文章生成システムのための システミック文法の開発環境の構築に関する研究

20－8

A Environment for Development of Systemic Grammar

熊野 正* 徳永 健伸 乾 健太郎 田中 穂積
KUMANO, Tadashi TOKUNAGA, Takenobu INUI, Kentaro TANAKA, Hozumi

東京工業大学 工学部
Department of Computer Science, Tokyo Institute of Technology

This paper describes a system for development of a systemic functional grammar (SFG) particularly for text generation. The system provides a graphical user interface which the user employs to describe a grammar. Given a grammar, the system translates it to that of the functional unification grammar (FUG) formalism. Surface generation is performed by unifying an input and the grammar. When debugging the grammar, the user actually attempts surface generation giving some inputs. In this process the system provides various information of intermediate states. This information would be useful for the user to find problems of the current grammar. The user can also modify the grammar through the GUI.

1. はじめに

自然言語生成の分野では，これまでに数多くの研究がなされ，多くの実験システムが構築されてきた．しかし，これらのシステムの多くは十分な言語資源を持たないため，そのまま実用システムに組み込むことは難しい．大規模な文法の開発は，システムの実用化にとって急務であり，効率的な開発のための環境を整える必要がある．

我々は，自然言語生成のための統一的な言語理論としてシステミック言語学 (*systemic functional linguistic*) を，その実装形式として機能単一化文法 (*functional unification grammar*; FUG) [4] を採用し，これまでにFUG を処理する単一化器の実装などを行った [7].

本稿では，システミック文法 (*systemic functional grammar*; SFG) [2] の開発を支援するシステム T^{3}G (TIT Toolkit for Text Generation) の構築について述べる．まず大規模な文法を構築する際に問題となる事柄を検討し，その検討を基に文法記述環境や文法デバッグ環境をグラフィカルユーザインターフェイスによって実装した．

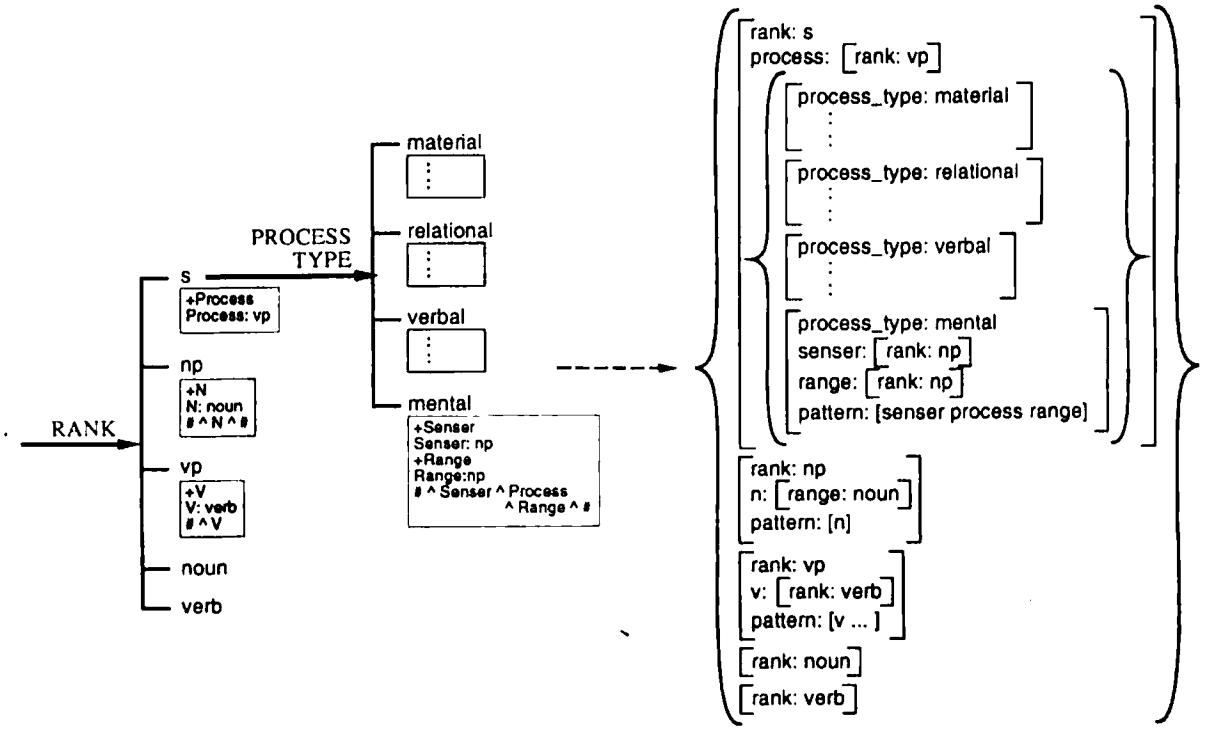
2. システミック文法と機能単一化文法

2.1. システミック文法と自然言語生成

大規模な文法を構築するためには，多様な言語表現を統一的な枠組の中で分析することが必要である．システミック言語学は，チョムスキー流の言語学が言語を構造的側面から分析しようとするのに対して，言語の機能的側面を中心に分析するアプローチである．これは自然言語生成に自然に適用することができると考えられる．実際，既に大規模な英語の生成用文法が開発されている [5,6]．しかしながら，日本語の文法に関する研究はまだ十分でなく，過去に開発された文法の多くは小規模なものである．

システミック言語学では，表層に現れる個々の言語表現を，互いに排他的な意味の集合の中から1つの意味を選択した結果であると考える．この選択は複数のプリミティブな選択の組み合わせとして実現される．個々のプリミティブな選択における選択肢の集合をシステムと呼ぶ．各選択肢には表層に対する制約が対応しており，これを表層化制約 (realization statement) と呼ぶ．システム間には依存関係があり，これをグラフで表現したものをシステムネットワークと呼ぶ．システミック言語学における文法 (lexico-grammar) はシステムネットワークとして記述される（図 1(a)）．

*連絡先：熊野 正　東京工業大学 工学部 情報工学科 徳永研究室
〒152 東京都目黒区大岡山 2-12-1　Tel: (03)5734-2831　E-mail: kumano@cs.titech.ac.jp



(a) システムネットワーク　　(b) FUG

図 1: システムネットワークの FUG への変換

2.2. 処理形式としての機能単一化文法

SFG を実装するための手法にはさまざまなものがある．例えば Penman システムでは手続き的に実装している [5]．これに対し，我々は Kasper の手法 [3] に基づいて，SFG を FUG として記述するアプローチを採っている．Kasper によると，システムネットワークとして表現された文法は，FUG の素性構造（機能記述 (functional description; FD)）に変換することができる（図 1）．

3. 大規模な文法の開発の問題点

3.1. システムネットワークの記述

システムネットワークを記述するとき，一般には紙に手で描いていくのが簡単であるが，ネットワークが大規模になると，一度描いたものを修正するのは困難になる．また，人手による FUG への変換も容易でない．大規模な文法の開発には，計算機上でシステムネットワークを記述するためのツールが必要である．

これまでに，HyperCard を用いたシステムネットワークの記述ツールなど開発されているが，これらのツールではネットワークの大局的な情報を得るのが困難である．従って，大規模なシステムネットワークを記述するのに適した文法エディタは，紙の上にネットワークを描いていくような操作性を持ち，かつシステム間の依存関係を容易に把握できる機能を持っていることが重要である．

3.2. システムネットワークのデバッグ

文法を開発する際，開発中の文法を用いて実際に生成を試みることによって，文法の問題点を発見することができる．しかしながら，例えば単一化器の処理系 (Prolog) のデバッガが出力するトレースログを見るだけでは問題の特定は容易でない．デバッグを容易にするためには，システムネットワーク上での選択の履歴を適切に表示する機能や，単一化の進行を文法開発者の指示に従って止めたり戻したりする機能，WFD の内容の変化を逐次的に表示する機能などが必要である．

4. 文法開発環境の実装

本システムの概要を図 2 に示す．システムの実装は，グラフィカルユーザインターフェイス部分は Tcl/Tk を使用し，その他の部分は SICStus Prolog を使用した．

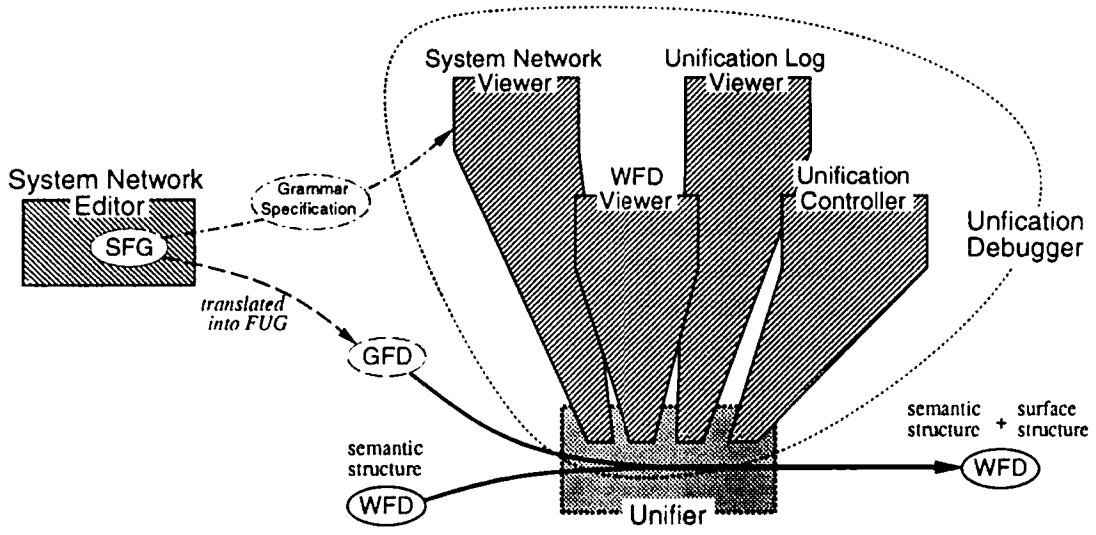


図 2: システムの概要

4.1. システムネットワークエディタ

文法開発者は，システムネットワークエディタを用いて，キャンバス上にシステムを配置し，システム間にリンクを張ることで，各素性・システム間の依存関係をグラフィカルに記述することができる．

エディタ上でシステムネットワークとして記述された文法は，Kasper の手法に従って FUG の機能記述（文法機能記述 (grammar functional description; GFD)）へ変換される．またシステムネットワークの情報は，デバッグ時にシステムネットワークビューア（後述）上に選択された素性を表示するのに用いられる．

4.2. 単一化器

単一化器は，意味情報（入力）をあらわす機能記述 (作業機能記述 (working functional description; WFD)) と GFD の単一化を行い，WFD に表層構造の情報を付加する．

生成システムは全ての可能な表層構造を生成する必要はなく，入力から生成可能な結果が 1 つ得られればよい．そのため単一化器は Prolog の処理の流れに沿って深さ優先に単一化を行う．この戦略は FUF [1] と同様である．

4.3. 単一化デバッガ

単一化ログビューア　単一化処理が意図した通りに進行しているかどうかをチェックするために，単一化の履歴を Prolog のデバッガと同様に逐次表示する単一化ログビューアを実装した（図 3）．

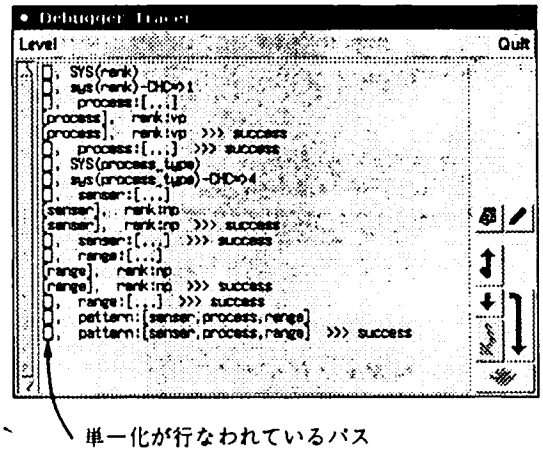


図 3: 単一化ログビューア

単一化コントローラ　文法の問題箇所を特定するためには，単に単一化のログを読むだけでは十分でなく，Prolog のデバッガのように単一化を 1 ステップづつ進行させたり，後戻りしたり，といった単一化のトレース機能があるほうがより効率的である．本システムは Prolog のデバッガと同様の機能を持つ単一化コントローラを備え，マウス操作によって *creep*, *skip*, *undo*, *redo* といったトレースコマンドを単一化器に与えることができる．

WFD ビューア　単一化が失敗する原因を特定するためには，単一化の進行に伴って WFD に付加される素性を確認したり，あるいは単一化の進行に追従して WFD の内容を表示する機能が必要である．単一化器の内部では WFD は Prolog のリスト構造で表現しているので，それをただ単に表示しただけでは必要な情報を把握するのが容易でない．また，一般に WFD は巨大な構造であり，構成素間の単一化 (conflation) の結果，素性の共有も頻繁に起こる．本システムでは，WFD を一般的な FD の表現形式にグラフィカルに表示する WFD ビューアを用意した．このビューアは，現在単一化が行われている部分を常にウィンドウの中央に表示する追尾機能や，構成素間の単一化のリンクを調べる機能，見る必要のないパスをマウス操作で閉じる機能などを備えている（図 4）．

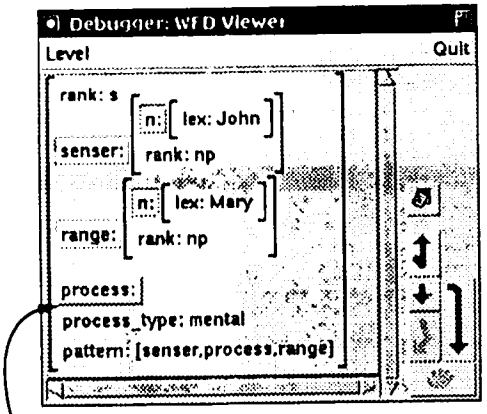


図 4: WFD ビューア

システムネットワークビューア　文法開発者は，作成したシステムネットワークを実際に見て，各システムにおいてどのような素性が選択されたかを確認しながらデバッグ作業を行えることが望ましい．本システムでは，システムネットワークエディタを使って記述したネットワークの上に各システムにおける素性の選択の結果を表示することができる．図 5 は現在素性 s を選択している様子を表している．

5. おわりに

本論文では，システムネットワークの形式で表現される大規模な生成用文法を構築する際の問題点を考察し，それらを解決するような文法開発環境の実装について述べた．本システムは，大規模な情報に GUI を用いて効率的にアクセスできることを目指して実装を行った．

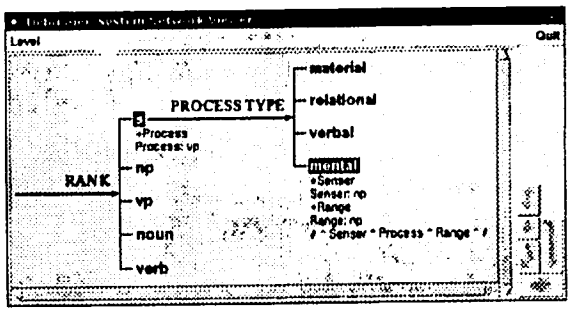


図 5: システムネットワークビューア

今後は，実際にこの開発環境を用いて文法を構築することによって，ツールを改善し，有用な機能を追加していく予定である．さらにこのシステムを使って大規模な文法を開発し，汎用的な文章生成システムを構築する予定である．

参考文献

[1] M. Elhadad. FUF: the universal unifier – user manual, version 5.0. Technical Report CUCS-038-91, Columbia University, 1991.

[2] M. A. K. Halliday. *An Introduction to Functional Grammar*. Edward Arnold, 1985.

[3] R. Kasper. Systemic Grammar and Functional Unification Grammar. In *Systemic Perspective on Discourse*, chapter 9, pp. 176–199. Ablex, 1987.

[4] M. Kay. Functional Unification Grammar: A formalism for machine translation. In *Proceedings of International Conference on Computational Linguistics*, pp. 75–78, 1984.

[5] W. C. Mann. An overview of the Nigel text generation grammar. In *Proceedings of Annual Meeting of the Association for Computational Linguistics*, pp. 79–84, 1983.

[6] W. C. Mann. An overview of the Penman text generation system. In *Proceedings of National Conference on Artificial Intelligence*, pp. 261–265, 1983.

[7] 熊野正．機能単一化文法のための文法開発環境の構築に関する研究，1993 年 2 月．卒業論文．