強化学習を利用した 対話的プロセスモデル構築支援

Interactive support for constructing process model using Reinforcement Learning

沖田勇馬 †， 飯島正 ‡
Yuma Okita†， and Tadashi Ijima‡
† 慶應義塾大学大学院 理工学研究科
‡ 慶應義塾大学 理工学部
†Graduate School of Science and Technology，Keio Univ.
‡Department of Science and Technology，Keio Univ.

要旨

近年，働き方改革の浸透と共に業務プロセス改革（BPR; Business Process Reengineering）が注目されている．一方でビジネスの規模拡大やプロセスの複雑化・多様化によって業務プロセスのモデリングは難しくなりつつある．迅速かつ正確なモデリングを目標とし，従来必要であったモデリングの専門家を介すことなく，対話的な操作を通じてユーザ単独でのモデリングを支援するシステムを提案する．同システムは，ユーザが入力した複数の実行系列に対し，プロセスモデル発見アルゴリズムによってプロセスモデルを生成し提示する．このプロセスモデル発見アルゴリズムに強化学習を使うことで, 洗練化・精緻化のサイクルを回した時の総合的な実行時間の短縮化とモデル生成の精度の向上を狙う. 同アルゴリズムのシステムでの初回実行時のモデル生成の精度について報告を行う.

1. はじめに

近年，働き方改革の浸透と共に業務プロセス改革（BPR; Business Process Reengineering）が注目されている. 本研究計画では, 与えられた複数の実行系列からからアルゴリズムによってプロセスモデルを構築し, 加えて生成モデルの構造やタスクの意味情報に基づくフィードバックから入力改善を促すシステムを提案してきた. 本報告では, その一部として例からモデルを構築するアルゴリズムに強化学習を利用する試みについて報告する. 第 2 節では構築支援システムの概要を説明し, 第 3 節では強化学習を利用したモデル構築アルゴリズムについて述べる. そこで強化学習を用いる理由とそのための工夫を説明し, 第 4 節では評価実験の結果と考察, 第 5 節ではまとめを述べる.

2. 対話的プロセス構築支援システム

2.1. 業務プロセスモデルの作成

業務プロセスモデルとは第一節で述べたように BPR の一環として, 業務の理解・共有・分析を目的とした業務の流れ（業務プロセス）の定義である．主にシステム開発や業務改善の場において活用され, 専門家が実際に業務を行っているユーザーからの情報提供を得て作成する. しかし情報社会の成熟により業務の流れは複雑になりつつあり, 業務プロセスのモデリングには多大な時間と困難を伴いつつある. 業務プロセスの作成を支援する研究としてはプロセスモデル記述言語がある. 統一モデリング言語である UML(United Modeling Language) を使用したケースや，業務プロセスモデリング・ビジネスサイドに特化して作られたグラフィカル表現記法 BPMN(Business Process Modeling Notation)．また, 数学的形式を起点とした PetriNet を業務プロセスに適応するケースや，この PetriNet から業務プロセスパターン表現に拡張された YAWL など，多種多様存在する.

しかし実際のプロセスモデルの作成にあたり, タスクを含む各種ノードの配置を支援する研究は行われておらず, 業務プロセスのモデル作成はその専門家に頼らざるを得ないのが実情である.

2.2. システム概要

本研究では従来必要であったモデリングの専門家を介すことなく，対話的な操作を通じてユーザー単独でのモデリングを支援するシステムを提案する．ユーザー自身がプロセスモデルを作成することで業務の知識を確実に反映させ，またシステムを介して迅速にモデル化することで，精緻化・洗練化のサイクルを素早く実行することを狙いとしている．従来のモデリングの専門家の役割を代替させるため，ユーザーによる入力から業務プロセスモデルを生成するモデリング機能と，ユーザー入力の改善を促すコンサルティング機能を持ち, 次の図 1 のような改善・構築・評価のサイクルを実行する.

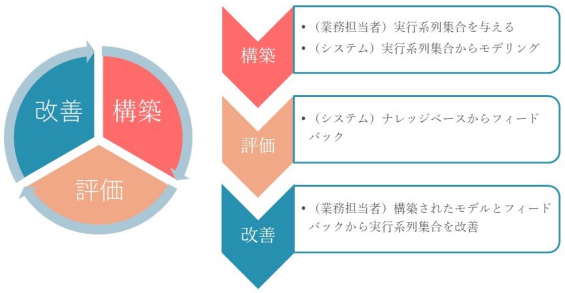


図 1: 対話的プロセスモデル構築支援システム

2.3. モデル構築アルゴリズム

この機能はプロセスモデル記述言語の知識と，モデルを構築するためのロジックを提供する．この機能では，業務の流れをいくつかの例（実行系列集合）から，出力として業務プロセスモデルを返す．本報告ではシステムが持つ機能の中でも, このモデル構築アルゴリズムに絞った研究について報告を行う.

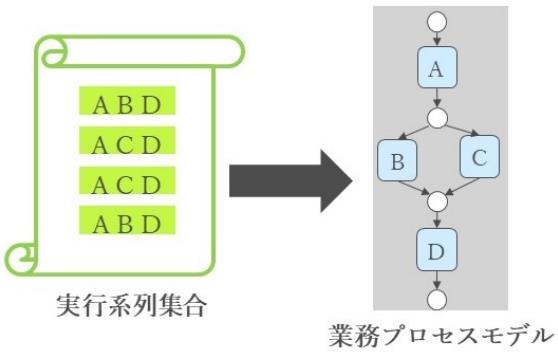


図 2: モデル構築アルゴリズム

類似した機能を追求した研究にプロセスマイニングがある. プロセスマイニングとは，情報システムのイベントログから有益な情報を抽出し活用する技術であり，情報システムのイベントログからシステムに内在する業務プロセスモデルを発見するアルゴリズム（Process Discovery）が研究されている．この技術は，情報システムを構築するための礎となった業務プロセスモデルに精通した技術者が失われてしまったり，アップデートの繰り返しでシステムの全容が把握できない場合に，情報システムの業務プロセスモデルを知るための一つの手段として期待されている.

本研究の目的は, ユーザーの与える実行系列を正としてユーザーの望む業務プロセスモデルを対話的に作成することである. 一方,Process Discovery は欠損やノイズを含むかもしれないイベントログから情報システムの基礎となった業務プロセスモデルを見つけることが目的なことに注意したい. 次節では,Process Discovery の技術をそのまま適用するよりも, よりふさわしいアルゴリズムの開発を試みる.

2.4. ペトリネットと WF-net

ペトリネット (Petri Net) とは，C.Petri の学位論文を出発点として展開されてきた，並列非同期同時進行複数のプロセスからなる離散事象システムの挙動を表現できるモデルである．並列分散システムや通信プロトコルばかりではなく化学反応やシステム生物学に至るまで幅広く応用されている．ペトリネットには，望ましい性質を備えるように，構造的に制約を加えたサブクラスが複数知られている. その内の一つが WF-net(ワークフローネット) であり, 業務プロセスを表現するために望ましい性質を持つこの記述言語を本研究では業務プロセスモデルの記述のために用いる.

3. 強化学習を用いたモデル構築アルゴリズム

3.1. 強化学習の学習モデル

強化学習とはある環境内におけるエージェントが, 現在の状態を観測し, 取るべき行動を決定する問題を扱う機械学習の一種である. 本報告における学習では以下のように問題を設定する.

環境 ：Place と Arc の組み合わせ

行動 ：Arc の追加

報酬 ：モデルが終端状態のときに評価関数の値を与える

エピソードの終端状態 ：全ての Transition が入出力 Arc を 1 つ以上持つ

解は状態価値関数が最も高い Place と Arc の組み合わせとし, 価値の同じ候補が複数存在する場合は,Place の数と Arc の数の和が最も少ないものを解として選択する. 具体的なイメージは次の図 3 の通りである.Arc をモデルに 1 本ずつ追加し, 終端状態に対して報酬を与える. 終端後は再度 Arc を追加する前の状態に戻し, 再び Arc を追加することを試みる.

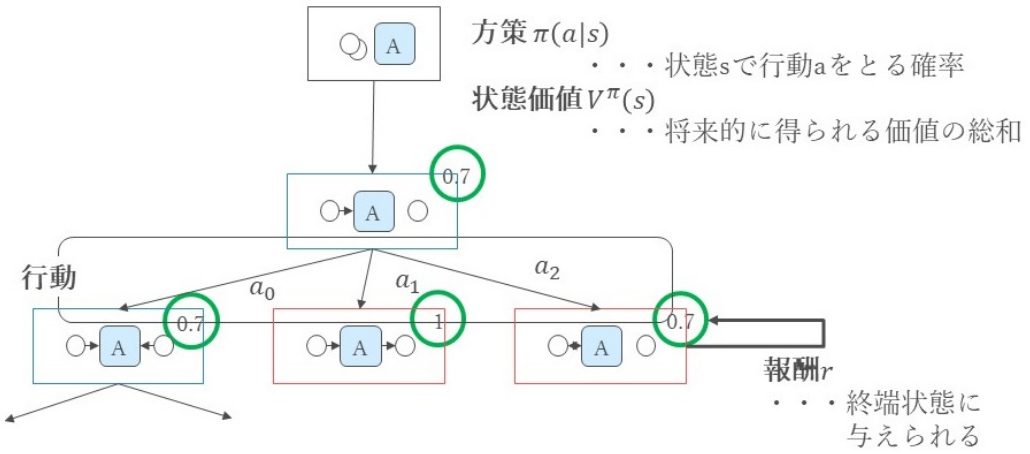


図 3: 強化学習のモデル

3.2. 強化学習の学習方法

学習方式としては以下の式で表される Q 学習を用いた. 方策の選択にはグリーディ法を用いた. 学習は 10000 回試行ごとに状態価値マップのサイズを計測し, 増分が規定数を下回ったときに学習を終了する. 学習は Place が 1 個の状態から行い, 学習終了時に規定の状態価値を上回る状態が得られない場合は,Place を 1 個追加して再度学習を繰り返す.

3.3. 評価関数

業務プロセスモデルの評価軸はプロセスマイニング研究の第一人者たる van der Aalst, Wil M.P. らが定める 4 つの特性 [1] の内,Fitness と Precision を評価関数として評価する.

- Fitness
  Fitness が優れているプロセスモデルは, イベントログ（提案システムの実行系列集合）と同様の動作を示すことができる.

- Simplicity
  プロセスモデルの簡潔さ, つまり Place と Arc の少なさを指す.

- Generalization
  プロセスモデルがイベントログに過剰適合しているか, そのモデルの汎化能力を指す指標である. 学習データのノイズや欠損の影響をどこまで排せるか重要となる.

- Precision
  プロセスモデルが未学習であるかの指標, つまりプロセスモデルが許されない動作を行ってしまう程度を指している.

本報告では Fitness と Precision に対応する 2 つの評価関数を用意した. Fitness を評価する評価関数は ProcessDiscovery 手法の中でもソフトコンピューティング手法の 1 種である GeneticAlgorithm for Process Mining[2] で用いられている次の評価関数 (1)(2) を用いる. この関数は各実行系列が一部だけでも実行できればそれを評価できることが特徴で, 実行時の Token の不足や余剰を Punisshment として減点する.

$$PF_{complete}\left(L,N\right)=\frac{allParsedActivities\left(L,N\right)-punishment}{numActivitiesLog\left(L\right)} \tag{1}$$

$$\begin{aligned} punishment &= \frac{allMissingTokens\left(L,N\right)}{numTracesLog\left(L\right)-numTracesMissingTokesn\left(L,N\right)+1} \\ &+ \frac{allExtraTokensLeftBehind\left(L,N\right)}{numTracesLog\left(L\right)-numExtraTokensLeftBehind\left(L,N\right)+1} \end{aligned} \tag{2}$$

Precision を評価する評価関数を次の式 (3) で表す. 出力モデルをランダムに実行し, モデルから出力されるログが学習データとなる実行系列集合に含まれる割合を示している.

$$PF_{precision}\left(L,N\right)=\frac{numParcedTraceInLog\left(L,N\right)}{numParsedTrace\left(N\right)} \tag{3}$$

Generalization について, 本報告で提案する対話的モデル構築支援システムは使用者の用意する実行系列集合を全て正でかつそれ以外を負として考える. 入力の誤りへの対応はコンサルティング機能に任せるためである. 従って Generalization は考慮しない.

Simplicity は上記 2 つの評価関数の値が規定値を上回る場合,Place と Arc の数を比較し少ないものを解として選択することで評価を行う.

3.4. インクリメンタルな学習

業務プロセスモデルの記述方式として提案システムで採用する WF-net はタスクの数によって状態数が

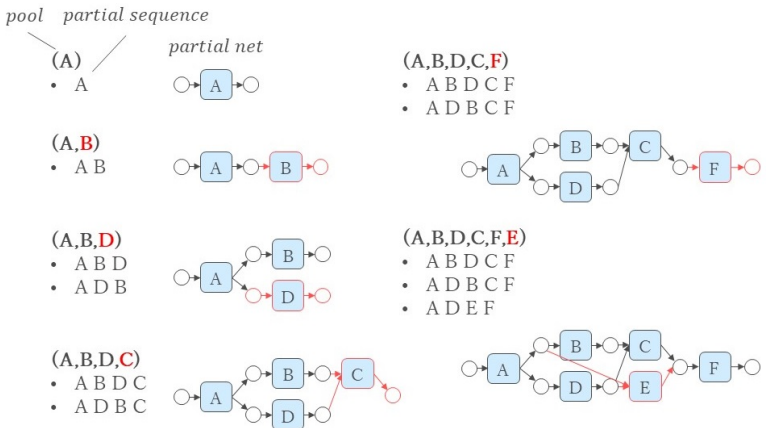
$$O\left(2^n\right)$$

に従って増加する. 単純に強化学習を行うだけでは, タスクの数が少ないプロセスモデルであっても学習時間が膨大なものになってしまう. 探索すべき状態数を減らす工夫として, 実行系列集合をいくつかの段階に分けて, 少しずつモデルを大きくする方式をとる. 具体的な手順を以下に示す.

Algorithm 1 インクリメンタルな学習

$pool=\{\},N=null$
$T=\{t|\exists_{\sigma\in S}t\in\sigma\}$
for all $t\ such\ that\ T$ **do**
  $pool=pool\cap t\left(\forall_{\sigma\in S}t\in\sigma\right)$
  $pS=makeParialSequence\left(pool,S\right)$
  $N=RL\left(N,pS\right)$
end for
$Incremental\ Learning\left(S\right)=N$

また Algorithm1 を視覚的に表したものが次の図 4 である. タスクの少ない状態から得られる部分モデルに Arc を付け足す形で学習を進めることで,Place と Arc の組み合わせを大きく減らすことができる.

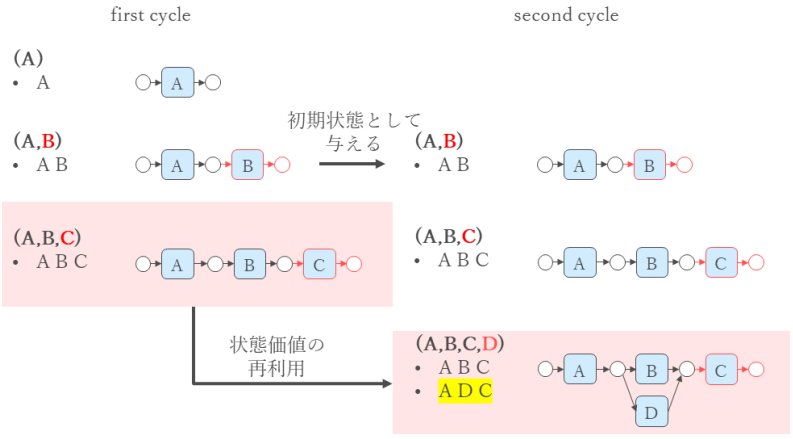


learning.jpg

図 4: インクリメンタルな学習

3.5.2 回目以降の学習

本報告で提案する対話的モデル構築支援システムはモデル構築・フィードバック・入力の修正を複数回繰り返すことを前提としている. 繰り返しで入力がさほど変更されないのであれば, 強化学習で得られた状態価値関数を再利用することで 2 回目以降の学習を高速に行うことができる. ただし本節で述べる機能は実装まで至っていないため, 構想を述べるだけに留める.



cycle.png

図 5: 2 回目以降の学習

4. 実験

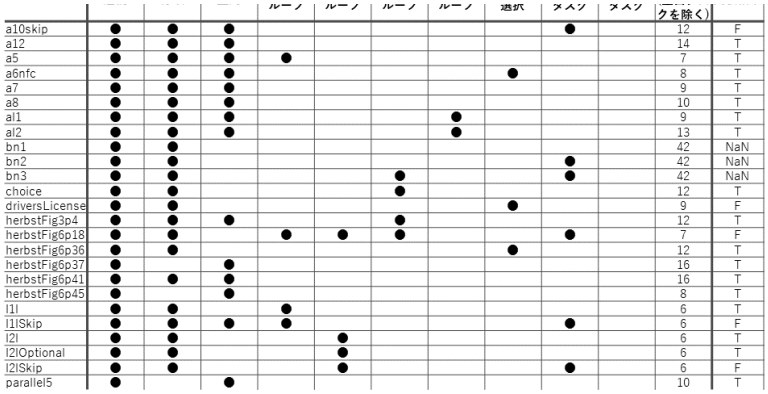
本節では 3.1 節から 3.4 節までに解説した初回の学習におけるアルゴリズムにおいて, モデルが正しく学習できることを確認する.

4.1. 実験方法

学習のための実験データとして Technische Universiteit Eindhoven の A. K. A. de Medeiros,A. J. M. M. Weijters,W. M. P. van derAalst は論文 [2] で ProcessDiscovery アルゴリズムの評価に用いられたプロセスモデルを使用する. 実験の手順としては, まずプロセスモデルをランダム 10,000 回動作させ, 十分なイベントログを用意する. これを学習アルゴリズムの入力データ, つまり十分な量のサンプルが集められた実行系列集合とし与える. そしてアルゴリズムから最終的に出力されるプロセスモデルが元のモデルと同じものになるかを True or False で判定を行う.

4.2. 実験結果と考察

実験結果は次の図 6 の様になった. 実験結果の T は True,F は False,NaN は実行時間が多すぎで実行できなかったことを示している.



				ループ	ループ	ループ	ループ	選択	タスク	タスク	クを除く)	
a10skip	●	●	●						●		12	F
a12	●	●	●								14	T
a5	●	●	●	●							7	T
a6nfc	●	●	●					●			8	T
a7	●	●	●								9	T
a8	●	●	●								10	T
al1	●	●	●				●				9	T
al2	●	●	●				●				13	T
bn1	●	●									42	NaN
bn2	●	●							●		42	NaN
bn3	●	●				●			●		42	NaN
choice	●	●				●					12	T
driversLicense	●	●						●			9	F
herbstFig3p4	●	●	●			●					12	T
herbstFig6p18	●	●		●	●	●			●		7	F
herbstFig6p36	●	●						●			12	T
herbstFig6p37	●		●								16	T
herbstFig6p41	●	●	●								16	T
herbstFig6p45	●		●								8	T
l1l	●	●		●							6	T
l1lSkip	●	●	●	●					●		6	F
l2l	●	●			●						6	T
l2lOptional	●	●			●						6	T
l2lSkip	●	●			●				●		6	F
parallel5	●		●								10	T

図 6: 実験結果

直列・分岐・並列・自己ループ・長さ 2 以上のループ・非自由選択のみを含むプロセスモデルであれば正しくモデルを作成することができたが, 元のモデルに空白タスクを含むプロセスモデルは正しい出力を得られなかった. これはアルゴリズムとして空白タスクを全く扱えていないためであり, 当然の結果といえる. 一部のモデルに関しては, 現実的な処理時間を超えても処理が終了しなかったため, 途中で学習を停止した. これは 3.4 節でも述べたようにタスクの数が増えるにつれて, 処理時間か指数関数的に増大するためで, 実際にタスクの数はいずれのモデルも 40 個と他のモデルと比べて 20 個から 30 個タスクが多かった.

4.3. まとめ

例からのプロセスモデル構築支援に強化学習を使う試みとして, 初回の学習は一部の構造を除き狙い通りにモデルを出力できることが確認できた. 一方で処理時間の長さは否めず, 今後は強化学習のテクニックを利用して上手く計算を行っていく. また,2 回目以降の学習に関するアルゴリズムは具体的な実装が今後の課題となる.

参考文献

[1] van der Aalst, Wil M.P. : “Process Mining:Discovery: Conformance and Enhancement of Business Processes.”, Springer-Verlag Berlin Heidelberg, 2011.

Data Mining and Knowledge Discovery, Vol.14, Issue2, Springer US, pp.245-304,2007.

[2] Alves de Medeiros, Ana Karla: “GeneticProcessMining.”, Beta Dissertation Series, D89, NUR 983, University Press Facilities, Eindhoven,2006.