

## 日本文推敲支援システムにおける書換え支援機能の実現方式†

林 良 彦<sup>††</sup> 菊 井 玄 一 郎<sup>††\*</sup>

文章作成において悪文を避け、わかりやすい文を作成するために多くのノウハウが提案されている。また、これらのノウハウを支援システムとして実現する試みも盛んに行われるようになってきた。しかし、文における表現の書換えまでを支援するシステムはまだ存在しない。われわれは、これらの推敲上のノウハウを規則ベースとして持つことにより、表現の書換えまでを支援するシステムの研究開発を進めている。本論文では、推敲を要しない文に対して高い計算コストをかけることが望ましくないという推敲支援システムの特性を考慮し、構文多義をオンデマンドで解消しながら、規則として定義された部分的な書換えを段階的に行う過程により、書換え支援機能を実現する方式を提案する。このような書換え支援機能の実現形態においては、部分的な書換えを段階的に実行するための制御および、部分的な書換えに伴う内部データの整合性維持が必要となる。本論文で提案する方式においては、アジェンダを実行制御の基本的枠組みとし、その上に制約伝播に基づくデータ整合性維持機構を実現している。オンデマンドの構文多義解消と制約伝播によるデータ整合性維持を組み合わせることで、必要なときに必要最小限の範囲で得た情報を最大限に利用することが可能となり、システム全体の処理効率の向上を図ることができる。

### 1. はじめに

日本語ワードプロセッサの普及にともない、文章を投入するという文書作成の第一段階における問題は、ほぼ解決された。しかし、投入された文書を推敲するという文書作成の第二段階の作業は、人手によって行わざるをえず、またその負荷も大きいため、文書作成におけるネックとなっている。このネックを解消するため、校正支援システム、あるいは、推敲支援システムと呼ばれるシステムが研究開発されている。しかし、これらのシステムの中心的機能は、いわゆる誤字脱字のチェックといった単語表記レベルの校正支援機能<sup>1)~3)</sup>や、文章全体の読みやすさを統計的に診断し得点化する可能性評価機能であり<sup>4), 5)</sup>、文における表現についての推敲を支援するもの<sup>6)~8)</sup>は少ない。また、文における表現の推敲支援を目的とするこれらのシステムにおいても、不適切な表現の存在箇所を指摘することはできても、それをどう書換えればよいのかを支援できるものは少なく、わずかに箱守ら<sup>9)</sup>が係り受けの曖昧性が少なくなるような語順の入れ替えについて検討している程度である。しかし、推敲作業が最終的には文を書換えることであることを考えると、推敲支援システムが推敲を要する表現を網羅的に抽出し、

それに対する妥当な書換え例を提示することができれば、推敲作業を効率化することができると考えられる。以上から、われわれは、書換え支援機能を持つ日本文推敲支援システムの研究開発を進めている。

本論文では、推敲支援システム実現上の要求条件に着目し、構文多義をオンデマンドで解消しながら部分的な書換えを段階的に行う過程により書換え支援機能を実現する方式を提案する。また、このような方式において特に問題となる、部分的な書換えを段階的に実行するための制御および、部分的な書換えに伴う内部データの整合性維持について、アジェンダを実行制御の基本的枠組みとしその上に制約伝播に基づくデータの整合性維持機構を実現する方式を提案する。

### 2. 書換え支援機能の実現形態

#### 2.1 推敲支援システムへの要求条件

システムが推敲作業の支援を行う方法として最も適切なものは、推敲を要する表現（以下、不適正表現と呼ぶ）を網羅的に検出し、それに対する書換え方を書換え例によって示すというものであると考えられる。むしろ、検出した不適正表現に対して妥当な書換え候補をシステムが提供できるというのがその際の前提となる。

一方、推敲支援システムが機械翻訳システムなどの他の自然言語処理システムと比べて大きく異なる点として、すべての表現、文を同一の深さの処理で扱う必要がないという点がある。すなわち、推敲の必要がない文については、そのことのみわかればよく、これを

† Design and Implementation of Rewriting Support Functions in a Japanese Text Revision System by YOSHIHIKO HAYASHI and GEN'ICHIRO KIKUI (NTT Communications and Information Processing Laboratories).

†† NTT 情報通信処理研究所

\* 現在 ATR 自動翻訳電話研究所

ATR Interpreting Telephony Research Laboratories

判別するために深い処理を行うことは望ましくない。また、推敲の必要がある文についても、なるべく低い計算コストで文中に含まれる不適正表現を特定できることが望ましい。

## 2.2 推敲支援規則の記述

悪文を避け、わかりやすい文を作成するために、多くの提案、ノウハウが示されている<sup>9)~11)</sup>。それらの多くは、文中にあるパターンAが存在すれば、それを別のパターンBに書換えることを示唆するものである。ここで、上記のパターンAが本稿でいう検出すべき不適正表現に相当し、上記のパターンBが本稿でいうシステムが提示すべき書換え候補に相当する。本論文では、不適正表現を検出するための規則、それに対する書換え候補を生成するための規則を合わせて推敲支援規則と呼ぶ。

さて、これらのパターンを規則として記述するために必要な情報について考えてみると、検出パターンの多くは、特定の性質を持つ複数の文節がある係り受け構造を成すこととして定義することができる。また、それに対する書換え候補の生成パターンは、係り受け構造の変形と文節に対する形態素調整によって実現することができる。例えば、マニュアル等の文章において好ましくないとされる<sup>11)</sup>「電源を投入しないとシステムを使えない」というような前件、後件双方に否定表現を含むような条件表現に対する推敲支援規則は、単純化して示せば、図1に示すような形で実現することができ、「電源を投入すればシステムを使える」および、「システムを使うためには電源を投入する必要がある」を書換え候補としてユーザに提示することができる。検出規則においてノードに付与された情報は、文節の性質に関する制約条件を示す。また、書換え規則においてノードに付与された情報は、書換えにおいて行うべき形態素調整を示す。以下では、これを形態素調整指令と呼ぶ。

## 2.3 書換え支援機能の実現方針

以上の議論をもとに、書換え支援機能をなるべく低い計算コストで実現するために以下の方針を採る。

(1) 部分的な書換えを段階的に行う：不適正表現ごとに、図1に示したような検出規則、書換え候補生成規則を用意しておく。原則として、検出された不適正表現1つずつに

対して影響する範囲内で部分的な書換え候補を生成し、それに対するユーザの選択により書換えを段階的に行っていく。これにより、検出された不適正表現すべてについての書換え候補を組み合わせる文全体の書換え候補を生成した場合の候補数の爆発を避ける。なお以下では、ある不適正表現に対してシステムが書換え候補を生成し、ユーザがそれを選択することにより部分的な書換えを遂行する過程を部分的書換えタスクと呼ぶ。また、部分的書換えタスクを段階的に実行する過程を推敲過程と呼ぶ。

(2) 早期に不適正表現の成立可能性を検出し、オンデマンドで係り受け多義の解消を行う：前節に述べたように、なるべく低く計算コストで不適正表現を特定することが必要である。しかし、係り受け構造を一意に定めるためには、意味解析や場合によっては文脈解析などの高い計算コストを要する処理を必要とする。よって、現実的な推敲支援システムを実現するためには、不適正表現の特定を、係り受け構造の多義が存在する状況下における不適正表現成立可能性の検出として行っておき、推敲過程において、オンデマンドで必要最小限の範囲で深い解析処理を適用し、多義解消を行っていくことが妥当であると考えられる。この際場合によっては、システムがインタラクティブであることから、ユーザを多義解消の資源として用いる<sup>12)</sup>

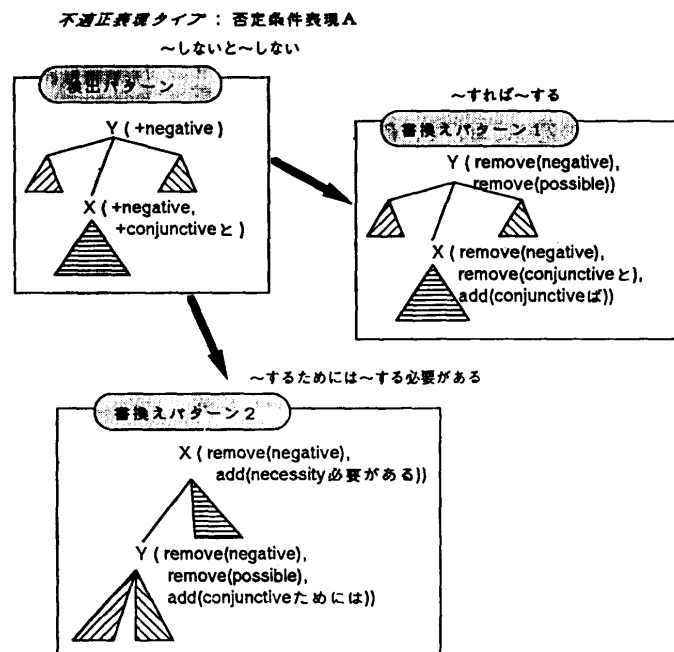


図1 推敲支援規則の例

Fig. 1 An example of revision support rule.

ことも可能である。

(3) 処理の重複を避ける：部分的な書換えを行う度に、それにより変化した入力文を新たな入力文と考えて、文解析処理、不適正表現検出処理を繰り返すのは、安全ではあるが非効率的な方法である。そこで、書換えに対応して文解析、不適正表現検出の結果を更新し、その時点での推敲対象文の文字列とこれらの内部データを常に無矛盾に保つことにより、1文に対して複数の書換え操作を行った場合に対しても、適切な書換え結果の保証と処理の効率化の両立を図る。

#### 2.4 システムの基本的構成

以上のような方針に基づく推敲支援システムの基本的な構成を図2に示す。

不適正表現検出部は、形態素解析結果および、多義の存在する文節間係り受け解析結果から、文中に含まれる不適正表現の可能性を一括して網羅的に抽出する。ここで、形態素解析部は、日本文音声変換システムに使われたもの<sup>14)</sup>をベースとしており、いくつかのヒューリスティックにより多義を解消し一意な結果を出力する。また、文節間係り受け解析部は、日英翻訳システムに使われたもの<sup>15)</sup>の一部を用いており、可能な2文節の係り受け関係を原則としてすべて出力する。いずれについても、実用上十分と考えられる精度を得ている。

推敲過程制御部は、スケジューラとデータ整合性維持機構とから構成される。前者は、部分的書換えの段階的実行を制御し、後者は、部分的書換えに伴って必要となるデータの整合性維持を行う。この過程において、係り受け多義解消部は、オンデマンドで係り受けの多義を解消し、その結果を要求元へ返す。部分的書換え実行部は、生成した書換え候補をユーザに取捨選択させることにより、部分的書換えタスクを実行し、差分解析部は、書換え前後の差分を解析し、それをデータ整合性維持機構に通知する。

本システムを実際利用するためには、推敲したい文章を本システムに送り、書き換えられた文章によって置き換えるユーザインタフェースが必要となる。現在実現しているプロトタイプシステムにおいては、汎用のテキストエディタ Emacs<sup>16)</sup>の1コマンドとして推敲機能を実行できるようにしている。

### 3. 部分的書換えの段階的実行

#### 3.1 部分的書換えタスクの実行前提条件としての不適正表現の検出結果

不適正表現検出部は、1つの不適正表現の検出結果を条件式として表現する。例えば、例文：「電源を投入しないと、システムを使えない。」に対する図1に示した規則による検出結果は、図3に示すように表現する。これは、文節2、4がともに否定表現を含む述語であり、これらの文節間に係り受け関係が成立していることが、「否定条件表現A」という不適正表現成立の必要条件であることを示している。この条件式の左辺は、節が1つのリテラルから構成される連言標準形をしている。以降では、連言で結ばれた各条件リテラルをプリミティブ条件と呼ぶ。プリミティブ条件は、文節の性質に関する条件と文節間の係り受け関係に関する条件に分けることができ、それぞれを文節性質条件、係り受け条件と呼ぶ。

すでに述べたように、システムは不適正表現の検出を多義が未解消な係り受け解析結果に対して行う。よって、係り受け条件は、ある時点で成立が確立していないことがありうる。また、部分的書換えを係り受け構造の変形と形態素調整によって行うため、文節性質条件、係り受け条件の双方とも、推敲過程の進行に伴って成立しなくなることがありうる。

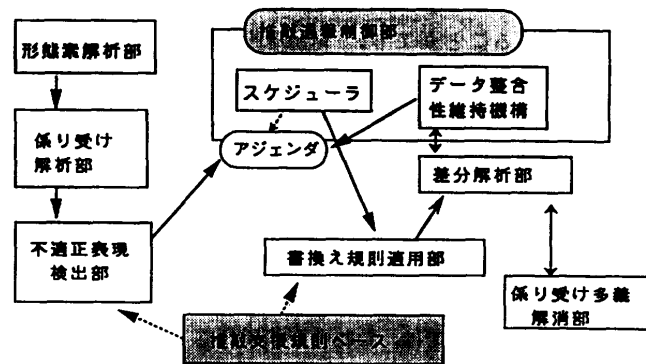


図2 システムの構成  
Fig. 2 Architecture of the system.

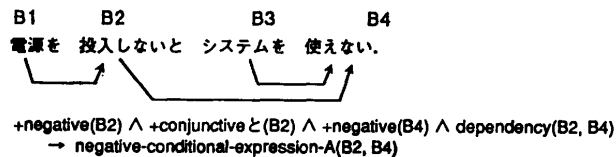


図3 不適正表現検出結果の例  
Fig. 3 An example of the detected inappropriate expression.

このことを不適正表現に対応する部分的書換えタスクの実行の観点からみれば、条件式の左辺は、タスクを実行するための前提条件であるといえる。すなわち、部分的書換えタスクの実行前提条件を構成するプリミティブ条件が1つでも満たされないことが判明した時点で、その部分的書換えタスクは無効なものとする必要がある。

### 3.2 アジェンダと推敲過程マネージャの設定

図3に示した条件式の右辺の不適正

表現に対応する部分的書換えについての知識は、不適正表現のタイプごとにオブジェクトとして定義しておき、推敲過程の開始時に不適正表現の検出結果に従ってインスタンスを生成する。生成されたインスタンスを部分的書換えタスク記述と呼ぶ。図3の例に対する部分的書換えタスク記述の概要を図4に示す。図4に示すように、対応する書換え候補生成規則名、書換えスコープ、静的な優先度などをあらかじめ指定しておく。さらに、検出結果からどの文節をスコープのヘッドとするかについての手続きをメソッドとして定義しておき、インスタンスを生成する際に検出結果からこれを求める。また、そのタスクの状態（有効／無効、実行済み／未実行）を保持するスロットを用意する。これらの部分的書換えタスク記述は、アジェンダ\*に登録され、適切なタイミングで実行されるようスケジューリングされる。

また、左辺で表現される部分的書換えタスクの実行前提条件がプリミティブ条件の連言で表現されることが、さらには、複数の不適正表現で共有されるプリミティブ条件が存在することに着目し、プリミティブ条件とアジェンダに登録されている各部分的書換えタ

```
(Partial-Rewriting-Task
: id 0
: type negative-conditional-expression-A
: rewrite-rule-name Negative-Conditional-Expression-A
: control ((static-priority 70))
: scope-head B2
: status ((active T) (executed NIL)))
```

図4 部分的書換えタスク記述の例

Fig. 4 An example of the partial rewriting task description.

\* 実行すべき／される可能性のある部分的書換えタスク記述を保持するデータ。

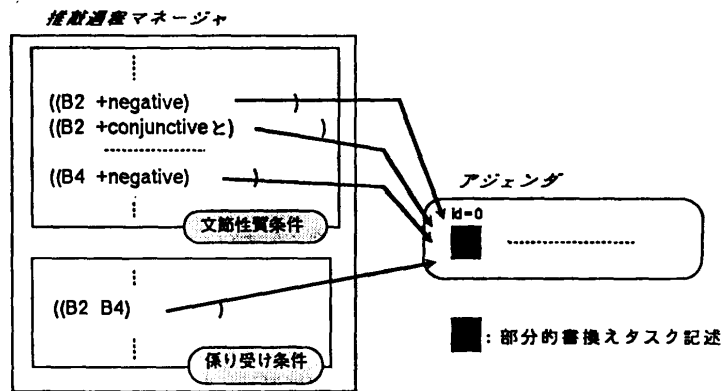


図5 推敲過程マネージャとアジェンダ

Fig. 5 An example of the Revision Process Manager and the Agenda.

クとの依存関係を推敲過程マネージャと呼ぶオブジェクトにより一括管理する。推敲過程マネージャは、文節性質条件とそれをプリミティブ条件に持つ部分的書換えタスクとの依存関係を管理する文節性質条件スロットと、係り受け条件とそれをプリミティブ条件に持つ部分的書換えタスクとの依存関係を管理する係り受け条件スロットの2つのスロットを持つ。図3の例に対する推敲過程マネージャの例を図5に示す。

### 3.3 部分的書換えタスクの段階的実行

部分的書換えタスクの段階的な実行は、以下のような手順で行われる。

(ステップ0) 部分的書換えタスクの選択：スケジューラにより、アジェンダに登録されている部分的書換えタスク群のなかから、無効化されていない未実行のタスクの中で最も高い優先度を持つものを次に実行すべき部分的書換えタスクとして選択する。もし、実行すべきタスクが存在しなければ、処理を終了する。

(ステップ1) 前提条件成立の確認：前提条件のうち、成立が未決定の係り受け条件があれば、係り受け多義解消部を起動し、その成立／非成立を決定する。

(ステップ2) 書換えスコープの決定：部分的書換えタスク記述に指定された書換えスコープに従って、部分的書換の及ぶ範囲を文節列として決定する。この際にも必要があれば、関連する部分について係り受け多義解消を実行する。

(ステップ3) 部分的書換え実行部の起動：部分的書換え実行部には、書換えスコープ内の文節列とそれらの文節間の係り受け可能性を示す二項関係の組が渡される。

(ステップ3-1) 書換え候補の生成：まず、スコープ内の文節間の係り受け可能性から、係り受け非交差

条件を考慮しながらスコープ内の係り受け構造を表現する依存構造木を生成する。一般に、スコープ内の係り受け構造は多義を持つので、依存構造木の列が生成される。次に、これらの木構造に対して、部分的書換えタスク記述に指定されている書換え候補生成規則を適用して、ユーザに提示する書換え候補を生成する。

(ステップ 3-2) ユーザ選択による書換の実行：システムによって提示された書換え候補のいずれかをユーザが選択することによって、部分的書換えを実行する。実行結果としては、書換えられたスコープ内の文節列および、依存構造木を二項関係の組に還元した文節間の係り受け可能性を返す。また、文節に対して適用された形態素調査指令のセットを返す。

(ステップ 4) 後処理：部分的な書換えが行われた場合、書換え前後の差分を差分解析部により解析し、データの整合性維持を行う。ステップ 0 へ行く。

#### 4. 部分的書換えに伴うデータの整合性維持

##### 4.1 整合性維持の対象となるデータ

推敲過程の進行に伴い、部分的な書換えによって推敲対象文の文字列が変化していく。この文字列と以下に示す 3 種類の内部データから成るデータの集合は、推敲過程において常に整合している必要がある。

- (1) 文節内の構造。
- (2) 文節間の係り受け可能性。
- (3) 部分的書換えタスクの実行前提条件。

このうち、文の文字列と(1)との間の整合性は、部分的書換え実行部により自然に達成される。文の文字列と(2)との間の整合性については、前節に示した手順のステップ 1、ステップ 2、ステップ 4 の各段階で、文節間の係り受け関係の成立／非成立が確定した時点で随時行う必要がある。このうち、ステップ 1、ステップ 2 によるものは、直接その情報を知ることができる。ステップ 4 によるものは、部分的書換え実行部の入出力の差分を解析することにより、書換えによって成立／不成立が確立した文節間の係り受け関係を把握することができる。

また、(2)と(3)との間の整合性は、(3)の実行前提条件が文節性質条件および、係り受け条件から成るため、(1)、(2)に連動して行う必要がある。この場合、係り受け条件は(2)と直接対応しているが、文節性質条件と(1)の間には間接的な対応しかない。そのため、文節性質条件についての整合性を維持するためには、適用された形態素調整指令とそれが影響する

文節性質条件との対応を考慮する必要がある。

##### 4.2 制約伝播によるデータの整合性維持機構

前章までの議論により、データの整合性を維持するためのトリガとなる情報は、係り受け多義解消部および、差分解析部から発せられる。本節では、そのような情報が発生したとき、それをシステムに新たに加えられた制約ととらえ、それを影響する範囲に伝播することによりデータの整合性維持を行う方式を提案する。提案する方式においては、データの整合性維持に関わるデータをオブジェクトとして実現し、制約の伝播をメッセージパッシングとして実現する。すなわち、あるオブジェクトにメッセージとしてある制約が送信された場合にそのオブジェクトがとるべきアクションを、メソッドとして定義しておく。本方式における基本的なオブジェクトは、文節オブジェクトと呼ぶオブジェクトと 3.2 節において導入した推敲過程マネージャの 2 つである。

###### 4.2.1 文節オブジェクト

入力文を構成する各文節に対して文節オブジェクトを割り付ける。文節オブジェクトのスロットとしては、文節内の構造を保持するための構造スロット、係り先となりうる文節オブジェクトのリストを保持する係り先候補リスト、係り元となりうる文節オブジェクトのリストを保持する係り元候補リストの 3 つのスロットを用意する。これにより、対象文に対するある時点における可能な係り受け構造が、ノードを文節とし、リンクを文節間の係り受け可能性とするネットワークとして内包的に表現<sup>12)</sup>される。また、いずれかの不適正表現検出結果のプリミティブ条件となっている文節性質条件名のリストを保持するスロット(文節条件名リスト)も用意し、それに文節条件の名のリストを保持しておく。なお、この情報は、不適正表現検出結果から部分的書換えタスク記述を生成する際に同時にセットしておく。

さて、文節オブジェクトにメッセージとして通知される制約と、それに対してとるべきアクションを整理すると次のようになる。今、注目する文節オブジェクトを X とする。

- (1) ある文節 Y へ係ることが確定した：

(1-a) X の係り先候補リストから Y 以外のものを除く。除去した各文節オブジェクトに対して、その係り元候補リストから X を除かせる。さらに、X からこれらの文節に係らないことを推敲過程マネージャに通知する。

(1-b) XとYの間に存在する各文節オブジェクトに対して、XからYへの係り受けが確定したことにより非交差条件を満たさなくなる係り受け関係を検出し、それを(2)のメッセージにより除去させる。

(2) ある文節Yへ係らないことが確定した：

(2-a) Yに対して、Yの係り元リストからXを除かせる。

(2-b) Xの係り先リストからYを除く。その結果、もし係り先リストの要素が1つになったら(これをZとする)、自分に対して、Zを引数として(1)のメッセージを送信する。

(2-c) 推敲過程マネージャに、XからYへの係り受けの不成立が確定したことを知らせる。

(3) 文節の内容が変更された：この場合、変更の原因となった形態素調整指令の系列が引数で渡される。

(3-a) 文節条件名リストに保持している各文節条件のうち、引数で与えられた形態素調整指令により影響を受ける可能性があるものを求める。これは、形態素調整指令とそれにより影響を受ける文節性質との関係を記述したテーブルをあらかじめ用意しておくことにより実現することができる。

(3-b) 上記のうち、条件成立を再度確認する必要があるものについては、対応する文節性質条件のチェックルーチン呼び出し、条件が成立しているかをチェックする。

(3-c) Xと成立しなくなった文節条件名の組を引数として、推敲過程マネージャにその旨を通知する。

なお以上において係り受け構造の多義の解消に関しては、非交差条件、係り先の唯一性条件という一般的に成立すると考えてよい絶対制約<sup>12)</sup>についてのみを扱っている。一方、ヒューリスティックに相当する選好的な規則は、係り受け多義解消部において扱っている。

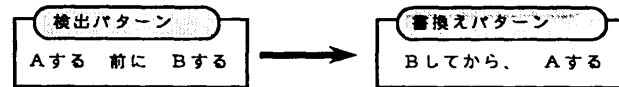
#### 4.2.2 推敲過程マネージャへのメッセージ

推敲過程マネージャにメッセージとして通知される制約と、それに対してとるべきアクションを整理すると次のようになる。

(1) 文節Xから文節Yへの係り受け不成立が確定した：X、Yの組に依存する部分的書換えタスクを係り受け条件スロットから検索し、それらの部分的書換えタスクをすべて無効化する。

(2) 文節Xが文節性質条件Pを満たさなくなっ

不適正表現タイプ：操作手順逆順表現



不適正表現タイプ：曖昧否定表現

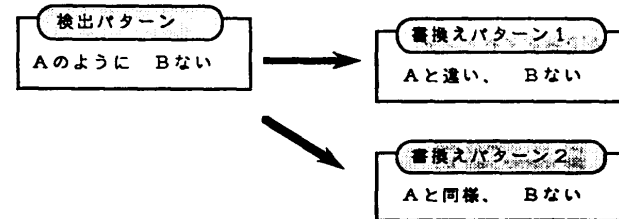


図6 推敲支援規則の例

Fig. 6 Other examples of revision support rules.

た：X、Pの組に依存する部分的書換えタスクを文節性質条件スロットから検索し、それらの部分的書換えタスクをすべて無効化する。

#### 4.3 データの整合性維持の例

以下では、図1に示した推敲支援規則に加え、図6に示す2つの規則\*が存在するものとし、例文：「システムを立ち上げる前に電源を投入しないと、システムを前回のようには使えない。」に対する係り受けネットワーク、推敲過程マネージャ、アジェンダが図7に示すように構成されているものとして、データ整合性維持の過程（特に部分的書換えタスクの実行前提条件について）を説明する。

まず、タスクに記述されている優先度を基にスケジューラがID=2のタスクを選択する。前提条件は、満たされるので、スコープ決めを行って書換え候補を提示する。スコープ決めを行う際に、文節3の係り先が問題になるため、係り受け多義解消部が起動され、これを解消する。今、文節3が文節5に係ることが確定したとすると、次の書換え候補が生成される。

候補1：「システムを立ち上げる前に電源を投入すれば、システムを前回のようには使える。」

候補2：「システムを前回のようには使うには、システムを立ち上げる前に電源を投入する必要がある。」

また、上記において、文節3が文節5に係ることが文節3のオブジェクトに通知されるので、さらに、文節3が文節8に係らないことが推敲過程マネージャに通知される。これにより、文節3から文節8への係り受けをプリミティブ条件として持つID=1のタスクが

\* 図1と同様に注釈付きの木構造の変換として実現されるが、図6では省略して記述してある。

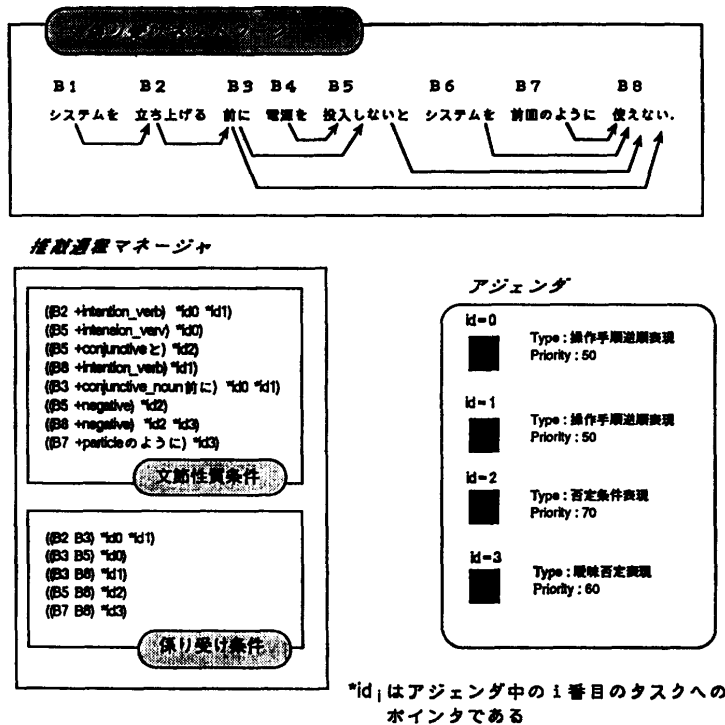


図 7 例文に対するデータ構造  
Fig. 7 Data structure for the example sentence.

無効化される。

ここで、ユーザが候補 2 を選択したとする。すると、文節 8 から否定表現が除去されたことが、文節 8 の文節オブジェクトを経由して推却過程マネージャに通知されるので、推却過程マネージャは、文節 8 が否定表現を持つという文節性質条件をプリミティブ条件として持つ ID=3 のタスクを無効化する。以上により、残るタスクは ID=0 となり、これが処理対象となる。このタスクの実行前提条件は満たされるので、さらに次の候補が生成される。

候補 3: 「システムを前回のように使うには、電源を投入してからシステムを立ち上げる必要がある。」

ユーザがこの候補を選択すると、最終的に例文は、候補 3 のように書換えられる。

## 5. 有効性の評価

### 5.1 評価実験の概要

本論文で提案した書換え支援機能の実現方式の有効性を調べるために、小規模な評価実験を行った。2つの文群 (A 群, B 群と呼ぶ) に対して実際に本システムにより書換え実験を行った。A 群は、現在のところ実現している推却支援規則が正しく働くことを確認す

るために作成した機能試験文群であり、全部で 30 文である。これに対し B 群は、すでに出版されている計算機システムの概説マニュアルからとった文群であり、全部で 61 文である。なお本実験は、推却支援システムとしてどれだけユーザにとって役立つかという観点ではなく、あくまで提案した方式の有効性を調べるという観点から行った。そのため、係り受け多義解消の失敗による影響を受けないよう、ユーザに対する問い合わせにより多義解消を行った。

表 1 に両群の文データの基本的な特性を示す。B 群は実際の文であるため、A 群に比べ長い文が多い。このため、推却過程に入る直前における曖昧さ推定数\* も B 群のほうがかなり大きい値となっている。

表 2 に本実験において 1 文あたりに何回の部分的書換えが行われたかの分布を示す。A 群のほうが書換えが生じ

た文が多いのはその性格から当然であるが、B 群においても約半数の文に対して書換えが生じている。

### 5.2 有効性の考察

#### (1) 書換えに伴う再処理の不要性

本方式の最大の利点の 1 つに、1 文について複数回の部分的書換えが生じるとき、部分的な書換えに伴う

表 1 評価対象文の特性  
Table 1 Characteristics of the sentence data.

| 特性データ    | A 群  | A' 群 | B 群    | B' 群   |
|----------|------|------|--------|--------|
| 平均文字数    | 29.6 | 32.0 | 44.7   | 59.4   |
| 平均文節数    | 6.1  | 6.4  | 7.8    | 10.8   |
| 平均曖昧さ推定数 | 5.4  | 4.8  | 1530.5 | 2465.8 |

表 2 部分的書換え回数の分布  
Table 2 Distribution of the number of rewriting operations per sentence.

| 書換え回数 | A 群 | B 群  |
|-------|-----|------|
| 0     | 5 文 | 28 文 |
| 1     | 16  | 20   |
| 2 以上  | 9   | 13   |

\* 文献 12) と同義で用いている。各文節ごとの係り先候補数を単純に掛け合わせたものである。実際の多義数は必ずこの値以下となる。

表 3 部分的書換えタスクの処理結果  
Table 3 Results of the partial rewriting tasks on the agenda.

| 処 理 結 果                | 説 明                                | A 群     | B 群     |
|------------------------|------------------------------------|---------|---------|
| 適用された<br>有効候補<br>リジェクト | ユーザがいずれかの候補を選択<br>ユーザがすべての候補をリジェクト | 34<br>3 | 48<br>7 |
| 無効化された<br>原因 a         | 他の書換えによって文節条件が不成立となったもの            | 9       | 5       |
| 原因 b                   | 他の書換え／係り受け多義解消によって係り受け条件が不成立となったもの | 2       | 7       |
| 原因 c                   | 係り受け条件の成立を確認しようとしたら成立しないことが判明したもの  | 2       | 2       |

再解析などの処理が不要なことがあげられる。複数回の部分的書換えが生じた文群（A' 群、B' 群とする）の基本的特性は、表 1 に示すとおりであり、A' 群については A 群全体とほぼ同等であるが、B' 群については B 群全体の平均をかなり上回っている。本方式における再解析および不適正表現の再検出の計算コストは、文節数の 2 乗に比例するから、この点は有効に働いていると考えられる。

## (2) 書換え不要文の早期判別

本方式では、係り受け構造の多義をすべて解消してから不適正表現検出処理を起動するのではなく、各文節の係り先の可能性を求めた段階で不適正表現検出処理を起動する。B 群における書換え不要文は 28 文であったが、そのうちの 23 文を不適正表現検出処理の結果から判別することができた。これら 23 文の平均曖昧さ推定数は、935.0 と B 群全体に比べて少ないが、これらの曖昧性を解消することなく書換えが不要であることを判別できることは、本方式の有効性を示すものである。

(3) 制約伝播による不要な書換えタスクの無効化  
不適正表現検出処理の結果に基づいてアジェンダに登録された部分的書換えタスクが推敲過程においてどの程度実行／無効化されたかについての結果を表 3 に示す。本方式の有効性に関するものは、無効化したタスクに関する部分である。原因 a～b は、無効化されたタスクとは直接関係しない間接的な原因によるものであるから、制約伝播によるデータ整合性維持の有効性を示すものと考えることができる。原因 c は、そのタスクの実行がスケジュールされ前提条件の確認を行う際に、はじめて係り受け条件の不成立を係り受け多義解消部から通知されるという直接的な原因である。このような原因によるタスク無効化は、結果的にはタスクを実行しないようにするために計算コストをさくことになるため、なるべく少ないことが望ましい。この点

を改良するためには、タスク自体の優先度だけでなく、前提条件である係り受け条件の尤度を取り入れたスケジューリング方略を検討する必要がある。

## 6. おわりに

本論文で提案した方式は、部分的な書換えを段階的に進めていく過程により、表現の書換えを支援する機能を推敲支援システムに実現するものである。本方式において特徴的なことは、構文多義を解消しない段階で部分的な書換えの可能性を抽出しておき、それらの実行に対する期待に駆動される形でオンデマンドに構文多義の解消を行いながら、実行可能な部分的な書換えを行っていくことである。これにより、部分的な書換えに伴う再解析などの重複した処理を避けられること、推敲を要しない文を早期に判別できること、さらに不適切な部分的書換えの実行を効率よく排除できることを示した。

現在までに、マニュアル文章むけに約 30 種類の不適正表現に対する推敲支援規則を実現した。今後は、さらに規則を充実させることと推敲支援システムとしての有効性の評価を並行して進めていく予定である。

本論文で提案した方式における問題点としては 2 点があげられる。1 つは、部分的書換えタスクの実行制御が主として静的な優先度に基づいているため、推敲過程の進行状況に応じた柔軟な実行制御が困難な点である。もう 1 つは、失敗からの回復機能を持っていないことである。例えば、ヒューリスティックの適用による多義解消が失敗したことによる不適切な書換え候補をユーザが誤って選択してしまうような場合に、その決定を覆すことが現在の方式ではできない。

これらの問題は、いずれもある決定（アジェンダからのタスクの選択、ユーザの書換え候補選択など）を一種の仮説と考えることにより解決できる可能性がある。そのために、ATMS<sup>13)</sup> などの仮説管理機構の併

用についても検討を進めていきたい。

**謝辞** 日頃熱心に議論してくださる NTT 情報通信処理研究所メッセージシステム研究部の皆様に感謝します。また、本研究の内容について貴重なコメントをいただいた知識処理研究部石田亨主幹研究員に感謝します。

### 参 考 文 献

- 1) 池原, 安田, 島崎, 高木: 日本文訂正支援システム (REVISE), 研究実用化報告, 36, No. 9, pp. 1159-1169 (1987).
- 2) 小山, 斎藤, 小林, 小山: 文章校正支援機能における日本語解析, 情報処理学会自然言語処理研究会資料, 69-2 (1988).
- 3) 鈴木, 武田: 日本語文書校正支援システムの設計と評価, 情報処理学会論文誌, Vol. 30, No. 11, pp. 1402-1412 (1989).
- 4) 建石, 小野, 山田: 日本文の読み易さの評価式, 情報処理学会文書処理とヒューマンインタフェース研究会資料, 18-4 (1988).
- 5) 大用: 次世代ワープロの決め手となるか 校正支援/可読性評価ツール, 日経バイト, 1988年3月号, pp. 96-104 (1988).
- 6) 安井, 中西, 小沢, 安西: 日本語スタイルチェッカーのユーザインタフェースに関する研究, 第38回情報処理学会全国大会論文集, 2J-5 (1989).
- 7) 林, 高木: 日本文推敲支援機能の実現方式, 1990年度人工知能学会全国大会論文集, pp. 399-402 (1989).
- 8) 箱守, 杉江, 大西: 日本語を対象とした文評価システムに関する研究, 情報処理学会自然言語処理研究会資料, 65-7 (1988).
- 9) 岩淵悦太郎 (編): 悪文 (第3版), 日本評論社 (1984).
- 10) 千早耿一郎: 悪文の構造, 木耳社 (1979).
- 11) テクニカルコミュニケーション研究会 (編): ユーザマニュアルのための文章・用字用語スタイルブック, 日経 BP 社 (1988).
- 12) 渡部, 丸山: 制約依存文法に基づいた日本語解

析支援システム, 情報処理学会自然言語処理研究会資料, 69-6 (1988).

- 13) de Kleer, J.: An Assumption-based Truth Maintenance System, *Artif. Intell.*, Vol. 28, pp. 127-162 (1986).
- 14) 宮崎, 大山: 日本文音声出力のための言語処理方式, 情報処理学会論文誌, Vol. 27, No. 11, pp. 1053-1061 (1986).
- 15) 白井 諭: 日英翻訳システム ALT-J/E におけるテーブル駆動型係り受け解析法, 第34回情報処理学会全国大会論文集, 5W-6 (1987).
- 16) Stallman, R., 竹内, 天海監訳: GNU Emacs マニュアル, 共立出版 (1988).

(平成2年8月15日受付)

(平成3年5月7日採録)



林 良彦 (正会員)

1959年生。1981年早稲田大学理工学部電気工学科卒業。1983年同大学院博士前期課程修了。同年日本電信電話公社入社。現在, NTT 情報通信処理研究所メッセージシステム研究部主任研究員。自然言語処理の研究に従事。1990年度人工知能学会全国大会優秀論文賞。人工知能学会会員。



菊井 玄一郎 (正会員)

1984年京都大学工学部電気工学科卒業。1986年同大学院修士課程(電気工学第2専攻)修了。同年NTT 情報通信処理研究所入所。1990年2月より(株)エイ・ティ・アール自動翻訳電話研究所データ処理研究室に出向。現在に至る。これまでに、機械翻訳、推敲支援システムなどの研究に従事。1990年度人工知能学会全国大会優秀論文賞。人工知能学会会員。