2G-03

チャットボットにおける ポリシーによるシナリオチェック方式

岩崎 信也† ‡　薦田 憲久‡　藤原 融‡

株式会社日立システムズ†　大阪大学大学院情報科学研究科‡

はじめに

チャットボットの開発では，シナリオと呼ばれるチャットボット用の対話テンプレートを作成しなければならない[1]．シナリオは，チャットボットの開発者により専用の GUI（Graphical User Interface）ツールを用いられて作成される[2]．しかし，シナリオを作成する中で対話の流れを誤って指定するなどの作成不良が発生する．不良発見には時間を要するうえ，見逃しが起こりやすい．そこで，本講演では，シナリオの作成不良の発見時間や見逃しを削減することを目的とし，シナリオ開発者がシナリオのあるべき状態をポリシーとして記述することで，シナリオの不良を機械的にチェックする方式を提案し，その効果を評価する．

チャットボットのシナリオ作成

チャットボットのシナリオでは，「利用者に対してどのような質問を提示するか」「利用者の返答によって，次の質問をどのようにするか」といった，ユーザーとの対話の内容や対話の流れを定義する．

シナリオは，ステップとその間の遷移，スロットで構成される．

(1)　ステップ

チャットボットが，ユーザーに発言するメッセージ（発話文）や選択肢文，ユーザーからの返答に対する処理をステップとして定義する．

(2)　ステップ間の遷移

条件付き遷移によってステップを接続することで対話の流れを構築する．

(3) スロット

ステップでのユーザーの発話や選択結果などの結果データを保存するための専用領域を定義する．定義したスロットは，別のステップや遷移条件で利用することができる．

シナリオの作成者がシナリオを作成する際，対話の流れを誤って指定するなどの作成不良が発生する．例えば，遷移条件を誤って記述し意図しない遷移となり対話が堂々巡りしたり，あるスロットを利用するステップがスロットに値を記録するステップより前に実行され参照不良となってしまったりすることなどがある．

不良を防ぐためには，シナリオの作成者がシナリオを作成しながら，随時対話を模擬し，シナリオが意図どおりに動くかチェックする．しかし，シナリオはチャットボットの専門知識を持たない作成者が作成・修正することが多く．シナリオのどこが原因かを発見するのに時間を必要とする．

ポリシーによるシナリオチェック

この問題に対して，シナリオのあるべき状態や動作をポリシーとして記述し，対話の流れが誤っている可能性がある箇所を機械的にチェックし，作成者に提示する方式を提案する．

この方法では，シナリオ向きのポリシー定義方法と，ポリシー違反の確認方法が必要である．

ポリシーは，定義 1 に示すように，シナリオのどこに対するポリシーなのかを指定する「対象」，対象のあるべき状態を指定する「条件」の組みで定義する．「対象」は，「要素」と「範囲」から成り，「要素」は，ポリシーの対象となるシナリオの構成要素を表し，「範囲」はポリシーの対象が，特定の要素なのか，すべての要素なのかを表す．「条件」は，「対象」がどのような状態のときにポリシーに準拠しているかを表し，対象の実行回数，読込回数，書込回数，順序条件のいずれかの条件を記述する．ポリシーの条件指定に応じ，計算量などからリアルタイムチェックと保存時チェックの 2 種類がある．

定義 1．ポリシー定義

ポリシー	＝	対象＋条件
対象	＝	要素＋範囲
範囲	＝	全体｜特定
要素	＝	ステップ｜スロット｜遷移
条件	＝	実行回数｜読込回数｜書込回数｜順序条件

† Hitachi Systems, Ltd.

‡ Graduate School of Information Science and Technology, Osaka University

表1 ポリシーの記述例

#	要素	範囲	条件
1	スロット	全体	書込回数：1回以下（リアルタイム）
2	ステップ	特定（ステップ[2]）	実行回数：1回（保存時）

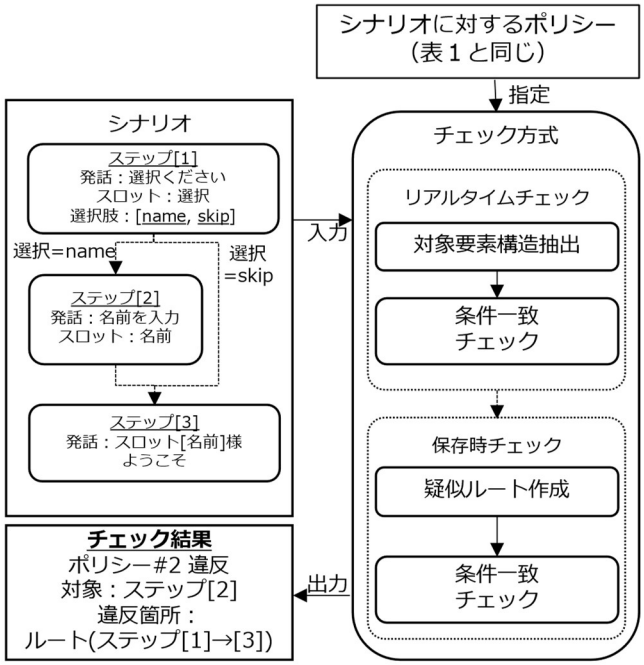

図1 ポリシーチェックの流れ

ポリシーの記述例を表1に示す．ポリシーは，シナリオを作成するためのGUIツールから記述する．一般的なシナリオで有効なポリシーは，テンプレートとして用意し，シナリオ作成者が必要に応じて修正する．

表1のポリシーのチェックを例に，図1左のシナリオのチェック処理を図1右に示す．

リアルタイムチェックは，シナリオの各要素をシナリオの編集のたびにチェックする．まず，ポリシーごとに対象要素を抽出し，その後，条件に一致するかを判定する．図1左のシナリオでは，スロット「選択，名前」にそれぞれステップ[1]，[2]が書き込むことを抽出し，条件一致をチェック，準拠しているため，表1の#1のポリシーは問題なしとなる．

保存時チェックは，シナリオの保存時に対話時のステップの流れ（ルート）の候補を網羅的に作成し，作成したルートから条件に一致するかを判定する．例えば，表1の#2「要素：ステップ，範囲：特定(ステップ[2])，条件：実行回数：1回」のポリシーでは，指定された対象（ステップ[2]）が実行されないルートがないかをチェックする．図1左のシナリオでは，ルートとして「ステップ[1]→[2]→[3]」，「ステップ[1]→[3]」の2つが作成され，「ステップ[1]→[3]」の流れでステップ[2]を実行していないため，ポリシー違反となる．

評価

提案方式であるポリシーによるチェック方式の有効性を，従来手法での不良の発見時間と，ポリシーの記述時間の比較で評価する．

従来のチェック方式では，誤った定義内容から，どのステップ・遷移・スロットに問題があるかを確認するには，原因となりそうな箇所に確認用のステップを追加して再チェックすることもあり，特にシナリオ作成に慣れていない作成者では時間を要する．このため，不良1件あたり数十分～数時間程度を発見・修正に要することが経験上分かっている．シナリオの規模により数個～数十個の不良が発生するため，シナリオ1件のチェックに要する時間は数時間～数十時間になり，その負担から十分にチェックせずに不良を見逃すことにもなる．

一方，提案方式におけるポリシーの記述は，標準テンプレートを調整すればよいため，1シナリオあたり10分程度で完了する．なお，シナリオのチェック時間は10秒程度である．

このように，作成シナリオの規模が小さく発生不良が2個，不良あたりの対応時間を10分とした場合でも，提案手法により従来と比べて50%程度の作業時間の削減が見込める．

まとめ

シナリオ開発者がシナリオのあるべき状態の指針をポリシーとして記述することで，シナリオ定義の誤りを機械的にチェックする方式を提案した．結果として，提案方式はシナリオの不良原因の発見時間や，不良の見逃しを削減することができ，ポリシーの記述時間も小さいことから有効性を確認できた．

参考文献

[1] 岩崎信也, 津村直哉：チャットボットサービスの変遷とそれを支える構成技術, 情報処理, Vol.62, No.10, e12-e18 (2021).

[2] S. Shukla, L. Liden, S. Shayandeh, E. Kamal, J. Li, M. Mazzola, T. Park, B. Peng, and J. Gao., 'Conversation Learner - A Machine Teaching Tool for Building Dialog Managers for Task-Oriented Dialog Systems', Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp. 343–349 (2020).