社団法人 情報処理学会　研究報告
IPSJ SIG Technical Report

2004－MBL－31　(10)
2004－ＩＴＳ－19　(10)
2004／11／11

1bit 状態遷移表シナリオ記述方式のスロット状態拡張に関する検討

関口 真理子 †　　荒金 陽助 ††　　阿部 伸浩 †

† NTT アイティ株式会社　　†† NTT 情報流通プラットフォーム研究所

あらまし カーナビや携帯電話などのモバイル端末の普及に伴い，スロットフィリング型音声対話による情報検索システムに注目が集まってきている．しかし，従来のシナリオ記述方式では，シナリオ追加・修正などの管理性，回線切断時の状態復帰などの頑強性において課題が残されている．これらの課題に対して筆者らは，1bit 状態遷移表を提案しているが，汎用性の面でいくつかの課題が残置されていた．そこで，本稿では「必須スロット数」,「スロットのデフォルト値」,「確認応答の要否」の導入を検討し，より汎用的なシナリオを表現可能な拡張 1bit 状態遷移表を提案する．

キーワード シナリオ記述方式, テレマティクス, 音声対話システム, 状態遷移表, 対話制御

A Study for Extended Scenario Description Method based on 1 bit State Transition Matrix

Mariko Sekiguchi†　　Yosuke Aragane††　　Nobuhiro Abe†

† NTT IT　　†† NTT Information Sharing Platform Laboratories

Abstract With the proliferation of mobile technique such as car navigation system and cellular phone, slot-filling based spoken dialog system for information retrieval is a topic of growing relevance. Generally, these systems with complex tasks need complex scenarios. So scenario design which has robustness and easiness to create/maintain scenarios is important. We have proposed 1 bit state transition matrix that can meets the requirements. But there are some problems left unsolved in its versatility. In this paper, we proposed an extended 1bit state transition matrix applying necessary slot number, default value, and the skip of confirmatory response of the system.

Keywords Scenario Description Method,Telematics, Spoken Dialogue System, State Transit M

1　はじめに

モバイル技術の急速な発達に伴い，ユビキタス環境からの情報アクセス技術への期待が高まってきている [1]．我々はその中でも特にテレマティクスをターゲットとしている．テレマティクスでは，運転しながらの情報アクセスとなるため，画面を注視することが出来ない，両手が塞がっている，など，入出力インタフェースの制限が厳しい．そこで，我々は音声インタフェースに注目した．テレマティクス環境では画面で現在の対話状況を把握することが困難であるため，常にその時の対話状況に応じた適切なガイダンスを出力して，ユーザの理解を助ける必要がある．また，ユーザの負担を軽減するために，ユーザの柔軟な発話を許容できることが望ましい [2]．このような音声インタフェースの特徴により，音声対話システムの対話シーケンスは複雑になる傾向があると考えられる．実サービスにおいては，様々なコンテンツに応じた対話シーケンスを構築する必要がある．従って，複雑なシナリオであっても追加/管理が容易で，かつ頑強性のあるシステム構築が要求される [3][4]．また，テレマティクス環境では通信状況が不安定であるため，対話中の回線切断に対応することも重要である [5]．従来の Web を用いたテ

レマティクスサービスでは，cookie を利用することでユーザ ID などのパラメータ保持を行ってはいるが，状態遷移の管理は行っておらず，再接続時にメインメニュー（トップメニュー）から再び階層をたどる必要があった．しかしこれは，時間的余裕が無く，再入力のハードルが高いテレマティクス環境においては，ユーザの満足度を極端に低下させる要因となりうる．そこで，回線切断の際に中断場所からのサービスの復帰が可能であるようなシステム構築が要求されるのである．

そこで我々は，上述の要求を満たすシナリオ記述方式として，1bit 状態遷移表を提案してきた [6]．これは，4 つのスロット状態を定義することで入力に対する状態遷移先を 0/1 の 1bit で表現することで，シナリオの追加/管理などの運用性や回線切断時のシナリオ状態再現を実現している．しかしながら，スロット数の動的管理や確認応答の煩雑さなどに課題があった．そこで本稿では，これらの課題を解決する拡張方式について検討する．

以下，2 章でテレマティクス用音声対話システムの為のシナリオ記述方式として状態遷移表に注目し，その利点と課題を述べる．次に，2 章で述べた課題を解決するシナリオ記述方式として，著者らが提案している 1bit 状態遷移表について，説明する．続いて，4 章では 1bit 状態遷移表の汎用化を目的とした拡張方式を提案する．最後に，5 章で本稿をまとめる．

2　シナリオ記述方式

2.1　スロットフィリング方式

テレマティクスサービスは主に運転中の使用を想定しているため，利用目的は「食事検索」や「駐車場検索」など，地点を検索してナビと連携をとるようなタスクが多い．これらのタスクでは，地点を検索するためのキーワードをユーザから取得することが対話の目的となる．このことから，音声対話システムとしてはスロットにキーワードを埋めていくスロットフィリング方式 [7][8] が適していると考えられる．スロットフィリング方式では，入力の手順は規定せず，スロットの状態を管理することでフローを進めることになる．従って文脈を細かく考慮した応答を返すシナリオを実現することは難しいが，ユーザが自由に要求を発声できるシナリオを，比較的簡単に実現することが可能である．

また，テレマティクスサービスでは，トンネルなどでの通信断が発生するため，中断した箇所からのフローの再開を実現する中断復帰機能が要求されるが，その点においてもスロットフィリング方式は適していると考えられる．端末側で常に現状態を保持しておき，再接続の際にその情報をサーバ側に送信することで，中断復帰機能の実装を容易に行うことが可能である．

S　「食事検索を行います．お店の種類を『和食か中華』のように 2 つお話ください」
U　「うどんか蕎麦」（一括入力）
S　「うどんか蕎麦ですか？」
U　「パスタか卵料理」（訂正入力）
S　「パスタか卵料理ですか？」
U　「卵料理はやめてピザにして」（部分否定訂正入力）
S　「パスタかピザですか？」
U　「はい」
S　「では，パスタかピザのお店を探します.」

図 2: 対話例 1

以上の特徴から，我々はスロットフィリング方式に注目した．また，スロットフィリング方式を実現するシナリオ記述方式として，可読性，保守性において優れていると考えられる状態遷移表に注目した．状態遷移表によるシナリオ記述方式の例を図 1 に示す．

2.2　スロットフィリング方式の課題

音声対話の特徴のひとつとして，柔軟な入力が可能である点があげられる．我々は，柔軟な入力として，一括入力，訂正入力 (上書き)，部分否定訂正入力を対象とした．このような発声を許すシステムの対話例を図 2 に示す．前節で述べたように，状態遷移表はテレマティクス用音声対話システムのシナリオ記述方式として適していると考えられる．しかし，上述のような柔軟な発声を許す複雑なタスクを対象にした場合，状態遷移表においても解決されないスロットフィリング方式の課題が残る．以下，課題として遷移先のチェックと条件分岐の抽出について述べる．

2.2.1　遷移先のチェック

状態遷移表では，全ての状態と入力に対し対話例を考え，シナリオ作成者が遷移先をチェックする必

state transition					process		
state	food	food food	yes	no	grammar	script	guidance
1	2	3	-	-	food.grm	-	“What type of food do you like?”
2	2	3	5	1	yn.grm	food.scr	“*slot*[0]?”

図 1: 状態遷移表の例

Case:1
U「蕎麦かうどん」
S「蕎麦かうどんですか？」
U「蕎麦じゃなくてパスタ」
S「パスタかうどんですか？」

Case:2
U「蕎麦かうどん」
S「蕎麦かうどんですか？」
U「蕎麦はやめてうどん」
S「うどんですか？」

図 3: 対話例 2

要がある．この状態遷移表の特徴により，以下に示す問題が起こる可能性がある．

- 複雑なタスクの場合，状態遷移表の規模が大きくなり，シナリオ作成者による遷移先のチェックに負荷がかかる．
- シナリオ作成者が遷移先を間違えて記述したことにより，矛盾が起こる．例としては，あるスロットへの入力に対し，遷移先として他のスロットが埋められた状態が記述されている場合などが挙げられる．
- シナリオ作成者が状態遷移ルールを明確に把握していなかった場合，一貫性の無いシナリオになる．

2.2.2 条件分岐の抽出

図 2 のような柔軟な発声を受理可能とするシステムの場合，スロットの内容と入力（認識結果）の関係によっては，同じ状態，入力であっても違う状態に遷移しなければならない場合がある．図 3 に例を示す．Case:1，Case:2 とも，値「蕎麦」が否定され，代わりの入力がなされている．しかし，Case:2 の場合は代わりに入力された値が既に入力されている値「うどん」と同じであったため，内容をマージする必要がある．その結果，Case:2 では入力項目数がひとつになり，入力項目数がふたつになる Case:1 とは違う状態に遷移することになる．他にも，部分否定訂正入力において否定対象が無かった場合など，状態と入力の関係によってはイレギュラーな遷移が必要になる場合が存在する．そこで，状態遷移表内に条件文を入れる，などの対処が必要となるが，その結果，以下のような問題が起こることが考えられる．

- 条件分岐が多くなればなるほど状態遷移表が複雑となり，シナリオ作成者による遷移先のチェックに負荷がかかる．
- 条件分岐はシナリオ作成者があらゆるユーザの入力を想定することで洗い出していくことになるため，条件分岐の「漏れ」が起こる．

3 1bit 状態遷移表

状態遷移表を用いたシナリオ記述方式は，他の方式と比較していくつかメリットがあるものの，状態遷移先チェックの煩雑さと，条件分岐を含む状態遷移ルールの多様さに課題が残る．本章では，これらの課題を解決する方式として，筆者らが提案している 1bit 状態遷移表について説明する．我々はまず，スロット状態によるシナリオ状態の決定，及び状態遷移ルールの定式化というアプローチをとることで解決を目指した．

スロットフィリング方式では，シナリオ状態を定義する重要なファクターとして「スロット状態」があげられる．そこで我々はスロット状態に注目し，音声対話システムにおけるスロット状態として 4 状態を定義した．さらに，シナリオ状態の表現記法として，スロット状態そのものを利用した．次節でスロット状態の定義について具体的に記述する．

また，テレマティクスサービスにおける情報検索を対象とした様々な対話例を調査した結果，このようにスロット状態に注目することで，2 章に記述した課題である条件分岐を含む状態遷移ルールが少数のルールで表現可能であると考えた．3.2 で，状態遷移ルールについて説明する．

対話例	シナリオ状態
＜初期状態＞	[XX]
S「お店の種類を『和食か洋食』のように二つ指定してください」	
U「蕎麦かうどん」（一括入力）	[YY]
S「蕎麦かうどんですか」	
U「いいえ」（逐次確認開始）	[SS]
S「「蕎麦を指定しましたか？」	
U「いいえ」	[SX]
S「うどんを指定しましたか？」	
U「そうじゃなくてパスタ」	[SX]
S「パスタを指定しましたか？」	
U「はい」	[ZX]
S「もうひとつのお店をお話ください」	
U「ラーメン」	[ZY]

図 4: スロットの状態遷移 (単一スロット種類×スロット数２)

対話例	シナリオ状態
＜初期状態＞	[XX]
S「お店の種類と場所を指定してください」	
U「目的地」	[XY]
S「目的地ですか」	
U「はい」	[XZ]
S「お店の種類を指定してください」	
U「和食」	[YZ]
S「和食ですか？」	
U「はい」	[ZZ]

図 5: スロットの状態遷移 (複数スロット種類×スロット数１)

3.1 シナリオ状態表現

各スロットの状態は以下のように変化するものとした．まず，初期状態は「空」である．受理可能な入力があると，空のスロットに入力単語をあてはめる．ただし音声入力の場合，ユーザの言い直しやシステムの誤認識が発生する可能性があり，入力された項目は必要に応じて確認をとる必要がある．従って，この時点でのスロット状態は「未確認」である．未確認スロットがあった場合，システムはユーザに確認を求める．ユーザが肯定した場合,「未確認」スロットは「決定」状態に設定される．また，一括入力後の「未確認」スロットが複数ある状態でシステムの確認に対しユーザが否定した場合，複数スロット全てが否定されたのではなく，一部のスロットのみが否定されている可能性が高いと思われる．そのため，全ての未確認スロットをリセットしてしまうのではなく，1 スロットずつその正誤を確認していくようにした．この状態を「逐次確認待ち」とする．

以上より，スロット状態として「X:空」「Y:未確認」「S:逐次確認待ち」「Z:決定」の 4 状態 (2bit) を定義した．なお，対話は，全てのスロットに値を入力，状態を「Z:決定」にすることを目的とし，進められる．

スロットフィリング方式では，シナリオ状態毎に，ガイダンスや認識文法などの対話制御情報を決定する．ここで上記で定義したスロット状態に注目したところ，これらの対話制御情報は，スロット状態によって一意に決まると考えられる．つまり，シナリオ状態は，そのシナリオに用意されたスロット状態の列によって表現可能であることになる．シナリオ状態を併記した対話例を図 4，図 5 に示す．なお，図 4 は単一種類のスロットを持つシナリオの対話例であり，一番目のスロット，二番目のスロット共に，お店の種類のスロットである．一方図 5 は，複数種類のスロットを持つシナリオの対話例であり，一番目のスロットをお店の種類，二番目のスロットを検索する場所とした．

図 4 の対話例のように，同じ種類のスロットが複数存在する場合は，そのスロットの並び順は特に意味を持たない．そこでスロット状態を一意に表現するため，スロット状態を表すレター (XYZS) の並び順に優先度を持たせ，同じ種類のスロットを常に，Z ＞ Y ＞ X ＞ S の順に並べ替えることとした．こうすることで，状態遷移表の状態の重複を無くすことが出来る．

3.2 状態遷移ルール

前節で述べたように，対話制御情報に注目すると，シナリオ状態はスロット状態のみで表現が可能であると考えられる．ただし，2 章で述べたように，スロットの内容と入力の関係によっては，状態遷移先が変更してしまうという課題は解決されない．しかし，ここでシナリオ作成者が状態遷移表を作成する際の思考に注目すると，一定のポリシーさえ決めておけば，現状態と入力から次の状態を一意に決める

ことが出来ている．つまりこれは，遷移先を決めるルールを定式化することが可能であれば，状態遷移先情報を状態遷移表（シナリオ）自体で持たず，システム側で計算することが可能であることを意味する．そこで，実際に状態遷移ルールの定式化を試みたところ，少数のルールで記述可能であった．以下に状態遷移ルールを示す．なおルールは，認識エンジンの性能やシステム設計者の設計方針に従い，サービスに応じて変更することも可能である．

1. 肯定 yes
   1-1. Y がある場合，Y を全て Z にする．
   1-2. S がある場合，一番左端の S を Z にする．
   (後処理) 同じ種類のスロット内で，同じ内容の Z があった場合はひとつにマージし，余ったスロットは空にして X に変更する．
2. 否定 no
   2-1. Y が複数ある場合，Y を全て S にする．
   2-2. Y がひとつある場合，Y を X にする．
   2-3. S がある場合，一番左端の S を X にする．
3. 部分指定訂正入力 $B_1B_2...notA_1A_2...$
   3-1. not の前の全ての項目 B_i に対し，同じ内容の Y が存在していた場合は，そのスロットを X にする．
   3-2. 3-1 以外の場合は，Y を全て X にする．
   (後処理) 単一/一括入力と同じ処理
4. 訂正入力 $notA_1A_2...$
   Y を全て X にする．
   (後処理) 単一/一括入力と同じ処理
5. 単一/一括入力 $A_1A_2...$
   (前処理 1) 逐次確認スロット S があり，入力 A_i が複数ある場合は，S を全て X にする．
   (前処理 2) 逐次確認スロット S があり，入力 A_i が一番左端の S とは違うスロット種類 だった場合は，S を全て X にする．
   (前処理 3)Y がある場合，Y を全て X にする．
   5-1. S がある場合，S に入力 A_1 を埋める．状態は S のまま，再び確認シーケンスへ進む．
   5-2. S がない場合，X に入力 A_i を埋め，Y にする．ただし，既に未確認スロット Y に同じ内容の入力があった場合は無視する．なお，確定スロット Z に同じ内容の入力があった場合も，X に入力 A_i を埋め，確認シーケンスに移る．その後，肯定された場合，肯定 yes のルールに従って同じ内容の Z がマージされる．
6. 後処理
   6-1. 同じ種類のスロット内で，Z>Y>X>S の順にソートする．

ただし，ルールの適用の際にはスロット情報が必要である．ここでスロット情報とは，スロット種類数，各スロット種類のスロット個数，スロット種類を示すスロット名のことであり，[店の種類][店の種類][予算] という 3 スロットが用意された食事検索の場合，スロット種類数は [店の種類] と [予算] のふたつ，スロット数はそれぞれ 2 個と 1 個である．また，それぞれ，[FOOD] [FOOD][PLACE] などといったスロット名が与えられる．スロット名は，ユーザの発声がどのスロット種類に対する入力なのかをシステムが識別するために使用される．具体的には，ユーザの発声に対し認識エンジンが音声認識を行い，スロット名をプレフィックスとして含む文字列を返すようにする．例として，ユーザの発声とその認識結果，さらにそれを状態遷移表で扱うために変換した結果の例を，図 6 に示す．このように状態遷移ルールを定式化したことにより，状態遷移表における状態遷移先情報には，遷移先そのものを記述する必要がなくなり，シナリオ作成者は，各状態において「その入力を受け付けるか否か」を表す 2 状態，1bit 情報のみを記述すれば良いことになる．図 7 に，従来の状態遷移表（A）と提案方式による状態遷移表（B）の例を示す．各列は，ユーザの入力の種類を示し，それに対応する各行は，シナリオの現状態を示している．入力における表現 “$input\_N$” は，システムに予め渡されているスロット種類のうちの N 番目のスロット種類に対する入力であることを示す．また，従来方式の入力における “$slot$” は，既にスロットに入力されている値を示す．セル内の情報は，従来方式においては遷移先状態の番号を，提案方式においてはその列の入力を受け付けるか否かを示す．提案方式では，例えば初期状態 XXX において入力 $input\_N$ が来ると，状態遷移ルール 5-2 に従い，ひとつのスロット X に値を入れ，その状態を Y にする．従って，シナリオ状態としては YXX となる．この状態でシステム側が出力する確認応答に対しユーザが肯定した場合，ルール 1-1 に従い Y のスロットを Z へと変換させるので，シナリオ状態としては ZXX となる．以下，状態遷移ルールに従い全スロットが埋まり状態 Z になるまで対話が続く．

ユーザの発声	認識エンジンの出力結果	状態遷移表用に変換した結果
現在地	PLACE_現在地	input_1
目的地で洋食	PLACE_目的地 FOOD_洋食	input_1 input_2
和食か中華	FOOD_和食 FOOD_中華	intput_2 input_2
和食じゃなくて中華	FOOD_和食 NOT FOOD_中華	intpu_2 NOT input_2
そうじゃなくてフレンチ	NOT FOOD_仏	NOT input_2

図 6: ユーザの発声, 認識結果, 状態遷移表の入力の関係

(A) 従来方式による状態遷移表

Scene 1		input_1 (if(input_1 = slot))	input_1 (if(input_1 ≠ slot))	input_1 and input_1 (if(...))	
A	initial condition	-	-B	-	
B	input_1 confirmation	B	D	G	
C	wait for next input	E	F	-	

(B) 提案方式による状態遷移表

Scene 1	input_1	input_1 input_1	input_1 not input_1	
XXX	1	1	0	
YXX	1	1	1	
ZXX	1	1	0	

図 7: 状態遷移表

3.3 1bit 状態遷移表の利点

以上のように，シナリオ状態を 2 ビット（＝4 状態）のスロット状態で表し，状態遷移ルールを定式化することにより，以下の効果が期待できる．

- 一定のルールに則った状態遷移表が作成される
- 状態遷移表に記述できる内容は全て正常系であるため，矛盾が生まれない
- シナリオ作成者は，状態/入力の条件による遷移先の振り分けを意識する必要が無くなる
- 状態遷移表の各列，各行が独立の関係であるため，修正が容易である

つまり，1bit 状態遷移表は，従来のシナリオ記述方式の課題であるシナリオの頑強性や追加/管理の容易性の向上に関して優れた特徴を持っていると考えられる．

なお，シナリオ記述方式として 1bit 状態遷移表を使用したプロトタイプシステムを構築済である．スロット情報が異なる 3 種類のシナリオを実装した．プロトタイプシステムでは，各シナリオが正しく動作することを確認し，1bit 状態遷移表の実現可能性を確認した．また，対話の中断復帰機能が容易に実装可能であることを確認した．また，被験者 6 名を対象に評価実験を行った結果，従来の状態遷移表と比較しての有効性が確認された [6].

4 拡張方式

4.1 1bit 状態遷移表の課題

3 章で述べたように，1bit 状態遷移表は，従来の状態遷移表の課題である状態遷移先チェックと条件分岐の抽出の煩雑さと状態遷移ルールの多様さについて，解決が可能であることが示されている．しかし，扱えるシナリオの汎用性においては，以下のような問題点があることと考えている．

まず，提案した方式では，全スロットを埋めることがシナリオの終了条件であるため，ユーザにとって検索するのに必要となる情報数以上のスロットがあった場合にも，システム側は全スロットに対するユーザの入力を要求することになる．特に，同じ種類のスロットが複数存在するシナリオ，例えば「和

食や中華」という発声を可能にするシナリオの場合，この制約があるため「和食」のみで検索することが出来ないという問題が起こる．また，ユーザによっては，日頃変わることの無い検索条件をデフォルト値として設定しておきたいという要求も考えられるが，本提案方式では毎回新規にユーザの入力を要求することになるという制約がある．また，例えば十分な認識性能を持つ認識エンジンを使用しているシステムでは確認応答なしで対話を進ませたい場合も考えられるが，本提案方式では必ず確認応答をしてユーザの肯定を得る必要がある．次節では，これらの課題を解決する拡張方式を提案する．

4.2 必須スロット数の設定

必須スロット数とは，スロット種類毎に設定される値で，そのスロット種類の中で少なくとも何個のスロットを埋める必要があるかを示す値である．この必須スロット数情報を追加することで，必須スロット数分スロットが埋まっていれば同じスロット種類の残りのスロットが埋まっていない場合でも，そのスロット種類に関して検索に必要な情報は取得できたものとして，シナリオを進めることが可能となる．

従来の 1bit 状態遷移表では，システムに渡す情報として，状態遷移表の他に“ スロット種類と各スロット種類のスロット個数 ”が必要である．これに加え，必須スロット数を導入する際には,“ スロット種類ごとの必須スロット数 ”も渡すことになる．また，3.2 節で示した状態遷移ルールに,“ 必須スロット数分以上状態が Z になったスロット種類があった場合は，状態が X である同じ種類のスロットを内容 NULL のまま，状態 Z に変更する ”というルールを加えることになる．以上の拡張/変更を行うことで，必須スロット数の設定が可能となる．

4.3 デフォルト値の設定

デフォルト値とは，各スロットに予め与えられている値のことであり，任意のスロットにデフォルト値を設定可能にすることで，ユーザが指定しなかったスロットに対しては入力を促さず，デフォルト値を代入して次の処理に進めることが可能になる．

システム側に予め渡しておくスロット情報としては,“ スロット種類毎のデフォルト値の要/不要フラグと必要な場合はそのデフォルト値 ”を追加することになる．また，状態遷移ルールの変更として，全ルールの後処理として,“ デフォルト値が設定されていないスロット種類に関し全てのスロットが Z で

対話例	シナリオ状態
＜初期状態＞	[XXXX]
S「お店の種類と場所をおっしゃってください.」	
U「和食」	[YXXX]
S「和食ですか？」	
U「はい」	([ZXXX]→)[ZZZX]
S「場所はどこにしますか？」	

[店の種類× 3(必須スロット数：1)]
[場所× 1(必須スロット数：1)]

図 8: 必須スロット数の導入例

対話例	シナリオ状態
＜初期状態＞	[XX]
S「お店の種類と場所をおっしゃってください.」	
U「和食」	[YX]
S「和食ですか？」	
U「はい」	[ZX]→[ZZ]
S「では現在地付近の和食のお店を検索します」	

[店の種類× 1(デフォルト：設定なし)]
[場所× 1(デフォルト：現在地)]

図 9: デフォルト値の導入例

あり，且つ，デフォルト値が設定されているスロット種類に関しそれぞれ全てのスロットが Z または全てのスロットが X である場合，残りのスロット X にデフォルト値を入力し状態を Z にする ”というルールを追加することになる．以上の拡張/変更を行うことで，デフォルト値の導入が実現可能となる．

4.4 確認応答の有無の設定

確認応答の有無とは，ユーザの入力に対する確認応答の要否を選択可能とする手法である．確認応答が無しに設定されたスロットに関しては，ユーザの入力があった場合は確認せずに入力が確定する．

確認応答の有無を設定するためには，システムに渡すスロット情報に“ スロット種類ごとの確認の要/不要フラグ ”を追加する必要がある．また，状態遷移ルールに,“ 確認が必要ないスロット種類に確認待ち状態 Y 若しくは S があった場合，そのスロットの状態を Z にし，同じ内容の Z があった場合はひ

対話例	シナリオ状態
＜初期状態＞	[XX]
S「お店の種類と場所をおっしゃってください.」	
U「目的地」	[XY]
S「目的地ですか？」	
U「はい」	[XZ]
S「お店の種類は何にしますか?」	
U「和食」	[YZ]→[ZZ]
S「では目的地付近の和食のお店を検索いたします」	

[店の種類× 1(確認：不要)]
[場所× 1(確認：要)]

図 10: 確認応答の有無の導入例

とつにマージする ”というルールを加えることになる．以上の拡張/変更を行うことで，確認応答の有無の設定が実現できる．

4.5 考察

1bit 状態遷移表シナリオ記述方式の拡張として,「必須スロット数の設定」,「デフォルト値の導入」,「確認応答の有無の指定」に注目した．これらの拡張を適用するには，提案方式に対して，システムに与えるスロット情報や状態定義方法に関しての拡張が必要となる．また，状態遷移ルールに関しても変更または追加が必要となる．ただし，ルール自体はシステム側に組み込まれるため，シナリオ作成者の作業としては，設定可能な初期値がいくつか増えるのみであり，状態遷移表を書く際の手順は変わらない．従って，拡張を行うことにより提案方式の利点を損なうことなく，汎用化を行うことが可能と考えられる．

5 終わりに

筆者らが提案している音声対話システム用シナリオ記述方式としての 1bit 状態遷移表は，作成したシナリオの頑強性，シナリオの作成/維持の容易性において利点を持つ．しかしながら，汎用性の面では,「全てのスロットが埋まらないとシナリオが終了しない」,「必ずユーザに確認応答を求める」，などの課題が残されていた．本研究では，これらの課題を解決する拡張方法として,「必須スロット数」,「デフォルト値」,「確認応答の有無」の導入を検討した．

参考文献

[1] 宮田博司, 柿原正樹: 新情報提供サービスについて, 自動車技術, Vol. 57, No. 2, pp. 54–59 (2003).

[2] 桂川景子, 柳拓良, 大野健, 渡部眞幸, 伊藤敏彦, 小西達裕, 伊東幸広: ドライブプラン作成・編集のための PC 版サブシステム DSP-PC の構成と評価, 情報処理学会論文誌, Vol. 44, No. 12, pp. 2990–3001 (2003).

[3] 青山一美, 平野泉, 菊地英明, 坪川拓史, 白井克彦: 音声対話システム汎用プラットフォームの検討, 情報処理学会研究報告, 2000-SLP-30,, No. 3, pp. 7–12 (2000).

[4] Umeda, M., Kogure, S. and Nakagawa, S.: Interpreter for Highly Portable Spoken Dialogue System, *In Proc. of 3nd SIGDIAL Workshop on Discourse and Dialogue*, pp. 105–114 (2003).

[5] Obuchi, Y., Nyberg, E., Mitamura, T., Duggan, M. and Judy, S.: Robust Dialog Management Architecture Using VoiceXML for Car Telematics systems, *In Proc. of Workshop on DSP in Mobile and Vehicular Systems* (2003).

[6] Sekiguchi, M., Aragane, Y. and Abe, N.: Proposal and Evaluation of 1-bit State Transition Matrices as Telematics Scenario Description Architecture, *The 6th International Symposium on Wireless Personal Multimedia Communications*, pp. 197–201.

[7] Webb, N., Roeck, D., Kruschwits, U., Scott, P., Steel, S. and Turner, R.: Natural Language Engineering: Slot-Filling in the YPA, *Workshop on Natural Language Interfaces, Dialogue and Partner Modelling, Fachtagung fur Kunstliche Intelligenz KI'99* (1999).

[8] Boye, J., Wiren, M., Tayner, M., Lewin, I., Carter, D. and Becket, R.: Language-Processing Strategies for Mixed-Initiative Dialogues, *In Proc. of IJCAI-99 Workshop on Knowledge and Reasoning in Practical Dialogue Systems* (1999).