人工知能学会研究会資料
SIG-J-9601-8(12/06)

単一化に基づく漸進的発話生成

Unification-based Incremental Utterance Generation

坂庭克幸　乾健太郎　徳永健伸　田中穂積
SAKANIWA Katsuyuki, INUI Kentaro, TOKUNAGA Takenobu and TANAKA Hozumi
東京工業大学大学院情報理工学研究科
Department of Computer Science, Tokyo Institute of Technology
2-12-1 Ôokayama Meguro Tokyo 152 Japan
{sakaniwa,inui,take,tanaka}@cs.titech.ac.jp

Abstract The human explanation provider in an explanatory dialogue gives the recipient a chance to present feedback based on more fine-grained levels of the utterance than any existing explanatory dialogue system does. Allowing the recipient to give frequent feedback improves the efficiency of communication. To realize such fine-grained explanatory utterances, the content planner needs to generate fine-grained surface speech act goals and the surface generator needs to incrementally generate the utterances of some appropriate grain-size. This paper presents the architecture of our incremental surface generator that can handle fine-grained speech act goals and can appropriately aggregates them to control the grain-size of the utterances. The system is implemented based on the techniques of unification grammars.

1 はじめに

何かを相手に説明するとき，どんな情報が受け手の理解を助け，また受け手の目的に適うかを，説明の与え手が正確に予測することは不可能である．与え手は受け手の信念・知識に関する正確なモデルを持っていないためである．対話的な説明が可能な環境では，対話者は何に関するどの程度詳細な情報が必要なのかに関する合意を相互作用的に作りあげることができる．Cawsey [2], Moore [8], Carletta [1] らの対話システムは対話のこの側面を取り扱った例である．

対話におけるこの合意形成過程は相互作用的であると同時に漸進的である．合意の形成には互いの信念に関するさまざまな情報の伝達が必要であるが，この情報伝達を一括的に行うのは効率的でない．しばしば，伝達すべき情報を細かく分割して漸進的に伝達するという戦略がとられる [9]．与え手は個々の情報伝達の成否について受け手からフィードバックをもらい，それに基づいて以後の発話内容を決める．つぎの発話 (u1) は，受け手にフィードバックを促す典型的な例である [1]．

(u1) G：富士見通りという道があるんですけど，
(u2) F：はい．
(u3) G：そちらを右に曲がって下さい．

話し手はしばしば特定の韻律パターンを用いて聞き手にフィードバックを促していることが報告されている [11]．(u1) の情報伝達が失敗する可能性のあるときは，このように相手にフィードバックを促して成否を確認することにより，失敗した場合に (u3) の情報の伝達が無駄になることを防ぐことができる．もし，(u1) の情報の伝達について受け手からのフィードバックを必要としないのであれば，(u4) のように発話することもできる．

(u4) G：富士見通りという道を右に曲がって下さい．
(u5) F：はい．

また，漸進的に情報を伝達する方が受け手に理解のための時間的余裕を与えることになる．したがって，対話の漸進性は受け手側の理解の負担を軽くするという点でも重要である．このように，対話的環境のもとでの説明生成において，漸進性は重要な基準の一つであると言える．

漸進的生成については，すでにさまざま方式が提案されている．たとえば，de Smedt と Kempen は漸進的に生成される概念の列から文法的に正しい文を生成するための文法を開発している [10]．また，Finkler の POPEL-HOW も漸進的に生成される内容から文を生成する [5]．しかしながら，これらの研究はモノローグの生成を対象としており，対話に特有な細かい粒度の漸進性と相互作用を実現するものではない．

Dohsaka らは，発話の間に一定時間以上の沈黙を置かないという時間的制約のもとで漸進的に発話を生成するモデルを提案している [3]．Dohsaka らのモデルは，従

[1] 本稿で用いる例はすべて NTT 情報通信網研究所で収集された道案内対話コーパスに基づいて作例である

ある．

2.1 発話ゴール

従来の対話システム（e.g. [1,8]）における表層発話行為は以下のように3つの観点から分解することができる．我々はこのような分解の結果得られる各々のゴールを発話ゴールとして扱う．3つの観点はシステミック言語学におけるメタ機能に当たる [6].

- 概念構成的ゴール：
  - 個体の参照：
    refer(inst)：個体 inst を参照する
  - 個体属性の伝達：
    attrib(inst,タイプ,type1)：個体 inst の型が type1 であることを伝達する
    attrib(inst,attr1,inst2)：個体 inst の属性 attr1 の値が個体 inst2 であることを伝達する
  - 参照のための個体属性の伝達：
    ref_attrib(ref1,inst1,attr1,inst2)：識別子 ref1 をもつ個体参照の発話ゴールを達成するために inst1 の属性 attr1 の値が inst2 であることを伝達する
- 対人関係的ゴール：
  - ムード：
    mood(inst,mood1)：個体 inst は mood1 というムードを持つ．
  - ゴールの確認：
    confirm(goal)：発話ゴール goal が達成されたかどうかについて聞き手にフィードバックを促す
- テクスト構成的ゴール：
  - ゴール間の順序：
    order(goal1,goal2)：発話ゴール goal1 を goal2 より先に達成する

2.2 表層生成部

表層生成部は発話の意味構造を受けとり，システミック文法を参照して発話の表層表現を生成する．システミック文法を CUF 上に実装する方法については Henschel らの文献 [7] を参照されたい．CUF を用いれば，システムネットワークの型階層を自然に記述することができる．

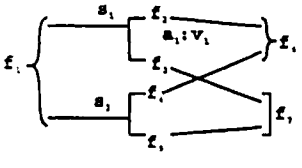

図 2: システムネットワーク

たとえば，図 2のシステムネットワークで表現された型階層は以下の節で定義できる．

(t1) s1 < f1. s2 < f1.
(t2) s1 = f2 | f3. s2 = f4 | f5.
(t3) f6 = f2 & f4
(t4) f7 < f3. f7 < f5.
(t5) f2 :: a1 : v1.

(t1) は素性 f1 がシステム s1 と s2 を包摂することを表す．(t2) はシステム s1 が素性 f2，f3 の選択であることを表す（s2 も同様）．(t3) は素性 f2，f4 がともに選択された場合，f6 も選択され，その逆も成り立つことを表す．(t4) は f3 または f5 が選択されると f7 も選択されることを表す．また，各素性が持つ属性に対する制約は (t5) のように記述する．

このように定義された型つき素性は原子意味構造および発話の意味構造を表現するオントロジーとして利用される．

対話では、(u6) のように文の途中に区切りを入れて相手にフィードバックを促す場合が少なくない．

(u6) 富士見通りという道を（ですね．）
(u7) はい．
(u8) 右に曲がって下さい．

表層生成部は、受け取った意味構造に付与された情報に依存して、1つの発話として表出したり、意味構造の部分構成素だけを表出したりすることができる．

2.3 意味構成部

意味構成部の作業は，発話ゴールの集合を原子意味構造の集合に変換する作業と原子意味構造を組み上げて発話の意味構造を生成する作業からなる．参照する資源は原子意味構造生成規則と談話状況である．談話状況は焦点スタックと個体の既出/未出の情報からなる．

2.3.1 原子意味構造生成規則

原子意味構造生成規則は1つまたは複数の発話ゴールを原子意味構造に変換する規則であり，次のように記述する．

$$S_g, S_c \rightarrow$$
$$(goal : [G_i, G_j, \ldots] \& ass : [F_k, F_l, \ldots])$$

ただし，S_gは発話ゴール $Goal_i$とその識別子 G_iの組 G_i：$Goal_i$の集合，S_cは規則の適用条件の集合，F_iは原子意味構造である．上の規則は，発話ゴール集合 S_gが与えられ，適用条件 S_cが成り立つとき，識別子 $G_i, G_j, \ldots$ に対応するゴール $Goal_i, Goal_j, \ldots$ が原子意味構造の集合 $[F_k, F_l, \ldots]$ によって達成できることを表す．

以下に典型的な規則の例を示す（"&" は素性構造の単一化を表す演算子である．文献 [4] 参照）．

(r1) G:refer(X) →
(goal:[G] & ass:[(原子意味構造 & 個体:X & 意味:Y)])

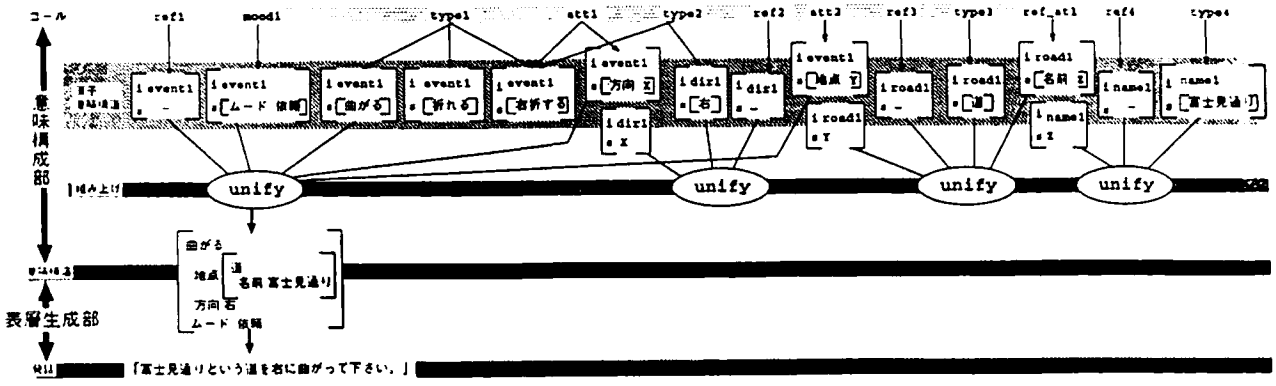

図 4: 発話 (u4) の生成

ref1: refer(event1)
ref2: refer(dir1)
ref3: refer(road1)
ref4: refer(name1)
type1: attrib(event1, タイプ, 曲)
type2: attrib(dir1, タイプ, 右)
type3: attrib(road1, タイプ, 道)
type4: attrib(name1, タイプ, 富士見通り)
att1: attrib(event1, 方向,dir1)
att2: attrib(event1, 地点,road1)
ref_at1: ref_attrib(ref3,road1, 名前,name1)
mood1: mood(event1, 依頼)

意味:名前:Z), (原子意味構造 & 個体:name1 & 意味:Z) が生成される. つぎに, 原子意味構造の組み上げを行う. 上の (原子意味構造 & 個体:road1 & 意味:名前:Z) は, 発話ゴール type3 から生成された (原子意味構造 & 個体:road1 & 意味:タイプ: 道) などと単一化される. 単一化で得られた意味構造の間には部分全体関係 event1 ≻ dir1, event1 ≻ road1, road1 ≻ name1 が成り立っているので, 制約 (c2) により表層生成部へは event1 に対応する意味構造のみを出力する.

load1 に焦点が当たっている対話状況では, ref_at1 のかわりに

att3: attrib(road1, 名前,name1)

が入力となる. このゴールに対し (r5) を用いると, 図 4 と同じ意味構造を出力する. 一方, (r6) を用いると

(u9) その道を右に曲がって下さい.
(u10) そこは富士見通りです.

という発話が生成される. この例では個体 road1 が 2 度参照されているが, これは制約 (c1) による. 制約 (c1) については次の例で説明する

発話 (u1) の生成過程を図 5に示す. 入力ゴールは図 4 に以下の確認のゴールを加えたものである. 発話 (u1)

conf1: confirm(ref3)

では個体 road1 を参照している. road1 は未出の具体物であるので, その存在を述べることによって参照できる. 「～があるのですが」という表現は, この参照の成否に関して相手にフィードバックを促す機能をもつ. またそれと同時に, road1 の参照はその後に続く event1 の説明の一部であり, 説明が継続することも伝えることができる. これを実現する原子意味構造生成規則は次のように記述できる.

(r9) G1:conf(G2), G2:refer(X), G3:attrib(_,_,X), 未出 (X), 具体物 (X) →
(goal:[G1],ass:[(原子意味構造 & 個体:G1 & 意味:(存在 & 対象:Y & ムード:継続)), (原子意味構造 & 個体:X & 意味: Y)])

この例では, road1 について制約 (c1) の適用条件が成り立ち, (原子意味構造 & 個体:road1 & 意味: X5) と (原子意味構造 & 個体:road1 & 意味: X7) の単一化は行わない. 結果として, road1 は 2 度参照される. ただし, 2 度目の参照では適切な代名詞化が必要である.

また, 図 5のゴール集合において, road1 が既出の個体である場合は,

(u11) (で,) 富士見通りという道なんですが
(u12) はい.
(u13) そこを右に曲がって下さい.

という対話が可能である. (u11) の実現には次のような原子意味構造生成規則を用意すればよい.

(r10) G1:conf(G2), G2:refer(X), G3:attrib(_,_,X), 既出 (X) →
(goal:[G1],ass:[(原子意味構造 & 個体:G1 & 意味:(提題 &対象:Z & ムード:継続)), (原子意味構造 & 個体:X & 意味:Z)])