音声対話用構文解析器の頑健性の評価

船越 孝太郎† 徳永 健伸† 田中 穂積†
† 東京工業大学大学院 情報理工学研究科 計算工学専攻
〒152-8552 東京都目黒区大岡山 2-12-1
E-mail: † {koh,take,tanaka}@cl.cs.titech.ac.jp

話し言葉に頻繁に現れる助詞落ち，倒置，自己修復などの不適格性は，音声対話を困難にする大きな要因の 1 つである．筆者らは，日本語におけるこれらの不適格性が複合して現れることを指摘し，解決法を示した．本論文では提案手法を実装し，新たに収集した音声発話データに対して評価を行なう．発話データの収集に際しては，小量の採集でも不適格性を多く含むように工夫を行なった．実際に音声認識結果に対して構文解析を行なった結果，対話システムが正しく解析できる発話が 171 発話から 322 発話に改善されることを確認した．

キーワード: 音声対話, 不適格性，助詞落ち，倒置，自己修復，言い直し，言い淀み

Evaluation of Robustness of a Parser for Speech Dialog Systems

Kotaro FUNAKOSHI† Takenobu TOKUNAGA†
† Department of Computer Science, Graduate School of Information Science and Engineering,
Tokyo Institute of Technology
2-12-1 Oookayama Meguro-ku, Tokyo 152-8552, JAPAN
E-mail: † {koh,take,tanaka}@cl.cs.titech.ac.jp

Ill-formedness in speech, such as postposition omission, inversion, and self-correction, is a major obstacle which makes speech dialog difficult. We proposed a method to handle these sources of Japanese ill-formedness in our previous paper. In this paper, we implement the proposed method and evaluate it by using newly collected speech data. We designed the experiment to obtain ill-formedness data effectively. Among 532 utterances in the corpus, introducing the proposed method increased the number of correct analysis from 171 to 322.

KeyWords: *Speech Dialog, Ill-Formedness, Postposition Omission, Self-Correction, Speech-Repair, Hesitation*

1 はじめに

話し言葉に頻繁に現れる助詞落ち，倒置，自己修復 (言い直し) などの不適格性 (ill-formedness) は，音声対話を困難にする大きな要因の 1 つである．筆者らは，日本語におけるこれらの不適格性が複合して現れることを指摘し，解決法を示した [?]．さらに小規模の対話コーパスに提案手法を人手で適用し，定性的な考察を行なった．しかし，この評価は人間による書き起こしに対するものであり，自動音声認識の結果に対するものではない．音声対話システムは自動音声認識の使用を前提にしており，自動音声認識結果に対する評価も重要である．

本論文では，[?] で提案した手法を，新たに収集した音声発話データに対して適用し，人間による書き起こしのみでなく，音声認識結果に対しても評価を行なう．先の論文では議論しなかった定量的な評価を行ない，実際の音声対話システム上での有用性を示す．

不適格性に関する研究の評価における問題点の 1 つは，データ収集の難しさである．助詞落ち，倒置，自己修復などは，比較的出現しやすい代表的な不適格性であるものの，それらが対話コーパス内の各発話 (文) で出現する割合は大きくない．Bear らは 10,000 文のコーパス中，607 文 (6%) が自己修復を含んでいたと報告している [?]．伝は使用した ATR 対話データベースの約 10%の文に，中野・島津は使用したコーパス約 15,000 ターン中に，704 個の自己

修復 (約 5%) があったと報告している [?],[?]．これに対し，Heeman・Allen が使用した Trains Corpus は，6163 ターン中に 1973 個 (32%) [1]と比較的多量の自己修復を含んでおり [?]，また，Levelt も人間同士の対話で言い直しの割合が 34%であったと報告している [?]．対話コーパス中の自己修復の割合は，タスクの設定などによって変動するものの，おおむね 2 割前後 (5%〜30%) であることが判る．山本らは，助詞落ちが名詞文節 4063 個に対して 171 個 (4%)，倒置が 1818 文中 32 個 (1.8%) であったと報告しており [?]，自己修復以上に少ない．

不適格性は，話者が発話に対して慎重になるほど減少すると経験的に予測できる．反対に，話者が自身の発話にあまり注意を払わない状況 (自由対話，いわゆるおしゃべり) や，発話以外の行為に自身の注意を振り向けなければならない状況 (特に時間的に制限された状況での思考や行動) などで増加する傾向がある．そこで我々は，被験者が発話内容を考えながら発話しなければならないように制限した状況で音声発話データ[2]を収集した．

次節では本論文で扱う不適格性について説明する．そして**??**節では**??**節で述べた不適格性を含む発話の解析手法を示す．**??**節では上で述べた音声発話データの収集方法との分析結果を説明し，それに対する実験結果を示す．**??**節で考察を行ない，**??**節で結論を述べる．

2　話し言葉の不適格性

本論文では，日本語を対象とし，話し言葉の不適格性のうち，助詞落ち，倒置，自己修復，言い淀みの 4 種を扱う．

2.1　助詞落ち

会話体の表現では，助詞の脱落が頻繁に起こる．さらに，自動音声認識を利用した対話システムの場合，音声認識器の誤認識による助詞の欠落も考慮しなければならない．

文節の末尾に来る助詞は，先行する内容語に比べてしっかりと発音されない傾向が強く，音声認識器が誤り易い箇所でもある．音声認識器の誤りには脱落，置換，挿入の 3 種があるが，ここでは脱落のみを対象とし，発話者による助詞落ちと同じ枠組で対処する．

山本らは,「は，が，を，に，へ」が助詞落ちの約 80%を占めると報告している [?]．本論文では，これに「も，の」を加えた 7 助詞を対象とする.「の」は，省略して発話されることはあまり無いが，誤認識によってその後の言語解析を妨げることが多い.「も」も，話者が省略することは無いが，誤認識によって脱落することが多い．ただし「も」の場合,「が，を，に」などの助詞が持つ格情報までしか回復できない．従って,「それを押して」と言わずに「それも押して」と言った場合などに発話者が伝達を意図した強調・反復性などの情報は失われる．

「から，まで，けど」など 2 音節以上ある助詞に関しては，内容語と同程度の精度で認識され，発話者が省略することも稀なので対象としない.「か，と，や，し」などの助詞は，現在我々の対話システムが対象とする言語表現の中では使われないため，本論文では対象としない．

2.2　倒置

「それを持ってきて．ここに.」という表現が,「ここにそれを持ってきて.」という倒置表現であるのか，それとも「それを持ってきて．ここに持ってきて.」という表現の省略であるのかを決めることは難しい．本論文では,「省略と考えた場合その省略部分が前の文と同じ内容になってしまう場合」[?] を倒置と認定する．そして,「倒置された文節は述部に係り，その述部は終止形／命令形で終る」ものとする．これは，[?] で使用された対話データベース内の倒置の特徴とも一致する．

2.3　自己修復

自己修復は一般に言い直しと呼ばれる現象である．ここでは，一般に言う総称としての「言い直し」を自己修復と呼び，自己修復を，言い足し，言い直し，リスタートの 3 つに分類する．

言い足し (addition, appropriateness repair) は，発話内容が間違っているのではないが，発話が伝達する情報の適切性に問題があると発話者が感じた場合に起こり,「赤い玉を押して，右のやつ」のように，情報を付け足す形の表現となる．言い直し (repair, error repair) は，言い誤りによって発話内容に誤りが生じた場合，あるいは話者が発話中に意図を変更した場合に起こり,「赤い玉を，ごめん，青い玉を押して」のように，情報を訂性する形の表現となる．言い足し・言い直しは，付加と訂性という機能面におけるはっきりとした違いがあるものの，表層的に

[1] abridged repair は除く．
[2] 各被験者が独立に発話するので，対話データではない．

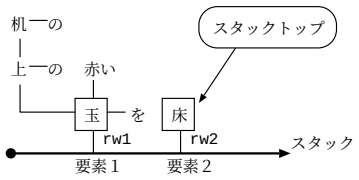

図 1: 「机の上の赤い玉を床」という入力を解析した場合のスタックの一例

は非常に似ているため，[?] のように modification repair として両者を一つにまとめて扱うこともある．しかし日本語の場合，両者の間で可能な表現に差があるため [?]，ここでは区別する．Levelt も，appropriateness repair と error repair の間で，編集表現 (editing expression/term) の分布に大きな差があるとして，両者の区別を重要なものだとしている [?]．編集表現とは,「ちがう，いや，ごめん」などの自己修復のマーカーとなる表現である．

リスタートでは，発話者はそれまでに発話した内容を全て破棄し，新たに発話を開始する．

2.4 言い淀み

言い淀みは，発話者が単語の発声を中断した，あるいは発声に失敗したために，発話内に単語断片が挿入されてしまう現象である．多くの場合言い淀みの直後には自己修復が入るが，必ずしもそうなるとは限らない．また，現在主流となっている自動音声認識は単語認識を基礎としているため，単語断片の認識は難しい．よって，言い淀みは自己修復の処理とは別に対処する．

3 解析手法

3.1 パーザと辞書

不適格性を解消するための処理はすべて，構文解析と平行してパーザの上で行う．以下，本論文で用いるパーザとパーザの使用する辞書について説明する．

3.1.1 係り受け解析を用いた漸進的な構文解析

構文解析の手法として，文節ベースの係り受け解析を採用した．我々の解析手法では内容語を重視し，機能語は内容語に付属するものと捉える．構文解析はスタックを用いて漸進的に行なう．

パーザは，解析過程で生成する複数の構文仮説を，別々のスタックに保持する．スタックの各要素には，依存関係で表現された構文木が納められる (図**??**)．以下，各スタックの要素である部分構文木のルートになっている語を「ルート語」と呼ぶ[3]．

パーザは音声認識器から単語列を受けとると，単語列の先頭から 1 つづつ，スタックにプッシュしながら解析を行なう．スタックにプッシュした語が機能語であった場合には，単純にスタックの上から 2 つ目の要素のルート語にその機能語を付ける．プッシュした語が内容語であった場合には，次にプッシュする予定の語が機能語でなければ，係り受け解析を行なう．係り受け解析はスタックトップの要素のルート語 ($rw2$) とトップのすぐ下の要素のルート語 ($rw1$) の関係を見て行う．すなわち，$rw1$ が $rw2$ に係り得るかどうかを調べる．$rw1$ が $rw2$ に係ることができないならば，スタックは変化しない．しかし，$rw1$ が $rw2$ に係ることができるならばこのスタックをコピーし，片方には $rw1$ が $rw2$ に係る仮説 ($H1$)，もう片方には $rw1$ が $rw2$ に係らない仮説 ($H2$) を保持する．仮説 $H1$ を保持するスタックでは，一度上 2 つの要素をスタックから取り出したあとに，新しい係り受け関係を作った要素 1 つだけをスタックトップに戻す．仮説 $H2$ を保持するスタックは変化しない．また，$H1$ については同様の操作を再帰的に行う．従って，[(君が) | (玉を)> という仮説スタック[4]に「押せ」という単語がプッシュされると，

[(君が) | (玉を) | (押せ) >
[(君が) | ((玉を) 押せ) >
[((君が) (玉を) 押せ) >

という 3 つの仮説スタックが生成される．

パーザは各仮説にスコアを与え仮説の数を制限するが，スコア付けの詳細については省略する．

3.1.2 文法表現と辞書

本手法では，文節構造以外の文法に相当するものは全て単語辞書の中に単語毎に用意する．

ある内容語 $c1$ がある内容語 $c2$ に係る時，$c1$ は $c2$ に対して特定の意味役割を担うと考える．例えば,「押して」という命令動詞の辞書のエントリは図**??**のように記述する[5]．

図**??**の第一行目は,「押して」という語が，左から順に，

- 動詞である
- 命令（IMP+）である

[3] 図**??**において四角で囲まれた語．
[4] [がスタックの底，| が要素間の区切り，> がスタックのトップ，( ) が係り受け関係を表す．
[5] 現在は語彙数が少ないため，活用語は必要な活用形毎に記述している．

```
押して VERB IMP+ PUSH+
    <OBJECT> 1 NOUN は|を|も  INSTANCE+
    <AGENT>  1 NOUN は|が|も  ANIMATE+ INSTANCE+
    <TO>     1 NOUN に|へ     LOCATION+
    <FROM>   1 NOUN から      LOCATION+
    <EXTENT> 1 ADV  -         DEGREE:*
    <SPEED>  1 ADV  -         SPEED:*
```

図 2: 命令動詞「押して」の辞書エントリ

```
押して VERB IMP+ PUSH+
    <OBJECT> 1 *  NOUN は|を|も|-  INSTANCE+
    <AGENT>  1 *  NOUN は|が|も|-  ANIMATE+ INSTANCE+
    <TO>     1 *  NOUN に|へ|-     LOCATION+
    <FROM>   1 *  NOUN から        LOCATION+
    <EXTENT> 1 *  ADV  -           DEGREE:*
    <SPEED>  1 *  ADV  -           SPEED:*
```

図 3: 助詞落ち・倒置への対応

- PUSH という動作を表す素性を持つ

ということを表す．第二行目以降は,「押して」に係ることのできる語の制約を役割毎に示している．例えば第二行目の，<OBJECT>(目的格) という役割に関する記述は，左から順に,

- 「押して」に対して 1 つしか存在しない
- 名詞しかこの役割は取れない
- 名詞についている助詞は「は，を，も」のどれか
- INSTANCE 素性を持っていなければならない

という事を表す.

??節で述べたパーザは，この辞書を用いて解析を行う．内容語 $c1$ が内容語 $c2$ に係ることができるかどうかは，$c1$ が $c2$ のエントリに示された役割の内のどれかを満たすことができるかどうかによって決まる．そしてパーザは，解析の段階で全ての係り受けに意味役割を割り当てる．

3.2 不適格性を含む発話の解析

3.2.1 助詞落ち・倒置

助詞落ち・倒置には，**??**節で説明したパーザと辞書を拡張することで対処する．

まず助詞落ちでは，形容詞など機能語が附属しない文節の係りのために用意した「無標 (-)」を名詞の係りに対しても適用する．すなわち，役割 <OBJECT> を取って動詞にかかる名詞の場合,「は，が，も」の他に，無標 (-) も許す．無標を許す係り受け関係は，**??**節で述べた通りである．

倒置には，後方への係りだけでなく前方への係りも可能にすることで対処する．まず辞書内の全てのエントリにおいて，各役割毎に B，F，*のいずれかで，可能な依存方向を指定する．B が後方依存のみ，F が前方依存のみ，*がどちらも可能 (ワイルドカード) であることを示す．そして，パーザが $rw1$ から $rw2$ への係り受け (**??**節参照) だけでなく，$rw2$ から $rw1$ への係り受けも処理できるようにする．これによって，倒置表現の解析が可能になる．

上記の修正を加えた，命令動詞「押して」の辞書エントリを図**??**に示す.

3.2.2 自己修復

自己修復の検出及び修正手法の詳細については [**?**] を参照されたい．紙面の都合上，本論文では概略のみを説明する．

自己修復は，係り受け解析と同様に，パーザが各仮説スタックの上 2 つの要素[6]を監視することで検出する．自己修復も，その場所で起きている可能性は検出できても保証はできないため，1 つの可能性としてしか捉えない．従って，自己修復 (の可能性) を検出した場合，仮説スタックを複製し，片方はそのままにし，もう片方に修正結果を保持する．

例として「馬は赤い玉を，青い玉を押して」という発話を処理する場合を考える．

[(馬は) | ((赤い) 玉を) | ((青い) 玉を) >

という仮説スタックができた時点で，あらかじめ与えられたパターンルールを基に，パーザは「赤い玉を」が「青い玉を」で言い直された可能性を検出する．そこで，不必要と思われる成分を除去し，

[(馬は) | ((青い) 玉を) >

という新しい仮説スタックを生成する．

上記の手法が解決できるのは文節以上の単位での自己修復であり，文節以下の単位での自己修復となる機能語の言い直しは扱えない．機能語の言い直しは，パーザが文節構造を形成する所で扱わなければならない (**??**節参照)．すなわち，スタックに機能語がプッシュされ，パーザがスタックの上から 2 つ目の要素のルート語にその機能語を付けようとした時に，既にそのルート語に機能語が附属していれば，機能語が言い直されたと考え新しい機能語に置換する．しかしながら，機能語のみの言い直しは非常に少ない[7]一方，フィラーの存在のために音声認識器が誤認識し，正しく認識された助詞の後に間違って認識された助詞が続くことがしばしば観察される．このような理由から，機能語のみの言い直しは無視する．

[6] 上から 2 つ目の要素が編集表現だった場合は，もう 1 つ下も含めて 3 つの要素を監視する.

[7] 機能語のみを間違えた時も，ほとんどの場合は文節単位で言い直す.

リスタートに対処するため，編集表現が出現するとパーザは空のスタックを生成する．

3.2.3 言い淀み

言い淀みの問題点は，言い淀みによって発生する単語断片を現行の認識エンジンで正しく認識することが困難なことである．言い淀みの処理は，自動音声認識によって誤認識された単語を無視することと捉え，一般の音声認識器の誤認識 (挿入及び置換) と同様に読み飛ばしで対処する．読み飛ばしを行なうのは内容語だけとする．

読み飛ばしの処理は，パーザが係り受け解析を行なうときに行なわれる．パーザは，他の単語と係り受け関係が作れない語を読み飛ばす．**??**節の係り受けの可能性の判定に際し，$rw1$ が $rw2$ に係れない場合に，もし $rw1$ が娘を持たないならば，$rw1$ は誤認識された可能性があると判断して，$rw1$ をスタックから除いた新しい仮説スタックを生成する[8]．

4 実験

自然言語理解システム「傀儡」[?] で使用される言語表現を対象として音声発話データを収集し，**??**節で述べた解析手法の評価を行なった．

4.1 発話データの収集

4.1.1 収集対象の発話

傀儡で用いられる表現を収集の対象とした．傀儡では，仮想世界内のソフトウエアロボットに対して音声言語により指示を与え，同世界内のオブジェクトの配置を変えることができる．仮想世界に存在するソフトウエアロボット (以下ロボット) は，馬，ニワトリ，雪だるま，カメラの 4 体で，それらが可能な動作は，

- (仮想世界上の物体を) 押す
- (特定の方向を) 向く
- (指示された場所あるいは方向に) 行く
- (指示された物体あるいは空間を) 映す

の 4 種類のみである．「映す」ことが可能なのはカメラだけで，その代わりにカメラは「押す」動作はできない．現在の所傀儡で用いられる言語表現は，ロボットに対するこれらの動作の命令表現だけである．

[8] この処理は，正確には，「他の単語と係り受け関係が作れない語」ではなく，「他の単語と係り受け関係を作らないと判断された語」を読み飛ばしているので，目的以上の仮説スタックを生成してしまう．

4.1.2 収集方法

データ収集は，5 人の日本人大学生 (男 4，女 1) に対して行なった．

??節で述べたように，不適格性を含むデータを多量に集めることは容易ではない．そこで，以下のような方法でデータ収集を行なった．

監督者を 1 名置き，被験者 1 名をモニタの前に座らせ，モニタに図**??**のような 2 次元画像を表示する．この画像には，傀儡の仮想世界を真上から俯瞰した図，すなわちロボットとオブジェクトの配置図が描かれている．ロボットは二等辺三角形で示され，二等辺間の頂点が向く方向がロボットの向きを表す．オブジェクトは赤・青 2 種の球でそれぞれ 2 つづつあり，赤・青の丸で表示されている．ロボットとオブジェクトの下にはそれぞれの名称が書かれている．

監督者がキーを押すと，0.5 秒前後の間隔を空けて「誰が，何を，どこに，どこから，どう」するのかを表すヒントがランダムな順序で画像上に表示されていく．被験者は，可能な限りこのヒントを追いかけながら，ヒントが示している命令表現を発話するように指示される．こうすることによって，被験者は頭の中で命令表現を整理してから発話するのでなく，必要な表現を考えながら発話することになるため，不適格性 (特に単純な単語の繰り返しでない複雑な自己修復) を含む発話が通常のコーパス採集を行なうよりも効率よく収集できると考えた．

図**??**の場合，左上の「うま」に動作主を示す枠が，右下の「赤い球」に目的物を示す枠が「これを」というメッセージと共に表示されている．左上には動作「押す」を指示するキーワード「PUSH」が，中央部の球の脇には目的位置を示す二重円が「ここに」というメッセージと共に表示されている．これらのヒントが標準的な語順で表示された場合，被験者は「馬はカメラの後ろの赤い玉を青い玉の左に押して」といった発話をする．「カメラの後ろの」の様な参照表現は，その時の空間配置に応じて被験者が適切な表現を考える．ロボットやオブジェクトの配置は，被験者が交替するごとに変更される．

動作を指定するキーワードも他の指示よりも先に表示され得るので，そのような場合に被験者が表示順序に忠実に発話すれば，倒置が起きることになる．また，被験者は自分の発話が不自然であったと感じたり間違っていると思えば，自由に言い直すように指示される．ただし，必ずしも言い直したり，倒置して発話する必要はない．被験者の発話内容が，モ

図 4: 被験者に与えた視覚刺激の例

発話のタイプ	正解	部分正解	不正解	合計
不適格性を含む	106	2	31	139
不適格性を含まない	393	1	3	396
合計	499	3	34	536

表 1: 書き起こしに対する構文解析結果

発話のタイプ	正解	部分正解	不正解	合計
不適格性と誤認識を含む	28	17	67	112
不適格性のみ含む	22	1	2	25
誤認識のみ含む	101	42	76	219
どちらも含まない	171	1	6	178
合計	322	61	153	536

表 2: 音声認識結果に対する構文解析結果

ニタ上の指示を完全に満たしていたかどうかも問題としない.

4.1.3 発話データ収集の結果

それぞれの被験者に対して 50 発話程度づつの収集を 2 回行ない，全部で 536 発話を収集した．発話の平均長は 9 語であった.

それぞれの不適格性の数は，助詞落ち 7 個，倒置 4 個，自己修復 153 個，言い淀み 49 個であった．不適格性を一つも含まない発話は 397 発話あった.

収集データに自動音声認識を適用した結果は，脱落 184 個，挿入 55 個，置換 300 個であった．認識エンジンには Advanced Media Inc.の AmiVoice を使用した．使用した文法は文節構造[9]だけを規定した緩いものである．語彙数は約 109 で，11 種類のフィラーを用意した．536 発話中 203 発話が正しく認識された (フィラーの認識は除く)．不適格性を含まず誤認識も起きなかった発話は 168 発話であった.

4.2 評価実験

4.2.1 評価実験の方法

??節のパーザを実装し，??節で収集したデータに適用した．パーザはスコアを付けて複数解を出力するが，評価には第 1 位の解のみを用いた.

構文解析の結果は次の 3 つに分類した.

[**正解**] パーザの出力する依存構文木が発話者の意図と一致している．解析時にパーザが付与する意味役割も話者の意図と一致しなければならない.「赤い玉を前の，前に押して」という発話の場合,

((赤い):COL 玉を):OBJ (前に):TO 押して)

となれば良い (COL は色，OBJ は目的格，TO は目標格を表す).

[9] (内容語 (機能語)*)+

[**部分正解**] 正解の構文木の部分木となっている.「赤い玉を前の，前に押して」という発話に対して,

((前に):TO 押して)

と出力された場合，部分正解とする.

[**不正解**] 構文木の構造や意味役割が発話者の意図と違う.

4.2.2 評価実験の結果

表??に書き起こしデータを構文解析した結果を示す．数値は発話数である.

不適格性を含む発話は，76%が正しく解析できた．不適格性を含み不正解となっている発話のほとんどは，自己修復を含んでいるにも関わらず自己修復と解釈しない方がスコアが高くなっていた．例えば「馬の前の，馬の後ろの玉を押して」という発話は「馬の後ろの」が「馬の前の」の言い直しであるとする仮説よりも,「馬の前の」が 2 回目の「馬」に係るとする仮説が優先される．これは，パーザが発話中のより多くの単語を用いる仮説を優先するためである．不適格性を含まないもので不正解となっている発話は,「馬は押して」などの発話で，主語と解析されるべき語が目的語として解析されてしまったものである．これは，主語にも目的語なり得る語が 1 つだけあった場合，目的語として解釈するほうが優先されるようになっているためである.

表??に音声認識結果を構文解析した結果を示す.

使用した音声認識器では，全て正しく認識できたと仮定した場合，すなわち書き起こしの場合 (499 発話) の 64.5%(322 発話) が正しく解析できた．しかし，本論文でパーザに導入した手法によって 151(=28+22+101) 発話が正しく解析できるようになっている．表??で，誤認識を含みながら正解となっている発話は，その約 2/3 が助詞の脱落が助詞落ち

として解決されたものである．残り約 1/3 は，誤認識が言い直しを受けている領域で起きていた．

5　考察

傀儡は不適格性を含む発話を解析することができない．本論文のパーザを使用した場合，音声認識が完全であれば収集したデータの約 20%にあたる 106 発話を新たに解釈できるようになることが表**??**より判る．実際に音声認識を使用した場合は，正解率が 32%(171 発話) から 60%(322 発話) に大きく改善された．

また傀儡の場合，誤認識のために文法的に不適格となってしまった発話に対しては何もすることができないが，本手法の場合は読み飛ばしによって部分的ではあるものの話者の発話内容を取り出すことができる．正確な認識結果を常に得ることはできないので，音声対話システムのスループットを高めるには，部分正解に分類した不完全な発話を適切に利用する必要がある．部分正解も含めた場合，システムが何らかの適切な行動をとることによってタスクのゴールに近づくことができる発話は 71.5%(383(=322+61) 発話) となり，傀儡が解析可能な発話 (171 発話) に対して大きく改善する．傀儡は発話が曖昧であった場合，省略解決によって解釈の選択を行なうが，部分正解の発話は音声認識結果中の単語の利用率などによって通常の省略表現と区別することができるので，構文解析結果を基にユーザへの聞き返しなどを行なうことで過剰な省略解決を抑制し，システムのスループットを高めることが可能になる．

本論文では，少ない発話数でも十分な数の不適格性を収集するための工夫をした．しかし，収集した発話データには助詞落ちは 7 個，倒置は 4 個しか含まれず，**??**節で述べた手法は，助詞落ちや倒置を誘発するには不十分であった．これらの不適格性が瀕出するのは，本論文のような制限された状況よりも，自由会話のような状況であると予想できる．

6　おわりに

本論文では，日本語の不適格性に対処する手法を実装し，評価を行なった．まず，不適格性として助詞落ち，倒置，自己修復，言い淀みを考え，それらに対処するための仕組みを用意した．助詞落ちと倒置には，係り受け解析を拡張することで対処した．自己修復に対しては，日本語の自己修復の特徴に対応した手法 [**?**] を実装した．言い淀みは，自動音声認識の誤認識と合わせて，読み飛ばしによって対処した．データの収集に際しては，発話の自然さを損なわずに，かつ効率的に不適格性を含む発話を収集できるように工夫をした．そして，実際の音声対話システムでの使用を踏まえた評価を行い，提案した手法が有効であることを確認した．

Nakatani と Hirschberg は，自己修復の検出に 300msec 前後の短めのポーズ情報が有効であったと報告している [**?**]．これは，直観的にも妥当に思われる．しかし，自己修復の存在する可能性がある個所に一定の範囲内の長さ (100–400msec) のポーズがあった場合に，そこに自己修復の存在を仮定する仮説を優先させてみたところ，21 発話で解析結果に改善が見られたのに対し，28 発話では悪化した．単純なポーズ情報の利用だけでは，自己修復の的確な検出は難しいと言える．今後は，収集した音声データをさらに詳しく解析し，正確な自己修復の検出のための音響／音韻的な情報の利用を検討する予定である．