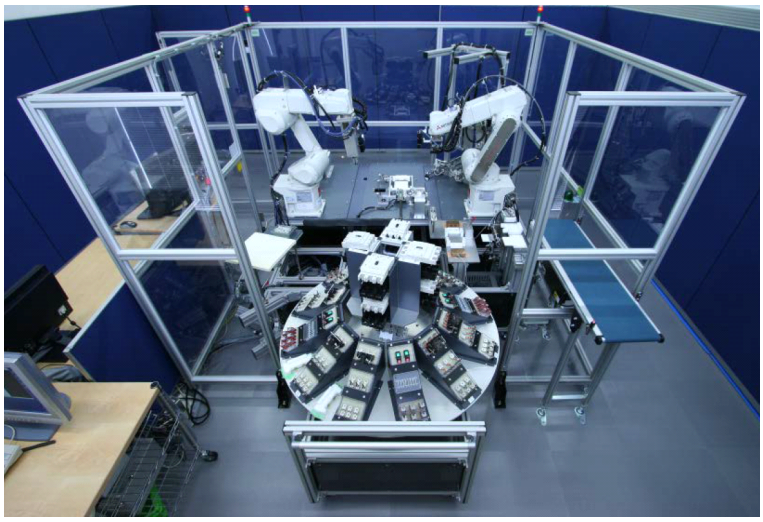


Fig. 1.1: Robotic cell production system (Mitsubishi Electric Corp.) [1]

うしたチョコ停への対処は FA(factory automation) システムの歩留まりに大きく影響する課題であり，リカバリー用の動作をプログラミングすることは，ロボットの立上げを担うシステムインテグレータ (SIer) にとって正常系のプログラミング以上に時間と労力を要する作業となっている．

しかしながら，複合的な要因によって発生するエラー状態の条件分けにより事前にリカバリー動作を網羅しておくことは実質的に不可能であり，ロボットが自律的にエラー状態から復帰できるケースはセンサの閾値に対してコマンドレベルの再試行が可能な場合，例えば手先の力覚センサ値にしたがってワークの再把持を試みるなどの単純な対応に限られるのが現状である．大半のロボットシステムはこうした反応的 (reactive) な対応の範疇を超えるチョコ停に対して，知識を有する人が駆け付けてエラー要因を取り除くまで待機しておくことを前提に設計されている．その理由としては，正常系の計画がエラー要因によって破綻した際，これまではリカバリーのために新たに必要となる動作を導出する指針となる修正戦略が体系化されていなかったことと，もしリカバリー動作が自動生成されたとしても，ロボットが当初の正常系の計画から動作を切り替えて実行することに対するリスクまで考慮した判断能力を有していなかったことが挙げられる．前者を解決するためには作業対象によらずロボットのマニピュレーションに伴うエラー状態の影響を表現する汎用的なフレームワークが，後者を解決するためにはそのフレームワークと実際のロボットシステムで収集可能な情報を結び付けて不確実性下でのリカバリーの実行方法を判断する仕組みが必要となる．これらの要素を備えることで実現されるロボットの熟考的 (deliberative) なエラー対応は，システムとしてエラーを完全に排除することよりも復旧の過程を重要視する近年の Safety-II の考え方 [4] に適合するものであり，Industry 5.0 のキーワードとしても掲げられている，持続可能でレジリエントな生産というこれからの FA 業界の在り方に不可欠なものといえる．

1.2　関連分野の研究動向

1.2.1　ロボットの行動計画に関する研究

作業レベルでの計画を管理しつつリアルタイムな動作の実行を志向したロボットの制御アーキテクチャとしては，1980 年代に Brooks が提唱したサブサンプション・アーキテクチャ (subsumption architecture)[5] を契機に，階層型アーキテクチャの研究が行われてきた．サブサンプション・アーキテクチャは物体回避といった

下位レベルから行動探索といった上位レベルまで，センサ入力に対して各階層が並列に処理を行い，上位層の解が得られた段階で下位層の出力を抑制・包摂する仕組みを備えることで，止まらないロボットの動作を実現する．後発の研究 [6][7] では，上位の問題解決に向けた各階層の選択と統合に関する改良が行われ，いかにして計算機上の行動計画と実環境にいるロボットの制御をシームレスにつなぐかに主眼が置かれてきた．近年では，産業用ロボットに対するこうした一連の計画問題が Task and Motion Planning (TAMP) として認知されるようになり [8][9]，作業レベルでの教示によって新規ユーザのロボット導入に対する敷居を下げるためのフレームワークの開発が進んでいる [10][11].

1.2.2　ロボットの自動リカバリーに関する研究

ロボットのエラーリカバリー技術は，エラーの検知，診断 (同定)，復旧のフェイズに分けることができる [12]．ここでの復旧とは，同定されたエラーに対して有効なロボットの動作を導出・決定し，正常な作業状態を実現することを指す．

エラーの検知については，時系列データとして取得されるロボットシステムのセンサデータをもとに，正常系からの乖離を早期に検知することが求められる．センサの性能に依存したデータのばらつきが存在するため，ある程度のマージンを持たせて異常と判定する閾値を設定することが通常である．時系列データのばらつきを考慮した機械学習的な異常検知手法としては，事前に相当な量の正常データを採取しておき，正常系のデータの分布からの乖離度合の累積値が閾値を超えた時点で異常と出力する方法が存在する [13]．近年では，教師なしでの特徴量の抽出 [14][15]，VAE (variational autoencoder) [16]，GAN (generative adversarial networks) [17] といった深層学習の適用で，特定のエラーに対して優れた検知性能が示されている．

エラー状態の診断については，ロボットアームに加わる力覚情報，モータの音，振動，電流値，カメラ画像などのマルチモーダル情報処理を行い，エラー状態の特徴量を抽出する分類問題として研究されている．サポートベクターマシンやニューラルネットワークを用いた教師あり学習がこれに相当する [18][19].

エラー状態からの復旧については，ペトリネットを用いてリカバリープロセスを導出した研究が存在する [20][21]．産業用ロボットの具体的な作業におけるエラー状態の類型化 [22] や階層化 [23] に取り組んだ研究も存在する．

1.2.3　ロボットの知識表現に関する研究

熟考的リカバリーのための推論を行う上でロボットが直面するのが，フレーム問題である．フレーム問題は，無限の可能性が考えられる現実世界において，与えられた課題に関係のある事象だけを選び出すことが人工知能にとっては非常に難しいという問題である．有限時間内に解を導くためには，どこまでの可能性を考慮し，それ以上は考えなくてよいというフレームを設定する必要がある．推論能力を持つロボットに必要となるのは，対象世界で実行すべき作業に関する概念間の関係を統一的な枠組みで表現すること，すなわちオントロジー (ontology)[24] を獲得することである．特に，グラフ表現によって明示的にロボットのオントロジーを獲得する研究が行われている [25][26]．FA 分野では，AI が下した判断を適用する際の安全性の担保のため，その判断根拠の説明可能性が重要であることを踏まえて，オントロジー的なフレームワークを適用した研究が出てきている [27][28].

1.3　本研究の目的

本研究では，ロボットが失敗した動作の意味を表す情報を活用してエラー状態の背後要因を推論し，検知された状態の不確実性に加え生産現場の状況の変動をも考慮した「合理的な」意思決定に基づく行動計画の修正ならびに実行を可能にするエラーリカバリーシステムの構築を目的とする．FA の領域における合理性は，意思決定者が認知・計算能力の限界により限られた合理性しか実現できないという限定合理性 (bounded rationality)[29][30] を，FA システムの資源制約下の合理性として取り扱うものとなる．FA の資源とは，不確実性下の行動判断に許容される時間資源や，実際のシステムが備えるセンサで収集可能な情報資源のことである．特に提案する熟考的リカバリー機能により，従来のように人の介入を待つことなく，自律的判断で作業を遂行するロボットの行動は，現場で入手可能な情報やそれらを処理する計算能力・時間といった数々の制約の中で合理性が担保されたものでなければならない．本研究ではこの点について，動的な生産状況下で機械側に委ねられるべき熟考の領域を判断するメタレベル意思決定の枠組みを導入することにより，持続可能な生産に向けて現場で真に有用な FA ロボットシステムをデザインすることを目指す．

1.4　本論文の構成

本論文の構成を以下に示す．まず，第 2 章では，提案システムの構成要素と，行動計画手法ならびにリカバリー処理フローの全体像について説明する．第 3 章では，検知されたエラー状態に対して行動計画を行うために，産業用ロボットの汎用動作の失敗に関する知識表現を用いた修正戦略の体系化に取り組む．第 4 章では，エラー検知に伴うセンサ情報の不確実性を考慮して適切な修正戦略を選択するための評価方法を構築する．第 5 章では，エラー状態の不確実性とリカバリー実行完了までの時間コストの制約下での意思決定が求められる生産現場の期待効用の観点から，意思決定モデルの判断基準を定めるメタレベル意思決定能力を有するシステムの制御方法について論じる．第 6 章では，複数台のロボットアームによる組立作業のシミュレーションによって提案システムの有効性を検証する．最後に，第 7 章では結論として，本論文で得られた知見をまとめるとともに，今後の課題と展望について述べる．

第 2 章

行動計画の修正を伴うエラーリカバリーシステムの構築

2.1 緒言

本章では，産業用ロボットの自律的なエラーリカバリーを実現する本研究の提案システムの構成要素と，そのリカバリー処理フローについて述べる．

まず，本研究で対象とするプランニングの定義について明らかにしておく．ロボットに関するプランニングの階層性と各層の抽象度を示す概念図を Fig. 2.1 に示す．人工知能分野におけるプランニング (自動計画，automated planning) とは，与えられた環境モデルの中で，操作子 (operator) と呼ばれる状態を遷移させる規則が与えられたとき，目標状態を達成するために必要な操作子の系列を導出することを指す．最も抽象的な Planner のレベルでは，対象とする作業に必要な操作子の系列を得ることが目的であり，これが行動計画 (action planning) または作業計画 (task planning) と呼ばれるものである．例えば，「本棚にある本を机に置く」という作業では「本棚の前に移動する」→「本を取り出す」→「机の前に移動する」→「持っている本を放す」という操作子の系列が必要になる．後述する半順序プランニングにおいては，この行動計画の段階で実行手順の自由度を残し，その下の Scheduler のレベルで実際に作業を行うロボットに依存するコストや空間的制約などを考慮して，実行手順を決定するためのスケジューリングが行われる．一方，ロボットに固有のプログラミング言語で実装されるコマンドにしたがって，そのロボットの制御周期での動作指令を生成することは，モーションプランニングまたは軌道生成 (主にロボットアームの場合) と呼ばれる．従来の産業用ロボットのコントローラ内で実施されるのはモーションプランニングのレベルであり，ロボットは自身や周辺環境に備え付けられたセンサのフィードバック情報を得ながら作業を進めていく．

本研究で新たにロボットシステムに組み込むことを考えるプランニング機能は，Fig. 2.1 のプランナで実施される行動計画と，スケジューラで実施されるスケジューリングを対象とする．本研究が対象とする FA ロボットのエラー状態とは，普段はうまくいく動作コマンドが対象ワークや環境の要因で正常終了に至らず一時停止している状態 (チョコ停) であり，ロボット自身が壊れているわけではないため，エラー要因の解消に向けて取るべき行動が導出できれば復旧し得るものである．提案システムでは，エラーによってロボットの作業が中断されたとき，Planner や Scheduler を介する行動計画のレベルまでさかのぼって修正を行い，自動リカバリーを実現する．そこで，実環境でセンサ信号として検知されたエラー情報から，それが正常系の行動計画にどのような影響を及ぼしており，リカバリーのためにはどのような操作子が新たに必要であるかを抽象的なレベルで推論・評価する体系を構築する．

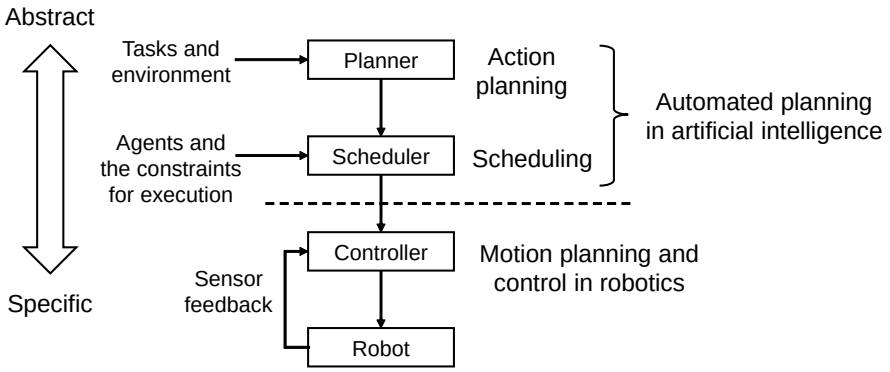


Fig. 2.1: Hierarchy of forward planning of robots.

2.2　関連研究

古典的プランニングの源流と言われる有名な手法に STRIPS[31] がある．STRIPS は全順序プランニング (total order planning) または線形プランニング (linear planning) と呼ばれる方式である．環境モデルの現在の状態に対して前提条件を満たす操作子 (operator) を適用することで状態が遷移する．STRIPS は与えられた目標状態が満たすべき条件を副目標 (sub-goal) に分解して，副目標を達成する操作子の系列を部分的に構築していき，最終的にそれらをひとつなぎにすることで，初期状態から目標状態に至る操作子の系列を一意な解として出力する．STRIPS は 1970 年代に開発された方式であるが，今なお実用的なプランニングの基礎として使用されている [32][33].

一方，行動計画の段階において実行手順の全順序は必ずしも必要でなく，自由度を残しておく方が修正の際に有利な場合がある．この考え方が，最小拘束の原理 (principle of least commitment) に基づく半順序プランニング (partial order planning) または非線形プランニング (non-linear planning) と呼ばれる手法である [34][35]. 半順序プランニングは，ロボットのように行動計画を実行する主体を考えたとき，階層的なプランニングアーキテクチャの構築を可能にする．すなわち，目標状態を達成する行動計画は必要最小限の制約をもって上位レベルで管理しておき，より具体的な下位レベルで各ロボットの実行手順を決定することができる．この階層性を活かした半順序プランナの実装例が存在する [36][37][38].

以上の先行研究は初期状態から目標状態を目指す順方向のプランニングである．一方，本研究では，エラー状態からの復帰，すなわち元の計画のどの時点までさかのぼって修正するかという逆方向のプランニング問題において，半順序プランニングの自由度を活用することを考える．

さらに，プランの実行段階において必要となる，最適な実行手順の決定に関するスケジューリング問題を解くために，本研究でも用いるヒューリスティック探索アルゴリズム [39][40] や遺伝的アルゴリズム [41] を適用した研究が存在する．近年では，タスクの実行順序の制約や作業空間の制約の下で，複数台ロボットのスケジューリング [42][43] や人とロボットの協調作業のスケジューリング [44][45] が実施されている．

2.3　行動計画システムの構成

本節では，まず，複数台のロボットの作業においてエラーリカバリーを達成するための行動計画システムの構成を示す．次に，提案システムにおいて活用される行動計画の手法である，半順序プランニングと最適スケジューリングの詳細について説明する．

2.3.1　リカバリープランニングのための階層型アーキテクチャ

本研究では，複数台のロボットが並行して進める作業の行動計画を中央集中的に管理するシステムを想定する．提案システムの構成を Fig. 2.2 に示す．

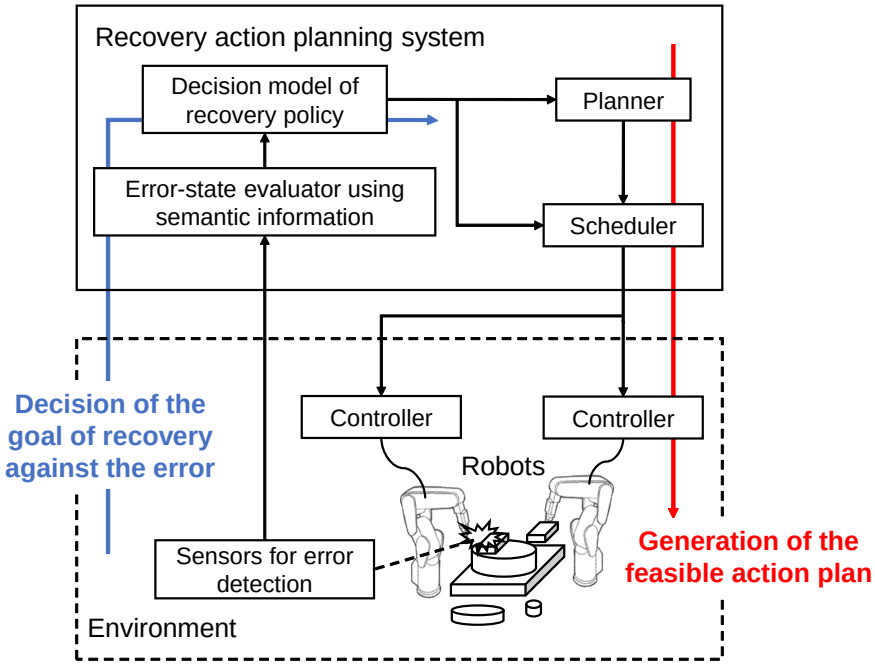


Fig. 2.2: Structure of the recovery action planning system.

作業環境では，各ロボットがコントローラから送信されるコマンドを逐次実行し，センサからのフィードバック情報を得る．コマンドを実行する順序やタイミングは，行動計画システム内のスケジューラで管理される．スケジューラの上位には，対象とする作業の目標状態を達成するために必要な行動を生成するプランナが存在する．作業環境でエラーが発生すると，センサが検出したエラーの情報と，正常に完了されなかったコマンドの情報が，行動計画システムに送信される．行動計画システムはまず，このエラー状態に関する意味情報の評価を行い，正常系に復旧するための方針を決定する．決定された方針は，新たな副目標のリストとしてプランナに送信され，各ロボットの作業再開のタイミングがスケジューラに通知される．プランナは，与えられた副目標を達成するプランステップとなるロボットの行動と，それらのステップの間の順序制約を導出する．スケジューラは，プランナで導出された順序制約を満たすように，各ロボットにプランステップを割り当て，再開後の作業手順を更新する．

2.3.2　半順序プランニングによる行動計画

半順序プランニングは，目標状態を達成する上で互いに独立に適用可能な操作子の実行順序の任意性を残して行動計画を記述する手法である．出力される半順序プランは，必要な操作子を表すプランステップ，プランステップ間の前提条件と事後条件の因果関係を表す因果リンク，その因果関係を充足する必要最小限の順序制約で構成される．一例として task1, task2, $\cdots$, task6 をプランステップとする半順序プランを Fig. 2.3 に示す．順序制約「$S \prec W$」は，プランステップ S がプランステップ W よりも先に実行される必要があることを表す．

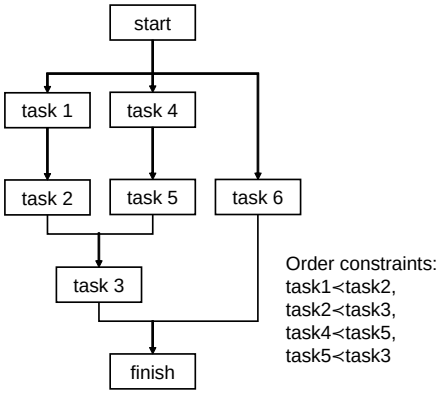


Fig. 2.3: Example of a partial order plan.

半順序プランニングのアルゴリズムを Algorithm 1 に示す．これは POP アルゴリズム (Partial Order Plan algorithm) と呼ばれ，出力されるプランに含まれるプランステップの数を最小にする．以下では，述語論理に基づくプログラミング言語である Prolog[39] の記法に則り，リスト中の英大文字で始まる各要素は変数を表す．

Algorithm 1 POP algorithm

Require: 初期状態を表す条件のリスト S0=[Cond1, Cond2, · · ·]，目標状態を表す条件のリスト Goals=[Goal1, Goal2, · · ·]

Ensure: 半順序プランのリスト Plan，順序制約のリスト Constraints

1: ある時刻からある時刻まで成立する条件のリスト TrueConds←[Cond1/0/T1, Cond2/0/T2, · · ·]
2: ある時刻で必要な条件のリスト OpenConds←[Goal1/Finish, Goal2/Finish, · · ·]
3: **while** OpenConds の条件のうち TrueConds で未成立のものが存在 **do**
4: OpenConds の先頭の要素 Goal/Time を取り出す
5: **if** TrueConds に要素 Goal/T1/T2 が存在 **then**
6: Constraints に T1<Time<T2 を追加
7: **else**
8: Goal を達成する Action/Time1 を Plan に追加
9: Action の前提条件 [Precond1, Precond2, · · ·] に対して，OpenConds に [Precond1/Time1, Precond2/Time1, · · ·] を追加
10: Action の事後条件 [Cond1, Cond2, · · ·] に対して，TrueConds に [Goal/Time1/Time2, Cond1/Time1/T21, Cond2/Time1/T22, · · ·] を追加
11: Constraints に Time1<Time≤Time2 を追加
12: **end if**
13: **end while**
14: **return** Plan, Constraints

2.3.3 最適スケジューリングによる実行手順の決定

前節の方法で導出された半順序プランは，エラーリカバリーに際して新たに要請されるプランステップを因果リンクに組み込むことで，実行済みのプランステップのすべてを初めからやり直すことなく，最小限の順序制約の追加で目標状態を達成するための計画を修正できるという利点を有する．一方，コントローラから各ロボットに動作コマンドを与える段階では，プランステップを実行する全順序が一意に決定されている必要がある．そこで，半順序プランからロボットコマンドの系列への橋渡しを担うスケジューラでは，順序制約を満たした上で作業時間が最小となる実行スケジュールを導出する．

m 台のロボットの作業時間最小化を目的とする最適スケジューリングは，A^* アルゴリズムを用いた探索で実現可能である [39]．このフェーズでは，各プランステップの実行に要する時間がコストとして定義される．一例として，Fig. 2.3 の半順序プランを 2 台のロボットで実行する場合のスケジュールを Fig. 2.4 のタイミングチャートに示す．網掛けで示される idle 時間は，task2 と task5 の両方が完了した後で task3 が実行可能になるという順序制約を満たすために挿入されている．作業時間が最小のスケジュールとは，全ロボットの idle 時間の総和が最小となるようにプランステップが配分されたスケジュールにほかならない．

A^* アルゴリズムは回の最適性を保証するヒューリスティックなグラフ探索アルゴリズムであり，探索空間での状態を表すノード n を経て目標ノードへ到達するときの評価関数を $f(n) = g(n) + h(n)$ として，総コストが最小となる経路を導く．ここで，$g(n)$ は出発ノードからノード n までに要したコスト，$h(n)$ はノード n から目標ノードまでの推定コストである．スケジューリングを実行するためには，半順序プランに含まれる

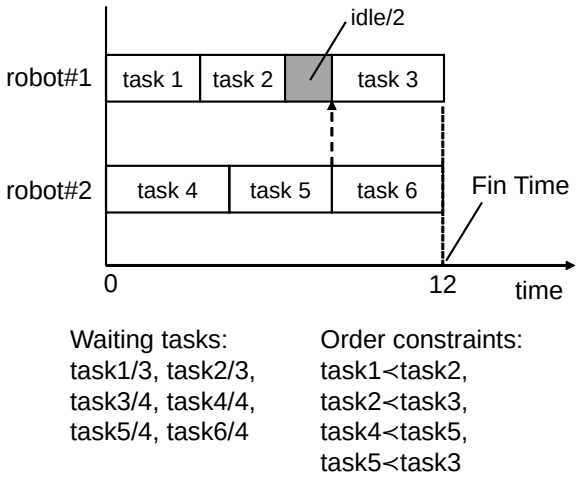


Fig. 2.4: Example of an optimal execution procedure generated from the partial order plan.

各々のプランステップを実際のロボットで実行する場合の時間コストを定めておく必要がある．今回の問題設定では，どのロボットで実行する場合も各ステップのコストは同じとする．この場合の最適スケジューリングのアルゴリズムを Algorithm 2 に示す．ここでは，実行コストが付与されたプランステップをタスクと呼んでいる．

Algorithm 2 Optimal scheduling by A^* search algorithm

Require: これから割り当てるタスクのリスト WaitingTasks

Ensure: m 台のロボットで実行するタスクのリスト ActiveTasks

現在できあがっているスケジュールの終了時間 FinTime←0
while WaitingTasks が空でない **do**
　WaitingTasks の先頭の要素 Task/Cost を取り出す
　if Task/Cost が現時点で実行可能 **then**
　　ActiveTasks の中で最も早く空くロボットに Task/Cost を割り当て，FinTime を更新
　else
　　ActiveTasks の中で最も早く空くロボットに何もしないというタスク idle/Cost1 を割り当てる
　end if
　WaitingTasks に残っているタスクのコストの和 S_D，ActiveTasks の終了時間の和 S_F を計算
　$FinAll = (S_D + S_F)/m$，Fin =max{ 現在の ActiveTasks の終了時間 } を計算
　if $FinAll > Fin$ **then**
　　$H = FinAll - Fin$
　else
　　$H = 0$
　end if
　$g(n) \leftarrow FinTime$，$h(n) \leftarrow H$ として，タスクを 1 つ割り当てた後のスケジュールを表すノード n の評価値 $f(n) = g(n) + h(n)$ とする
　探索木を一段伸ばし，これまでに展開されたノードの集合のうち最も $f(n)$ の小さいノードに進む
end while
return ActiveTasks

2.4　リカバリー行動計画時の処理フロー

本節では，ロボットが動作実行中にエラーを検知してから作業を再開できるようになるまでの提案システムの処理フローを説明する．提案システムのリカバリー行動計画の処理フローを Fig. 2.5 に示す．このフローに含まれる各ステップの実施内容を以下で説明する．

2.4.1　作業中のエラーの検知

ロボットシステムが備えるセンサがエラー値を検知することが，リカバリー処理のトリガーとなる．産業用ロボットの内部状態や周囲の状態を知るためのセンサとして，力覚センサやビジョンセンサがよく用いられる．それぞれのセンサには，ロボットの正常動作の教示段階でエラーと判断する閾値が設定されている．リカバリー行動計画システムは，センサが検知したエラー情報を，正常に実行完了されなかった動作コマンドの情報とともに受信する．

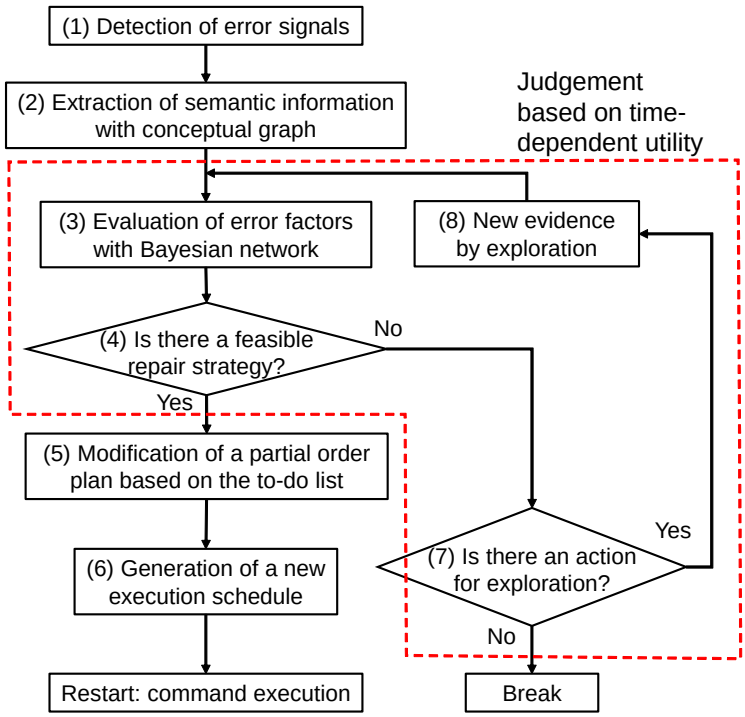


Fig. 2.5: Recovery process of the proposed system.

2.4.2　エラーの意味情報の抽出

動作コマンドの類型に応じて，格文法に基づく意味ネットワークである Conceptual Graph(第 3 章) のフレームが呼び出される．リカバリー行動計画システムは，エラーの情報を用いて各ノードの属性値を例示化 (instantiation) することでエラー要因の候補の属性を抽出する．これによって，検知されたエラーが正常系の行動計画に与える影響と，復旧するために解消しなければならない問題が明らかになる．

2.4.3　エラー要因の評価

センシングの不確実性により，行動計画システムが意味ネットワークによって解釈したエラー状態と，実際に生じているエラー状態は必ずしも一致しない．復旧のための行動計画の修正戦略を決定する上でエラーの確信度を得るため，抽出されたエラー要因の候補と，そのエビデンスとなる情報の関係を表すベイジアンネットワーク (第 4 章) を用いて，与えられた事前確率と条件付き確率の表から事後確率を計算する．

2.4.4　修正戦略の決定

エラー要因の属性ごとにリカバリーの目標状態を定める汎用的な修正戦略のうち，事後確率から採用すべきと判断されたエラー要因を解消する戦略を，リスクと時間遅れのある意思決定問題に要求される生産現場の状況を反映した時間依存効用関数 (第 5 章) に基づいて決定する．修正戦略が示すリカバリーの目標状態は，半順序プランナへの入力となる To-do リストに挿入される．現時点の情報では不確実性が許容されず実行すべき修正戦略が決定できない場合には，エラー要因の候補に関する新たな情報を得るための探索動作を実行するこ

とをスケジューラに要請する．

2.4.5　半順序プランの修正

プランナは，To-do リストの目標状態が達成されるように，元の半順序プランの未実行部分を修正し，スケジューラに送信する．リカバリーのために追加されるプランステップの影響を受けない部分の半順序構造は維持される．

2.4.6　実行スケジュールの修正

スケジューラは，プランナから受信した半順序プランの順序制約のもとで，リカバリーに必要な動作を組み込んで，作業再開時点から正常系の作業完了までの時間が最小となる新たな実行スケジュールを生成し，コントローラに送信する．

2.4.7　探索動作継続の判断

実行すべき修正戦略が決定できず，なおかつこれ以上の新たな情報を得る探索動作が指定できない場合には，すべてのロボットの動作の実行を中断する．提案するリカバリーシステムは，プランナに定義されている行動の組合せによりエラー要因を解消できる範囲で活用されることを想定しているため，ハードウェアの故障や機構的なロック状態に陥った場合には動作の実行を停止し，人の介入を待つことになる．

2.4.8　追加エビデンスによる再評価

新たな情報を取り入れてエラー要因を吟味する余地があるとして，探索動作が指定された場合には，リカバリー行動計画の作成前にスケジューラを介して，その動作を実行する．エビデンスとしてのセンサ情報が追加されることによって，エラー要因の候補の属性を表すベイジアンネットワークの各ノードの事後確率の値が更新される．

2.5　結言

本章では，行動計画レベルの修正を経て目標状態を達成するエラーリカバリーシステムを構築した．正常系における目標状態までの順方向のプランニングの観点では，半順序プランニングと最適スケジューリングによって，人手による行動計画から各ロボットの実行スケジュールの生成までのプロセスは不要となる．実行スケジュールに含まれる任意の動作でエラーが検知されたとき，提案システムはエラーの意味情報に基づいて動作失敗の背後にあるエラー要因を抽出・評価し，適切な修正戦略を決定する．修正戦略が決定されると，半順序プランの修正によってリカバリーのために必要なプランステップとその因果関係が明らかになり，それらを組み込んで再度最適化された実行スケジュールが生成される．この新たな実行スケジュールに従って，各ロボットは作業を再開できる．また，エラー要因の候補の評価値から修正戦略が決定できない場合には，探索動作の継続可否の判断を行い，ロボットが自律的にリカバリー可能な範囲を見定めることができる．

次章では，エラーリカバリーのための行動計画という逆方向のプランニング問題において，作業全体の半順序性を活用した修正を行うための修正戦略を構築する．この修正戦略を作業対象によらない汎用的なものとするために，上位概念化されたロボットの動作の類型とそれに伴うエラーの類型を定義することを考えていく．

第 3 章

ロボットの汎用動作における失敗の意味情報に基づく修正戦略の構築

3.1 緒言

本章では，産業用ロボットの汎用動作の失敗に関する知識表現と，作業対象によらず適用可能な修正戦略の体系化を目指す．

第 1 章で述べたように，リカバリーを達成するために必要な行動を推論する上で，エラー停止した作業状態におけるどこまでの要素を考慮すればよいかというフレーム問題が立ち上がってくる．そこで，失敗したロボットの動作が正常系の計画に与える影響を、抽象的な行動のレベルで記述する格フレームを導入し，それによって抽出されるエラー要因の類型に対して修正戦略を構築する．

3.2 関連研究

ロボットの行動に関するフレームを定義する上では，そのフレームで表現できる述語概念の網羅性が問題となる．Schank が 1970 年代に提唱した概念依存理論 (conceptual dependency theory) では，数ある自然言語に共通な述語概念が 11 個の類型で規定されている [46]．Bækgaard らは，述語にかかるインタラクション要素の意味的役割 (semantic role) の定義にしたがった類型化を行っている [47]．これらの研究ではいずれも，述語概念は，自然言語においてその述語に係る格の構造に着目することで，少数の意味素 (semantic primitive) の集合に閉じて定義できることが示されている．この知見は近年でも，自然言語処理分野において画像や音声といった膨大なデータを効率よく体系的に処理するために活用されている [48]．環境とのインタラクションを伴うロボットの行動計画で意味論的知識表現を活用した研究も存在する [49][50].

本研究では，ロボットの動作を表現する上で必要となる格の関係を，上位概念化された行動に対して構築することで，その類型の動作が現在の作業環境において失敗した要因すなわち状態遷移が正しく行われなかった要因を含む範囲を切り出し，フレーム問題を解消することを考える．

3.3 Conceptual Graph による状態の表現

本研究では，ロボットの行動による作業状態を表現するフレームワークとして，格文法に基づいて構成される意味ネットワークである Conceptual Graph (CG) を適用する [51][52]．格文法は，ある文章が示す概念を，

中心となる動詞すなわち述語概念と格関係で結び付けられる名詞の組合せとして解釈する．格関係は日本語の助詞に相当するもので，主格 (誰が)，対象格 (何を)，場所格 (どこで)，手段格 (何を用いて)，といった種類がある．これらを組み合わせることで，自然言語のどんな文章でも述語概念を中心とするグラフ表現を可能にしたものが CG である．CG は，具体的な概念を表す概念ノード (concept node) と，概念間の格関係を表す関係ノード (relation node) で構成される．

簡単な例として，“A cat catches a mouse.” という文章に対応する CG を Fig. 3.1 に示す．この CG は catch という述語に対して，主語となる cat が主格を表す関係 “Agnt” で，目的語を表す mouse が対象格を表す関係 “Ptnt” で結びついている．そして，cat が mouse を catch した結果として cat が mouse を持っていることが所有格を表す関係 “Poss” で示されている．このように CG は動作の前提条件と事後条件を記述するのに適したネットワークであり，計算機上で時間的・空間的な状態の拡がりを推論することが可能である [53].

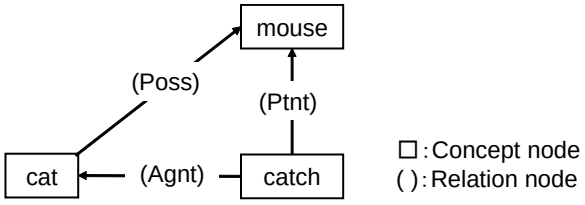


Fig. 3.1: Example of CG that represents a sentence of natural language.

ロボットの汎用動作を表現するために提案する CG のフレームを Fig. 3.2 に示す．Fig. 3.2 のそれぞれの概念ノードに照合される属性値の説明を Table 3.1 に，関係ノードで接続された概念ノードが表す格関係の説明を Table 2 3.2 に示す．

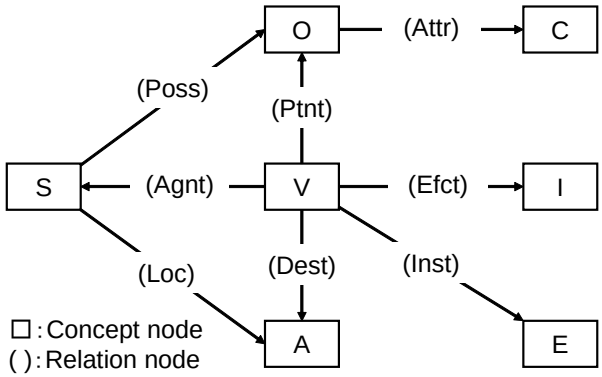


Fig. 3.2: Proposed CG framework to describe general robotic actions.

これらの格の関係は，作業状態を遷移させる行動の意味を表現するために必要な関係である．ロボットがある行動に失敗したとき，Fig. 3.2 の一部の格を含むフレームが，その述語が環境に対して与える意味的役割のフレームとして呼び出される．そのフレーム中の概念ノードにロボットコマンドの情報とセンサ情報を参照して決まる値が代入されて，1 つの作業状態を表す CG が例示化 (instantiation) される．この CG の格関係で示される情報が，修正戦略を導く上で重要となる失敗の意味情報 (semantic information) である．

ロボットの作業中の CG の例示化の一例を Fig. 3.3 に示す．図中の 2 台のロボットアームが実行している組立作業の詳細設定については第 6 章で説明する．今，“robot#1” と呼ばれる図の手前側にいるロボットが “gearA” というワークを把持した状態で，所定の組み付け位置である “*pt*” という場所に移動する動作 “move”

Table 3.1: Instantiated attributes on concept nodes.

Node	Explanation
S	An operating robot
V	An action
O	An operated workpiece
C	Condition of the workpiece
A	A spot where the robot exists
E	Equipment
I	Another interfering robot or workpiece

Table 3.2: Case grammar described by relation nodes.

Relation	Explanation
V-Agnt→S	V is executed by S
V-Ptnt→O	V operates O
V-Dest→A	V reaches A
S-Poss→O	S possesses O
S-Loc→A	S exists at A
O-Attr→C	O is under C
V-Inst→E	V is executed with E
V-Efct→I	V affects I

を実行しようとして，手先の力覚センサで異常値を検出し停止したという状況を想定する．このとき，robot#1 を pt に移動させる move という動作の格フレームを用いて，「robot#1 は gearA を把持している (Poss)」「ワークの把持状態が異常である (Attr)」「他のロボットやワークとの干渉はない (Efct)」という情報から，例示化が完了する．これより，「robot#1 が正しい状態で gearA を把持して干渉なく pt に到達する」という move が正常系において果たすべき状態遷移に対して，「gearA の状態が正しくないために正常に状態遷移できなかった」というエラー状態が表される．

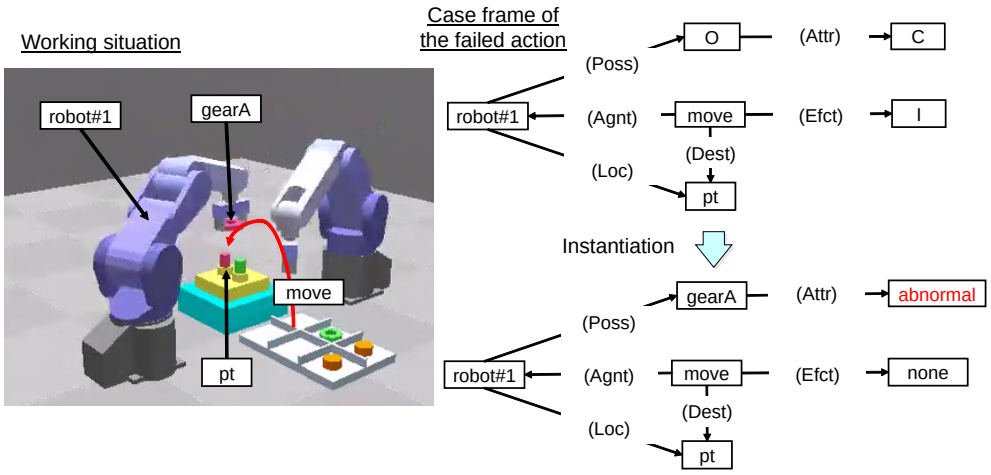


Fig. 3.3: Example of the instantiation of CG against the failed action.

3.4　産業用ロボットの汎用動作の抽出

本節では，Fig. 3.2 のフレームで表現可能な動作の類型の網羅性について説明する．産業用ロボットの動作コマンドは，メーカごと・機種ごとに専用のプログラミング言語で定義されているが，状態を遷移させる行動のレベルで捉えたとき，物体操作に不可欠ないくつかの汎用動作が存在する．それらの動作は CG で記述する際に必要な格の構造の違いにより，次の 3 種類の行動の型に集約される．Fig. 3.2 の CG がこれらの汎用動作に伴うエラーを表現できるように構成されていることにより，提案システムにおいてエラーに関わりのある事柄の抽出すなわちフレーム問題の克服が可能となる．

1 つ目は，ロボット自身が移動する “Move” 型の動作である．2 つ目は，ロボットが物体をつかむ “Grasp” 型の動作である．3 つ目は，ロボットが組立等のために物体に力を加える “Apply” 型の動作である．先に述べた Schank の概念依存理論の類型にもこの 3 つが入っている (ただし，Apply は Propel という名称である)．第 6 章のシミュレーションで用いる三菱電機 (株) 製産業用ロボット MELFA であれば，専用プログラミング言語 MELFA BASIC において，Move 型のコマンドとして指定した教示点への関節補間動作 Mov や直線補間動作 Mvs，Grasp 型のコマンドとして Hand Open/Close，Apply 型のコマンドとして力覚制御を用いた押し付け動作等が存在する．ロボットがフィーダへ移動し，ワークを把持し，製品を組み立てるステージへ移動し，所定の位置にワークを挿入する，といった一連の組立作業は，Move 型，Grasp 型，Apply 型の動作の組合せで遂行される．これらが成功裏に遂行されるための要件を規定する CG のフレームについて次に説明する．

3.4.1　Move 型の動作

Move 型の動作を規定する CG のフレームを Fig. 3.4 に示す．Move を記述するためには移動先を表す場所格 Dest が必要となる．Move が適用される前提条件は移動先の場所が他のロボットに占有されていないことであり，事後条件は主体のロボットが移動先の場所を占有することである．

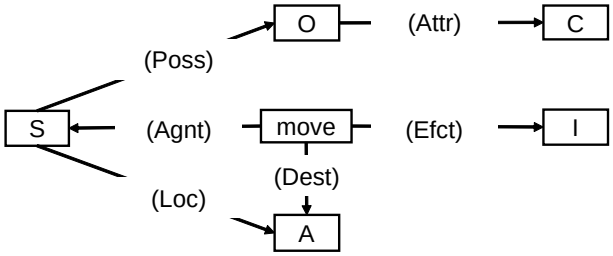


Fig. 3.4: CG frame to describe error factors during “Move” operation.

3.4.2　Grasp 型の動作

Grasp 型の動作を規定する CG のフレームを Fig. 3.5 に示す．Grasp を記述するためには把持対象のワークを表す対象格 Ptnt が必要となる．Grasp が適用される条件は主体のロボットが何も所有せず対象ワークの置かれている場所にいる事であり，事後条件は主体のロボットが対象ワークを所有することである．

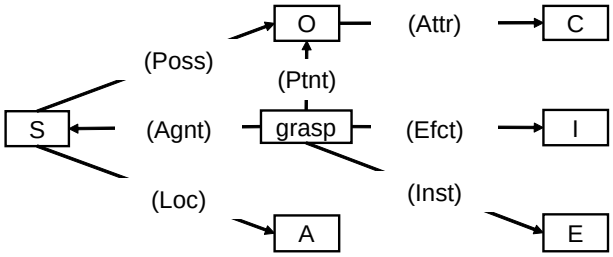


Fig. 3.5: CG frame to describe error factors during “Grasp” operation.

3.4.3　Apply 型の動作

Apply 型の動作を規定する CG のフレームを Fig. 3.6 に示す．Apply が適用される前提条件は主体のロボットが対象ワークを所有して，そのワークに力を加えて放すべき場所にいることであり，事後条件はロボットがワークを手放している事である．機械部品のはめ込みやネジ締め動作は Apply 型に分類される．

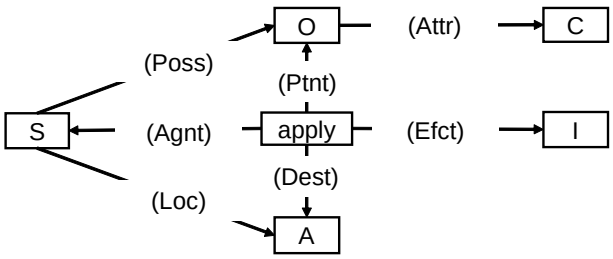


Fig. 3.6: CG frame to describe error factors during “Apply” operation.

3.5　汎用動作の実行に伴うエラー状態の類型化

提案システムでは，エラー検知時に実行しようとしていた Move，Grasp，Apply 型のいずれかの行動について，前節で示した CG のフレームが呼び出され，動作プログラムの情報とセンサ情報から概念ノードの値が例示化されることでエラー状態が表現される．それぞれの類型の CG で区別可能な，エラー要因を示す格関係とそのシチュエーションについて次に説明する．

3.5.1　Move 実行中のエラー要因

Move 型に分類される動作の実行中に生じ得るエラー要因を Table 3.3 に示す．動作プログラムの情報から，動作主体を表す Agnt と，移動先を表す Dest の格関係は確定的である．また，Move が成功すればロボットがその移動先にいるという Loc の格関係が成立していないことも確定的である．Move 型の Fig. 3.4 のフレームに含まれる残りの格関係が不確定的なエラー要因の候補となる．

Table 3.3: Distinguishable error factors of “Move” by CG and corresponding situations.

Factor	Situation of error detection
Poss	Missing workpiece
Attr	Abnormal condition of a workpiece
Efct	Interference of another workpiece or robot

Move 実行中に Poss の格関係がエラー要因となる場合は，運搬するワークを落とし把持力が消失した状況を表す．Attr の格関係がエラー要因となる場合は，ワークの状態が不適切で運搬中に異常な力がかかった状況を表す．Efct の格関係がエラー要因となる場合は，何らかの障害物に干渉したことで目標の移動地点に到達できずにいる状況を表す．

3.5.2　Grasp 実行中のエラー要因

Grasp 型に分類される動作の実行中に生じ得るエラー要因を Table 3.4 に示す．動作プログラムの情報から，動作主体を表す Agnt と，対象ワークを表す Ptnt の格関係は確定的である．また，Grasp が成功すればロボットがそのワークを所有しているという Poss の格関係が成立していないことも確定的である．Grasp 型の Fig. 3.5 のフレームに含まれる残りの格関係が不確定的なエラー要因の候補となる．

Table 3.4: Distinguishable error factors of “Grasp” by CG and corresponding situations.

Factor	Situation of error detection
Loc	Abnormal position of a robot
Attr	Abnormal condition of a workpiece
Inst	Abnormal setting of equipment
Efct	Interference of another workpiece or robot

Grasp 実行中に Loc の格関係がエラー要因となる場合は，ロボットの位置が想定されるワークの座標に合っ

ていない状況を表す．Attr の格関係がエラー要因となる場合は，ワークの状態が不適切で把持した際に異常な力がかかった状況を表す．Inst の格関係がエラー要因となる場合は，ツールの設定が不適切で所定の手先位置にいるにもかかわらずワークに到達できずにいる状況を表す．Efct の格関係がエラー要因となる場合は，何らかの障害物に干渉したことでワークに到達できずにいる状況を表す．

3.5.3　Apply 実行中のエラー要因

Apply 型に分類される動作の実行中に生じ得るエラー要因を Table 3.5 に示す．動作プログラムの情報から，動作主体を表す Agnt と，対象ワークを表す Ptnt，ワークを放す場所を表す Dest の格関係は確定的である．Apply 型の Fig. 3.6 のフレームに含まれる残りの格関係が不確定的なエラー要因の候補となる．

Table 3.5: Distinguishable error factors of “Apply” by CG and corresponding situations.

Factor	Situation of error detection
Poss	Missing workpiece
Loc	Abnormal position of a robot
Attr	Abnormal condition of a workpiece
Efct	Interference of another workpiece or robot

Apply 実行中に Poss の格関係がエラー要因となる場合は，ワークを把持できていない状況を表す．Loc の格関係がエラー要因となる場合は，ロボットの位置が想定される取り付け位置に合っていない状況を表す．Attr の格関係がエラー要因となる場合は，ワークの状態が不適切で取り付ける際に異常な力がかかった状況を表す．Efct の格関係がエラー要因となる場合は，何らかの障害物に干渉したことでワークを取り付けられずにいる状況を表す．

3.6　エラー要因の格に基づく修正戦略の構築

前節において，いずれの行動の型でも，その行動の記述に含まれる格関係によってエラー要因を類型化することができた．行動計画のレベルでエラー要因を抽出したことのメリットとして，エラーの検知手段やコマンドレベルでのリカバリー方法は機種に依存するとしても，たとえば障害物に干渉しているのであればその障害物を取り除くことで Efct の格関係を修復する必要があるというように，汎用性のある推論が可能になる．エラー要因と一対一対応で修復すべき格関係を表した修正戦略を Table 3.6 に示す．

Table 3.6: Repair strategies against error factors.

Error factor	Repair strategy
Missing workpiece	“Poss” repair by regrasping the workpiece
Abnormal position of a robot	“Loc” repair by moving the robot
Abnormal condition of a workpiece	“Attr” repair by resetting the workpiece
Abnormal setting of equipment	“Inst” repair by resetting the equipment
Interference of another robot	“Efct” repair by moving the robot
Interference of another workpiece	“Efct” repair by removing the workpiece

例示化された CG が示すエラー要因に対して，該当する修正戦略を適用することで，リカバリーの To-do リストとしてプランナに送信する副目標が定まる．半順序プランナは，元の正常系の半順序構造を活用しつつ，与えられた副目標を達成するための新たなプランステップを導出する．したがって，CG が真のエラー要因を示していれば，必要最小限の修正でリカバリー可能な行動計画を得ることができる．

3.7　結言

本章では，ロボットの汎用動作を表すために格文法に基づく意味ネットワークである Conceptual Graph を導入した．深層格の構造に着目して，物体操作を実行するロボットの動作が Move，Grasp，Apply の 3 つの型に集約できることを示した．3 種類の動作の類型のそれぞれに対して，行動計画の上で適用されるときの前提条件と事後条件が存在することを踏まえて，エラー要因の候補となる格関係を整理した．それによって作業対象によらない修正戦略を構築することができた．

本章で体系化された修正戦略は，CG が実環境での真のエラー要因を示している場合に効果を発揮する．一方で，実際のエラーの要因にそぐわない修正戦略が適用されてしまった場合には，ロボットはエラーリカバリーを達成できないのみならず、誤ったリカバリー行動を実行することによって新たなエラーを引き起こす可能性もある．次章では，不確実性を伴うエラーの要因の候補に対して，リカバリー行動計画の生成と実行を判断するために，例示化された CG の属性値の尤もらしさを評価する方法を考える．

第 4 章

エラー状態に関する不確実性を考慮した修正戦略の選択

4.1 緒言

本章では，ロボットが作業する実環境でのセンサ値によるエラー検知からエラー要因の同定に至る際の不確実性を考慮して，適切な修正戦略を選択するための評価方法を構築する．

前章で体系化されたエラー要因の格関係に基づく修正戦略は，発生したエラー状態が CG によって正しく記述されている場合に機能するものである．しかしながら，実際のロボットシステムでエラー検知に用いられるセンサ値から CG の概念ノードに代入される属性値に変換するとき，正常と異常の境界となる閾値に対するセンサ値のばらつき等によって，例示化された CG が実態にそぐわなくなっている可能性が存在する．こうしたセンサ値のばらつきの原因としては，センサ自体の性能により混入する機械的・電気的ノイズや，検知対象のロボットとワークの位置の繰り返し誤差があり，完全に排除することはできない．したがって，リカバリー行動計画のための副目標の設定に際しては，常に不確実性の下での判断を行うことになる．

提案システムでは，前章で導入した汎用性のある CG のフレームワークと，FA ロボットシステムに依存するセンシング手段すなわちエラー要因のエビデンスの入手手段を組み合わせて，動的に構成されるベイジアンモデルに基づきエラー要因の候補を評価する．

4.2 関連研究

センサによる状態観測の不確実性を考慮した自律ロボットの設計にベイズ理論を適用した研究は数多く存在する [54][55]．産業用ロボットの制御に関するものでは，視覚情報が得られない環境での触覚フィードバックに基づくモーションプランニング [56] や，エンドエフェクタによる把持可能性の評価 [57] にベイズ理論を用いた研究が存在する．人と機械の協調作業に関する研究では，人の行動により不確実性をもって遷移していく状態を評価するために，動的なベイジアンネットワークを適用した例も存在する [58][59]．本研究では，ベイジアンネットワークを単なる事後確率計算のための手段として使うのみならず，CG によって抽出される汎用的な意味情報と実際のロボットシステムが備えるエビデンス観測手段とを結びつけることで，システム依存の適切な評価モデルを動的に構築することを考える．

4.3　ベイジアンネットワークにおける事後確率の計算

ベイジアンネットワーク (Bayesian network, 以下 BN) は，複数の状態変数間の確率的な因果関係を有向グラフ構造で表したものである．BN に含まれるある変数の状態が観測されたとき，ほかの変数の状態を条件付確率として推論することができる [60][61]．提案システムにおいて明らかにしたい，あるエラー要因が発生するという状態変数 X_i と，そのエラー要因に関する異常値をセンサが検知するという状態変数 Y_j からなる BN の例を Fig. 4.1 に示す．

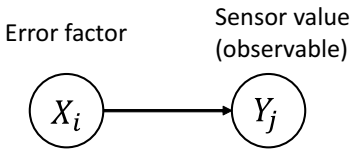


Fig. 4.1: Example of Bayesian network to represent the relationship between an error factor and a sensor value.

事前確率 $P(X_i)$ の大きさは，何の情報もない段階で X_i であることをエージェントが信じる信念 (belief) の強さを表している．実際のロボットシステムで観測できるのは Y_j であり，これをエビデンスとして事後確率 $P(X_i|Y_j)$ が求められたとき，新たな情報によって信念が更新されたという．ベイズの定理より，事後確率 $P(X_i|Y_j)$ は次式で求められる．

$$P(X_i|Y_j) = \frac{P(Y_j|X_i)P(X_i)}{P(Y_j)} = \frac{P(Y_j|X_i)P(X_i)}{\sum_{X_i} P(Y_j|X_i)P(X_i)}. \tag{4.1}$$

(4.1) 式の計算に必要な事前確率 $P(X_i)$ および条件付確率 $P(Y_j|X_i)$ はロボットシステムごとに与えられるものとする．具体的には，現場においてシステムの立上げ・試運転の段階でエラー要因 X_i の発生頻度から $P(X_i)$ が設定され，そのエラー要因に対するセンサの検知性能から $P(Y_j|X_i)$ が設定される．これらの設定値は生産開始後でも変更可能である．たとえば，数日から数週間スパンでのエラー発生頻度をデータベースに記録しておき，毎朝に設定した値でその日の自動生産を実施することが考えられる．

循環のない 3 つ以上のノードを持つ BN では，信念の伝播 (belief propagation) によって因果関係の上流と下流の双方向から推論が可能である．信念の伝播を表す BN の例を Fig. 4.1 に示す．

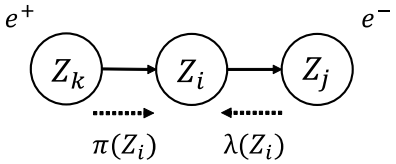


Fig. 4.2: Belief propagation of Bayesian network.

状態変数 Z_i が下流側の Z_j で観測されるエビデンス e^- に与える影響を表したメッセージ $\lambda(Z_i)$ は次式で求められる．

$$\lambda(Z_i) = P(e^-|Z_i) = \sum_{Z_j} P(e^-|Z_j)P(Z_j|Z_i). \tag{4.2}$$

一方，上流側の Z_k で観測されるエビデンス e^- が Z_i に与える影響を表したメッセージ $\pi(Z_i)$ は次式で求められる．

$$\pi(Z_i) = P(Z_i|e^+) = \sum_{Z_k} P(Z_i|Z_k)P(Z_k|e^+). \tag{4.3}$$

このとき，Z_i の事後確率はメッセージ $\lambda(Z_i)$ とメッセージ $\pi(Z_i)$ の積を正規化したものとして得られる．Z_i に接続される上流または下流のノードが 2 つ以上の枝を持つ場合でも同様の計算が可能である．

4.4　センサ情報に基づくベイジアンモデルを用いたエラー要因候補の評価

第 3 章で導入した Fig. 3.2 の CG のフレームに対応する BN のフレームを Fig. 4.3 に示す．この BN は，Table 4.1 に挙げるエラー要因となり得る深層格の種類を表す状態変数 $X_i(i = 1, 2, 3, 4, 5)$ と，Table 4.2 に挙げるエラー検知手段となり得るセンサの種類を表す状態変数 $Y_j(j = 1, 2, 3)$ で構成される．

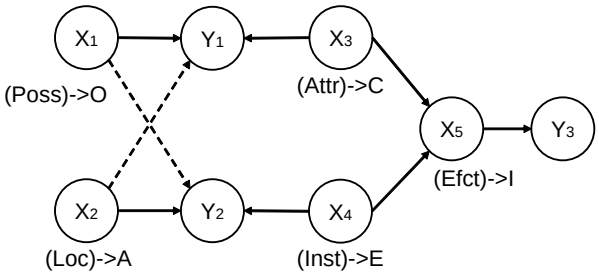


Fig. 4.3: Proposed BN frame of BN to evaluate the attribute values of CG.

Table 4.1: Variables to represent error factors in CG.

Node	Explanation
X_1	Missing workpiece
X_2	Abnormal position of a robot
X_3	Abnormal condition of a workpiece
X_4	Abnormal setting of equipment
X_5	Interference of another workpiece or robot

Table 4.2: Variables to represent available evidence.

Node	Explanation
Y_1	An abnormal value of a force sensor
Y_2	An abnormal value of a motor-encoder feedback
Y_3	An interference detection of a vision sensor

エラー検知手段としては産業用ロボットシステムでよく利用されるセンサを想定する．Y_1 はロボットの手先に備えられる力覚センサで，実際に手先に印加されている力とモーメントの値から，把持しているワークの

有無 (Poss)，ロボットの所定位置からのずれ (Loc)，ワークの状態の異常 (Attr) の検知手段となる．Y_2 はロボットの各関節に内蔵されるモータとエンコーダのフィードバック値で，関節のフィードバック位置とトルクに換算可能なフィードバック電流から，把持しているワークの有無 (Poss)，ロボットの所定位置からのずれ (Loc)，ワークを把持するためのツール設定値のずれ (Inst) の検知手段となる．Y_3 はハンドアイとしてロボットごとに手先に備えられたり，設備全体を俯瞰的に監視するためセルの天井部に設置されたりするビジョンセンサで，物体の位置関係の画像情報からロボットとワークの干渉 (Efct) を検知する手段となる．

Fig. 4.3 中の X_1 と X_2 における破線矢印は，信念の伝播を扱う BN が開ループ構造となるように，ロボットシステムが備えるセンサに合わせてエビデンスの入手手段を選択できることを示している．力覚センサとビジョンセンサの 2 つをエラー検知に用いるシステムの場合，前章で整理した Move 型，Grasp 型，Apply 型のエラー要因の候補を評価する BN は，それぞれ Fig. 4.4，Fig. 4.5，Fig. 4.6 のように構築される．これらを用いて，汎用性のある CG のエラー要因の記述と実際のロボットシステムで収集可能な情報とを関連付けて不確実性を評価することが可能である．

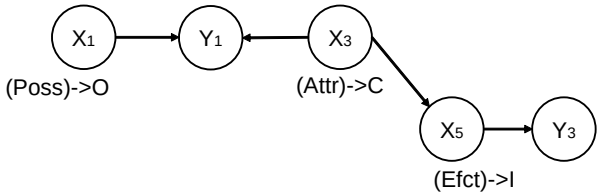


Fig. 4.4: BN to evaluate error factors of “Move” by using force sensor and vision sensor.

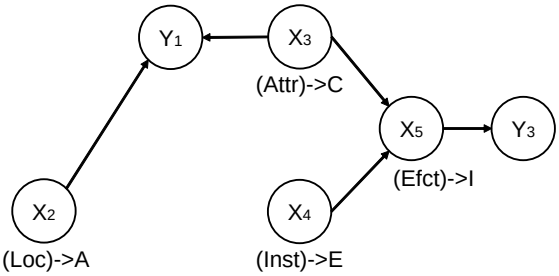


Fig. 4.5: BN to evaluate error factors of “Grasp” by using force sensor and vision sensor.

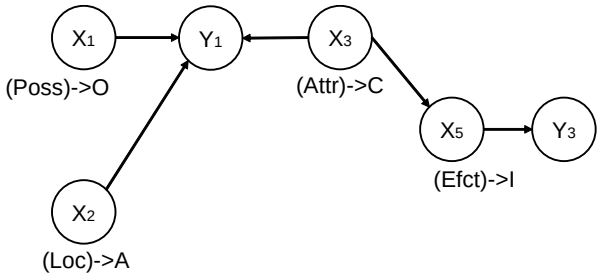


Fig. 4.6: BN to evaluate error factors of “Apply” by using force sensor and vision sensor.

一例として，第 3 章の CG の例示化の説明で取り上げたエラー状態において構築される BN を Fig. 4.7 に示す．これは，失敗した “move” という動作について，CG の例示化の際にセンサ情報を参照して決定した (すなわち不確実性を含む) 格の関係と，実際のロボットシステムが備えるセンサを紐づける BN である．この場合，ロボットの手先の力覚センサで「robot#1 は gearA を把持している (Poss)」「ワークの把持状態が異常である (Attr)」ことを検出しており，さらにシステムが備えるビジョンセンサを参照することで「他のロボットやワークとの干渉はない (Efct)」ことに対する追加のエビデンスを得ることができる．このように，入手可能な情報資源の下で抽出されたエラー要因の尤もらしさを評価するネットワークが構築される．

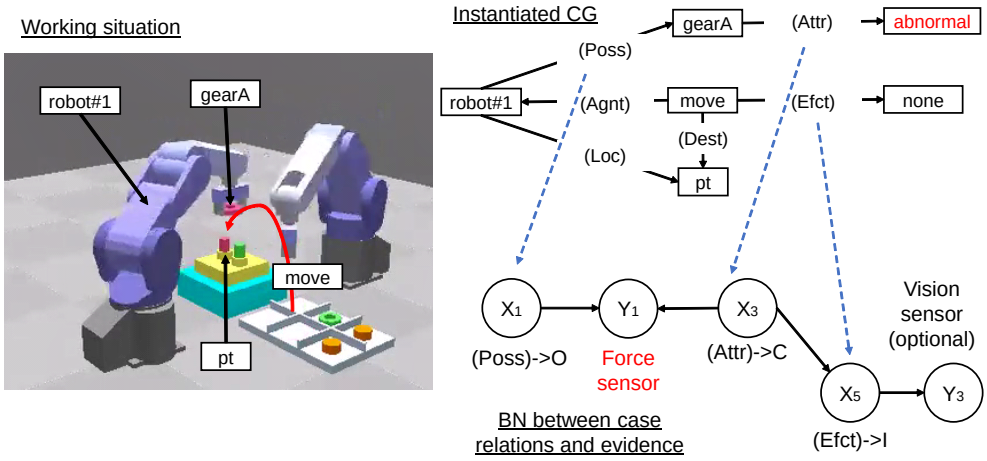


Fig. 4.7: Example of BN between case relations in CG and evidence.

4.5 結言

本章では，不確実性のあるエラー要因の候補と，そのエビデンスとなるセンサ情報の関係を表すベイジアンネットワークを構成して，実際にそのエラーが生じていると判断する信念としての事後確率を求める方法を体系化した．提案する BN のフレームはエラー要因の深層格を抽出する CG と対応付けられており，汎用性のあるエラー要因の記述とロボットシステム依存のセンシング手段を組み合わせて，修正戦略を選択するために必要な因果関係を含む BN を動的に構築することができる．

得られた事後確率の使い方として，最も確率の大きいエラー要因の候補に対する修正戦略を採用することは 1 つの意思決定の方法である．一方で，より知的な意思決定の基準を定めるメタレベル意思決定問題としては，新たなエビデンス収集にかかるコストに対する情報価値を考慮して，判断を下すまでにどれだけの時間を費やし，どこまでの不確実性を許容するかを，生産現場の状況に即して制御することが要求される．次章では，時間依存効用理論に基づく意思決定システムの制御方法について考える．

第 5 章

生産現場の時間依存効用に応じた行動計画システムの制御

5.1 緒言

本章では，エラー状態の不確実性と実行完了までの時間コストを伴うリカバリー行動計画に対して，変動的な時間圧の下での意思決定を迫られる生産現場での効用の観点から，どこまでの熟考を許容するモデルを用いて意思決定を行うべきかというメタレベル意思決定システムの制御方法について論じる．これは，環境の不確実性を低減するための情報がノーコストで即座に得られるわけではなく，収集可能な情報量やその処理能力，時間資源に限界があるという FA の制約下で，ロボットの自律的行動の合理性を担保するために不可欠な要素である．

産業用ロボットを含む FA システムには，その日の生産計画を達成するためにプラントの上流で定められた目標稼働率が存在する．チョコ停の期間が長引くにつれラインやセルの稼働率が下がっていき，やがて生産計画が破綻することになる．エラーリカバリー行動計画は，要求されるデッドラインまでに実行完了できる場合に価値があるものといえる．対象作業の投入資源に対して所定の生産性を発揮するために今，どこまでリカバリーの実行を先送りにすることが許され，追加コストを支払ってでも不確実性を低減するためのセンシング行動をとるべきか否かは，エラー発生時にシステムがどれだけタイトな生産状況に直面しているかに関わってくる．したがって，エラー要因の候補の不確実性を評価して修正戦略を選択する提案システムは，確率的な評価値に対する意思決定モデルを現場の時間圧に応じて切り替えて運用されるべきであると考える．このことから，リスクだけでなく利得を得るまでの時間の要素を考慮した時間依存効用 (time-dependent utility) の理論を導入する．

5.2 関連研究

知能システムに要請される真に合理的な意思決定能力とは，熟考に計算資源を費やすことによる解の質の向上と実行の機会損失コストのトレードオフを扱う，有限な資源管理 (resource-management) を含めた意思決定能力であるとされる [62]．Breese らは，3 つの意思決定モデル：目標状態に到達可能な選択肢を場当たり的に選ぶ Feasible planning method (方策 F)，コストの期待値に関するモデルを使って最適な選択肢を判断する Basic probabilistic model (方策 B)，不確実性低減のための追加の情報収集の要否まで含めた判断を行う Information probabilistic model (方策 I) を定義した [63]．これらのうち最適な方策の遷移を示した図を模写し

たものが Fig. 5.1 である．横軸は右にいくほど熟考を打ち切るまでの時間の猶予が大きい場合，縦軸は上にいくほど実行した行動の失敗時の損失が大きい場合である．意思決定を行う環境のリスクが小さく熟考に許される時間が短い場合には方策 F が優位であるが，リスクが大きくなるにつれて，あるいは許容時間が増加するにつれて方策 B，さらには方策 I が優位になることが示されている．

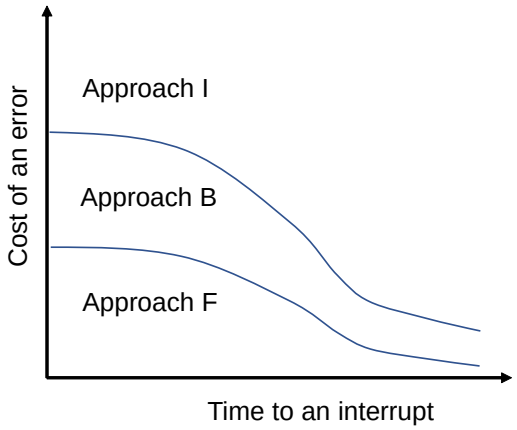


Fig. 5.1: Change of optimal reasoning policies [63].

時間依存効用理論は，医療現場のように意思決定を先延ばしにすることが最終的に得られる利得に影響を及ぼす問題を定式化するために研究されてきた [64][65]．Baucells らは，リスクと時間遅れのあるくじ (lottery) に対する心理的距離という概念を導入し，時間 t 後に確率 p で利得 x が得られるくじ」が「今すぐ確率 $p * exp[-r(x)]$ で利得 x が得られるくじ」と等価であるしている [66]．Epper らは，時間遅れに対する純粋な時間選好と，時間遅れによって発生する潜在的なリスクに対するリスク選好の二つで将来利得を割り引くモデルを提案している [67]．本研究では，FA において資源制約が存在するために，こうしたリスクと時間遅れを考慮した意思決定を行わなければならないことに着目し，現場にとって最適な意思決定モデルを導出することを考える．

5.3　期待効用に基づく意思決定

提案するエラーリカバリーシステムでは，コストを時間の単位で考え，与えられたデッドラインからリカバリー完了までに要した時間コストを差し引いた残り時間が利得であるとする．エラーリカバリーに関するリスク選好は以下のように考えられる．

第 3 章から第 4 章にかけて議論してきた通り，発生したエラー要因に適合する修正戦略が選択されたとき，システムは最小コストでリカバリー可能な行動計画を実行できる．一方で，作業環境で得られる情報資源に限りがあるため，CG と BN によって評価された第 1 候補以外のエラー要因が実際には発生している可能性も存在する．第 1 候補以外のエラー要因のためにリカバリー行動計画が最後まで実行できなくなれば，その時点から新たにリカバリー行動計画を立て直すためにより大きなコストを被ることになる．代替案としては，第 2 候補以下のエラー要因に対する修正戦略も組み込んだ行動計画を求めることが考えられる．この場合，第 1 候補のエラー要因だけで解決できる状態であったならば冗長な行動を取ることになり利得は小さくなるが，第 2 候補以下のエラー要因が付随する状態であったとしてもリカバリーを完了できる．

それでは，第 2 候補以下の事後確率がどの程度であれば考慮に入れるべきか，という問いがリスク選好であ

る．たとえば，事後確率が 0.1 以下なら第 2 要因を無視してよいと考える人もいれば，0.05 以下でも無視すべきでないと考える人もいるかもしれない．これがリスクに対する態度の違いであり，相対的に前者はリスク志向的 (risk seeking)，後者はリスク回避的 (risk averse) と呼ばれる．同定方法は後述するが，確率的に得られる利得に対するその人の満足度を表す効用関数の形状が下に凸であるほどリスク志向的，上に凸であるほどリスク回避的になる．例として，利得 x に対する効用関数を $U(x) = C \cdot x^D$ で表現するとき，2 つのパラメータ C と D によるリスク選好を Fig. 5.2 に示す．行動経済学では，人は利得の期待値ではなく期待効用にしたがって意思決定を行うものとされる．

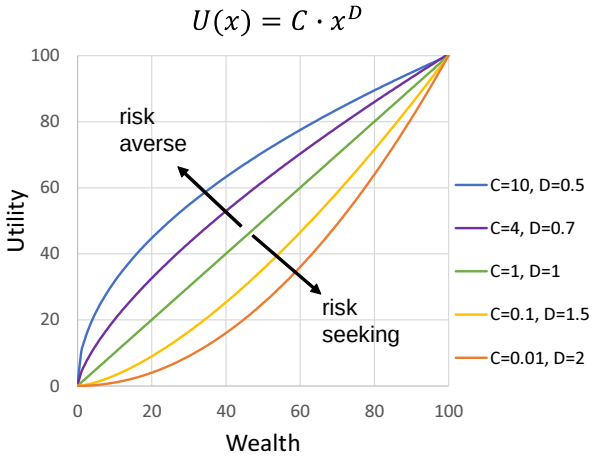


Fig. 5.2: Attitudes toward risk and shapes of the utility function.

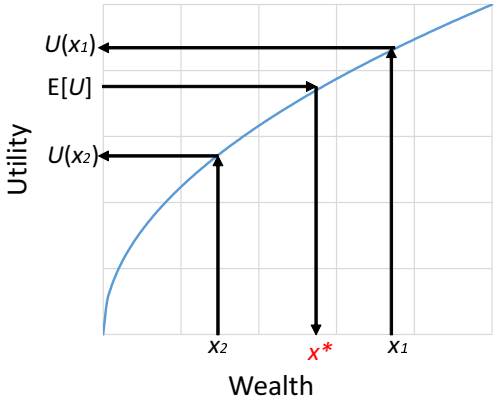


Fig. 5.3: Example of expected utility and the certainty equivalent in a risk averse utility function.

ある効用関数の下での期待効用と確実性等価 (certainty equivalent) の概念図を Fig. 5.3 に示す．確率 α_1 で利得 x_1，確率 α_2 で利得 x_2 が得られるときの期待効用は次式で求められる．

$$E[U(x)] = \alpha_1 \cdot U(x_1) + \alpha_2 \cdot U(x_2). \tag{5.1}$$

このとき，期待効用と同じ効用水準を与える確実な利得，すなわち意思決定者がこのくじのために支払ってもよいと思う価格を表す確実性等価 x^* は次式で求められる．

$$x^* = \arg(E[U(x)]). \tag{5.2}$$

提案システムが扱うエラーリカバリーにおいては，デッドラインすなわち利得ゼロに至るまでの効用の減り方に現場のリスク選好が反映される．第 1 候補のエラー要因のみを考慮した行動計画に対して上式で求められる確実性等価 x^* が，第 2 候補以下のエラー要因まで考慮することで確実に得られる見込みのある利得より大きければ，第 1 要因のみを考慮して計画を立てることが意思決定者にとって合理的となる．さらに，重要顧客の納期や品種切替等によって一日の中でも変動する時間圧の大小に依存して，現場で受け入れられるリスクに対する態度も変わってくることが考えられる．こうした時間選好を含む効用関数の決め方について次節で説明する．

5.4　時間依存効用関数の同定

Baucells らは，時間 t 後に確率 p で利得 x が得られるくじの距離を $\tau = -ln[p] + r(x) \cdot t$ と定義し，決定者にとっての価値を $V(x,p,t) = exp[-d(\tau)] \cdot U(x)$ で表現した [66]．ここで $r(x)$ は確率割引率，$d(\tau)$ は心理的距離関数と呼ばれる．時間依存効用関数 $U(x)$ の形状を定めるための N 個の内分点は，確実性等価に関する質問を繰り返すことで以下のように求められる．

1. $p_0 = 1$ とする．任意の x_0，p_1 をとり，$(x_0, 1, t_0)$ が $(x_0, p_1, 0)$ と等価になる t_0 を尋ねる．
2. $(x_1, 1, 0)$ が $(x_0, p_1, 0)$ と等価になる x_1，$(x_1, 1, t_1)$ が $(x_1, p_1, 0)$ と等価になる t_1 を尋ねる．
3. $k = 2, 3, \cdots, N$ に対して，$(x_k, 1, 0)$ が $(x_{k-1}, p_1, 0)$ と等価になる x_k，$(x_k, 1, 0)$ が $(x_0, p_k, 0)$ と等価になる p_k，$(x_k, 1, t_k)$ が $(x_k, p_1, 0)$ と等価になる t_k を尋ねる．
4. $U(x_k) = exp*[-k \cdot d(\tau_1)] \cdot U(x_0)$，$r(x_k) = \tau_1 / t_k$，$d(-ln[p_k]) = k \cdot d(\tau_1)$ として，$(x_0, p_0, t_0), (x_1, p_1, t_1), \cdots, (x_N, p_N, t_N)$ をプロットする．ただし，$\tau_1 = -ln[p_1]$ であり，$d(\tau_1) > 0, U(x_0) > 0$ でスケーリングする．

この手順にしたがって求められた (x_k, p_k, t_k) の例を Table 5.1 に示す．$x_1 = 85$ は，この意思決定者にとって確率 0.9 で 100 の利得が得られることと確実に 85 の利得が得られることが等価であることを表し，$t_1 = 16$ は，時間遅れ 16 で確実に 85 の利得が得られることと今すぐ確率 0.9 で 85 の利得が得られることが等価であることを表す．Table 5.1 の数値をプロットすることで得られる効用関数 $U(x)$ を Fig. 5.4，確率時間割引率 $r(x)$ を Fig. 5.5，心理的距離関数 $d(\tau)$ を Fig. 5.6 に示す．ここでは $d(\tau_1) = 1, U(x_0) = 1$ でスケーリングした．この例では $U(x)$ が下に凸の形状を示すことにより，どちらかといえばリスク志向的な態度で意思決定が行われる．リスク回避的な意思決定者に同様の質問をすると，確率 0.9 で 100 の利得が得られることと等価であると考える確実な利得 x_1 はもっと小さい値を示すことになる．

提案システムの運用に際して，エラー要因の不確実性の評価値に対して修正戦略を判断するための時間依存効用関数は，その日ごとの生産現場の状況を考慮してチューニングされるべきものである．現場監督者が上記の同定方法で効用関数の形状を決める (x_k, p_k, t_k) 設定するとき，同じ人でも現場の状況によって異なる回答をすることが考えられる．作業品質のばらつきが大きく予期せぬエラー事象が懸念されるような状況，あるいは自動復旧に失敗した場合にリセットをかけられる人がすぐ駆け付けられないような状況に置かれていると，よりリスク回避的な態度を示すことが予想される．

Table 5.1: Example of identification of risk appetite and time appetite.

k	x_k	p_k	t_k
0	100	1	20
1	85	0.9	16
2	65	0.8	12
3	45	0.7	10
4	20	0.6	8

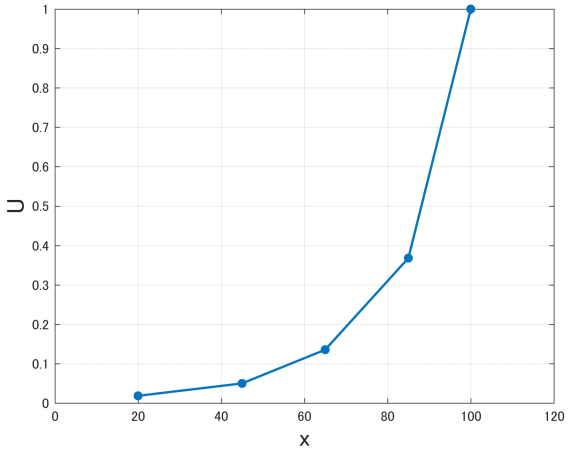


Fig. 5.4: Plot of utility U against the lottery's wealth x.

5.5　生産現場における意思決定モデルの運用

時間依存効用関数を用いた不確実性下での意思決定は，Fig. 5.1 に示した Breese らの最適方策の変遷の中でも熟考度合いの大きい方策 B や方策 I のレベルに位置付けられるものである．本節では，提案するエラーリカバリーシステムがどの程度熟考的なモデルを採用して意思決定を行うべきかというメタレベル意思決定問題について，FA の因子と対応付けて考察する．

生産現場のエラーリカバリーにおいて必要とされる熟考度合いの観点から，Fig. 5.1 に対応する最適方策の変遷を示した図が Fig. 5.7 である．時間的猶予を示す横軸には対象作業のサイクルタイムが対応し，失敗のコストを示す縦軸には製品の価格が対応する．安価な製品を短い周期で大量に生産する現場というのは比較的トライアンドエラーが許される状況であり，熟考により作業再開が遅延することは望ましくなく，エラー要因の第 1 候補に対するリカバリー行動を即決するモデルが採用される．一方，一品物の高価な製品を組み立てているような現場では，リカバリーの行動を実行することに慎重さが求められるため，熟考的な意思決定モデルが採用される．このときの熟考モデルに時間依存効用関数を組み込むことで，作業対象に応じて投入される FA システムの資源と動的な生産状況の時間圧を考慮して，エラー状態の可能性をどこまで深堀するかの合理的判断が実現される．

現場では，Fig. 5.7 を参照して方策 F，方策 B，方策 I のどれを用いるべきかをまず選定し，方策 B または方策 I のモデルを用いる場合には時間依存効用関数を設定する．それぞれの意思決定モデルの下でのリカバ

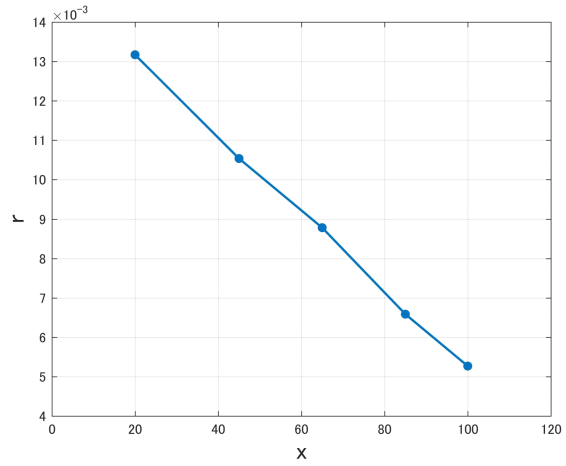


Fig. 5.5: Plot of probability discount rate r against the lottery's wealth x.

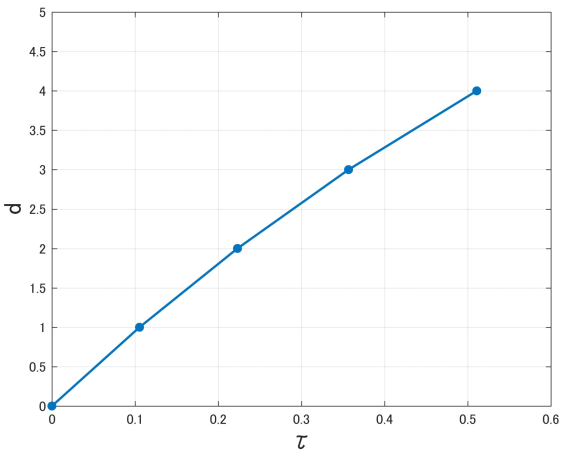


Fig. 5.6: Plot of psychological distance d against the lottery's distance τ.

リー処理と生成される行動計画の差異を Fig. 5.8 に示す．検出されたエラーに対し，CG によるエラー要因の抽出と BN による事後確率の評価が行われるところまでは共通の処理である．方策 F では，毎回エラー要因の第 1 候補のみに対するリカバリー行動計画が選択される．方策 B では，エラー要因の第 1 候補のみに対するリカバリー行動計画か，第 2 要因以下まで考慮したリカバリー行動計画かが，そのときの時間依存効用によって選択される．方策 I では，エラー要因の第 1 候補のみに対するリカバリー行動計画か，第 2 要因以下まで考慮したリカバリー行動計画か，さらにそれらを判断するのに追加のセンシングを実行すべきか否かが，そのときの時間依存効用によって選択される．

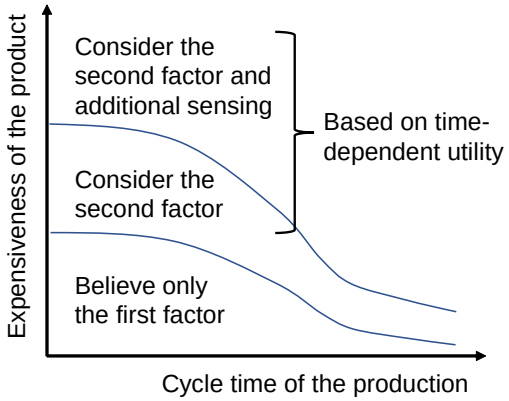


Fig. 5.7: Changes of optimal decision making models for error recovery on production sites.

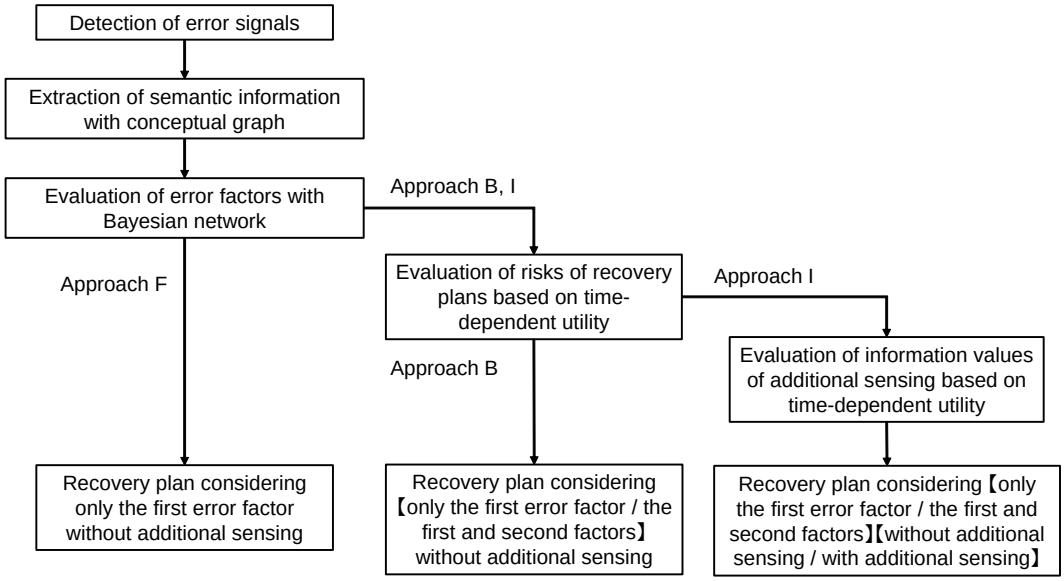


Fig. 5.8: Recovery process under the individual decision making models.【A/B】represents that the proposed system can select A or B.

仮に無限の時間猶予と必ず真の情報が得られるセンサが与えられていれば，取り得る情報収集手段を尽くして可能性のあるすべてのエラー要因をつぶすことが合理的である．しかしながら，実際の FA システムでは許容される時間資源・情報資源の範囲で意思決定を行うことになるために，ある段階で熟考を打ち切って実行に移す必要があり，そうした限定合理性がこの枠組みで保証される．

5.6　結言

本章では，生産現場の状況に応じて，熟考的エラーリカバリーに伴う資源管理を含めた合理的意思決定を行うための時間依存効用の考え方を導入した．同定された時間依存効用関数により，エラーリカバリーの実行に伴うリスクと変動する時間圧の下で，作業環境に適応した行動計画を求めることが可能である．さらに，現場でどの程度の熟考を伴う意思決定モデルを採用すべきかについては，対象作業のサイクルタイムと製品価格という自動生産に関わる因子により，指針が決められることを示した．次章では，複数台ロボットによる組立作業の具体例で，提案システムの動作を評価する．

第 6 章

複数台ロボットによる組立作業のエラーリカバリーシミュレーション評価

6.1 緒言

本章では，提案システムの有効性を検証するために実施した，2 台のロボットアームによる組立作業のシミュレーションについて論じる．作業対象としては一般性を持たせるため，実機での組立作業のベンチマークとしてよく用いられているギヤユニットの組立を模擬する [68]．その行動計画は Move 型，Grasp 型，Apply 型の動作の組合せで構成される．

検証事例として発生させるエラーは，実機に備え付けられるセンサの存在を仮定して検知されるものとする．提案する CG と BN のフレームワークを用いて修正戦略が決定され，エラー要因に応じたリカバリー行動計画が生成されることを示した上で，設定された時間依存効用関数に対する時間圧をパラメータとして変動させたときのロボットの挙動をシミュレーションする．

6.2 組立作業の問題設定

設計した作業環境におけるロボットとワークの初期状態を Fig. 6.1 に示す． モデルの描画・シミュレーションには三菱電機 (株) 製の産業用ロボット MELFA 用ソフトウェア「RT ToolBox3」を用いた．Fig. 6.1 で描画されている 2 台のロボットは，ともに MELFA の垂直多関節 (6 自由度) 型 4kg 可搬の RV-4FRL という機種に自作の電動ハンドを装着したもので，機能的に等価である．各ワークは区切られたフィーダパレットに置かれている．プランニング空間としては，動作主体の robot#1 と robot#2 はホームポジション h_1，h_2，フィーダ内の各ワークの把持位置 fa_1，fa_2，fa_3，fb_1，fb_2，fb_3，ベースユニット上の挿入位置 pt の間を移動でき，すべてのワークを把持・挿入できるよう事前に教示されているものとする．

組立順序の模式図と目標状態を Fig. 6.2 に示す． 赤色で示されているワーク [shaftA, gearA, nutA] のセットと，緑色で示されているワーク [shaftB, gearB, nutB] のセットは独立に組み立てることができるため，この作業の行動計画は半順序性を有する．個々のワークについて，ロボットがフィーダへ移動し (Move)，ワークを把持し (Grasp)，ベースユニットへ移動し (Move)，ワークを挿入する (Apply) という一連の行動を実行することで組立作業が進められる． 空間的な制約として，1 つの場所を占有できるロボットは 1 台とする．これより，ベースユニット pt にいるロボットが別の場所に移動するまで，次に取り付けられる部品を把持している他方のロボットは待機しておくことになる．

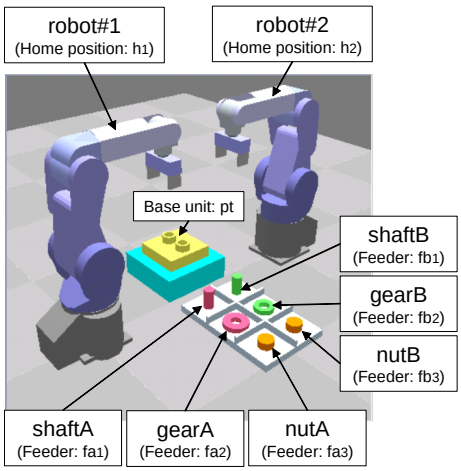


Fig. 6.1: Initial state of assembly tasks.

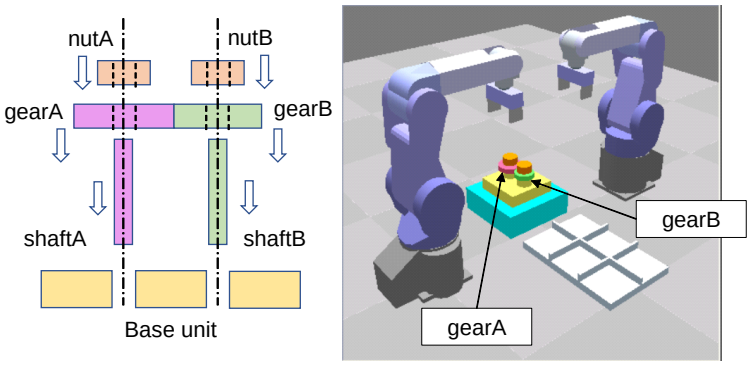


Fig. 6.2: Assembling sequence and the goal.

この組立作業の正常系の半順序プランを Fig. 6.3 に示す．プランステップとなる操作子 move の引数は移動先の場所，grasp と apply の引数は操作対象のワークを示している．Order constraints は Fig. 6.2 の目標状態を達成するための順序制約のリストを示している．

この半順序プランに基づいて生成される実行スケジュールでは，ベースユニットの場所を占有できるロボットは 1 台だけという空間的制約と，プランステップに対応するロボットコマンドの実行時間が与えられ，最後のナットの取付が完了するまでの総作業時間が最小となるように robot#1 と robot#2 にプランステップが割り当てられる．今回のシミュレーションでは，一律に Move 型の動作の実行時間を 2，Grasp 型の動作の実行時間を 3，Apply 型の動作の実行時間を 4(単位：sec) と設定する．正常系の各ロボットの実行スケジュールを Fig. 6.4 に示す．ガントチャートの灰色網掛け部分は空間的制約を満たすために挿入されるロボットの待機時間を示している．robot#1 が最後に実行する move(h_1) は，robot#2 がベースユニットに入れるように退避しておく必要があることから挿入される．

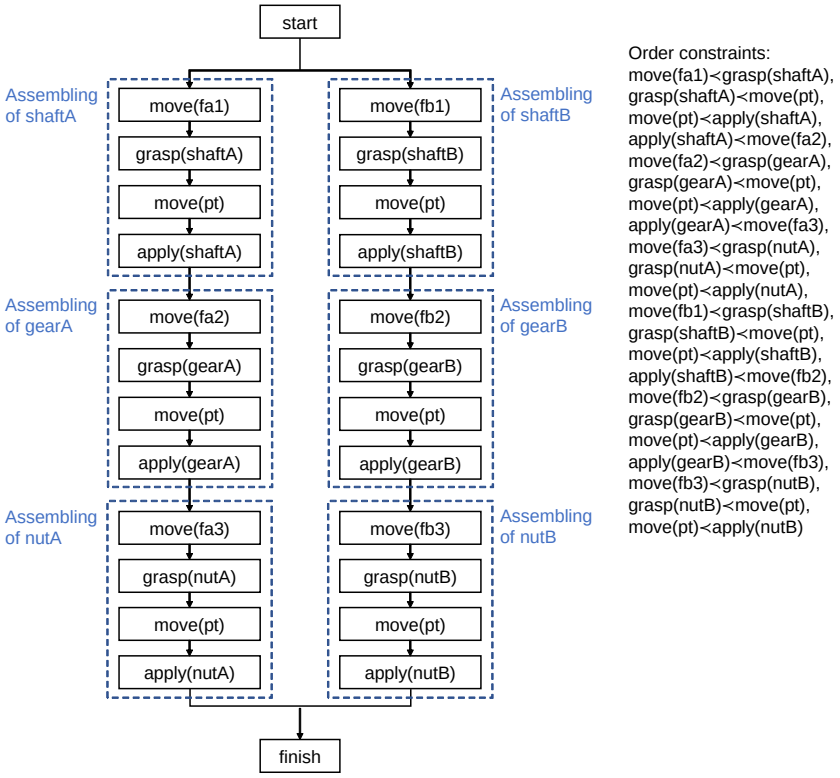


Fig. 6.3: Partial order plan of the normal system.

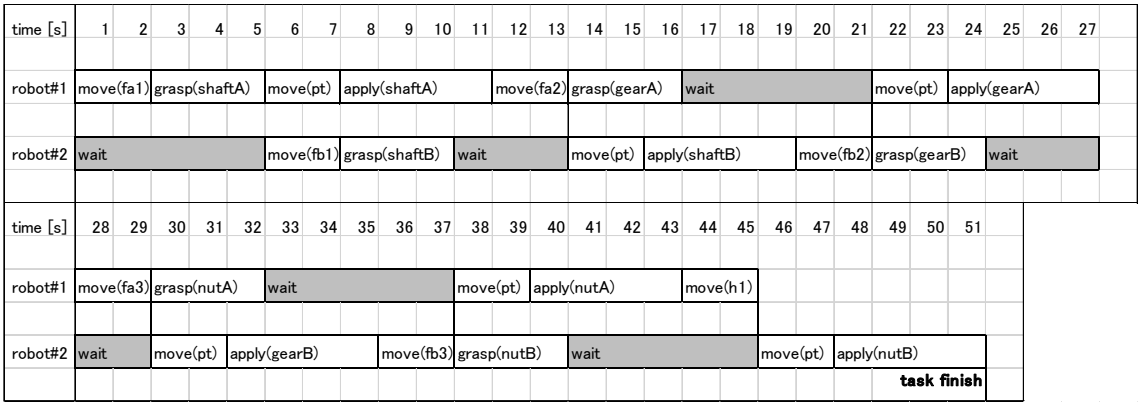
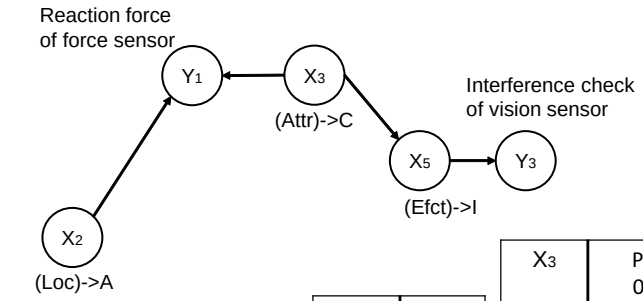


Fig. 6.4: Execution procedure of the normal system.

6.3　提案システムのエラーリカバリー事例

次に，エラーリカバリーのシミュレーションの設定について説明する．この作業では追加センシングの要否まで検討する熟考的アプローチを用いる．2 台のロボットはそれぞれ作業状態に関するエビデンスの収集手段として，アーム先端部分に力覚センサとビジョンセンサ (ハンドアイカメラ) を備えるとする．同時に発生し得るエラー要因として，ロボットの位置の異常 (Loc)，操作対象のワークの状態の異常 (Attr)，別のワークとの干渉 (Efct) の 3 つが存在するとする．このとき，提案する BN のフレームは Fig. 6.5 のように構成され，事後

確率の計算に必要な事前確率 $P(X_2)$，$P(X_3)$ と条件付確率 $P(Y_1|X_2, X_3)$，$P(X_5|X_3)$，$P(Y_3|X_5)$ の表が与えられる．Fig. 6.5 の中の表の値は想定されるセンサの性能より便宜的に設定した値であり，0 が正常，1 が異常であることを表す．



X2	X3	P(Y1|X2, X3) 0	1
0	0	0.90	0.10
0	1	0.80	0.20
1	0	0.10	0.90
1	1	0.10	0.90

X2	P(X2)
0	0.90
1	0.10

X3	P(X3)
0	0.80
1	0.20

X3	P(X5|X3) 0	1
0	0.90	0.10
1	0.20	0.80

X5	P(Y3|X5) 0	1
0	0.80	0.20
1	0.10	0.90

Fig. 6.5: BN of simulation system with probability tables.

以下では，エラー停止するロボットコマンドは同じであるが，その要因が異なる 2 つの事例でリカバリーに至るまでの提案システムの熟考的対応をシミュレーションする．ここでは部分的な作業のやり直しを可能にするために，把持しているワークをその場で放す Grasp 型の逆操作のコマンド “release” と，取り付けられているワークに力を加えて引き抜く Apply 型の逆操作のコマンド “remove” を準備しておく．変動する時間圧の下での利得 $x = (\text{デッドライン } T_d) - (\text{リカバリーの実行時間})$ に対する効用関数として，リスク回避的効用関数 $U_1(x) = 10 \cdot x^{0.5}$ とリスク志向的効用関数 $U_2(x) = 0.01 \cdot x^2$ $(0 \leq x \leq 100)$ の 2 種類を設定し，両者の対応を比較する．

6.3.1　Case 1: 動作主体のロボットの位置の異常によるエラー

1 つ目の事例では，動作主体のロボットの位置の異常を要因とするエラーが後続の挿入動作に影響を及ぼす場合を扱う．エラー検知のシナリオとしては，robot#2 の apply(gearB) の動作実行中に，力覚センサで異常値を検知して停止命令がかかったとする．このとき例示化される CG と BN を Fig. 6.6 に示す．Apply 型の CG に含まれるエラーの深層格は Loc であり，所定の挿入位置に対して robot#2 の現在の位置がずれているために動作に失敗したことを示している．

CG が示すエラーの第 1 要因を真としたときの半順序プランを Fig. 6.7 に示す．正常に実行完了しているプランステップは灰色網掛けで示されている．この事例では，エラー検知の直前に実行された move(pt) という移動動作がエラー要因を生んでおり，これを正常に完了することで apply(gearB) が再び実行できるようになる．正常系で robot#1 に割り当てられているプランステップからの因果リンクの追加はない．

次に，追加のエビデンスとして BN に含まれるビジョンセンサの値を確認し，正常であった $(Y_3 = 0)$ 場合にエラー要因の候補の事後確率がどう変化するかを Fig. 6.8 に示す．追加センシング前は Loc のエラーの事後確率が $P(X_2 = 1|Y_1 = 1) = 0.45$，Attr のエラーの事後確率が $P(X_3 = 1|Y_1 = 1) = 0.27$，Efct の

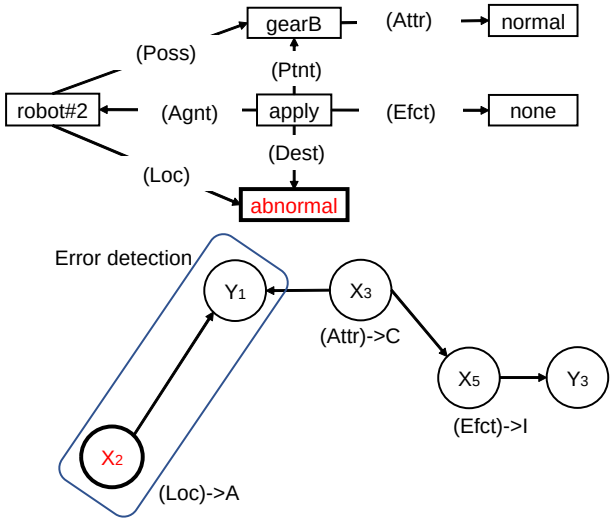


Fig. 6.6: Instantiated CG and BN in Case 1.

エラーの事後確率が $P(X_5 = 1|Y_1 = 1) = 0.29$ である．追加センシング後は Loc のエラーの事後確率が $P(X_2 = 1|Y_1 = 1, Y_3 = 0) = 0.48$，Attr のエラーの事後確率が $P(X_3 = 1|Y_1 = 1, Y_3 = 0) = 0.11$，Efct のエラーの事後確率が $P(X_5 = 1|Y_1 = 1, Y_3 = 0) = 0.05$ であり，Loc の格関係の異常によるエラーであるという確信度が大きくなる．

追加センシングなしで第 1 候補である Loc の格関係のみを考慮して対応する修正戦略を決定した場合に生成される実行スケジュールを Fig. 6.9 に示す．この場合，robot#2 が move(pt) を再試行することでエラー状態が解消される．Loc の格関係に加え，追加センシングなしで第 2 候補である Attr の格関係まで考慮した場合に生成される実行スケジュールを Fig. 6.10 に示す．この場合，robot#2 は Attr の格関係を修復するために gearB の把持を実行するところからやり直す．追加センシングを実行した上で第 1 候補である Loc の格関係のみを考慮した場合に生成される実行スケジュールを Fig. 6.11 に示す．この場合，robot#1 が時間コスト 2.0 s を費やしてビジョンセンサの情報を得る．いずれの場合もエラー状態を解消した後，当初の目標状態に向けて残っているプランステップは作業時間が最小となるように割り当て直される．

この事例で第 1 要因のみ考慮する Fig. 6.9 のリカバリー動作の実行時間は 6.0 s であり，第 2 要因まで考慮する Fig. 6.10 の実行時間は 8.0 s である．これらの時間は，考慮する要因に対するリカバリー行動をロボットコマンドとして実行する際に必要な時間である．仮に第 2 要因の影響があり Fig. 6.9 の実行スケジュール通りに復旧できなかったとすると，そこから Fig. 6.10 のスケジュールに切り替えて復旧するのに 14.0 s の実行時間を要する．これらをデッドライン T_d から差し引いた残りを確率的に得られる利得とするとき，T_d を変動させて効用関数 U_1 ならびに U_2 の下で採用されるスケジュールを Table 6.1 に示す．時間圧が比較的小さい $T_d = 100$ s のときは，いずれの効用関数でも第 2 要因まで考慮したスケジュールが採用される．時間圧が比較的大きい $T_d = 50$ s のときは，リスク回避型効用関数 U_1 では依然として第 2 要因まで考慮される一方，リスク志向型効用関数 U_2 では第 1 要因のみ考慮されるようになる．いずれの効用関数でも追加センシングを行うスケジュールが採用されないのは，新たに得られる情報の価値に対してセンシングコスト 2.0 s が受け入れられないためである．

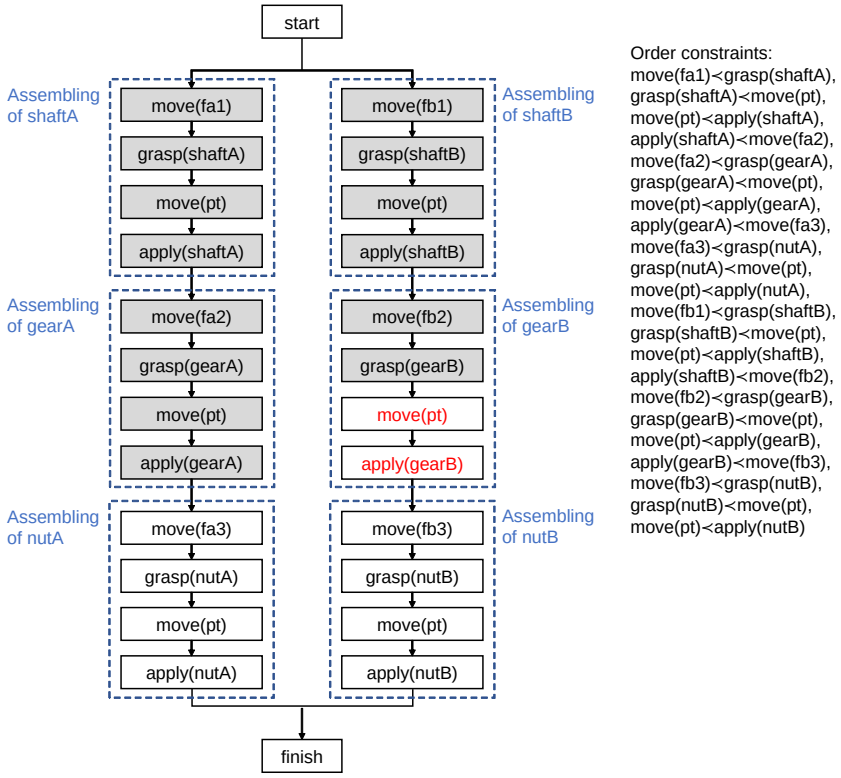


Fig. 6.7: Partial order plan after error detection in Case 1.

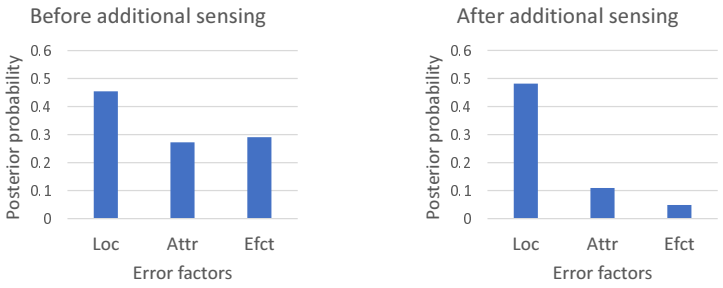


Fig. 6.8: Posterior probability before/after additional sensing with the vision sensor (Y_3) in Case 1.

これらのリカバリープランの選択がどのようなロボットの動作の違いに表れるかを Fig. 6.12 に示す．各シーンの図中の吹き出しは，Fig. 6.9〜Fig. 6.10 の実行スケジュールで対応する動作を表している．robot#2 の apply(gearB) の動作実行中に力覚センサで異常検知したシーン (A) は共通である．第 1 要因のみ考慮した Fig. 6.9 のスケジュールでは，即座に robot#2 が move(pt) を再試行するシーン (B) が見られる．一方，第 2 要因まで考慮した Fig. 6.10 のスケジュールでは，robot#2 が gearB の把持を実行するところからやり直すシーン (C) が見られる．この場合，robot#1 が並行してエラーの影響を受けない作業を進めておけるために，リカバリーが完了後に残っている nutB の組付けを代替することができ，シーン (C) と (E) の違いが見られる．いずれの場合も，システムが必要と判断したリカバリー動作を当初の実行スケジュールに過不足なく統合して，目標状態を達成することができる．

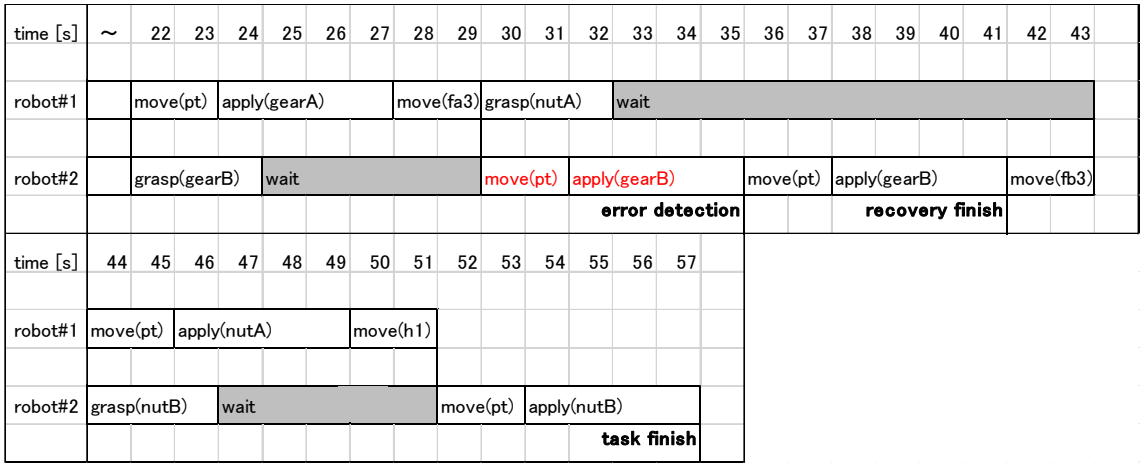


Fig. 6.9: Recovery plan considering only the first error factor without additional sensing in Case 1.

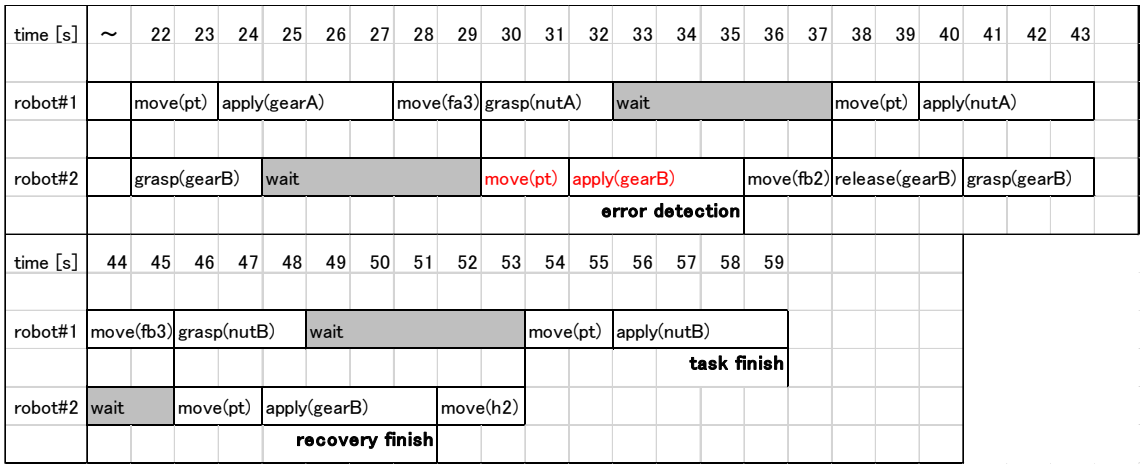


Fig. 6.10: Recovery plan considering the first and second error factors without additional sensing in Case 1.

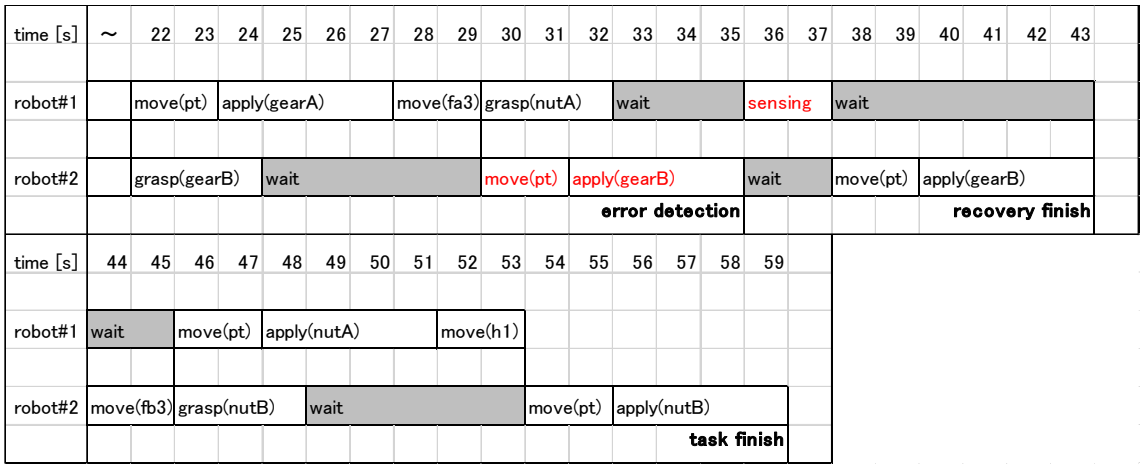


Fig. 6.11: Recovery plan considering only the first error factor with additional sensing in Case 1.

Table 6.1: Change of the rational recovery plan in Case 1. This is a test case that the posterior probability of the second factor is updated from 0.26 to 0.10 by additional sensing and the sensing cost is 2.0 s.

Utility function	Recovery plan when $T_d = 100$	Recovery plan when $T_d = 50$	Acceptable sensing cost when $T_d = 100$
U_1 (risk averse)	Fig. 6.10	Fig. 6.10	1.1
U_2 (risk seeking)	Fig. 6.10	Fig. 6.9	1.2

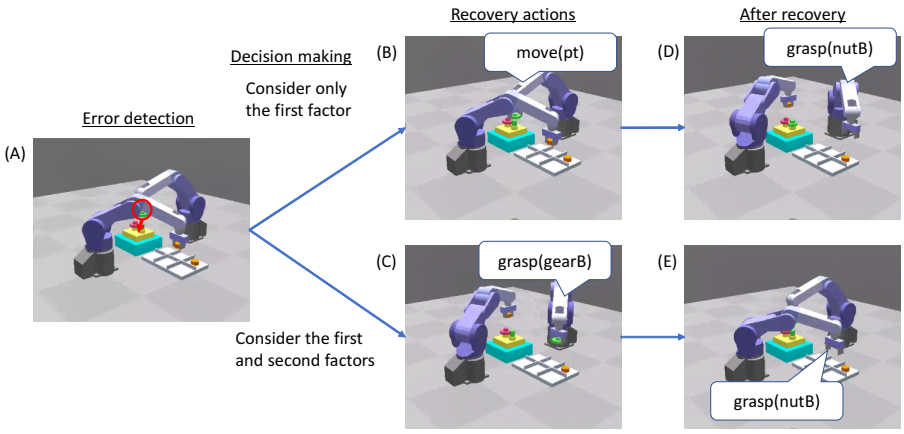


Fig. 6.12: Differences of actions of the robots according to the decision in Case 1.

6.3.2　Case 2: 先に挿入されたワークとの干渉によるエラー

2 つ目の事例では，他方のロボットによって先に挿入されたワークとの干渉を要因とするエラーが挿入動作に影響を及ぼす場合を扱う．エラー検知のシナリオとしては，robot#2 の apply(gearB) の動作実行中に，ビジョンセンサで干渉物を検知して衝突前に停止命令がかかったとする．このとき例示化される CG と BN を Fig. 6.13 に示す．Apply 型の CG に含まれるエラーの深層格は Efct であり，所定の挿入位置への経路上で gearA が干渉しているために動作に失敗したことを示している．

CG が示すエラーの第 1 要因を真としたときの半順序プランを Fig. 6.14 に示す．この事例では，robot#1 によって実行された apply(gearA) という挿入動作がエラー要因を生んでおり，これを正常に完了することで apply(gearB) が再び実行できるようになるという因果リンクが追加される．

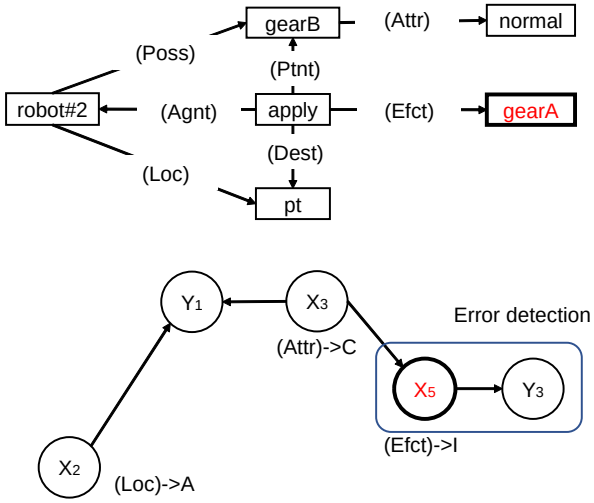


Fig. 6.13: Instantiated CG and BN in Case 2.

次に，追加のエビデンスとして BN に含まれる力覚センサの値を確認し，正常であった ($Y_1 = 0$) 場合にエラー要因の候補の事後確率がどう変化するかを Fig. 6.8 に示す．追加センシング前は Loc のエラーの事後確率が $P(X_2 = 1|Y_3 = 1) = 0.10$，Attr のエラーの事後確率が $P(X_3 = 1|Y_3 = 1) = 0.41$，Efct のエラーの事後確率が $P(X_5 = 1|Y_3 = 1) = 0.59$ である．追加センシング後は Loc のエラーの事後確率が $P(X_2 = 1|Y_3 = 1, Y_1 = 0) = 0.01$，Attr のエラーの事後確率が $P(X_3 = 1|Y_3 = 1, Y_1 = 0) = 0.39$，Efct のエラーの事後確率が $P(X_5 = 1|Y_3 = 1, Y_1 = 0) = 0.57$ であり，Loc の格関係の異常は事後確率が十分に小さいとしてエラー要因の候補から外すことができる．

追加センシングなしで第 1 候補である Efct の格関係のみを考慮して対応する修正戦略を決定した場合に生成される実行スケジュールを Fig. 6.16 に示す．この場合，robot#1 が一旦 gearA を取り外し，apply(gearA) を再試行することでエラー状態が解消される．Efct の格関係に加え，追加センシングなしで第 2 候補である Attr の格関係まで考慮した場合に生成される実行スケジュールを Fig. 6.17 に示す．この場合，robot#1 が apply(gearA) を再試行する間に，robot#2 は Attr の格関係を修復するために gearB の把持を再試行する．追加センシングを実行した上で第 1 候補である Efct の格関係と第 2 候補である Attr の格関係を考慮した場合に生成される実行スケジュールを Fig. 6.18 に示す．この場合，robot#2 が時間コスト 2.0 s を費やして力覚センサの情報を得る．

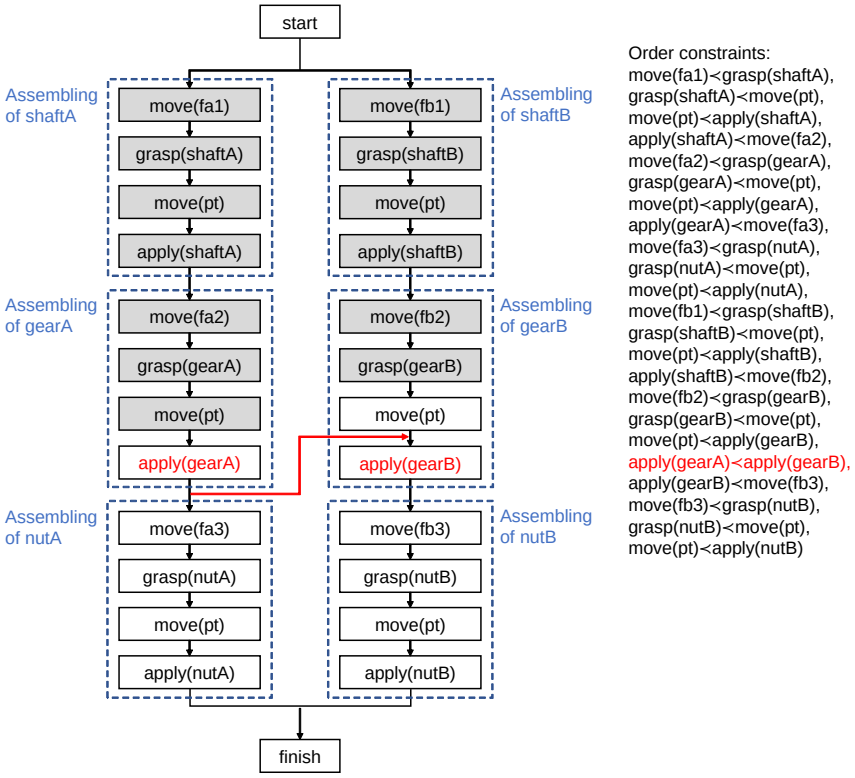


Fig. 6.14: Partial order plan after error detection in Case 2.

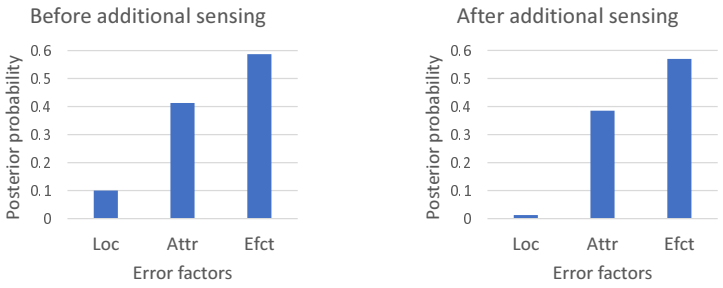


Fig. 6.15: Posterior probability before/after additional sensing with the force sensor (Y_1) in Case 2.

この事例で第 1 要因のみ考慮する Fig. 6.16 のリカバリー動作の実行時間は 20.0 s であり，第 2 要因まで考慮する Fig. 6.17 の実行時間は 25.0 s である．仮に第 2 要因の影響があり Fig. 6.16 の実行スケジュール通りに復旧できなかったとすると，そこから Fig. 6.17 のスケジュールに切り替えて復旧するのに 45.0 s の実行時間を要する．これらをデッドライン T_d から差し引いた残りを確率的に得られる利得とするとき，T_d を変動させて効用関数 U_1 ならびに U_2 の下で採用されるスケジュールを Table 6.2 に示す．時間圧が比較的小さい $T_d = 100$ s のときは，いずれの効用関数でも追加センシングを実行した上で第 2 要因まで考慮したスケジュールが採用される．時間圧が比較的大きい $T_d = 50$ s のときは，リスク回避型効用関数 U_1 では追加センシングなしで第 2 要因まで考慮される一方，リスク志向型効用関数 U_2 では追加センシングなしで第 1 要因のみ考慮されるようになる．この事例では Case 1 に比べて失敗したときのコストが大きいため，時間圧が小さいうち

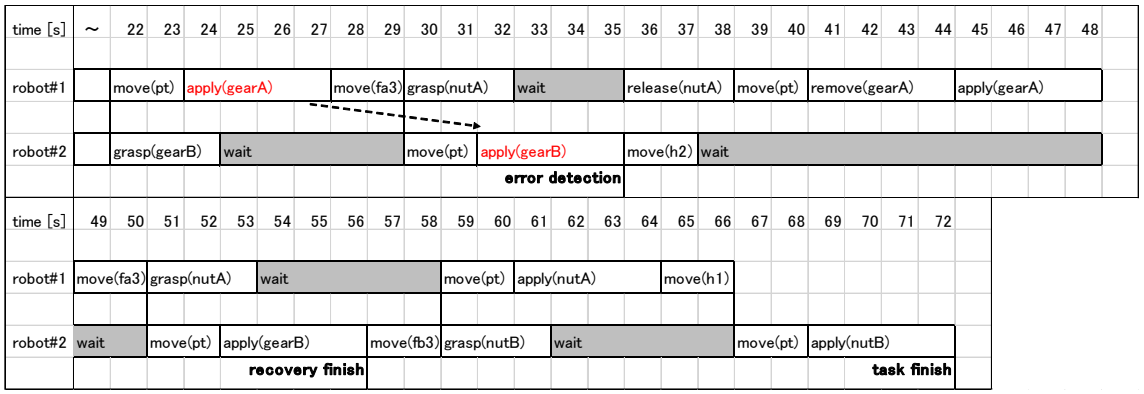


Fig. 6.16: Recovery plan considering only the first error factor without additional sensing in Case 2.

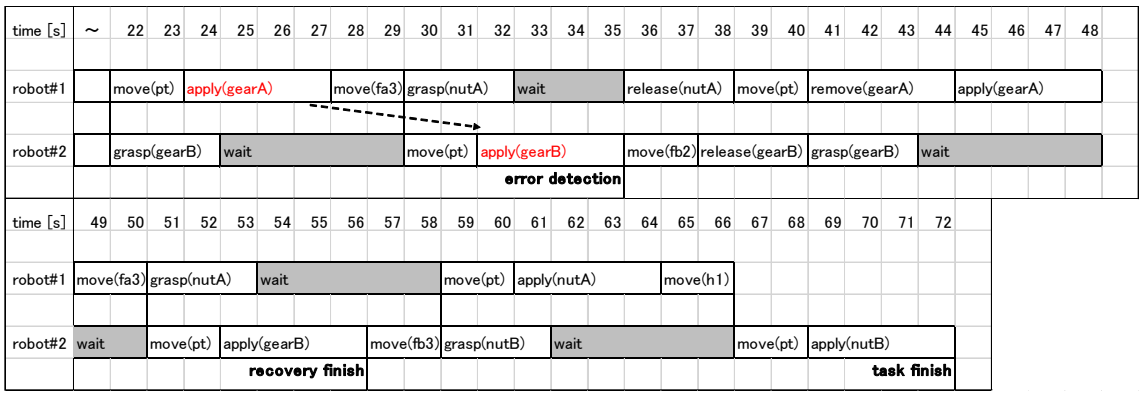


Fig. 6.17: Recovery plan considering the first and second error factors without additional sensing in Case 2.

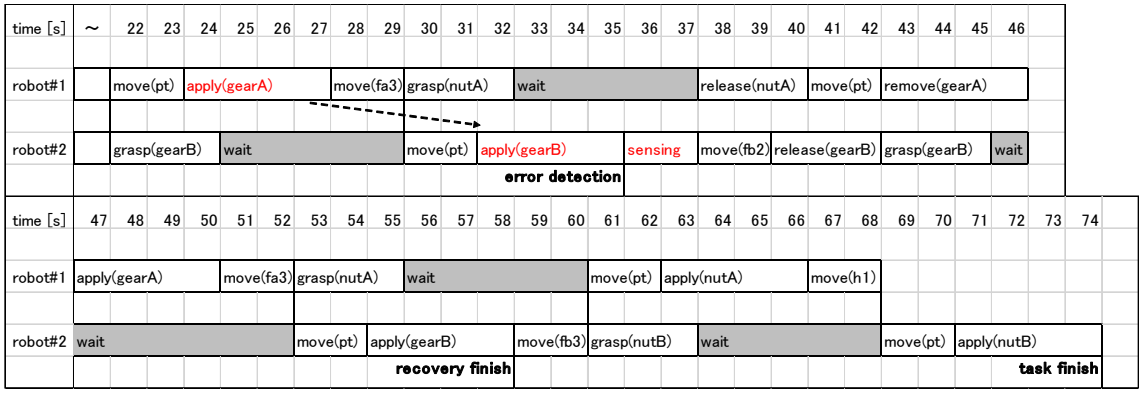


Fig. 6.18: Recovery plan considering the first and second error factors with additional sensing in Case 2.

は新たに得られる情報の価値に対するセンシングコスト 2.0 s が受け入れられるという結果が示されている．

Table 6.2: Change of the rational recovery plan in Case 2. This is a test case that the posterior probability of the second factor is updated from 0.20 to 0.10 by additional sensing and the sensing cost is 2.0 s.

Utility function	Recovery plan when $T_d = 100$	Recovery plan when $T_d = 50$	Acceptable sensing cost when $T_d = 100$
U_1 (risk averse)	Fig. 6.18	Fig. 6.17	2.2
U_2 (risk seeking)	Fig. 6.18	Fig. 6.16	2.2

これらのリカバリープランの選択がどのようなロボットの動作の違いに表れるかを Fig. 6.19 に示す．各シーンの図中の吹き出しは，Fig. 6.16〜Fig. 6.18 の実行スケジュールで対応する動作を表している．robot#2 の apply(gearB) の動作実行中にビジョンセンサで異常検知したシーン (A) は共通である．第 1 要因のみ考慮した Fig. 6.16 のスケジュールでは，robot#2 は干渉が解決され次第 apply(gearB) を再試行できるようにホームポジション h_2 に退避しておき，robot#1 が gearA を取り除くシーン (B) が見られる．一方，第 2 要因まで考慮した Fig. 6.17 のスケジュールでは，robot#1 が gearA を取り除く間に robot#2 が gearB の把持を再試行するシーン (C) が見られる．さらに，追加のセンシングを行うべきと判断された Fig. 6.18 のスケジュールでは，robot#2 が手先を所定の位置に移動させて力覚センサの値を確認するシーン (D) が見られ，その間 robot#1 は動作再開を保留する．しかる後に第 2 要因まで考慮すべきとして，robot#1 が gearA を取り除く間に robot#2 が gearB の把持を再試行するシーン (E) が見られる．リカバリー完了後に残りの作業を進めるシーン (F)(G)(H) の手順は共通である．

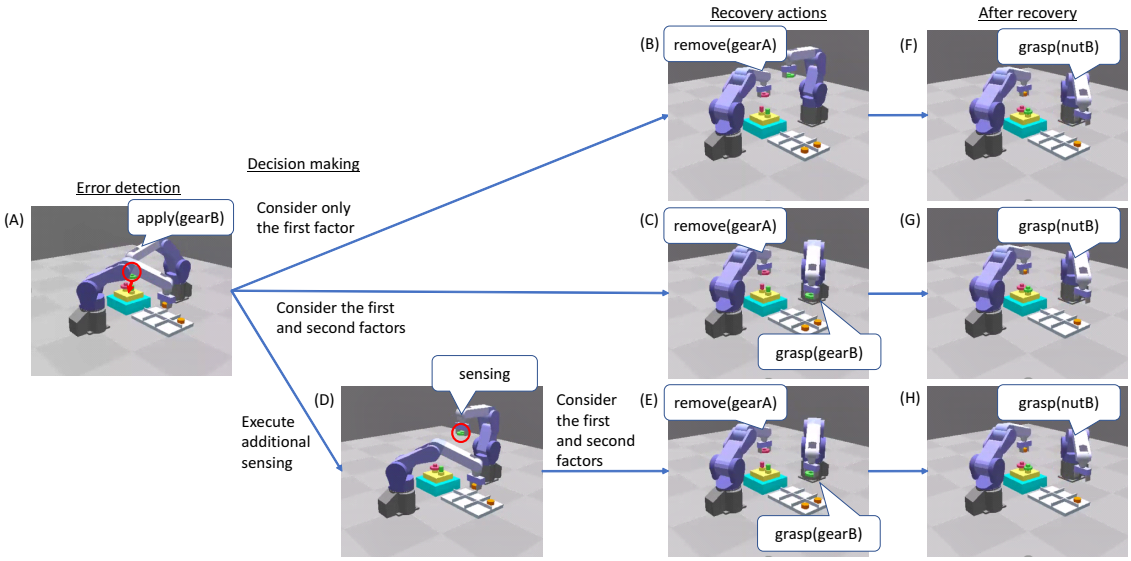


Fig. 6.19: Differences of actions of the robots according to the decision in Case 2.

6.4　複雑な作業に対するエラーリカバリー行動計画の汎用性の検証

本章で検証してきた 2 台のロボットアームによるギヤユニット組立は，一般的な FA のロボットセルに見られる作業事例であり，他の電気製品や電子基板の生産システムでも同様にエラーリカバリーを実行することができる．一方で，作業内容としては工数が比較的少なく単純な設定であったため，より複雑なエラー事象が想定される作業において，提案システムがどのようにリカバリー行動計画を実行するのかを本節でまとめる．

6.4.1　3 台以上のロボットによる作業の実行

まず，第 2 章で述べた A^* アルゴリズムによる最適スケジューリングは作業を実行するロボットの台数によらず適用可能であり，ロボットが 3 台以上の場合であっても提案システムは実行可能な解を導くことができる．また，実際のロボットセルでは，機種の異なる複数のロボットが導入される場合が考えられる．この場合にも汎用動作の格関係に基づく修正戦略が機能するが，再スケジューリングの段階においてすべてのワークをすべてのロボットが操作できるとする前提条件は成立しない可能性がある．すなわち，早く手が空いたロボットが元々他のロボットに割り当てられていたプランステップを代替するにあたっては，教示済みのワークに限られるという実行上の制約下で，最も作業時間の短くなる実行手順が探索される．

6.4.2　Conceptual Graph で表現される状態の拡張性

複数のエラー要因に対して CG で記述される状態の拡張性と，そのときの修正戦略の適用方法について考える．エラーが検出された動作の類型による 1 つの CG に含まれる複数の格関係が異常を示している場合には，これまでにも評価してきた通り，それぞれの要因に対する修正戦略を適用して半順序プランナに渡される To-do リストを構成すればよい，干渉を示す格関係 Efct については，複数のロボットやワークと同時に干渉する場合があり，個別にそれらを取り除くことが To-do リストに追加される．干渉物が単なる障害物として存在するワークではなく，並行して動作している他のロボットや，他のロボットが操作しているワークであった場合には，まずエラーを検知したロボットの動作についての CG を例示化した後，干渉物についての CG を例示化する．robot#2 の移動中に干渉するとしてエラー検知された相手が robot#1 であった場合の CG の拡張例を Fig. 6.20 に，robot#1 が挿入しようとしているワークであった場合の CG の拡張例を Fig. 6.21 に示す．こうした場合には，エラーを検知した動作から見て末端のエラー要因から修正戦略を適用することで To-do リストを構成する．Fig. 6.20 のエラーでは，robot#1 がいる位置を正常化した上で，robot#1 が robot#2 の Move 動作に及ぼしている干渉を解消する．Fig. 6.21 のエラーでは，robot#1 が把持している gearA の状態を正常化した上で，gearA が robot#2 の Move 動作に及ぼしている干渉を解消する．

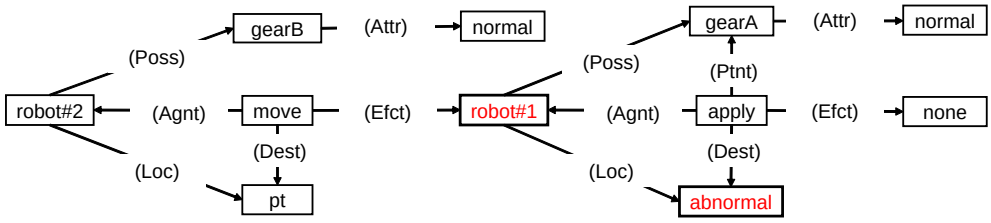


Fig. 6.20: Example of expansion of CG against interference of an operating robot.

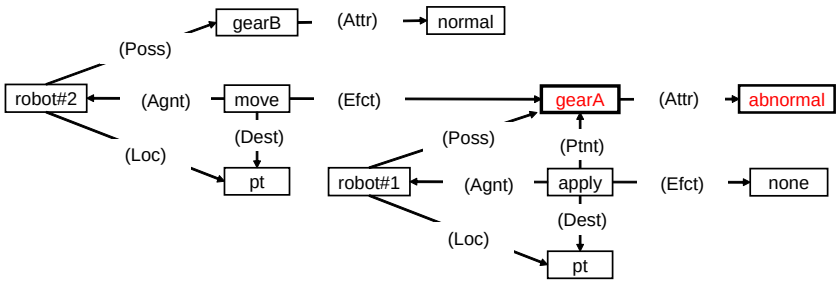


Fig. 6.21: Example of expansion of CG against interference of an operated workpiece.

6.4.3　プランナが対象とする作業環境の拡張性

提案システムのリカバリー行動計画は，半順序プランナで事前に定義されている作業空間の中で状態遷移を行う操作子の適用によって導出されるものである．本章の具体例では，Fig. 6.1 に示されているホームポジション，フィーダの位置，ベースユニットの位置に相当するエリアでの物体操作が対象となるため，それ以外の場所に落としてしまったワークをロボットが拾い上げるような動作は，基本的にプランナとしては推論できない．しかしながら，ビジョンセンサの情報に従ってロボットの可動域内にあるワークの位置に手先を合わせにいくビジュアルフィードバック制御技術 [69][70] 等と組み合わせて運用できる場合，たとえば組立場所の周辺エリアをプランナに登録しておき，ワーク喪失による Poss の格関係のエラーを周辺エリアへの Move 動作と Grasp 動作の組合せで解決することが可能である．

6.4.4　再帰的なリカバリー行動計画と提案システムの限界

リカバリー動作中に新たなエラーが発生した場合の深さ方向の拡張性，すなわち再帰的なリカバリー行動計画について考える．この場合には，1 つ前に検知されたエラーに対するリカバリー行動計画を完了することを目標として，修正戦略を適用する．結果的に，k 番目 ($k = 1, 2, \cdots$) のエラーのリカバリー行動計画に対して，$k+1$ 番目のエラーのリカバリー行動計画が入れ子構造となった実行スケジュールが得られる．一例として，前節で取り上げた Case 1 で robot#2 の位置 (Loc の格関係) の異常を解消する Move 動作中に，把持していた gearB の状態 (Attr の格関係) の異常が新たに検知されたときに生成される実行スケジュールを Fig. 6.22 に示す．robot#2 は一旦 gearB のフィーダに戻って再把持を行い，2 回目のエラーを検知した Move 動作を完了した後，1 回目のエラーを検知した Apply 動作を完了することで，最終的に正常系へと復帰する．

一方で，$k+1$ 番目のリカバリー行動計画を熟考する際に参照されるセンサ情報も，k 番目のリカバリー行動計画と同様に不確実性を含むものである限り，有限回の試行で必ず正常系への復帰が完了することの保証はできない．提案システムはロボットが行動可能である限り次にとるべき行動を導出しようとするが，いずれはデッドライン T_d に達してそれ以上の試行に価値がなくなるか，あるいは現場で設定された最大試行回数に達して以降は人が駆け付けるまで待機するよう指令を受ける等して，ロボットの自律的対応の打ち切りを迎えることになる．

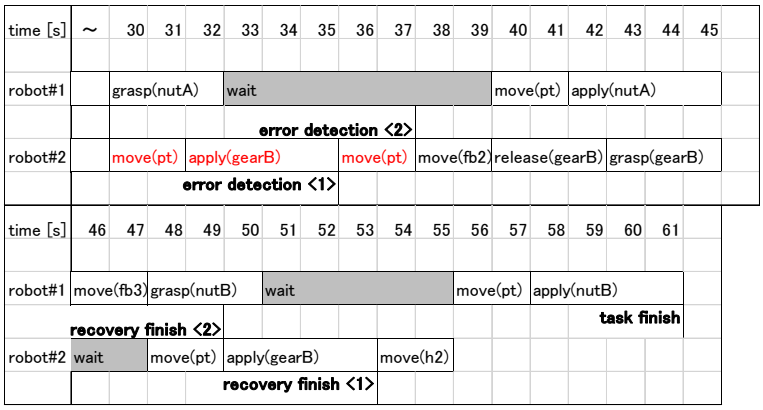


Fig. 6.22: Example of recursive recovery plan in Case 1.

6.5 結言

本章では，2 台のロボットによる組立作業のシミュレーションを行い，前段の行動から波及するエラー要因に応じて，実行可能なリカバリー行動計画が生成されることを確認した．同種のエラーでも要求されるリカバリー完了までの時間圧が変化したとき，効用関数に応じて考慮すべき不確実性要因と許容可能なセンシングコストが異なり，その判断がロボットの実行する動作に反映されることを示した．さらに，複数のエラー要因が CG で記述される場合の拡張性や，リカバリー動作中にエラーが発生した場合の再帰性について，段階的に正常系まで復帰できることを検証した．以上の結果により，提案する熟考モデルでは，実行時間コストと情報価値のトレードオフである追加センシングの要否まで含めた合理的意思決定能力を用いて，作業を継続可能であることが示された．

第 7 章

結論

7.1 本論文のまとめ

本論文では，産業用ロボットの失敗の意味情報を活用した行動計画レベルでのエラーリカバリーシステムの構築，ならびにそのエラーリカバリー機能を運用する現場での行動計画の生成と実行に関するリスクを考慮した，FA の資源制約下での合理的意思決定モデルの制御について論じた．各章で得られた成果を以下に記す．

第 2 章では，対象作業の行動計画の全容を管理するプランナと、プランナで導出された順序制約の下で複数のロボットの動作実行手順を決定するスケジューラを介して、下位のコントローラにより各ロボットの動作を制御する階層構造を持つ，リカバリー行動計画システムを構築した．本研究で対象とする行動計画とは，環境モデルにおいて与えられた目標状態を達成する操作子の系列を導出することであり，リカバリー問題では正常系から乖離した状態であるエラー状態から，どの時点までさかのぼって修正するかという逆方向の推論能力が求められることを示した．このとき不要な出戻りを避け，元の計画にエラー要因を解消する操作子とそれに付随する制約条件を追加することができるよう，作業の自由度を残した表現が可能な半順序プランニングを適用した．一方で，作業再開時には各ロボットの動作コマンドレベルで実行手順が一意的に定まっている必要があることから，スケジューラでは A^* アルゴリズムを用いて，作業完了までの時間が最小となる実行スケジュールが導出されることを示した．提案システムによるエラー信号の検知から作業再開までのフローチャートを構成し，実行可能なリカバリー行動計画を得るために必要なエラー要因の評価ステップを示した．

第 3 章では，エラー状態に直面したロボットはリカバリーのために作業空間にあるどこまでの要素を考慮すればよいかというフレーム問題に対処するため，エラー要因を解釈してリカバリーの目標を定めるための汎用的な修正戦略の体系化に取り組んだ．ロボットの動作に伴うエラー要因を体系的に記述するために，格文法に則って構成される意味ネットワークである Conceptual Graph を適用し，エラー状態の例示化の過程を示した．産業用ロボットの動作コマンドはメーカーごとのプログラミング言語で記述されているが，物体操作を行うために不可欠な汎用動作として捉えたとき，ロボット自身が移動する “Move”，ロボットが物体をつかむ “Grasp”，ロボットが物体に力を加える “Apply” の 3 つの述語に集約されることを示した．行動計画のレベルでこれらの述語の前提条件と事後条件を整理し，動作の状態を記述するのに必要十分な Conceptual Graph の格フレームを定義した．このフレームに含まれるノードと，検知されたエラー動作の情報を照合することで，再び前提条件を充足してリカバリーに至るまでに何をしなければならないかが明らかとなる．これがエラーの背後にある深層格に関する意味情報であり，これによって修復すべき格関係という形で作業対象によらない修正戦略を定義することができた．Move 型、Grasp 型、Apply 型の動作でエラーが生じる状況と対応する格関係を分析して，修正戦略の網羅性を確認した．

第 4 章では，実環境で検知されるエラー情報がセンシングの不確実性を含むものであることから，実際のエラー状態にそぐわない修正戦略が適用されてしまうリスクに対して，抽出されたエラー要因の尤もらしさを評価する枠組みを構築した．確率的な評価を行うため，Conceptual Graph で抽出されたエラー要因の候補となる格関係と，実際のシステムが備えるセンサ情報を結びつけるベイジアンネットワークを導入した．ロボットシステムが備える力覚センサやビジョンセンサなどの情報資源に対して，提案するフレームを用いて事後確率を更新するベイジアンネットワークを動的に構成し，信念の伝播によってエラー要因の尤もらしさを評価できることを示した．

第 5 章では，算出されたエラー要因の評価値に対して，今どれだけの不確実性を許容でき，さらに時間コストを費やし不確実性低減策を取り入れた計画を求めるべきか否かについて，行動判断に許容される時間資源やシステムが備えるセンサによる情報資源といった FA 領域の資源制約下での合理的意思決定を行うために期待効用理論を導入した．生産現場では，一日の中でも変動するデッドラインまでにどれだけ早くリカバリーが完了するかに価値があることから，効用関数は時間依存効用であるべきことを示した．追加のセンサ情報を取得してエラー要因の評価値が更新されるまで判断を保留している間にも，早期にリカバリーを完了することで得られるはずの利得は減少していく．このような状況でリカバリー動作が失敗するリスクに対する態度を表す時間依存効用関数の形状について，時間 t 経過後に確率 p で利得 x が得られるくじと等価になる，現時点で確実に得られる利得に掛る質問を行うことにより同定する方法を示した．さらに，エラーリカバリーシステムがどのレベルの熟考モデルを用いて意思決定を行うべきかというメタレベル意思決定問題について，時間的猶予に関わる対象作業のサイクルタイムと，失敗時の損失の大きさに関わる製品の価格という 2 軸で，対象作業のエラーリカバリーに要求される熟考レベルを整理した．安価な製品を短い周期で生産する状況では，最も尤もらしいエラー要因に対する修正戦略を短い推論時間で判断するモデルを採用し，より慎重に行動を判断することが求められる高価な製品を長い周期で生産する状況では，時間依存効用に基づく判断モデルを組み込むことで，現場に適した合理的意思決定を制御できることを示した．

第 6 章では，提案システムの有効性を検証するため，2 台のロボットアームによるギヤユニットの組立作業を題材とするシミュレーション実験の結果を示した．複数台協調セルを模擬した中央ステージを占有できるロボットは 1 台のみという空間的制約の下で，半順序プランに基づき生成された実行スケジュールにしたがって作業を進める途中でエラーを発生させ，同じロボットコマンドでもエラー要因が異なる複数の事例を作成し，リカバリーに至るまでの熟考的対応の結果を比較した．修正戦略の不適合によるリカバリー失敗のリスクが小さい事例では不確実性低減のための追加センシングは実行されず，同じセンシングコストでも失敗のリスクの大きい事例では時間圧が大きくなってくるまで追加センシングを行うという結果が得られた．さらに，リスク回避的な効用関数で評価する場合とリスク志向的な効用関数で評価する場合とで，時間圧の変化に対して採用される行動計画に差異が見られ，いずれも最終的にはリカバリーを達成する行動が実行されることを確認した．こうした行動の違いは，FA の現場で所定の生産性を発揮するために投入可能な時間資源・情報資源の制約下での合理的判断に基づくものであることが，提案システムでは保証される．さらに，複数のエラー要因が発生した場合やリカバリー動作中にエラーが発生した場合であっても，完了できなかった動作の格の関係で記述されるエラー状態である限りは提案システムの範疇で，正常系に至るまでの再帰的な対処が可能であることを検証した．

7.2　課題と展望

最後に，生産現場で提案システムを導入し運用する上での，本論文の各章の項目に関連する課題を整理し，将来の FA に向けた本研究の展望を述べる．

Conceptual Graph の格フレームを用いたエラー要因の抽出においては，ノードを照合する際の正常と異常の閾値について，適用対象のシステムが備えるエラー検知手段のセンサに合わせて設定する必要がある．たとえば干渉の有無については，ビジョンセンサで取得された画像から干渉ありと判断するプログラムを介して，そのとき干渉しているロボットやワークの ID が Efct の格関係にマッチングされる．複数の干渉物がある場合や，他の格関係の異常が同時に検知された場合には，それらを解消する修正戦略を 1 つずつ適用し，多段階的にリカバリーを進めることができる．

ベイジアンネットワークを用いたエラー要因候補の事後確率の算出においては，事前確率と条件付確率の表を与える必要がある．センサの性能や周辺設備に依存するこれらの情報は，初めは人が経験的に入力することになるとしても，生産準備段階および生産実施段階で工場のサーバ等に蓄積されていく稼働データとの連携により，運用中に精度を高めていくことが考えられる．

時間依存効用関数の設計においては，本論文で示した同定手順にしたがって質問が投げかけられるインタフェースを介して，その日の生産計画を把握した現場監督者がチューニングを行うことを想定している．個人のリスクに対する考え方のバイアスが回答に表れることが考えられ，その現場にとって最適な効用関数になっているかは様々な状況でのデータを集めて評価することが必要である．将来的には生産条件に対する効用関数の自動最適化が課題である．

本論文のシミュレーション実験では，プランステップを実行するロボットの自由度を活かすため，機能的に等価な 2 台のロボットですべてのワークを操作することができることを仮定した．実際には，異なる機種や専用ハンドを備えたロボットを用いる自動化セルが少なくなく，リカバリーで使用する動作について個別に教示作業を実施する必要がある．こうした導入時のユーザの負担を軽減するため，動作コマンドレベルでの軌道生成技術や教示済の動作からの模倣・転移学習技術と組み合わせていくことが考えられる．

提案システムの実用化により，機械側で生産現場の状況を考慮してエラー要因を解消する行動計画レベルのリカバリーの実現が期待される．FA ロボットの新機能として製品化を目指す上では，確率的な評価値に基づくリカバリー動作の実行を許容することが 1 つの重要課題であり，安全性の担保のため作業中に動作変更する範囲に制限がかけられるなど，本論文で示したほど柔軟な対応のバリエーションが認められないかもしれない．それでも，従来の設備稼働時間の 10%以上を占めるとも言われてきたチョコ停の大部分で動作の継続が可能になることで，ロボットシステムの稼働率を飛躍的に向上させる余地があると考えられる．

昨今では，熟練作業者の高齢化や，世界的な新型コロナウイルス感染症（COVID-19）の影響により，これまでのように安定して作業品質の高い労働力を確保・育成することはもはや困難であることが浮き彫りになってきた．本研究で得られた成果は，環境に応じたエラーリカバリーの自動化に伴い，前工程の作業品質のばらつきや品種切り替え等の外乱的要因の影響を機械側の対応でケアすることができ，これからの FA に求められる持続可能でレジリエントな生産に貢献できる．

謝辞

本研究の機会を与えて頂き，学部四回生から修士課程修了までの三年間，さらに社会人博士として再び在籍した三年間にわたって懇切丁寧なご指導を賜りました，京都大学大学院工学研究科の椹木哲夫教授に心より御礼申し上げます．また，本論文の審査において副査を務めてくださいました，同研究科の松野文俊教授，松原厚教授からは，的確かつ有益な御意見，御助言を賜りました．厚く御礼申し上げます．

業務と並行して学位取得を目指すことに対し御理解，御支援を頂きました，三菱電機株式会社先端技術総合研究所の自律制御システム開発プロジェクトグループほか職場の皆様に深謝します．本研究で得られた知見を活かし，今後ともより一層精進していく所存です．

最後に，日頃より応援してくれた妻，暖かく見守ってくれた母，相談に乗ってくれた友人たち，そしてこれまでお世話になった全ての方々に感謝の意を表します．

参考文献

[1] 野田, 奥田, 田中, 永谷, 北明, 堂前, 椹木, 横小路, 堀口, 幸田, 宇津野, 松久, 水山, 小森, 泉井, 西脇: 次世代セル生産を実現するロボット知能化技術, 第 10 回計測自動制御学会システムインテグレーション部門講演会論文集, pp.766-769, 2009.

[2] 白土, 堂前, 奥田: 産業用ロボットの実用化技術最前線, システム/制御/情報, Vol. 60, No. 12, pp. 528-533, 2016.

[3] 加知, 吉本, 北上, 小泉: 生産情報との連携による工場エネルギー管理システムとその実装評価, 計測自動制御学会論文集, Vol. 50, No. 4, pp. 319-327, 2014.

[4] Hollnagel, Wears, Braithwaite: From Safety-I to Safety-II: a white paper, The resilient health care net: published simultaneously by the University of Southern Denmark, University of Florida, USA, and Macquarie University, Australia, 2015.

[5] Rodney A. Brooks: A Robust Layered Control System for a Mobile Robot, Robotics and Automation, IEEE Journal, Vol. 2, No. 1, pp. 14-23, 1986.

[6] Erann Gat: On Three-Layer Architectures, Artificial intelligence and mobile robots, pp. 195-210, 1998.

[7] L. Zhou, Z. Qidan, Y. Dongmei, Z. Xiuping: Research on subsumption architecture and application in dynamic programming of robot, 2004 International Conference on Intelligent Mechatronics and Automation, pp. 522-526, 2004.

[8] L. P. Kaelbling and T. Lozano-Pérez: Hierarchical task and motion planning in the now, 2011 IEEE International Conference on Robotics and Automation (ICRA), pp. 1470-1477, 2011.

[9] N. T. Dantam, Z. K. Kingston, S. Chaudhuri, L. E. Kavraki: Incremental Task and Motion Planning: A Constraint-Based Approach, Robotics: Science and systems, Vol. 12, 2016.

[10] Cashmore, Fox, Long, Magazzeni, Ridder, Carrera, Palomeras, Hurtos, Carreras: ROSPlan: Planning in the Robot Operating System, Twenty-Fifth International Conference on Automated Planning and Scheduling, 2015.

[11] 原田, 万, ラミレス, 山野辺, 辻: データベースに基づく産業用ロボットの作業動作計画, 日本ロボット学会誌, Vol. 37, No. 8, pp. 679-682, 2019.

[12] Peter Loborg: Error Recovery in Automation - An Overview, AAAI-94 Spring Symposium on Detecting and Resolving Errors in Manufacturing Systems, Stanford, Ca, USA, pp. 94-99, 1994.

[13] D. Romeres, D. K. Jha, W. Yerazunis, D. Nikovski, H. A. Dau: (2019, June). Anomaly detection for insertion tasks in robotic assembly using Gaussian process models, 2019 18th European Control Conference (ECC), pp. 1017-1022, 2019.

[14] A. Munawar, P. Vinayavekhin, G. De Magistris: Spatio-temporal anomaly detection for industrial robots

through prediction in unsupervised feature space, 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 1017-1025, 2017.

[15] F. Cheng, A. Raghavan, D. Jung, Y. Sasaki, Y. Tajika: High-accuracy unsupervised fault detection of industrial robots using current signal analysis, 2019 IEEE International Conference on Prognostics and Health Management (ICPHM), pp. 1-8, 2019.

[16] T. Chen, X. Liu, B. Xia, W. Wang, Y. Lai: Unsupervised anomaly detection of industrial robots using sliding-window convolutional variational autoencoder, IEEE Access, Vol. 8, pp. 47072-47081, 2020.

[17] H. Lu, M. Du, K. Qian, X. He, K. Wang: GAN-based Data Augmentation Strategy for Sensor Anomaly Detection in Industrial Robots, IEEE Sensors Journal, 2021.

[18] D. Park, H. Kim, Y. Hoshi, Z. Erickson, A. Kapusta, C. C. Kemp: A multimodal execution monitor with anomaly classification for robot-assisted feeding, 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 5406-5413, 2017.

[19] V. Narayanan, R. B. Bobba: Learning based anomaly detection for industrial arm applications, In Proceedings of the 2018 Workshop on Cyber-Physical Systems Security and Privacy, pp. 13-23, 2018.

[20] N. G. Odrey, G. Mejia: An augmented Petri Net approach for error recovery in manufacturing systems control, Robotics and Computer-Integrated Manufacturing , Vol. 21, No. 4, pp. 346-354, 2005.

[21] G. Chang, D. Kulic: Robot task error recovery using Petri nets learned from demonstration, Advanced Robotics (ICAR), 2013 16th International Conference on. IEEE, pp. 1-6, 2013.

[22] 松野, 白土, 黄, 福田: ロボットによるネジ締め作業における異常検出アルゴリズム, 日本ロボット学会誌, Vol. 30, No. 8, pp. 804-812, 2012.

[23] A. Nakamura, K. Nagata, K. Harada, N. Yamanobe, T. Tsuji, T. Foissote, Y. Kawai: Error Recovery Using Task Stratification and Error Classification for Manipulation Robots in Various Fields, Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference, pp. 3535-3542, 2013.

[24] 溝口, 池田, 來村: オントロジー工学基礎論 : 意味リンク, クラス, 関係, ロールのオントロジー的意味論, 人工知能学会誌, Vol. 14, No. 6, pp. 1019-1032, 1999.

[25] Lim, Suh, Suh: Ontology-based unified robot knowledge for service robots in indoor environments; IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, Vol. 41, No. 3, pp. 492–509, 2010.

[26] Olszewska, Barreto, Bermejo-Alonso, et al.: Ontology for autonomous robotics, 2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN), pp. 189–194, 2017.

[27] R. K. Gupta, B. Gurumoorthy: Feature-based ontological framework for semantic interoperability in product development, Advanced Engineering Informatics, Vol. 48, p. 101260, 2021.

[28] L. Wang, J. Hodges, D. Yu, R. S. Fearing: Automatic modeling and fault diagnosis of car production lines based on first-principle qualitative mechanics and semantic web technology, Advanced Engineering Informatics, Vol. 49, p. 101248, 2021.

[29] H. A. Simon: Bounded rationality in social science: Today and tomorrow, Mind & Society, Vol. 1, No. 1, pp. 25-39, 2000.

[30] 田中, 北野: サイモンの「限定合理性」の持つ意味と意義, 文学部心理学論集, No. 4, pp. 7-18, 2010.

[31] R. Fikes, N. Nilsson: STRIPS: A new approach to the application of theorem proving to problem solving, Artificial intelligence, Vol. 2, No. 3-4, pp. 189-208, 1971.

[32] T. Nguyen, S. Kambhampati: A heuristic approach to planning with incomplete strips action models, International Conference on Automated Planning and Scheduling, Vol. 24, No. 1, pp. 190-198, 2014.

[33] D. Aineto, S. Jiménez, E. Onaindia: Learning STRIPS action models with classical planning, International Conference on Automated Planning and Scheduling, Vol. 28, No. 1, pp. 399-407, 2018.

[34] Daniel S. Weld: An introduction to least commitment planning, AI magazine, Vol. 15, No. 4, pp. 34-56, 1994.

[35] 馬場口, 山田: 人工知能の基礎, pp. 78-94, 昭晃堂, 1999.

[36] H. L. Younes, R. G. Simmons: VHPOP: Versatile heuristic partial order planner, Journal of Artificial Intelligence Research, Vol. 20, pp. 405-430, 2003.

[37] A. Coles, A. Coles, M. Fox, D. Long: Forward-chaining partial-order planning, International Conference on Automated Planning and Scheduling, Vol. 20, No. 1, pp. 42-49, 2010.

[38] P. Bechon, M. Barbier, G. Infantes, C. Lesire, V. Vidal: HiPOP: Hierarchical Partial-Order Planning, STAIRS, pp. 51-60, 2014.

[39] Ivan Bratko: Prolog Programing for Artificial Intelligence (4th Edition), pp. 280-298, Pearson Education Canada, 2011.

[40] M. Pranzo, D. Pacciarelli: An iterated greedy metaheuristic for the blocking job shop scheduling problem, Journal of Heuristics, Vol. 22, No. 4, pp. 587-611, 2016.

[41] P. T. Zacharia, N. A. Aspragathos: Optimal robot task scheduling based on genetic algorithms, Robotics and Computer-Integrated Manufacturing, Vol. 21, No. 1, pp. 67-79, 2005.

[42] E. Bischoff, F. Meyer, J. Inga, S. Hohmann: Multi-robot task allocation and scheduling considering cooperative tasks and precedence constraints, Proceedings of the 2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 3949-3956, 2020.

[43] A. Tika, N. Gafur, V. Yfantis, N. Bajcinca: Optimal scheduling and model predictive control for trajectory planning of cooperative robot manipulators, IFAC-PapersOnLine, Vol. 53, No. 2, pp. 9080-9086, 2020.

[44] T. Bänziger, A. Kunz, K. Wegener: Optimizing human–robot task allocation using a simulation tool based on standardized work descriptions, Journal of Intelligent Manufacturing, Vol. 31, No. 7, pp. 1635-1648, 2020.

[45] S. K. K. Hari, A. Nayak, S. Rathinam: n approximation algorithm for a task allocation, sequencing and scheduling problem involving a human-robot team, IEEE Robotics and Automation Letters, Vol. 5, No. 2, pp. 2146-2153, 2020.

[46] Roger C. Schank: Conceptual dependency: A theory of natural language understanding, Cognitive Psychology, Vol. 3, No. 4, pp. 522-631, 1972.

[47] L. Bækgaard, P. B. Andersen: Using Interaction Scenarios to Model Information Systems, Aarhus School of Business, Denmark, 2008

[48] J. C. Macbeth, D. Gromann, M. M. Hedblom: Image Schemas and Conceptual Dependency Primitives: A Comparison, Joint Ontology Workshop, 2017.

[49] C. Galindo, J. A. Fernandez-Madrigal, J. Gonzalez, S. Alessandro: Robot task planning using semantic maps, Robotics and autonomous systems, Vol. 56, No. 11, pp. 955-966, 2008.

[50] M. Tenorth, M. Beetz: Representations for robot knowledge in the KnowRob framework. Artificial Intelligence, Vol. 247, pp. 51-169, 2017.

[51] John F. Sowa: Conceptual Graph Summary, Conceptual Structures: Current Research and Practice, pp. 3-66,

Ellis Horwood, New York London Toronto, 1992.

[52] John F. Sowa: Knowledge Representation: Logical, Philosophical, and Computational Foundations, pp. 502-510, Brooks Cole Publishing Company, 2000.

[53] Roger T. Hartley: A Uniform Representation for Time and Space and Their Mutual Constraints, Computers & Mathematics with Applications, Vol. 23, No. 6, pp. 441-457, 1992.

[54] O. Lebeltel, P. Bessiere, J. Diard, E. Mazer: Bayesian robot programming, Autonomous Robots, Vol. 16, No. 1, pp. 49-79, 2004.

[55] J. A. Ting, A. D'Souza, S. Schaal: Bayesian robot system identification with input and output noise, Neural Networks, Vol. 24, No. 1, 99-108, 2011.

[56] B. Saund, S. Choudhury, S. Srinivasa, D. Berenson: The blindfolded robot: A bayesian approach to planning with contact feedback, International Symposium on Robotics Research (ISRR), 2019.

[57] R. Jain, T. Inamura: Learning of tool affordances for autonomous tool manipulation, 2011 IEEE/SICE international symposium on system integration (SII), pp. 814-819, 2011.

[58] Karim A. Tahboub: Intelligent human-machine interaction based on dynamic bayesian networks probabilistic intention recognition, Journal of Intelligent and Robotic Systems, Vol. 45, No. 1, pp. 31-52, 2006.

[59] A. Xu, G. Dudek: Optimo: Online probabilistic trust inference model for asymmetric human-robot collaborations, 2015 10th ACM/IEEE International Conference on Human-Robot Interaction (HRI), pp. 221-228, 2015.

[60] 本村陽一: ベイジアンネットワーク：入門からヒューマンモデリングへの応用まで, 日本行動計量学会第7回春のセミナー, 2004.

[61] 渡辺, 萩原, 赤穂, 本村, 福水, 岡田, 青柳: 学習システムの理論と実現, pp. 75-95, 森北出版, 2005.

[62] 椹木哲夫: 意思決定論に基づく異種情報源の融合と協調, 計測と制御, Vol. 32, No. 3, pp. 229-236, 1993.

[63] J. S. Breese, M. R. Fehling: Decision-Theoretic Control of Problem Solving: Principles and Architecture, Breese, Fehling: Decision-theoretic control of problem solving: Principles and architecture, arXiv preprint arXiv: 1304.2343, 2013.

[64] E. Horvitz, G. Rutledge: Time-Dependent Utility and Action Under Uncertainty, Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence, pp. 151-158, 1991.

[65] G. Bastos, F. Ramos, L. Sauza, C. Ribeiro: TIME-DEPENDENT UTILITY DECISION MAKING USING THE TIMDP MODEL, ResearchGate, 2011.

[66] M. Baucells, F. H. Heukamp: Probability and time trade-off, Management Science, Vol. 58, No. 4, pp. 831-842, 2012.

[67] T. Epper, H. Fehr-Duda, A. Bruhin: Viewing the future through a warped lens: Why uncertainty generates hyperbolic discounting, Journal of risk and uncertainty, Vol. 43, No. 3, pp. 169-203, 2011.

[68] 横小路, 横井: World Robot Summit 製品組立チャレンジ, 計測と制御, Vol. 56, No. 10, pp. 798-804, 2017.

[69] 橋本浩一: ビジュアルフィードバック制御と今後, 日本ロボット学会誌, Vol. 27, No. 4, pp. 400-404, 2009.

[70] 荒井翔悟: ビンピッキング & キッティングのためのロボットシステム, 日本ロボット学会誌, Vol. 37, No. 10, pp. 938-942, 2019.

関連業績一覧

学術論文

[1] 松岡, 椹木, 堀口, 中西: 階層型プランニングシステムによる産業用ロボットの自律的エラーリカバリー, 計測自動制御学会論文集, Vol. 53, No. 1, pp. 80-89, 2017.

[2] 松岡, 椹木, 前川: 失敗の深層意味情報に基づくロボットのエラーリカバリー行動計画, 計測自動制御学会論文集, Vol. 57, No. 1, pp. 25-36, 2021.

[3] S. Matsuoka, T. Sawaragi: Recovery Planning of Industrial Robots Based on Semantic Information of Failures and Time-Dependent Utility, Advanced Engineering Informatics 掲載予定 (2021-12-19 Accept).

国際学会発表 (査読あり)

[1] S. Matsuoka, T. Sawaragi, Y. Horiguchi, H. Nakanishi: Hierarchical planning for error recovery in automated industrial robotic systems, Proceedings of 2016 IEEE International Conference on Systems, Man, and Cybernetics, pp. 1406-1410, 2016.

国内学会発表

[1] 松岡, 椹木, 堀口, 中西: 組立ロボットのエラーリカバリーのための半順序性を活用した階層型プランニング, 第 59 回システム制御情報学会研究発表講演会講演論文集, 152-4, 2015.

[2] 松岡, 椹木, 堀口, 中西: 階層型プランニングシステムによる産業用ロボットの自律的エラーリカバリー, 第 16 回計測自動制御学会システムインテグレーション部門講演会講演論文集, pp. 449-452, 2015.

[3] 松岡, 椹木, 前川: 失敗の深層意味情報に基づくロボットのエラーリカバリー行動計画, 第 20 回計測自動制御学会システムインテグレーション部門講演会講演論文集, pp. 2409-2413, 2019.

賞罰

[1] 第 16 回計測自動制御学会システムインテグレーション部門講演会 (SI2015) 優秀講演賞, 2015.