修 士 論 文

法令文の論理式への変換
-原子文について-

北陸先端科学技術大学院大学
北陸先端科学技術大学院大学情報科学研究科情報処理学専攻

北田 安希雄

2006 年 3 月

修 士 論 文

法令文の論理式への変換
-原子文について-

指導教官　島津明 教授

審査委員主査　島津明 教授
審査委員　東条敏 教授
審査委員　白井清昭 助教授

北陸先端科学技術大学院大学
北陸先端科学技術大学院大学情報科学研究科情報処理学専攻

410037 北田 安希雄

提出年月: 2006 年 2 月

概 要

我々の社会の構造や機能の基本的部分は各種の法律や法規によって明示的に記述されている。したがって法規や法律は、社会の構造や機能を使う情報システムを規定する一種の仕様と見ることができる。したがって、これを形式的に表現することができるなら法推論等により、情報システムを検証することができる。このためには自然言語の法令文は計算機が推論することのできる論理表現で表される必要がある [1]。本研究は、自然言語で書かれた法令文書を入力として、その法令文書に書かれた内容を述語論理式に変換することを目的とする。

我々は、法令文を論理表現に変換する方法として、法令文全体の論理構造への変換、要素の原子文への変換という段階的な方式を考え、本研究は原子文への変換を行う。

原子文は、述語動詞と名詞との意味的な関係、すなわち深層格を表現するものであるので、それぞれの述語動詞に対する格解析が必要となる。また、法令文に出現する表現、例えば、「改善するよう努める」や「有罪であると認める」といったように、「(述語動詞) + よう + (述語動詞)」や「(述語動詞) と (述語動詞)」といった構造においては述語動詞が述語動詞の対象や目的となることがあり、論理式には、これらことを表現する必要がある。さらに、「区民に対する警察署の協力」といった句では、「協力」の動作主格は「警察署」、対象格は「区民」となっている。このようなサ変名詞がとる深層格も、論理式で表現する必要がある。本研究では以上に述べた構造の解析や格解析を行う。

まず、実際の法令文を基にして格フレーム辞書を構築する。千代田区生活環境条例 全 28 条と富山県条例第 54 号「情報通信技術の利用に関する条例」全 10 条に出現する 129 種類、計 431 個の述語動詞から、格フレーム辞書を構築した。辞書に載せた内容は、これらの述語動詞がどのような名詞を深層格としてとっているか、その名詞が深層格として取られた頻度、深層格の名前、そして付随する表層格である。この辞書の構築は、深層格を判断する必要があるため人手で行った。この結果、129 種類の述語動詞の情報を持つ格フレーム辞書ができた。

次に、構築した辞書を用いて格解析を行う。格解析の手法は、まず、法令文を JUMAN [2]、KNP [3] により、形態素解析、構文解析を行う。次に、格フレーム辞書を参照して、格解析の対象とする述語動詞がどのような深層格をとりうるのかチェックし、とりうる深層格としてのスコア付けを、格の候補の文節に対して行う。そして、スコアが閾値を超え、最も高い文節を深層格として決定する。スコアは基本的には、その文節が格フレーム辞書の表層格に一致する格助詞を持つかどうか、主辞となる名詞と格フレーム辞書内の深層格となっている名詞群との意味の類似度、その文節と述語動詞との間にある読点の数などによって決める。加えて、法令文の特徴も考慮する。例えば、法令文に出現する多くの述語動詞が、文頭にある「〜は、」という文節にある名詞を動作主格としているので、この文頭の「〜は、」という文節には動作主格としてのスコアを高く付ける。

最後に、格解析結果から述語動詞や名詞に変数を過不足なく割り振り、それらの関係を宣言する原子文を生成する。

上記に基づいて開発したシステムで、「千代田区生活環境条例」の３条～１２条に出現する述語動詞に対して格解析を行ったところ正しく解析できたのは、７１個の普通動詞のうち６６個、２６個のサ変名詞のうち１６個、２５個の連体修飾語となる述語動詞のうち２０個であった。なお、ここでいうところの「正しく解析できた」とは、「正しい原子文を生成するのに必要な格解析を正確に行えた」こととしている。主な誤りの原因は、構文解析の誤りと、格となる名詞がシソーラスになかったことである。また、「広島市ぽい捨て等の防止に関する条例」全２０条に出現する述語動詞に対しても格解析を行ったところ正しく解析できたのは、６６個の普通動詞のうち４２個、２５個のサ変名詞のうち７個、３２個の連体修飾語となる述語動詞のうち１０個であった。ここでの誤りの原因の約6割が、述語動詞が格フレーム辞書にないことであった。その他の原因としては、千代田区条例に現れた際にとっていた深層格の名詞と、広島市条例に現れた際にとっていた深層格の名詞の類似度が低いために解析を誤った例が6例あった。

これらの実験によって、上記の格フレーム辞書、および格解析により、法令文からその意味を表現する原子文がある程度生成できることが確かめられた。今後は、格フレーム辞書のカバレージを高くすることによって格解析の精度を向上する必要がある。

目 次

図 目 次

表 目 次

第1章　はじめに

1.1　背景と目的

我々の社会の構造や機能の基本的部分は各種の法律や法規によって明示的に記述されている。したがって法規や法律は、社会の構造や機能を使う情報システムを規定する一種の仕様と見ることができる。したがって、これを形式的に表現することができるなら法推論により、情報システムを検証することができる。実際、推論を法令文に関して行うシステムとして、ある事柄が違法であるかを判断する法律エキスパートシステムや、法律文中の矛盾性を発見するシステムが開発されている [7]。しかし、推論のためには自然言語の法令文は計算機が推論することのできる論理表現で表される必要があり [1]、それらのシステムは扱う内容を人手で論理式に変換する必要がある。。そこで本研究においては、自然言語で書かれた法令文書を入力として、その法令文書に書かれた内容を述語論理式に変換することを目的とする。

我々は法令文を論理表現に変換する方法として、法令文全体の論理構造への変換、要素の原子文への変換という段階的な方式を考えており (図 1.1)、本研究では原子文への変換を行う。なお、法令文全体の論理構造を捉えるシステムは江尻（島津研究室）が研究、開発する予定である。江尻のシステムと本研究で開発するシステムを併用することにより、法令文をから論理式への変換を行う。

原子文は、述語動詞と名詞との意味的な関係、すなわち深層格を表現するものであるので、それぞれの述語動詞に対する格解析が必要となる。また、法令文に出現する表現、例えば、「改善するよう努める」や「有罪であると認める」といったように、「(述語動詞) + よう + (述語動詞)」や「(述語動詞) と (述語動詞)」といった構造においては述語動詞が述語動詞の対象や目的となることがあり、論理式には、これらことを表現する必要がある。そこで、本研究で行う格解析では、これらの構造の解析も行う。さらに、「区民に対する警察署の協力」といった句では、「協力」の動作主格は「警察署」、対象格は「区民」となっている。「生じるゴミ」といった句では、「生じる」の対象格は「ゴミ」となっている。これらのようなサ変名詞がとる深層格や、連体修飾語と被連体修飾語の格関係も論理式で表現する必要があり、本研究ではこれらの解析も行う。

これらの格関係の解析のために、本研究では法令文特有の性質を分析し、その性質を考慮した格フレーム辞書の構築、および、格解析を行う。例えば、本研究では実際の法令文を基にした格フレーム辞書を構築する。これは、法令文には「ある述語にはある名詞が格になり易い」といった法令文特有の性質があるとの考えの下、実際の法令文を基にするこ

1.

(a) 土地、建物又は工作物を所有する者は、

(b) それらの清潔を保ち、

(c) 良好な生活環境を保全するよう努めなければならない。

⇓

2.

(a) 所有 (e1) ∧ 人 (x1) ∧ agt(e1, x1) ∧ (土地 (x2) ∨ 建物 (x2) ∨ 工作物 (x2)) ∧ obj(e1, x2)

(b) 保つ (e2) ∧ agt(e2, x1) ∧ obj(e2, e3) ∧ 清潔だ (e3) ∧ obj(e3, x2)

(c) 努める (e4) ∧ agt(e4, x1) ∧ obj(e4, e5) ∧ 保全 (e5) ∧ agt(e5, x1) ∧ obj(e5, x4) ∧ 生活環境 (x4) ∧ 良好 (e6) ∧ obj(e6, x4)

⇓

3.

∀ x1, x2, e1 所有 (e1) ∧ 人 (x1) ∧ agt(e1, x1) ∧ (土地 (x2) ∨ 建物 (x2) ∨ 工作物 (x2)) ∧ obj(e1, x2) →
∃ e2, e3, e4, e5, e6, x3 保つ (e2) ∧ agt(e2, x1) ∧ obj(e2, e3) ∧ 清潔 (e3) ∧ obj(e3, x2) ∧ O (努める (e4) ∧ agt(e4, x1) ∧ obj(e4, e5) ∧ 保全 (e5) ∧ agt(e5, x1) ∧ obj(e5, x3) ∧ 生活環境 (x3) ∧ 良好 (e6) ∧ obj(e6, x3))

1 から 2 へは、意味的に切り分けられた部分からそれぞれ原子文を生成する。
2 から 3 へは、様相演算子、限量子、含意等を用いて生成された原子文を統合している。
なお、agt は動作主格、obj は対象格を表す。O は義務の様相演算子である。

図 1.1: 法令文から論理式への変換

とによってこの法令文特有の性質を格解析に利用しようと考えたためである。また、法令文はかなりの頻度で文頭にある「〜は、」という文節を持っており、法令文に出現する多くの述語動詞が、この文頭の「〜は、」という文節にある名詞を主格としていた。そこで、この文頭の「〜は、」という文節には主格になり易い名詞がある、という法令文特有の性質を考慮した格解析を行う。そして、このような格解析の結果から原子文を生成する。

1.2 本論文の構成

本論文では以上の背景ならびに目的から、法令文から原子文を生成するシステムの構築手法を提案する。

本論文では、まず第２章で格フレーム辞書の構築と格解析を扱った関連研究、また法令文の論理表現に関する関連研究について紹介する。

第３章では、格解析を行うための格フレーム辞書を実際の法令文を基にして構築する手法と、その構築した辞書と JUMAN [2]、KNP [3] の解析結果を用いた格解析 (サ変名詞の格解析、連体修飾の関係の解析を含む) の手法を述べる。

第４章では、格解析の結果を用いて原子文生成をするシステムについて述べる。

第５章では、開発したシステムに対する評価実験を行う。

最後に、第６章では本研究のまとめと問題点ならびに今後の課題について述べる。

第2章　関連研究

2.1　格解析

自然言語文から原子文を生成するには、述語の決定、述語に対する項を決定する格構造解析が必須となる。そのため、格フレーム辞書が必要となる。河原 [5] らは新聞記事のコーパスを構文解析し、構文的曖昧性のない述語項構造のみを抽出クラスタリングすることによって、表層格の格フレーム辞書の自動構築を行っているこの辞書を１次格フレーム辞書として用いて、コーパスに対する格解析を行い、新たに分かる情報を抽出し、２次格フレーム辞書を構築し、この辞書によって二重主語構文、連体修飾の外の関係、格変化といった複雑な言語現象を解析することを可能にしている。

また、笹野 [6] らも新聞記事のコーパスから収集した名詞句の意味解析を国語辞典の定義文を利用して行い、その結果を用いて名詞句の関係解析のための表層格の名詞格フレーム辞書の自動構築を行っている。

本研究では、法令文特有の性質を考慮し、実際の法令文（千代田区生活環境条例と富山県条例第 54 号「情報通信技術の利用に関する条例」）から、人手で述語とその深層格の関係を抽出することによって深層格の情報まで載せた格フレーム辞書を構築する。ここでいうところの法令文特有の性質とは、法令文のある述語動詞には、ある種の名詞を格としてとりやすいといった特徴があるという考えである。実際の法令文から辞書を構築するのは、その特徴を考慮した格解析を行うためである。

また本研究では、法推論等により情報システムを検証することを目的として、格解析の結果から原子文を生成する。

2.2　法令文の論理表現

従来より文法の特徴や辞書を用いて、自動で言語を論理式に変換する手法はある [4] が、曖昧性の扱い、格構造の決定に関しては以前として問題がある。吉野 [7] は法令文書を表すための論理式の表現について議論しており、論理表現には、フレームとデータ、そしてフレームとデータの関係を表すスロットによる表現、「もし〜ならば〜である」という知識の集合による表現、意味ネットワーク、そして論理式などがあるとしている。本研究においては、Prolog といった人工知能用言語への応用や、様相論理や時相論理による記述力の高さのメリットを持つ論理式で法令文を表現することにする。岩本ら [8] は法律文の言語モデルを示し、それにより法律文が持つ言語情報と論理情報の両方を共に記述する表

表 2.1: 法律文における深層格

動作主格 sub	文の主語となるもの 「は」「が」「も」
対象格 obj	文の目的語となるもの 「を」「に」
受け手格 rec	文の目的語の受け手となるもの 「に」「に対して」
条件格 con	法律の成立条件となるもの 「場合には」「ときに」「限り」 「を条件として」
時格 tim	時間的制限を表すもの 「時に」「日に」「までに」「から」
場所格 loc	場所的制限を表すもの 「で」「に」
範囲格 abo	法律の対象とする範囲を表すもの 「につき」「について」
原因格 res	原因や理由を表すもの 「により」「によって」
方法格 way	方法や手段を表すもの 「に従って」「によって」
相手格 par	文の主語の相手となるもの 「と」

現枠組を述べている。田中ら [9][10] は、法令文が前提条件となる要件部とその要件に対して帰結となる効果部から成るという性質 (要件効果構造) を考慮して、法律文の構造を分析している。平松、[11] は、要件効果構造を考慮した法令文の構造、主として並列構造を解析するシステムを開発している。また、長野ら [12] は法令文の分析の結果得られた法令文における全ての格関係 (表 2.1) を示している。本研究においては、条件格以外の格関係を原子文として表現することにした。条件格を表現しないのは、論理式では条件格は原子文ではなく、含意 (→) で表現されるべきであると考えたからである。この解析および論理式への変換は、法令文全体の論理構造を解析する江尻（島津研究室）のシステムによって行う。また、本研究では、表 2.1 のような述語動詞と名詞の関係を表す格関係だけではなく、述語動詞と述語動詞の関係を表す格関係も用いる。これは、法令文では述語動詞が述語動詞の対象や目的となることがよくあるからである。本研究で用いる述語動詞と述語動詞の関係を表す格関係としては、(述語動詞) + よう + (述語動詞)」の構造を表す goa(goal の略) と、「(述語動詞) と (述語動詞)」の構造を表す inc(incident の略) である。

第3章　法令文の格解析

3.1　格フレーム辞書の構築

本研究では、千代田区生活環境条例 全 28 条と富山県条例第 54 号「情報通信技術の利用に関する条例」全 10 条に出現する計 431 個、129 種類の述語動詞から、格フレーム辞書を構築した。辞書に載せた内容は、これらの述語動詞がどのような名詞を深層格としてとっているか、その名詞が深層格として取られた頻度、深層格の名前、そして付随する表層格である。なお頻度は、名詞が並列句で出現した場合、並列句として現れた名詞の数で割っている。例えば、「駅、公園、又は道路を清掃する」の「駅」「公園」「道路」の頻度はそれぞれ 0.33 回である。

法令文に出現する表現、例えば、「改善するよう努める」や「有罪であると認める」といったように、「(述語動詞) + よう + (述語動詞)」や「(述語動詞) + と + (述語動詞)」といった構造においては述語動詞が述語動詞の対象や目的となることがあり、論理式には、これらことを表現する必要がある。そこで、これらの構造を解析するために、格フレーム辞書にはこれらの構造をとりうる述語動詞であるかどうかを記述する。

「図る」と「認める」と「管理」の辞書の例をそれぞれ表 3.1、3.2、表 3.3 に示す。「認める」の辞書の例にある inc が、「(述語動詞) + と + (述語動詞)」の構造を「認める」という述語動詞がとりうるという情報である。また、「図る」の辞書の例にある goal が、「(述語動詞) + よう + (述語動詞)」の構造を「図る」という述語動詞がとりうるという情報である。

この辞書の構築は、深層格を判断する必要があるため人手で行った。この結果、129 種類の述語動詞の情報を持つ格フレーム辞書ができた。

3.2　格解析の手法

本研究における格解析は次のような手順である。

1. 法令文を JUMAN [2]、KNP [3] により、形態素解析、構文解析を行う。
2. 格フレーム辞書を参照して、格解析の対象とする述語動詞がどのような深層格をとりうるのかチェックする。
3. もし、2. において対象としている述語動詞が、「(述語動詞 1) + よう + (述語動詞 2)」や「(述語動詞 1) + と + (述語動詞 2)」の構造をとりうる述語動詞 2 であるとわかっ

表 3.1: 格フレーム辞書の例 「図る」

深層格	表層格	名詞	頻度
agt	ガ	区	2
		県	2
		者	1
obj	ヲ	連携	2
		合理化	1
		向上	1
		改善	2
		推進	1
		啓発	1

表 3.2: 格フレーム辞書の例 「認める」

深層格	表層格	名詞	頻度
agt	ガ	区長	8
obj	ヲ	地域	2
		地区	1
		者	1
inc	ト	述語動詞	1

表 3.3: 格フレーム辞書の例 「管理」

深層格	表層格	名詞	頻度
agt	ガ	飼い主	0.5
		者	4.5
		国	0.5
		東京都	0.5
obj	ヲ	動物	1
		土地	0.33
		建物	0.33
		者	1
		道路	1
		場所	2
goal	ヨウ	述語動詞	＊

て、かつそのような述語動詞 1 があるならその述語動詞を inc、もしくは goal とする。

4. とりうる深層格としてのスコア付けを、述語動詞に直接係っている各文節および、被連体修飾名詞に対して行う。
5. スコアが閾値を超え、最も高い文節を深層格として決定する。閾値は 10 点とした。
6. 4. でスコアが閾値を超える文節がないならば、3. と 4. を対象としている述語動詞に直接係っていない文節に対しても行う。ただし、対象としている述語動詞より後方にある、被連体修飾名詞を除く文節は除外する。
7. 5. でもスコアが閾値を超える文節がないならば、その深層格となる文節は存在しないということにする。
8. 時格と場所格は全ての述語動詞がとりうると考えたので、辞書に載っていなくても、まだ格として決定されておらず、かつそれらしい語があるなら時格、場所格として決定した。

なお、3. において inc もしくは goal となる他の述語動詞が見つかった場合、格フレーム辞書に obj をとりうるという情報があったとしても 4. において obj のスコア付けは行わない。これは、述語動詞が obj と inc または、obj と goal の格を供にとるという事例がほとんどなかったので、この性質を考慮したためである。

8. において時格、場所格として決定する条件は表 3.4 に示す。

各文節へのスコアの付け方は、以下のとおりである。

- 法令文に出現する多くの述語動詞が、文頭にある「〜は、」という文節にある名詞を動作主格としていたので、この文頭の「〜は、」という文節には動作主格としてのスコアを 10 点加えている。また、「〜は、」が、対象格となる場合も数例あったの

表 3.4: 語 w を時格、場所格として決定する条件

時格とする条件	場所格とする条件
w が述語動詞に直接係っている (w が述語動詞に直接係られている)	
w の意味がシソーラスで、 「時間」を上位ノードとして持つ	w の意味がシソーラスで、 「場所」を上位ノードとして持つ
(w が被連体修飾語でないなら) w が「に」を伴う	(w が被連体修飾語でないなら) w が「で」「に」「において」「における」を伴う

で、対象格としてのスコアには 5 点を加えている。

- 法令文において文頭に「何人も、」があった場合、その文に出現するほとんどの述語動詞の動作主格は「何人も、」となっていたので、文頭の「何人も、」には動作主格としてのスコアを 10 点加えている。
- 格フレーム辞書の表層格に一致する格助詞を持つならば、+10 点。
- 副助詞、接続助詞を持つならば、＋ 5 点。
- 文頭の「〜は、」以外の文節においてスコア付けの対象としている文節から格解析の対象としている述語動詞の間にある読点の数を 3 倍した数をスコアから引いている。これは、文頭の「〜は、」以外の文節は、読点を超えたところにある述語動詞の格にはなりにくいという性質を考慮したものである。
- スコア付けの対象としている文節の名詞に対して、格フレーム辞書内の深層格となっている名詞群との意味の類似度によって、スコアを付ける。
- 対象としている述語動詞が連体修飾語であった場合のその被連体修語にはスコアを 20 点加えた。これは、被連体修飾語はかなりの頻度で連体修飾語となる述語動詞の何らかの格になるからである。すなわち、連体修飾語と被連体修飾語の格関係は、被連体修飾語と格フレーム辞書内の深層格となっている名詞群との意味の類似度によってよってのみ決まることとなる。

スコア付けの対象としている文節の主辞 w と、格フレーム辞書にある名詞群 C との意味の類似スコア $sim\_score(w, C)$ は以下のように求める。

まず、単語 e_1, e_2 間の類似度 $sim_e(e_1, e_2)$ を、日本語語彙大系のシソーラスを利用して、以下のように定義する。

$$sim_e(e_1, e_2) = max_{x \in s_1, y \in s_2} sim(x, y)$$

$$sim(x,y) = \frac{2L}{l_x + l_y}$$

ここで、x, yは意味属性であり、s_1, s_2 はそれぞれ e_1, e_2 の日本語語彙大系における意味属性の集合である。$sim(x,y)$ は意味属性 x, y 間の類似度であり、l_x, l_y は x, y のシソーラスの根からの階層の深さ、L は x と y の意味属性で一致している階層の深さを表す。類似度 $sim(x,y)$ は 0 から 1 の値をとる。

スコア付けの対象としている文節の主辞の名詞を w, 格フレーム辞書にある名詞群 C のそれぞれの名詞を $c_1, c_2 \ldots c_n$、その名詞それぞれの頻度を $f_1, f_2 \ldots f_n$ とすると、意味の類似スコア $sim\_score(w,C)$ は

$$sim\_score(w,C) = \frac{\sum_{i=1}^{n} sim_e(w,c_i) \times f_i}{\sum_{i=1}^{n} f_i} \times 10$$

とする。類似スコア $sim\_score(w,C)$ は 0 から 10 の値をとる。

このスコア付けにおいて、格助詞が最も影響するようにしてあるのは、日本語においては格助詞が最も深層格を決める上での手がかりになると考えたためである。

スコア付けの対象とする文節の選び方としては、格となりうる文節は「(名詞) × n + (助詞) × m」という形をとっているので、このような形をとらない文節は、スコア付けの対象から除外した。スコア付けの対象から除外するとは、その文節が確実に格にはならないと決定することである。さらに、「(名詞) × n + (助詞) × m」という形をとっていても、名詞に支配される文節は、スコア付けの対象から除外した。ここでいう「支配する」ということについて説明する。例えば、「北海道に住んでいる兄の到着を私は待っている」という文では、「兄」が「到着」に係っており、「住んでいる」はその「兄」に係っており、さらに「北海道に」はその「住んでいる」に係っている。このような関係において、「到着」が、「兄」と「住んでいる」と「北海道に」を「支配する」ということにする。日本語では、ある述語動詞の格として、他の述語動詞に支配される語がなりうることはあるが、名詞に支配される語が格になることはほとんど見受けられない。この性質を考慮して、名詞に支配される語は、スコア付けの対象外とした。また、ある文節が格として採用されたなら、その文節に支配される語はその他の格の候補からは除外した。これは、日本語では格となる語に支配される語がその他の格となることがほとんどないという性質を考慮したためである。

3.3 格フレームの選び方

格フレームの選び方は、解析対象の述語動詞 v に格フレーム $F_1, F_2, \ldots$ を用いて、格解析を行う。それぞれの解析結果を、$R_1, R_2, \ldots$ として、格フレーム一致スコア FrameAlignment(R_i, F_i) が最も高くなる場合のの F_{max} を v に対応する格フレームとし、v に対する最終的な格解析結果を R_{max} とする。

3.3.1 格フレーム一致スコアの計算方法

格フレーム一致スコア $FrameAlignment(R_i, F_i) =$

格の一致度 $CaseAliginment(R_i, F_i)$ × 意味の類似度 $MeanSim(R_i, F_i)$

とする。

格の一致度 $CaseAliginment(R_i, F_i)$ と、意味の類似度 $MeanSim(R_i, F_i)$ の計算方法を説明するために、格フレーム F_i が、格 $c_{i1}, c_{i2}, \ldots, c_{in}$ を持ち、そのうち解析結果 R_i は、$c_{i1}, c_{i2}, \ldots, c_{il}$ の格を持ち、それぞれ格となる名詞は $r_{i1}, r_{i2}, \ldots, r_{il}$ とする。また、これらの格の辞書にある頻度の合計を $f_{i1}, f_{i2}, \ldots, f_{in}$ とする。$CaseAliginment(R_i, F_i)$ は、「対応付けられた格の頻度の合計 / 格フレーム全ての格の頻度の合計」とし、計算式は以下のようになる。

$$CaseAliginment(R_i, F_i) = \frac{\sum_{j=1}^{l} f_{ij}}{\sum_{j=1}^{n} f_{ij}}$$

意味の類似度 $MeanSim(R_i, F_i)$ の計算式は、以下のようになる。

$$MeanSim(R_i, F_i) = \frac{\sum_{j=1}^{l} f_{ij} \times sim\_score(r_{ij}, c_{ij})}{\sum_{j=1}^{l} f_{ij}} \times \frac{1}{10}$$

3.4 格解析の繰り返し

(3.2) で述べた解析の結果、格解析の対象とした述語動詞の格となる文節が解析されるわけであるが、格として決まった語がサ変名詞であった場合、そのサ変名詞も他の名詞を格としてとりうる。このサ変名詞が格として何をとるかという情報は、原子文の生成には必要な情報であるので、そのサ変名詞を対象にした格解析を行う。

第4章　原子文の生成

本研究では、格解析結果から述語動詞や名詞に変数を過不足なく割り振り、それらの関係を宣言する原子文を生成する。

格解析の結果は、解析対象の述語動詞がどの文節を格としてとっているかという情報であるので、まず、その文節のどの部分を原子文に反映させるか決める必要がある。これには、JUMAN によって自立語と判断された単語を繋げることとした。これにより、例えば「環境美化活動」が変数に割り振られる原子文は「活動 (x1)」のように主辞を表現するだけではなく、「環境美化活動 (x1)」といったように全てを表現することにした。これは全てを表現することが、より法令文の内容を正確に表現した論理式であると考えたためである。

またこのプログラムは、KNP によって並列句と解析された語句を、論理輪記号によって原子文にすることも行う。例えば、「妊婦、障害者、又はけが人」を「妊婦 (x1) $\vee$ 障害者 (x1) $\vee$ けが人 (x2)」にするといった具合である。

格解析の結果から、原子文を生成する手順を以下に示す。なお本研究では、格を持つ語には変数 $e_1, e_2 \ldots e_m$ を割り振り、格を持たない語には変数 $x_1, x_2 \ldots x_n$ を割り振るものとする。

1. 文の末尾から、格を持っている文節を探していく。
2. 格を持っている文節があり、なおかつその文節に対してまだ変数が割り振られていないなら、その文節の主辞に対して e_m が割り振られたことを宣言する原子文を生成する。
   例:走る (e_1)
3. 2. で見つかった文節が持つ格となる文節に変数が割り振られたことを宣言する原子文を生成する。
   例:私 (x_1)
   ただし、その格となる文節にまだ、変数が割り振られてない場合に限る。
   なお、生成された原子文は全て論理積 ($\wedge$) で結ばれる。
   例:走る (e_1) $\wedge$ 私 (x_1)
   この段階で、格となる文節が他の文節を並列句としてとるという構文解析の結果があるなら、その文節にも同じ変数を割り振り、論理和 ($\vee$) でそれらの式を結び、結ばれた式を（）でまとめたものを宣言する。
   例: (私 (x_1) $\vee$ 彼 (x_1))
4. 2. で宣言された格を持つ文節と、3. で宣言された格である文節に割り振られた変数を

引数として、それらの関係すなわち深層格を宣言する原子文を生成する。
例:agt(e_1, x_1)

5. 文頭まで 2.3.4. を行う。

文頭ではなく、文末から 2.3.4 を行う理由は、日本語ではより意味的に重要な述語動詞が後方にある傾向があり、それらの意味的に重要な動詞の原子文から並んでいるほうが見やすいという考えである。

第5章　評価実験

5.1　原子文生成例

本研究で開発したシステムで、実際に法令文を格解析し、その結果から原子文を自動生成した例をいくつか以下に示す。

千代田区第３条第１項
区は、安全で快適なまちを実現するため、具体的な諸施策を総合的に推進しなければならない。

⇓

推進 (e1) ∧ 区 (x1) ∧ agt(e1, x1) ∧ 諸施策 (x2) ∧ obj(e1, x2)

「推進しなければならない」に対してシステムを用いた場合

実現 (e1) ∧ 区 (x1) ∧ agt(e1, x1) ∧ まち (x2) ∧ obj(e1, x2)

「推進しなければならない」に対してシステムを用いた場合

まち (ひらがな表記) は、シソーラスにないので語の意味によるスコアが付かないのだが、
このように構文が単純であれば、格解析および原子文への変換が可能である。

千代田区第３条第３項
区は、第１項に規定する施策の計画及び実施に当たっては、関係行政機関と協力し、密接な連携を図らなければならない。

⇓

当たる (e1) ∧ 区 (x1) ∧ agt(e1, x1) ∧ (計画 (e2) ∨ 実施 (e2)) ∧ obj(e1, e2) ∧ agt(e2, x1) ∧ 施策 (x3) ∧ obj(e2, x3) ∧ 規定 (e3) ∧ agt(e3, x1) ∧ obj(e3, x3) ∧ 項 (x4) ∧ loc(e3, x4)

「当たっては、」に対してシステムを用いた。

千代田区第５条第３項
事業者等は、この条例の目的を達成するため、区及び関係行政機関が実施する施策に協力しなければならない。

⇓

協力 (e1) ∧ 事業者等 (x1) ∧ agt(e1, x1) ∧ 施策 (x2) ∧ obj(e1, x2) ∧ 実施 (e2) ∧ (区 (x3) ∨ 関係行政機関 (x3)) ∧ agt(e2, x3) ∧ obj(e2, x2)

「協力しなければならない」に対してシステムを用いた。

千代田区第８条第３項
区は、違法駐車等の防止に関して広く区民等、事業者等及び関係行政機関の協力を求め、必要な施策を実施しなければならない。

⇓

求める (e1) ∧ 区 (x1) ∧ agt(e1, x1) ∧ 協力 (e2) ∧ obj(e1, e2) ∧ agt(e2, x1) ∧防止 (e3) ∧ obj(e2, e3) ∧ (区民等 (x4) ∨ 事業者等 (x4) ∨ 関係行政機関 (x4)) ∧ par(e2, x4) ∧ agt(e3, x1) ∧ 駐車等 (x5) ∧ obj(e3, x5)

「求め」に対してシステムを用いた。

千代田区第９条第１項
何人も、公共の場所においてみだりに吸い殻、空き缶等その他の廃棄物を捨て、落書きをし、又は置き看板、のぼり旗、貼り札等若しくは商品その他の物品を放置してはならない。

⇓

放置 (e1) ∧ 人 (x1) ∧ agt(e1, x1) ∧ 物品 (x2) ∧ obj(e1, x2)

「放置してはならない。」に対してシステムを用いた。放置の動作主格は「何人も、」であるが、本研究では「何人も、」は「人」で表現することにした。しかし、議論の余地は残る。

千代田区第１２条第３項
ごみの散乱の原因となるおそれのある物の製造、加工、販売等を行う者は、その散乱の防止 について、区民等に対する意識の啓発を図るとともに、回収及び資源化について必要な措置を講じなければならない。

⇓

行う (e1) ∧ 者 (x1) ∧ agt(e1, x1) ∧ (製造 (x2) ∨ 加工 (x2) ∨ 販売等 (x2)) ∧ o bj(e1, x2)

「行う」に対してシステムを用いた。

広島市第３条第１項
市は、吸い殻、空き缶等のぽい捨て、飼い犬のふんの不回収、落書き等美観を害する行為及び喫煙により他人の身体を害する行為の防止に関する施策を実施しなければならない。

⇓

実施 (e1) ∧ 市 (x1) ∧ agt(e1, x1) ∧ 施策 (x2) ∧ obj(e1, x2)

「実施しなければならない」に対してシステムを用いた。

広島市第７条第２項
飲料を自動販売機により販売する者は、当該飲料に係る容器の回収用の箱等を当該自動販売機に附置し、及び当該箱等を適正に管理するよう努めなければならない。

⇓

販売 (e1) ∧ 者 (x1) ∧ agt(e1, x1) ∧ 飲料 (x2) ∧ obj(e1, x2)

「販売する」に対してシステムを用いた。

広島市第８条第１項
空き地を所有し、又は管理する者は、当該空き地への吸い殻、空き缶等のぽい捨ての防止に努めなければならない。

⇓

管理 (e1) ∧ 者 (x1) ∧ agt(e1, x1) ∧ 空き地 (x2) ∧ obj(e1, x2)

「管理する」に対してシステムを用いた。

広島市第９条第１項
何人も、屋外の場所において、喫煙をしようとするときは、携帯用灰皿を携帯するよう努めなければならない。

⇓

喫煙 (e1) ∧ 人 (x1) ∧ agt(e1, x1)

「喫煙」に対してシステムを用いた。

広島市第１０条第１項
市長は、ビラ、パンフレットその他これらに類する物が屋外の公共の場所において散乱しているときは、当該ビラ等を配布し、又は配布させた者に対し、当該散乱しているビラ等を速やかに回収するよう指示することができる。

⇓

散乱 (e1) ∧ ビラ等 (x1) ∧ agt(e1, x1)

「散乱している」に対してシステムを用いた。

広島市第１１条第３項
何人も、屋外の公共の場所において、飼い犬を連れている場合に当該飼い犬がふんをしたときは、当該ふんを回収しなければならない。

⇓

回収 (e1) ∧ 人 (x1) ∧ agt(e1, x1) ∧ ふん (x2) ∧ obj(e1, x2)

「回収しなければならない」に対してシステムを用いた。

5.2 解析結果

本研究で開発したシステムで、格フレームを構築する際に基にした「千代田区生活環境条例」の 3 条〜12 条に出現する 71 個の述語動詞に対して格解析を行った。また、「広島市ぽい捨て等の防止に関する条例」全 20 条に出現する 67 個の述語動詞に対しても行った。解析結果を表 5.1 に示す。

表 5.1: 解析結果

		正解	誤り				
			KNP による構文解析ミス	対象語が語彙大系にない	述語動詞が辞書にない	辞書の格不足	その他
千代田区	普通動詞	66	2	0	0	0	3
生活環境条例	連体修飾	20	2	1	0	0	2
(３条〜１４条)	サ変名詞	16	3	2	0	0	5
広島市	普通動詞	42	3	0	13	3	5
ぽい捨て防止条例	連体修飾	10	1	0	17	0	4
(３条〜２０条)	サ変名詞	7	0	0	7	1	10

両条例とも、１条は条例の目的、２条は語句の定義であって
特殊な文章であると判断し、解析の対象から除外した。

本研究では、法令文の内容を正確に表現する原子文を生成することを目的としているので、表 5.1 における正解とは正しい原子文を生成するのに必要な格解析を行えたこととしている。すなわち、ある述語動詞に対して格となりうる文節を過不足なく発見できたということである。「辞書の格不足」とは、辞書を構築する際には現れなかった深層格が、解析の際に述語動詞に付随したため正しく解析できなかった例である。例えば、「推進する」

は千代田区条例に現れた際には、動作主格と対象格しかとっていなかったが、広島市条例に現れた際には方法格もとっていた。

また、「(述語動詞) + よう + (述語動詞)」と「(述語動詞) と (述語動詞)」の構造を、千代田区条例では 7 例、広島市条例に対する解析では 13 例が解析された。

千代田区条例に対する解析において、語彙大系になかったため解析誤りの原因となった語は、「まち (ひらがな表記)」「まちづくり」「障害物」である。これらの語が主辞となる文節では、意味のスコアが付かないので、スコアが閾値に達せず、格であるのに本システムでは格として採用されずに解析誤りとなることがあった。

千代田区条例に対する解析における表 5.1 の「その他」の事例をいくつか以下に挙げる。

- 法令文には使役形や受動態はあまり現れないので、実装システムではこれらを考慮していない。このため使役形の述語動詞によって誤った例が 1 例あった。
- 「ごみの散乱の原因となるおそれのある物の製造、加工、販売等を行う者は、その散乱の防止 について、区民等に対する意識の啓発を図るとともに、回収について必要な措置を講じなければならない。(第１２条第３項)」に出現する「回収」の対象格は、「おそれのある物」であるが、名詞に支配されているため格の候補から外れてしまった。このような例は 2 例あった。
- 「散乱した場合 (第１３条第３項)」の「散乱する」は「場合」という名詞に係っているが、これらは格の関係にはなっておらず、正しい解析ができなかった。
- 述語動詞とその格の文章が別れており、文脈解析を行わなければ格を解析できない例が 1 例あった。
- 格であるのに、主辞となる語の意味が辞書に載っている名詞の意味と離れていることと、副助詞しか付随していないために、スコアが閾値を超えず格として採用されない例が 1 例あった。

広島市条例に対する解析において、辞書にないため解析ができなかった述語動詞は、「持ち帰る」「類する」「居住する」「準用する」「協働する」「支持する」「走行する」「携帯する」「害する」(「害す」はある) である。

広島市条例に対する解析における表 5.1 の「その他」の事例をいくつか以下に挙げる。

- 千代田区条例に現れた際にとっていた深層格の名詞と、広島市条例に現れた際にとっていた深層格の名詞の類似度が低いために解析を誤った例が 6 例あった。例えば「防止する」の格フレーム辞書にある対象格の名詞群は「駐車」と「散乱」であるが、これらの語と広島条例で現れた際に「防止する」の対象格となっていた「行為」は類似度が低い。
- 実装システムで考慮していない使役形と受動態が原因によって、解析できなかった例が 4 例あった。

- そもそも論理式でどのように表現するべきか決めていない例が 3 例あった。
- 「何人も、屋外の場所において、吸い殻、空き缶等を生じさせたときは、これを居住する場所等に持ち帰り、又はごみ箱、飲料に係る容器の回収用の箱等に捨てるよう努めなければならない。(第６条第１項)」において、「捨てる」の対象格は「吸い殻、空き缶等」であるが、構文構造上、本研究の格解析ではこのような例を扱うのは難しい。
- 「製造を行う者」において、「製造」の動作主格は「者」であるが、このような構造は本研究で実装したシステムでは解析できない。このような例が 3 例あった。

第6章　まとめ

本研究では、格フレーム辞書を実際の法令文を基に構築し、それを用いた格解析を行うシステム、およびその解析結果から原子文を生成するシステムを開発した。またシステムに対する評価実験によって、上記の格フレーム辞書、および格解析により、法令文からその意味を表現する原子文がある程度生成できることが確かめられた。

広島市条例に対する格解析の失敗の原因として「述語動詞が格フレーム辞書にない」という理由が目立つ。したがって、より大量の法令文から格フレーム辞書を構築していくことによって、格フレーム辞書のカバレージを高くしていくことが今後の最も重要な課題である。。その他の、今後の課題を以下に挙げる。

- 使役形、受動態へ対処する必要がある。
- 千代田区条例に対する解析においても、広島市に対する解析においてもサ変名詞の解析精度は他のものに比べて低い。これは、サ変名詞が持つ格は格助詞ではなく、「の」といった接続助詞しか付随しないことも原因ではある。しかし、動詞として現れるより、名詞として現れた時のほうが、格の数が少ない傾向にあることも挙げられる。例えば、「区民は公園を清掃しなければならない」では、「清掃」に対して動作主格、対象格があるが、「公園の清掃をする」といった使われ方の場合、動作主格が問題にならない場合が多い。このようなサ変名詞の特徴を考慮した格フレーム辞書および、格解析が必要かもしれない。
- また、本研究における実装システムでは「環境美化活動」といった複合的な名詞は「環境美化活動」という１つの名詞としてみなすが、法令文の意味をより正確に論理式で表現するには、「環境」と「美化」と「活動」の関係を解析する必要がある。
- 「快適なまちを実現するため、」や「迅速に」といった副詞句、副詞節の解析、および原子文への変換を行う必要がある。
- 図 5.1 において「公共の場所」の「公共」が原子文に反映されていない。これは、「(名詞) + の + (名詞)」といった名詞句の解析が行われていないからである。このような名詞句の解析も今後の課題である。
- いくつかの自然言語の表現を、論理式でどのように表現するか決めなければならない。例えば千代田区条例でも現れた「ボランティアとしての協力」や「不特定多数の者」といった表現である。

- 本研究では、法令文に現れた名詞をそのまま変数を割り振るための、述語として用いた。このため、例えば「道」と「通り」がほぼ同じ意味を持つということがシステムが判断できない。推論を行う際の、このような常識的な知識の扱い方を考える必要がある。

謝辞

本研究を進めるにあたり、多大なご支援、ご指導を頂いた島津先生に深くお礼申し上げます。さらに、貴重なご助言を頂いた白井清昭助教授、山田寛康助手、中村誠助手に厚く感謝致します。

参考文献

[1] 片山卓也, 検証進化可能電子社会 -情報科学による安心な電子社会の実現-, 情報処理, vol.46, No.5, PP515-521 2005.

[2] 黒橋禎夫, 河原大輔, 日本語形態素解析システム JUMAN version 4.0 使用説明書, 東京大学大学院情報理工学系研究科, 2003.

[3] 黒橋禎夫, 日本語構文解析システム KNP version 2.0 b6 使用説明書, 京都大学大学院 情報学研究科, 1998.

[4] Daniel Jurafsky, James H. Martin, SPEECH and LANGUAGE PROCESSING, Prentice Hall, pp501-544, 2000.

[5] 河原大輔, 黒橋禎夫, 格フレーム辞書の漸次的自動構築, 自然言語処理, Vol.12, No.2, pp.109-131 2005.

[6] 笹野遼平, 河原大輔, 黒橋禎夫. 名詞格フレーム辞書の自動構築とそれを用いた名詞句の関係解析, 自然言語処理, Vol.12, No.3, pp.129-144 2005.

[7] 吉野一, 法律エキスパートシステムの基礎, ぎょうせい, pp12-pp24, 1986.

[8] 岩本秀明, 野村浩郷, 法律文の自然言語処理について, 情報処理学会研究報告 91 巻 37 号, 自然言語処理研究報告第 91-NL-83 号, pp7-14 1991.

[9] 田中規久雄, 川添一郎, 成田一, 法律条文の標準構造 -自然言語による法知識処理をめざして-, 情報処理学会研究報告第 93 巻 97 号, pp79-pp86, 1993.

[10] 田中規久雄, 法律効果規定部の意味機能について, 情報処理学会研究報告 98 巻 21 号, pp1-8, 1998.

[11] 平松寛司, 永井秀利, 中村貞吾, 野村浩郷, 要件効果構造に基づく法律文統語構造解析, 自然言語処理, pp41-pp48, 1997.

[12] 長野馨, 永井秀利, 中村貞吾, 野村浩郷, 動詞の機能に基づく法律文の制限言語モデル, 情報処理学会研究報告, 第 93 巻, 第 41 号, 自然言語処理研究報告 第 93-NL-95 号, pp25-32, 1993.

付 録A　構築した格フレーム辞書

本研究で構築した格フレーム辞書を付録として載せる。

表 A.1: 構築した格フレーム辞書

述語動詞	深層格	表層格	名詞:頻度 (回)
当たる	agt	＜ガ＞	区:1
	obj	＜ニ＞	実施:0.5, 計画:0.5
ある	agt	＜ガ＞	障害:1, 必要:4, おそれ:1, 地区:1, 届出:0.5, 協定:1.5, 者:1
	loc	＜ニ：デ＞	交通:1, 地域:1, 地区:1
受ける	agt	＜ガ＞	区:1, 者:3
	obj	＜ヲ＞	委託:1, 命令:2, 認証:1
及ぼす	agt	＜ガ＞	活動:1
	obj	＜ヲ＞	影響:1
	rec	＜ニ＞	青少年:1
応じる	agt	＜ガ＞	区:1
	obj	＜ニ＞	必要:1
行う	agt	＜ガ＞	区:2.33, 区民:0.33, 者:4.33, 区長:2, 機関:4
	obj	＜ヲ＞	支援:1, 活動:3, 啓発:1, 清掃:1, 指定:1, 顕彰:1, 製造:0.33 加工:0.33 販売:0.33 提出:0.5, 販売:0.5, 申請:6
	abo	＜ニツイテ＞	防止:1
	loc	＜デ＞	地域:1
代える	agt	＜ガ＞	機関:1
	obj	＜ヲ＞	もの:1
	rec	＜ニ＞	署名:1
掲げる	obj	＜ヲ＞	措置:1
	loc	＜ニ＞	次:2

述語動詞	深層格	表層格	名詞:頻度 (回)
科する	obj	＜ヲ＞	過料:1
	rec	＜ニ＞	法人:0.5, 人:0.5
限る	agt	＜ガ＞	区長:1
	obj	＜ヲ＞	終日:0.5, 時間帯:0.5
害す	agt	＜ガ＞	者:2
	obj	＜ヲ＞	風俗:1, 環境:2
聴く	agt	＜ガ＞	区長:3
	obj	＜ヲ＞	意見:3
暮らす	agt	＜ガ＞	区民:1
	loc	＜ニ＞	(まち)
講じる	agt	＜ガ＞	区民:1, 団体:0.5, 者:1.5, 警察署:1
	obj	＜ヲ＞	措置:4
	abo	＜ニツイテ＞	回収:0.5, 合理化:0.5
定める	obj	＜ヲ＞	事項:6, 期限:1, 日:1, 地区:1, 方法:1, もの:1
	loc	＜デ＞	規則:9, 条:1,
	agt	＜ガ＞	条例:1, 区長:1
	abo	＜ニ＞	(まちづくり)
妨げる	obj	＜ヲ＞	請求:1
従う	agt	＜ガ＞	者:2
	obj	＜ニ＞	命令:2
資する	agt	＜ガ＞	区民:1, 団体:0.5, 者:0.5
	obj	＜ニ＞	実現:2
示す	agt	＜ガ＞	標識:1
	obj	＜ヲ＞	こと:1
生じる	obj	＜ガ＞	土砂:0.33, 廃材:0.33
	way	＜ニヨリ＞	工事:1
生ずる	obj	＜ガ＞	ごみ:1
	way	＜ニヨリ＞	工事:1
処する	obj	＜ヲ＞	者:4
	rec	＜ニ＞	過料:3, 罰金:1
捨てる	agt	＜ガ＞	者:1
	obj	＜ヲ＞	者:3, 吸殻:2
	loc	＜ニ：デ＞	場所:1, 土地:0.33, 建物:0.33, 物:0.33 上:1, 内:1

述語動詞	深層格	表層格	名詞:頻度 (回)
する	agt	＜ガ＞	人:1, 区民:1, 飼い主:0.5, 管理者:0.5, 者:1
	obj	＜ヲ＞	落書き:1, 禁煙:1, 行為:1, 廃棄:1
損なう	agt	＜ガ＞	状況:1
	obj	＜ヲ＞	環境:1
備える	obj	＜ヲ＞	ファイル:1
	loc	＜ニ＞	計算機:1
高める	agt	＜ガ＞	区民:1
	obj	＜ヲ＞	意識:1
保つ	agt	＜ガ＞	者:2
	obj	＜ヲ＞	場所:1
	rec	＜ヲ：ニ＞	清潔:2
努める	agt	＜ガ＞	区:5, 区民:3, 団体:1, 者:5, 区長 2, 県:3
	obj	＜ニ＞	啓発:1, 整備:2, 改善:1, 除去:1, 確保:1, 活動:1, 回収:0.5, 化:0.5, 実施:1, 周知:1
	goa	＜ヨウ＞	述語動詞
届け出る	agt	＜ガ＞	者:1
	obj	＜ヲ＞	協定:1
	rec	＜ニ＞	区長
伴う	agt	＜ガ＞	ごみ:1
	obj	＜ニ＞	活動:1
取り組む	agt	＜ガ＞	区民:0.5, 者:0.5
	obj	＜ニ＞	美化:1
なる	agt	＜ガ＞	区:1, 行為:1
	rec	＜ト＞	迷惑:1, 一体:1
図る	agt	＜ガ＞	区:2, 者:1, 県:2
	obj	＜ヲ＞	連携:1, 啓発:1, 向上:1, 改善:1, 推進:1, 合理化:1
罰する	obj	＜ヲ＞	者:1
認める	agt	＜ガ＞	区長:8,
	obj	＜ヲ＞	地域:2, 地区:1, 者:1
	inc	＜ト＞	述語動詞
みなす	obj	＜ニ＞	申請:1
	rec	＜ト＞	もの:1
	inc	＜ト＞	述語動詞

述語動詞	深層格	表層格	名詞:頻度 (回)
命じる	agt	＜ガ＞	区長:1
	obj	＜ヲ＞	措置:1
	rec	＜ニ＞	者:1
設ける	agt	＜ガ＞	者:1
	obj	＜ヲ＞	設備:1
求める	agt	＜ガ＞	区:3
	obj	＜ヲ＞	協力:1.5, 負担:1, 参加:0.5
有する	agt	＜ガ＞	者:1
	obj	＜ヲ＞	土地:1
著しい	agt	＜ガ＞	散乱:1
	loc	＜デ＞	地域:1
多い	agt	＜ガ＞	駐車:1
ない	agt	＜ガ＞	必要:1
委託	agt	＜ガ＞	者:1
違反	agt	＜ガ＞	者:4
	obj	＜ニ＞	規定:4
	loc	＜ニオイテ＞	内:1
回収	agt	＜ガ＞	者:4
	obj	＜ヲ＞	物:1
	way	＜ニヨリ＞	設備:1
解除	agt	＜ガ＞	区長:8
	obj	＜ヲ＞	地区:8
改善	agt	＜ガ＞	区:2
	obj	＜ヲ＞	環境:1, 点:1
確保	agt	＜ガ＞	区:1, 県:1
	obj	＜ヲ＞	場所:1, 性:1
活動	agt	＜ガ＞	区民
	obj	＜＞	整備:1
	loc	＜デ＞	地区:1
加工	agt	＜ガ＞	者:1
	obj	＜ヲ＞	物:1
管轄	agt	＜ガ＞	警察署:1
	obj	＜ヲ＞	地区:1

述語動詞	深層格	表層格	名詞:頻度 (回)
簡素化	agt	＜ガ＞	県:1
	obj	＜ヲ＞	手続き:1
管理	agt	＜ガ＞	者:4.5, 飼い主:0.5, 国:0.5, 東京都:0.5
	obj	＜ヲ＞	動物:1, 土地:0.33, 建物:0.33, 者:0.33, 場所:2, 道路:1
	goa	＜ヨウ＞	述語動詞
該当	agt	＜ガ＞	者:3
	obj	＜ニ＞	号:1, 条:1
喫煙	agt	＜ガ＞	区民:1, 者:1
	loc	＜デ＞	上:1, 内:1
規定	obj	＜ヲ＞	施策:1, 者:1, 区長:1, 申請:1, 書面:1, 規則:1, 地域:1
	agt	＜ガ＞	区:1, 区長:1
	loc	＜ニ＞	項:2, 条:1, 規定:1
	rec	＜ト＞	地区:1
協議	agt	＜ガ＞	区:1, 区長:3, 団体:0.25, 区民:0.25, 者:0.25, 機関:0.25
	par	＜ト＞	機関:1, 警察署:3
	abo	＜ニ＞	(まちづくり)
協力	agt	＜ガ＞	区:2, 区民:3.33, 者:1.33, 機関:1.33
	obj	＜ニ＞	施策:2, 美化:0.5, 活動:1.5
	par	＜ト＞	機関:2
	abo	＜＞	防止:1
記録	obj	＜ヲ＞	申請:1
	loc	＜ヘ＞	ファイル:1
禁止	obj	＜ヲ＞	行為:1
計画	agt	＜ガ＞	区:1
	obj	＜ヲ＞	施策:1
啓発	agt	＜ガ＞	区:1, 者:2, 区長:1
	obj	＜ヲ＞	区民:1, 意識:2, 者:1
	abo	＜ニ＞	推進:1
顕彰	agt	＜ガ＞	区:1
	obj	＜ニ＞	貢献:1
建築	agt	＜ガ＞	者:1
	obj	＜ヲ＞	施設:1
貢献	rec	＜ヘ＞	美化:0.5, 浄化:0.5

述語動詞	深層格	表層格	名詞:頻度 (回)
向上	agt	＜ガ＞	区民:0.5, 者:0.5
	obj	＜ヲ＞	意識:1
行動	agt	＜ガ＞	区民:1
公表	agt	＜ガ＞	区長:1, 知事:1
	obj	＜ヲ＞	事実:1, 状況:1
	way	＜ニヨッテ＞	方法:1
告示	agt	＜ガ＞	区長:4
	obj	＜ヲ＞	事項:3, 協定:1
告発	agt	＜ガ＞	区長:1
	obj	＜ヲ＞	者:1
合理化	agt	＜ガ＞	県:1
	obj	＜ヲ＞	手続き:1
参加	agt	＜ガ＞	区民:1
	obj	＜ニ＞	美化:0.5, 活動:0.5
散乱	obj	＜ガ＞	ごみ:1, チラシ:1, 吸殻:0.5, 空き缶 0.5
支援	agt	＜ガ＞	条例:1, 区:2, 区長:2
	obj	＜ヲ:ニ＞	行動:1, 活動:2
資源化	agt	＜ガ＞	者:2
	obj	＜ヲ＞	者:1, 容器:0.33, 包装:0.33, 袋:0.33
施行	obj	＜ヲ＞	条例:1
指定	agt	＜ガ＞	区長:10
	obj	＜ヲ＞	地域:1, 地区:11
	rec	＜ト＞	地区:1
使用	agt	＜ガ＞	組織:2
	obj	＜ヲ＞	組織:1, 電子計算機:1
自覚	agt	＜ガ＞	団体:0.5, 者:0.5
	obj	＜ヲ＞	責任:1
実現	agt	＜ガ＞	条例:1, 区:3, 区民:2
	obj	＜ヲ＞	区:1, 協定:1
実施	agt	＜ガ＞	区:3, 区長:2, 機関:1
	obj	＜ヲ＞	施策:3, 措置:1, 支援:1
指導	agt	＜ガ＞	区:1
	goa	＜ヨウ＞	述語動詞

述語動詞	深層格	表層格	名詞:頻度 (回)
従事	agt	＜ガ＞	者:1
	obj	＜ニ＞	活動:1
周知	agt	＜ガ＞	者:1
	abo	＜ニ＞	債務:1
	rec	＜ニ＞	者:1
所有	agt	＜ガ＞	者:1
	obj	＜ヲ＞	土地:1, 建物:1, 工作物:1
処理	agt	＜ガ＞	者
	obj	＜ヲ＞	物:1
浄化	obj	＜ニ＞	環境:3, 地区:1
除去	agt	＜ガ＞	区:2
	obj	＜ヲ＞	物:2
推進	agt	＜ガ＞	区:1, 区民:1, 区長:1, 県:1
	obj	＜ヲ＞	施策:1, 活動:1, 美化:0.5, 浄化:0.5, 利用:1
請求	obj	＜ヲ＞	賠償:1
	rec	＜ニ＞	もの:1
清掃	agt	＜ガ＞	区:1, 者:2
	obj	＜ニ＞	場所:1, 道路:1
製造	agt	＜ガ＞	者
	obj	＜ヲ＞	物
整備	agt	＜ガ＞	条例:1, 区民:1, 区:2, 者:1
	obj	＜ヲ＞	環境:3, 灯:1, 体制:0.5, 内容:0.5
設置	agt	＜ガ＞	区長:2
	obj	＜ヲ＞	標識:1, 協議会:1
占有	agt	＜ガ＞	者:1
	obj	＜ヲ＞	土地:0.33, 建物:0.33, 者:0.33
阻害	obj	＜ヲ＞	育成:1
存続	obj	＜ヲ＞	指定:2
達成	agt	＜ガ＞	区民:1, 者:1
	obj	＜ヲ＞	目的:2
調整	agt	＜ガ＞	団体:0.25, 区民:0.25, 者:0.25, 機関:0.25
	abo	＜ニツイテ＞	(まちづくり)
提供	agt	＜ガ＞	者:1
	obj	＜ヲ＞	土地:1, 資金:0.5, 場所:0.5

述語動詞	深層格	表層格	名詞:頻度 (回)
締結	obj	＜ヲ＞	協定:3
	agt	＜ガ＞	者:2
提出	agt	＜ガ＞	人:1
	obj	＜ヲ＞	物:2
適用	obj	＜ヲ＞	規定:1
到達	agt	＜ガ＞	申請:1
	rec	＜ニ＞	機関:1
認証	agt	＜ガ＞	区長:2, 区:1
	obj	＜ヲ＞	協定:3
把握	agt	＜ガ＞	区:2
	obj	＜ヲ＞	状況:1, 点:1
配布	agt	＜ガ＞	者:2
	obj	＜ヲ＞	物:1, チラシ:2.5, パンフレット:0.5
配慮	agt	＜ガ＞	人:1
	obj	＜ニ＞	美観:1
販売	agt	＜ガ＞	者:3,
	obj	＜ヲ＞	物:1, 飲料:1, 食料:1
	loc	＜デ＞	場所:1
飛散	agt	＜ガ＞	土砂:0.33, がれき:0.33, 廃材:0.33
	loc	＜ニ＞	場所:1
美化	obj	＜ヲ＞	環境:4
負担	obj	＜ヲ＞	経費:1
変更	agt	＜ガ＞	区長:8
	obj	＜ヲ＞	地区:8
歩行	agt	＜ガ＞	区民:1
	loc	＜ニオイテ＞	場所:1
放置	obj	＜ヲ＞	物品:1, ふん:1, 貼り札:0.5, チラシ:0.5, 看板:1
	loc	＜ニ＞	場所:2, 土地:0.33, 建物:0.33, 物:0.33, 地域:1
	agt	＜ガ＞	飼い主:0.5, 管理者:0.5
保持	obj	＜ヲ＞	環境:1
保全	agt	＜ガ＞	者:1
	obj	＜ヲ＞	環境:1
防止	agt	＜ガ＞	区:1, 者:2, 区長:1
	obj	＜ヲ＞	駐車:2, 散乱:2

述語動詞	深層格	表層格	名詞:頻度 (回)
優先	agt	＜ガ＞	警察署:1
	obj	＜ニ＞	地域:1
要請	agt	＜ガ＞	区:1, 区長:1
	obj	＜ヲ＞	管理:1
	rec	＜ニ＞	管理者:1
	goa	＜ヨウ＞	述語動詞
利用	obj	＜＞	施設:1, 技術:4
	agt	＜ガ＞	者:1
	loc	＜ニ＞	手続き:2
流出	agt	＜ガ＞	土砂:0.33, がれき:0.33, 廃材:0.33
	loc	＜ニ＞	場所:1
連携	agt	＜ガ＞	区:1
	par	＜ト＞	機関:1
清潔だ	obj	＜ヲ＞	土地:0.33, 建物:0.33, 工作物:0.33
	loc	＜＞	周辺:1
著しい	agt	＜ガ＞	散乱:1
	loc	＜デ＞	地域:1
多い	obj	＜ガ＞	駐車:1, 者:1
	loc	＜デ＞	地域:1
ない	obj	＜ガ＞	必要;1

付 録B　格解析と原子文の生成を行うプログラム

本研究で作成した格解析と原子文の生成を行うプログラムを付録として載せる。

```
#!/opt/nlp/bin/perl



### knp の解析結果のファイル名をコマンドから与え、読み込ませる。

$tmp_knp = shift(@ARGV);
$hyouji = shift(@ARGV); #係り先表示をどうするかの変数。1 なら表示する。

undef($/); # $/を undef しとけば、<>はファイル全てを読み込む。（デフォだと 1 行だけ）
open(KIN, "<$tmp_knp") || die "knp :  $!";

$knp_result = <KIN>;

$/ = "\n";   # $/は元にもどす。（戻さないと<STDIN>すら動かない。）

use KNP; # KNP のパッケージ (サブルーチン) を読み込む

# KNP を取り扱うためのオブジェクトを生成
$knp = KNP->new();

# KNP の解析結果を読み込む
$knp->read_sentence($knp_result)->set_kakarimoto_bunsetu->set_bunsetu_type;

close(KIN);




### 格フレーム辞書のセッティング。

$/ = ""; #  $/に空文字列を入れて、<>の区切りを空行にする。
open(DIC, "dic11.txt") || die "dic :  $!";
while(<DIC>){
  chop($_);
  @dic = ();
  @dic = split(/\n/,$_);
```

```
@verb = split(/[\s\t]+/,$dic[0]);

if($verb[0] =~ /(\d)$/){        #for 多義性
  $ex{$`} = 1;
  $tagi{$`} += 1;                #その述語動詞が辞書にあり、かつ多義性がある印
}


$ex{$verb[0]} = 1;             #その述語動詞が辞書に存在するっていう目印

# 辞書の頭文字によって、種類別のハッシュに格納
$i = 1;
while($i <= $#dic) {
  if($dic[$i] =~ /^A/){
    $A{$verb[0]} = $dic[$i];
    if($dic[$i] =~ /^A\*/){
      $A_op{$verb[0]} = 1;
    }else{
      $A_op{$verb[0]} = 0;
    }
  }
  elsif($dic[$i] =~ /^O/){
    $O{$verb[0]} = $dic[$i];
    if($dic[$i] =~ /^O\*/){
      $O_op{$verb[0]} = 1;
    }else{
      $O_op{$verb[0]} = 0;
    }
  }
  elsif($dic[$i] =~ /^R/){
    $R{$verb[0]} = $dic[$i];
    if($dic[$i] =~ /^R\*/){
      $R_op{$verb[0]} = 1;
    }else{
      $R_op{$verb[0]} = 0;
    }
  }
  elsif($dic[$i] =~ /^L/){
      $L{$verb[0]} = $dic[$i];
    if($dic[$i] =~ /^L\*/){
      $L_op{$verb[0]} = 1;
    }else{
      $L_op{$verb[0]} = 0;
    }
  }
  elsif($dic[$i] =~ /^P/){
    $P{$verb[0]} = $dic[$i];
    if($dic[$i] =~ /^P\*/){
      $P_op{$verb[0]} = 1;
    }else{
      $P_op{$verb[0]} = 0;
    }
  }
```

```
    elsif($dic[$i] =~ /^W/){
      $W{$verb[0]} = $dic[$i];
      if($dic[$i] =~ /^W\*/){
        $W_op{$verb[0]} = 1;
      }else{
        $W_op{$verb[0]} = 0;
      }
    }
    elsif($dic[$i] =~ /^a/){
      $a{$verb[0]} = $dic[$i];
      if($dic[$i] =~ /^a\*/){
        $a_op{$verb[0]} = 1;
      }else{
        $a_op{$verb[0]} = 0;
      }
    }
    elsif($dic[$i] =~ /^r/){
      $r{$verb[0]} = $dic[$i];
      if($dic[$i] =~ /^r\*/){
        $r_op{$verb[0]} = 1;
      }else{
        $r_op{$verb[0]} = 0;
      }
    }
    elsif($dic[$i] =~ /^T/){
      $T{$verb[0]} = $dic[$i];
      if($dic[$i] =~ /^T\*/){
        $T_op{$verb[0]} = 1;
      }else{
        $T_op{$verb[0]} = 0;
      }
    }
    elsif($dic[$i] =~ /^Gv/){
      $GG{$verb[0]} = $dic[$i];
      $Gv{$verb[0]} = 1;
    }
    elsif($dic[$i] =~ /^Ev/){
      $EE{$verb[0]} = $dic[$i];
      $Ev{$verb[0]} = 1;
    }
    $i++;
  }
}

$/ = "\n";   # $/は元にもどす。(戻さないと<STDIN>すら動かない。)

close(DIC);
```

日本語語彙大系のセッティング

```
use lib '/home/i4001/a-kitada/perl';
use NTTTH;

# 日本語語彙体系のオープン
NTTTH::OpenFiles;
# 各カテゴリの根ノードからのパスを読む (類似度の計算に必要)
NTTTH::RegistPath;




### 配列の値が最大の添え字を返すサブルーチン
sub max_hai {
  my($i,$max_score);
  $max_score = 0;
  $i =  1;
  while($i <= $#_){
    $max_score = $i if $_[$max_score] <= $_[$i];
  } continue {
    $i += 1;
  }
  return ($max_score);
}




### 初めの要素が最大なら１を、そうでないなら０を返すサブルーチン
sub max_case {
  my($fst,@hoka) = @_;
  my($n);
  foreach $n (@hoka) {
    return(0) if $fst < $n;
  }
  return(1);
}




### 副助詞の文節かどうか判断するサブルーチン
  # 文節番号を引数にして、副助詞なら１、そうでないなら０を返す。
sub fukujosi {
  my($n) = @_;
  my($i, $end);
  $i = $knp->bp_first($n);
  $end = $knp->bp_last($n);
```

```
  while ($i <= $end){
    return(1) if ($knp->pos_list($i) =~ /(副助詞|接続助詞)/);
    $i += 1;
    }
  return(0);
}


#名詞を支配する名詞を候補からはずすサブルーチン
sub meisi_kesi {
  my($nk, $maeC, $hazusi) = @_;
  if($knp->bunsetu_type($nk) =~ /C/){
    if($maeC == 1){
      $hazusi = 1;
    }else{
      $maeC = 1;
    }
  }elsif($knp->bunsetu_type($nk) =~ /Y/){
    $maeC = 0;
  }

  if($hazusi == 1){
    $score[$nk] = -1;
  }

  my ($nmoto_k);
  $nmoto_k = $knp->kakarimoto_bunsetu($nk);
  return (0) if ($nmoto_k == 0);
  my (@ndk);
  @ndk = split(/\s+/,$nmoto_k);
  foreach $nddk (@ndk) {
    meisi_kesi($nddk, $maeC, $hazusi);
  }
}
```

```
### 格のスコア付けをするサブルーチン
sub Case_scorer {
  my($di,$begin,$end,$AorO) = @_;

  my($score_max,$category,$head_hyouki,$head_yomi,$head_no,$scase,$i,$imi_nasi,$buntou_ha,
  $j,$similarity,$max_similarity_pairs,$s_sum,$s_div,@dimi,@s_case,@imi,@ddi,@cat_no,$sou,
  $bp_last,$nannbito_mo,$kakatte_masu,$saki_bun);
  @ddi = split(/\t+/,$di);

  $imi_nasi = 0;
  if(exists $ddi[2]){
  @imi = split(/,\s/,$ddi[2]);
```

```
}else{
 $imi_nasi = 1;
}



#まず、全文節のスコアを０点にセット

@score = ();



#対象にしている述語動詞が名詞に係っているなら、その名詞にもスコアを付ける。
$saki_bun = $knp->kakarisaki_bunsetu($end);
if ($knp->bunsetu_type($saki_bun) =~ /C/){
  $kakatte_masu = $saki_bun +1;
  $score[$saki_bun] = 20;
}else{
  $kakatte_masu = $end;
}



  # 主格ならば、文頭の「は、」に重み (10 点) を付ける。
$buntou_ha = 0;
$nannbito_mo = 0;
$i = $begin;
out:
while($i < $end){
  if ($knp->bunsetu_type($i) =~ /C:(\d)+:ハ/) {
    if ($knp->pos_list($knp->bp_last($i)) =~ /読点/) {
      $buntou_ha = $i;
      last out;
    }
  }elsif ($knp->bunsetu_type($i) =~ /C:(\d)+:モ/) {
    $bp_last = $knp->bp_last($i);
    if ($knp->pos_list($bp_last) =~ /読点/ && $knp->base($bp_last-2) eq "人" &&
    $knp->base($bp_last-3) eq "何") {
      $buntou_ha = $i;
      $nannbito_mo = 1;
      last out;
    }
  }

} continue {
  $i += 1;
}

unless ($buntou_ha == 0) {
  $begin = $buntou_ha + 1;
  if ($AorO == 2) {
    $score[$i] += 10;
  }elsif ($AorO == 1 && $nannbito_mo == 0) {
```

```
    $score[$i] += 5;
  }
}



  #表層格のスコア付け
$i = $begin;
while($i < $end){
  # new 助詞がないならスコアをマイナスにする
  unless ($knp->bunsetu_type($i) =~ /C/){
    $score[$i] = -1;
    next;
  }
  @s_case = split(/:/,$knp->bunsetu_type($i));

  if ($ddi[1] =~ /$s_case[2]/) {
    $score[$i] += 10;
  }elsif(fukujosi($i) == 1){
    $score[$i] += 5;
  }else{
    $score[$i] += 0;
  }

} continue {
$i += 1;
}



#読点をまたぐことによるスコア引き (3 点ずつ)
$i = $end-1;
$j = 0;
while($i >= $begin){
  if($knp->pos_list($knp->bp_last($i)) =~ /読点/){
    $j += 3;
  }
  $score[$i] -= $j;
$i -= 1;
}



#名詞に支配される名詞を格の候補から除外
$i = 1;
while($i < $end){
  meisi_kesi($i, 0, 0);
  $i += 1;
}



#(ここまで、文頭の「は」があるなら、それ以降を考えてきた)
```

```
#(ここからは、A か O であれば文頭も考え、そうでなければ考えない)

if($AorO == 0){
  $i = $buntou_ha + 1;
}else{
  $i = 1;
}


#新意味のスコア付け
$s_sum = 0;
$s_div = 0;


while($i < $kakatte_masu){
  if ($score[$i] < 0){
    next;
  }
  $head_no = $knp->get_bunsetu_head($i);
  $head_hyouki = $knp->hyoki($head_no);
  $head_yomi = $knp->yomi($head_no);
  $head_yomi =~ tr/ア-ン/あ-ん/;


  $category = $NTTTH::Word2Cat{"$head_hyouki,$head_yomi"};

  if($category eq '' || imi_nasi == 1){
    next;
  }
  foreach $im (@imi) {
    @dimi = split(/:/,$im);
    ($similarity,$max_similarity_pairs) =
    &NTTTH::SimilarityWord("$head_hyouki,$head_yomi","$dimi[0],*",1);
    next if($similarity < 0);
    $s_sum += ($similarity * $dimi[1]);
    $s_div += $dimi[1];
  }
  if($s_div == 0){  #0 で割ってしまうバグを抑える
    $s_div = 1;
  }
  $s_sum = ($s_sum / $s_div) * 10;

  $score[$i] += $s_sum;

} continue {
$s_sum = 0;
$s_div = 0;
$i += 1;
}



#KNP の係り先のスコア付け
```

```
  $i = $begin;
  while($i < $end){

    next if ($score[$i] < 0);

    $ko = $knp->kakarisaki_bunsetu($i);
    if($ko == -1){
      next;
    }elsif($end == $ko && $knp->kakari_type($i) eq "D"){
      $score[$i] += 100;
      next;
    }

    $mago = $knp->kakarisaki_bunsetu($ko);
    if($mago == -1){
      next;
    }elsif($end == $mago && $knp->kakari_type($ko) eq "D"){
    #  $score[$i] += 2;
      next;
    }

    $himago = $knp->kakarisaki_bunsetu($mago);
    if($himago == -1){
      next;
    }elsif($end == $himago && $knp->kakari_type($mago) eq "D"){
    #  $score[$i] += 1;
      next;
    }


  } continue {
  $i += 1;
  }

  #print "knp 後 $score[6] 点\n";

  return (@score);
}




### 格の用例数の合計を計算するサブルーチン
sub Yorei_sum {
  my($di) = @_;
  my(@ddi, @imi, @dimi, $y_sum);
  @ddi = split(/\t+/,$di);
  @imi = split(/,\s/,$ddi[2]);
  foreach $im (@imi) {
    @dimi = split(/:/,$im);
    $y_sum += $dimi[1];
  }
```

```
return ($y_sum);
}
```

```
### 格フレームとの意味の類似度を計算するサブルーチン
sub M_sim {
  my($bun, $di) = @_;
  #print "サブ$di\n";
  my($head_no, $head_hyouki, $head_yomi, @ddi, @imi, @dimi, $m_sim, $similarity,
  $max_similarity_pairs);
  $head_no = $knp->get_bunsetu_head($bun);
  $head_hyouki = $knp->hyoki($head_no);
  $head_yomi = $knp->yomi($head_no);
  $head_yomi =~ tr/ア-ン/あ-ん/;

  @ddi = split(/\t+/,$di);
  @imi = split(/,\s/,$ddi[2]);
  foreach $im (@imi) {
    @dimi = split(/:/,$im);
    $category = $NTTTH::Word2Cat{"$head_hyouki,$head_yomi"};
    next if($category eq '');
    ($similarity,$max_similarity_pairs) =
    &NTTTH::SimilarityWord("$head_hyouki,$head_yomi","$dimi[0],*",1);
    next if($similarity < 0);
    $m_sim += $similarity * $dimi[1];
    #print "m_sim:$m_sim similarity:$similarity dimi[1]:$dimi[1]\n";
  }
return ($m_sim);
}
```

```
### スコアの理由を表示するサブルーチン
sub scorer_cause {
  my($bun, $di, $shu, $jutugo) = @_;
  my($i, $total, @s_case, $head_no, $head_hyouki, $head_yomi, $category, @cat_no,$sou,
  @ddi, @imi, $imi_nasi, $toutensuu, $s_sum, $s_div, @dimi, $similarity,
  $max_similarity_pairs, $ha,$bun_last,$saki_bun, $kakatte_masu);
  @ddi = split(/\t+/,$di);
  $imi_nasi = 0;
  if(exists $ddi[2]){
  @imi = split(/,\s/,$ddi[2]);
  }else{
   $imi_nasi = 1;
  }

  $ha = 0;
  $total = 0;
```

```
 # 主格ならば、文頭の「は、」に重み (10 点) を付ける。(表示)
if ($shu == 2) {
    if ($knp->bunsetu_type($bun) =~ /C:(\d)+:ハ/) {
      if ($knp->pos_list($knp->bp_last($bun)) =~ /読点/) {
        $total += 10;
        print "文頭「は」(agt としての)   +10 点\n";
        $ha = 1;
      }
    }elsif ($knp->bunsetu_type($bun) =~ /C:(\d)+:モ/) {
      $bun_last = $knp->bp_last($bun);
      if ($knp->pos_list($bun_last) =~ /読点/ && $knp->base($bun_last-2) eq "人"
      && $knp->base($bun_last-3) eq "何") {
        $total += 10;
        print "文頭「何人も、」    +10 点\n";
        $ha = 1;
      }
    }

}elsif ($shu == 1) {
    if ($knp->bunsetu_type($bun) =~ /C:(\d)+:ハ/) {
      if ($knp->pos_list($knp->bp_last($bun)) =~ /読点/) {
        $total += 5;
        print "文頭「は」(obj としての)   +5 点\n";
        $ha = 1;
      }
    }
}



$saki_bun = $knp->kakarisaki_bunsetu($jutugo);
if ($saki_bun == $bun && $knp->bunsetu_type($saki_bun) =~ /C/){
  $kakatte_masu = $saki_bun;
  $total += 20;
  print "述語の係り先   +20 点\n";
}

#表層格のスコアの情報表示
  if($ha == 0 && $kakatte_masu == 0){
    @s_case = split(/:/,$knp->bunsetu_type($bun));
    if ($ddi[1] =~ /$s_case[2]/) {
      $total += 10;
      print "格助詞一致   +10 点\n";
    }elsif(fukujosi($bun) == 1){
      $total += 5;
      print "副 or 接続助詞 +5 点\n";
    }else{
      print "格助詞不一致  0 点\n";
    }
  }
```

```
#意味のスコアの情報表示
  $head_no = $knp->get_bunsetu_head($bun);
  $head_hyouki = $knp->hyoki($head_no);
  $head_yomi = $knp->yomi($head_no);
  $head_yomi =~ tr/ア-ン/あ-ん/;


  $category = $NTTTH::Word2Cat{"$head_hyouki,$head_yomi"};
  if($category eq ''){
    print "「$head_hyouki」が語彙体系にない    0 点\n";
  }elsif($imi_nasi == 1){
    print "意味が格フレーム辞書にない    0 点\n";

  }else{
    $s_sum = 0; $s_div = 0;
    foreach $im (@imi) {
      @dimi = split(/:/,$im);
      ($similarity,$max_similarity_pairs) =
      &NTTTH::SimilarityWord("$head_hyouki,$head_yomi","$dimi[0],*",1);
      next if($similarity < 0);
      $s_sum += ($similarity * $dimi[1]);
      $s_div += $dimi[1];
    }
  if($s_div == 0){      #0 で割ってしまうバグを抑える
    $s_div = 1;
  }
  $s_sum = ($s_sum / $s_div) * 10;
  $total += $s_sum;
  printf "意味スコア    +%5.2f 点\n", $s_sum;
}


#knp 係り先スコアの情報表示
  $ko = $knp->kakarisaki_bunsetu($bun);
  unless ($ko == -1){
    $mago = $knp->kakarisaki_bunsetu($ko);
    unless ($mago == -1){
      $himago = $knp->kakarisaki_bunsetu($mago);
    }
  }


  if($m_end == $ko && $knp->kakari_type($bun) eq "D"){
 #   $total += 3;
 #   print "knp 範囲内    +3 点\n";
 # }elsif($m_end == $mago && $knp->kakari_type($ko) eq "D"){
   # $total += 2;
   # print "knp 孫        +2 点\n";
 # }elsif($m_end == $himago && $knp->kakari_type($mago) eq "D"){
   # $total += 1;
   # print "knp ひ孫      +1 点\n";
  }else{
    print "knp スコープ外    0 点\n";
```

```
    }


  #読点をまたぐことによるスコア引き (3 点ずつ)(表示)

  $i = $jutugo - 1;
  $toutensuu =0;
  unless ($ha == 1){
    while($i >= $bun){
      if($knp->pos_list($knp->bp_last($i)) =~ /読点/){
        $toutensuu += 1;
      }
      $i -= 1;
    }
  }

  $total -= ($toutensuu * 3);
  $i = ($toutensuu * 3);
   print "読点またぎ$toutensuu 回    -$i 点\n";


  ##合計
  printf "合計 %5.2f 点\n\n", $total;
  return (0);
}



### ある文節に係る文節のスコアを再帰的に-1 していくサブルーチン
sub kakarimoto_kesi {
  my($k) = @_;
  my ($moto_k);
  $moto_k = $knp->kakarimoto_bunsetu($k);
  return (0) if ($moto_k == 0);
  my (@dk);
  @dk = split(/\s+/,$moto_k);
  foreach $ddk (@dk) {
    $A_score[$ddk] = -1;
    $O_score[$ddk] = -1;       $R_score[$ddk] = -1;
    $L_score[$ddk] = -1;       $P_score[$ddk] = -1;
    $W_score[$ddk] = -1;       $a_score[$ddk] = -1;       $r_score[$ddk] = -1;
    $T_score[$ddk] = -1;
    kakarimoto_kesi($ddk);
  }
}






sub case_analysis{
  my($m_begin, $m_end, $pri) = @_;
```

```
my($Ev, $Gv, $A, $O, $R, $L, $P, $W, $a, $r, $T, $case_no, $case_head,
   $m_head_no, $m_head, @hissu_reserved, $moto_meishi, @d_m_m, $case_no2, $case_head2,
   $alig_num, $alig_den, $m_sum, $case_alig, $mean_sim, $frame_alig, $i, @tagi_score,
   $tag_no);
     ###変数の初期化
$Ev = 0;
$Gv = 0;
$A = 0;
$O = 0;
$R = 0;
$L = 0;
$P = 0;
$W = 0;
$a = 0;
$r = 0;
$T = 0;
$case_no = 0;
$case_head = ();
@A_score = ();
@O_score = ();
@R_score = ();
@L_score = ();
@P_score = ();
@W_score = ();
@a_score = ();
@r_score = ();
@T_score = ();


$m_head_no = $knp->bp_first($m_end);
$m_head = $knp->base($m_head_no);

if($tagi{$m_head} >= 1 && $pri >= 0){
  $i = 1;
  while($i <= $tagi{$m_head}) {
    $tagi_score[$i] = case_analysis(1, $m_end, -$i);
    $i += 1;
  }
$tag_no = max_hai(@tagi_score);
$i = 1;
print "述語に多義性あり。スコア計算しフレーム選択します。\n";
while ($i <= $#tagi_score){
  print "$m_head$i:$tagi_score[$i] 点\n";
  $i += 1;
}
$m_head = "$m_head$tag_no";
print "格フレーム$m_head を選択\n\n\n";
}



if ($pri <= 0){
  $pri = -$pri;
```

```
  $m_head = "$m_head$pri";
  $pri = -$pri;
}



print "            述語動詞 = 「$m_head」\n" if($pri == 1);


print "\n";
print "   〜格フレーム辞書の内容〜\n" if(exists $A{$m_head} && $pri == 1);
print "$A{$m_head}\n" if(exists $A{$m_head} && $pri == 1);
print "$O{$m_head}\n" if(exists $O{$m_head} && $pri == 1);
print "$R{$m_head}\n" if(exists $R{$m_head} && $pri == 1);
print "$L{$m_head}\n" if(exists $L{$m_head} && $pri == 1);
print "$P{$m_head}\n" if(exists $P{$m_head} && $pri == 1);
print "$W{$m_head}\n" if(exists $W{$m_head} && $pri == 1);
print "$a{$m_head}\n" if(exists $a{$m_head} && $pri == 1);
print "$r{$m_head}\n" if(exists $r{$m_head} && $pri == 1);
print "$T{$m_head}\n" if(exists $T{$m_head} && $pri == 1);
print "$GG{$m_head}\n" if(exists $GG{$m_head} && $pri == 1);
print "$EE{$m_head}\n" if(exists $EE{$m_head} && $pri == 1);



 # 主格 (A) のスコア付け。
if (exists $A{$m_head}) {
  @A_score = Case_scorer($A{$m_head},$m_begin,$m_end,2);  # 最後の 1 は、主格であること
  #をあることを示す。主格は、文頭の「は」を重視するから
   $i = $m_begin;
   print "主格 (A) \n   文節 NO  スコア    主辞\n" if ($hyouji ==1 && $pri == 1);
   while($i < $m_end){
     next if ($A_score[$i] < 100);
     printf "       %d     %f 点     %s\n",$i, $A_score[$i]-100,
     $knp->hyoki($knp->get_bunsetu_head($i)) if ($hyouji == 1 && $pri == 1);
   } continue {
   $i += 1;
   }
    if ($A_op{$m_head} == 0){
      $A_score[0] = 1;
    }else{
      $A_score[0] = -1;
      $i = $m_begin;
      while($i < $m_end){
        next if ($A_score[$i] < 100);
        $A_score[$i] = $A_score[$i] - 97;
      } continue {
      $i += 1;
      }
    }
} else {
  $A_score[0] = -1;
}
```

```
# 対象格 (O) のスコア付け。
 if (exists $O{$m_head}) {
   @O_score = Case_scorer($O{$m_head},$m_begin,$m_end,1);
    $i = $m_begin;
   print "対象格 (O) \n   文節 NO  スコア    主辞\n" if ($hyouji == 1 && $pri == 1);
    while($i < $m_end){
      next if ($O_score[$i] < 100);
      printf "       %d     %f 点     %s\n",$i, $O_score[$i]-100,
      $knp->hyoki($knp->get_bunsetu_head($i)) if ($hyouji == 1 && $pri == 1);
    } continue {
    $i += 1;
    }
     if ($O_op{$m_head} == 0){
       $O_score[0] = 1;
     }else{
       $O_score[0] = -1;
       $i = $m_begin;
       while($i < $m_end){
         next if ($O_score[$i] < 100);
         $O_score[$i] = $O_score[$i] - 97;
       } continue {
         $i += 1;
       }
     }
 } else {
   $O_score[0] = -1;
 }



# 受け手格 (R) のスコア付け。
 if (exists $R{$m_head}) {
   @R_score = Case_scorer($R{$m_head},$m_begin,$m_end,0);
    $i = $m_begin;
    print "受け手格 (R) \n   文節 NO  スコア    主辞\n" if ($hyouji == 1 && $pri == 1);
    while($i < $m_end){
      next if ($R_score[$i] < 100);
    printf "       %d     %f 点     %s\n",$i, $R_score[$i]-100,
    $knp->hyoki($knp->get_bunsetu_head($i)) if ($hyouji == 1 && $pri == 1);
    } continue {
    $i += 1;
    }
   if ($R_op{$m_head} == 0){
       $R_score[0] = 1;
     }else{
       $R_score[0] = -1;
       $i = $m_begin;
       while($i < $m_end){
         next if ($R_score[$i] < 100);
         $R_score[$i] = $R_score[$i] - 97;
       } continue {
```

```
        $i += 1;
      }
    }
 } else {
   $R_score[0] = -1;
 }



# 場所格 (L) のスコア付け。
 if (exists $L{$m_head}) {
   @L_score = Case_scorer($L{$m_head},$m_begin,$m_end,0);
    $i = $m_begin;
   print "場所格 (L) \n   文節 NO  スコア    主辞\n" if ($hyouji == 1 && $pri == 1);
    while($i < $m_end){
      next if ($L_score[$i] < 100);
      printf "       %d     %f 点     %s\n",$i, $L_score[$i]-100,
      $knp->hyoki($knp->get_bunsetu_head($i)) if ($hyouji == 1 && $pri == 1);
    } continue {
    $i += 1;
    }
   if ($L_op{$m_head} == 0){
       $L_score[0] = 1;
     }else{
       $L_score[0] = -1;
       $i = $m_begin;
       while($i < $m_end){
         next if ($L_score[$i] < 100);
         $L_score[$i] = $L_score[$i] - 97;
       } continue {
         $i += 1;
       }
     }
 } else {
   $L_score[0] = -1;
 }



# 相手格 (P) のスコア付け。
 if (exists $P{$m_head}) {
 @P_score = Case_scorer($P{$m_head},$m_begin,$m_end,0);
    $i = $m_begin;
    print "相手格 (P) \n   文節 NO  スコア    主辞\n" if ($hyouji == 1 && $pri == 1);
    while($i < $m_end){
      next if ($P_score[$i] < 100);
    printf "       %d     %f 点     %s\n",$i, $P_score[$i]-100,
    $knp->hyoki($knp->get_bunsetu_head($i)) if ($hyouji == 1 && $pri == 1);
    } continue {
    $i += 1;
    }
 if ($P_op{$m_head} == 0){
       $P_score[0] = 1;
```

```
    }else{
      $P_score[0] = -1;
      $i = $m_begin;
      while($i < $m_end){
        next if ($P_score[$i] < 100);
        $P_score[$i] = $P_score[$i] - 97;
      } continue {
        $i += 1;
      }
    }
 } else {
   $P_score[0] = -1;
 }



# 方法格 (W) のスコア付け。
 if (exists $W{$m_head}) {
   @W_score = Case_scorer($W{$m_head},$m_begin,$m_end,0);
    $i = $m_begin;
   print "方法格 (W) \n   文節 NO  スコア    主辞\n" if ($hyouji == 1 && $pri == 1);
    while($i < $m_end){
      next if ($W_score[$i] < 100);
      printf "       %d     %f 点     %s\n",$i, $W_score[$i]-100,
      $knp->hyoki($knp->get_bunsetu_head($i)) if ($hyouji == 1 && $pri == 1);
    } continue {
    $i += 1;
    }
     if ($W_op{$m_head} == 0){
       $W_score[0] = 1;
     }else{
       $W_score[0] = -1;
       $i = $m_begin;
       while($i < $m_end){
         next if ($W_score[$i] < 100);
         $W_score[$i] = $W_score[$i] - 97;
       } continue {
         $i += 1;
       }
     }
 } else {
   $W_score[0] = -1;
 }



# 範囲格 (a) のスコア付け。
 if (exists $a{$m_head}) {
   @a_score = Case_scorer($a{$m_head},$m_begin,$m_end,0);
    $i = $m_begin;
   print "範囲格 (a) \n   文節 NO  スコア    主辞\n" if ($hyouji == 1 && $pri == 1);
    while($i < $m_end){
      next if ($a_score[$i] < 100);
```

```
      printf "       %d     %f 点     %s\n",$i, $a_score[$i]-100,
      $knp->hyoki($knp->get_bunsetu_head($i)) if ($hyouji == 1 && $pri == 1);
    } continue {
    $i += 1;
    }
     if ($a_op{$m_head} == 0){
       $a_score[0] = 1;
     }else{
       $a_score[0] = -1;
       $i = $m_begin;
       while($i < $m_end){
         next if ($a_score[$i] < 100);
         $a_score[$i] = $a_score[$i] - 97;
       } continue {
         $i += 1;
       }
     }
 } else {
   $a_score[0] = -1;
 }



# 範囲格 (r) のスコア付け。
 if (exists $r{$m_head}) {
   @r_score = Case_scorer($r{$m_head},$m_begin,$m_end,0);
    $i = $m_begin;
   print "範囲格 (r) \n   文節 NO  スコア    主辞\n" if ($hyouji == 1 && $pri == 1);
    while($i < $m_end){
      next if ($r_score[$i] < 100);
      printf "       %d     %f 点     %s\n",$i, $r_score[$i]-100,
      $knp->hyoki($knp->get_bunsetu_head($i)) if ($hyouji == 1 && $pri == 1);
    } continue {
    $i += 1;
    }
     if ($r_op{$m_head} == 0){
       $r_score[0] = 1;
     }else{
       $r_score[0] = -1;
       $i = $m_begin;
       while($i < $m_end){
         next if ($r_score[$i] < 100);
         $r_score[$i] = $r_score[$i] - 97;
       } continue {
         $i += 1;
       }
     }
 } else {
   $r_score[0] = -1;
 }
```

```
 # 時格 (T) のスコア付け。
  if (exists $T{$m_head}) {
    @T_score = Case_scorer($T{$m_head},$m_begin,$m_end,0);
     $i = $m_begin;
    print "時格（T）\n   文節 NO  スコア    主辞\n" if ($hyouji == 1 && $pri == 1);
     while($i < $m_end){
       next if ($T_score[$i] < 100);
       printf "       %d     %f 点     %s\n",$i, $T_score[$i]-100,
       $knp->hyoki($knp->get_bunsetu_head($i)) if ($hyouji == 1 && $pri == 1);
     } continue {
     $i += 1;
     }
      if ($T_op{$m_head} == 0){
        $T_score[0] = 1;
      }else{
        $T_score[0] = -1;
        $i = $m_begin;
        while($i < $m_end){
          next if ($T_score[$i] < 100);
          $T_score[$i] = $T_score[$i] - 97;
        } continue {
          $i += 1;
        }
      }
  } else {
    $T_score[0] = -1;
  }



# goa(目的格) があるなら対象格を無効に
if ($Gv{$m_head} == 1){
print "$m_head 1\n" if($pri == 1);
$moto_Gv2 = $knp->kakarimoto_bunsetu($m_end);
@dG2 = split(/\s+/,$moto_Gv2);
#print "$m_head:$m_end moto_Gv 2 $moto_Gv 2\n" if($pri == 1);
  foreach $ddG2 (@dG2) {
    $Ghead_no2 = $knp->get_bunsetu_head($ddG2);
    $Ghead2 = $knp->base($Ghead_no2);
    if ($Ghead2 =~ /よう/){
      print "$m_head 2\n" if($pri == 1);
      @O_score = ();
      $O_score[0] = -1;
    }
  }
}




  ## knp 範囲内の最高スコア探し

while ($A_score[max_hai(@A_score)] >= 110 || $O_score[max_hai(@O_score)] >= 110
```

```
|| $R_score[max_hai(@R_score)] >= 110 || $L_score[max_hai(@L_score)] >= 110
|| $P_score[max_hai(@P_score)] >= 110 || $W_score[max_hai(@W_score)] >= 110
|| $a_score[max_hai(@a_score)] >= 110 || $r_score[max_hai(@r_score)] >= 110
|| $T_score[max_hai(@T_score)] >= 110){

   if (max_case($A_score[max_hai(@A_score)],$O_score[max_hai(@O_score)],
   $R_score[max_hai(@R_score)],$L_score[max_hai(@L_score)],$P_score[max_hai(@P_score)],
   $W_score[max_hai(@W_score)],$a_score[max_hai(@a_score)],$r_score[max_hai(@r_score)],
   $T_score[max_hai(@T_score)]) == 1){
     $A = max_hai(@A_score);  $A_s = $A_score[$A];
     @A_score = ();           $A_score[0] = -1;
     $O_score[$A] = -1;       $R_score[$A] = -1;
     $L_score[$A] = -1;       $P_score[$A] = -1;
     $W_score[$A] = -1;       $a_score[$A] = -1;       $r_score[$A] = -1;
     $T_score[$A] = -1;
     kakarimoto_kesi($A) if($A < $m_end);

   }elsif (max_case($O_score[max_hai(@O_score)],$A_score[max_hai(@A_score)],
   $R_score[max_hai(@R_score)],$L_score[max_hai(@L_score)],$P_score[max_hai(@P_score)],
   $W_score[max_hai(@W_score)],$a_score[max_hai(@a_score)],$r_score[max_hai(@r_score)],
   $T_score[max_hai(@T_score)]) == 1){
     $O = max_hai(@O_score);  $O_s = $O_score[$O];
     @O_score = ();  $O_score[0] = -1;
     $A_score[$O] = -1;       $R_score[$O] = -1;
     $L_score[$O] = -1;       $P_score[$O] = -1;
     $W_score[$O] = -1;       $a_score[$O] = -1;       $r_score[$O] = -1;
     $T_score[$O] = -1;
     kakarimoto_kesi($O) if($O < $m_end);

   }elsif (max_case($R_score[max_hai(@R_score)],$O_score[max_hai(@O_score)],
   $A_score[max_hai(@A_score)],$L_score[max_hai(@L_score)],$P_score[max_hai(@P_score)],
   $W_score[max_hai(@W_score)],$a_score[max_hai(@a_score)],$r_score[max_hai(@r_score)],
   $T_score[max_hai(@T_score)]) == 1){
     $R = max_hai(@R_score);  $R_s = $R_score[$R];
     @R_score = ();  $R_score[0] = -1;
     $O_score[$R] = -1;       $A_score[$R] = -1;
     $L_score[$R] = -1;       $P_score[$R] = -1;
     $W_score[$R] = -1;       $a_score[$R] = -1;       $r_score[$R] = -1;
     $T_score[$R] = -1;
     kakarimoto_kesi($R) if($R < $m_end);

   }elsif (max_case($L_score[max_hai(@L_score)],$O_score[max_hai(@O_score)],
   $A_score[max_hai(@A_score)],$R_score[max_hai(@R_score)],$P_score[max_hai(@P_score)],
   $W_score[max_hai(@W_score)],$a_score[max_hai(@a_score)],$r_score[max_hai(@r_score)],
   $T_score[max_hai(@T_score)]) == 1){
     $L = max_hai(@L_score);  $L_s = $L_score[$L];
     @L_score = ();  $L_score[0] = -1;
     $O_score[$L] = -1;       $R_score[$L] = -1;
     $A_score[$L] = -1;       $P_score[$L] = -1;
     $W_score[$L] = -1;       $a_score[$L] = -1;       $r_score[$L] = -1;
     $T_score[$L] = -1;
     kakarimoto_kesi($L) if($L < $m_end);
```

```
}elsif (max_case($P_score[max_hai(@P_score)],$O_score[max_hai(@O_score)],
$A_score[max_hai(@A_score)],$R_score[max_hai(@R_score)],$L_score[max_hai(@L_score)],
$W_score[max_hai(@W_score)],$a_score[max_hai(@a_score)],$r_score[max_hai(@r_score)],
$T_score[max_hai(@T_score)]) == 1){
  $P = max_hai(@P_score);  $P_s = $P_score[$P];
  @P_score = ();  $P_score[0] = -1;
  $O_score[$P] = -1;       $R_score[$P] = -1;
  $L_score[$P] = -1;       $A_score[$P] = -1;
  $W_score[$P] = -1;       $a_score[$P] = -1;       $r_score[$P] = -1;
  $T_score[$P] = -1;
  kakarimoto_kesi($P) if($P < $m_end);

}elsif (max_case($W_score[max_hai(@W_score)],$A_score[max_hai(@A_score)],
$R_score[max_hai(@R_score)],$L_score[max_hai(@L_score)],$P_score[max_hai(@P_score)],
$O_score[max_hai(@O_score)],$a_score[max_hai(@a_score)],$r_score[max_hai(@r_score)],
$T_score[max_hai(@T_score)]) == 1){
  $W = max_hai(@W_score);  $W_s = $W_score[$W];
  @W_score = ();  $W_score[0] = -1;
  $A_score[$W] = -1;       $R_score[$W] = -1;
  $L_score[$W] = -1;       $P_score[$W] = -1;
  $O_score[$W] = -1;       $a_score[$W] = -1;       $r_score[$W] = -1;
  $T_score[$W] = -1;
  kakarimoto_kesi($W) if($W < $m_end);

}elsif (max_case($a_score[max_hai(@a_score)],$A_score[max_hai(@A_score)],
$R_score[max_hai(@R_score)],$L_score[max_hai(@L_score)],$P_score[max_hai(@P_score)],
$W_score[max_hai(@W_score)],$O_score[max_hai(@O_score)],$r_score[max_hai(@r_score)],
$T_score[max_hai(@T_score)]) == 1){
  $a = max_hai(@a_score);  $a_s = $a_score[$a];
  @a_score = ();  $a_score[0] = -1;
  $A_score[$a] = -1;       $R_score[$a] = -1;
  $L_score[$a] = -1;       $P_score[$a] = -1;
  $W_score[$a] = -1;       $O_score[$a] = -1;       $r_score[$a] = -1;
  $T_score[$a] = -1;
  kakarimoto_kesi($a) if($a < $m_end);

}elsif (max_case($r_score[max_hai(@r_score)],$A_score[max_hai(@A_score)],
$R_score[max_hai(@R_score)],$L_score[max_hai(@L_score)],$P_score[max_hai(@P_score)],
$W_score[max_hai(@W_score)],$a_score[max_hai(@a_score)],$O_score[max_hai(@O_score)],
$T_score[max_hai(@T_score)]) == 1){
  $r = max_hai(@r_score);  $r_s = $r_score[$r];
  @r_score = ();  $r_score[0] = -1;
  $A_score[$r] = -1;       $R_score[$r] = -1;
  $L_score[$r] = -1;       $P_score[$r] = -1;
  $W_score[$r] = -1;       $a_score[$r] = -1;       $O_score[$r] = -1;
  $T_score[$r] = -1;
  kakarimoto_kesi($r) if($r < $m_end);

}elsif (max_case($T_score[max_hai(@T_score)],$A_score[max_hai(@A_score)],
  $R_score[max_hai(@R_score)],$L_score[max_hai(@L_score)],$P_score[max_hai(@P_score)],
  $W_score[max_hai(@W_score)],$a_score[max_hai(@a_score)],$r_score[max_hai(@r_score)],
  $O_score[max_hai(@O_score)]) == 1){
    $T = max_hai(@T_score);  $T_s = $T_score[$T];
```

```
      @T_score = ();  $T_score[0] = -1;
      $A_score[$T] = -1;       $R_score[$T] = -1;
      $L_score[$T] = -1;       $P_score[$T] = -1;
      $W_score[$T] = -1;       $a_score[$T] = -1;       $r_score[$T] = -1;
      $O_score[$T] = -1;
      kakarimoto_kesi($T) if($T < $m_end);
  }
}
```

#knp スコープ内であるので＋１００点としていたスコアを元に戻す。

```
foreach (@A_score){
  if ($_ >= 100){
    $_ -= 100;
  }
}

foreach (@O_score){
  if ($_ >= 100){
    $_ -= 100;
  }
}

foreach (@R_score){
  if ($_ >= 100){
    $_ -= 100;
  }
}

foreach (@L_score){
  if ($_ >= 100){
    $_ -= 100;
  }
}

foreach (@P_score){
  if ($_ >= 100){
    $_ -= 100;
  }
}

foreach (@W_score){
  if ($_ >= 100){
    $_ -= 100;
  }
}

foreach (@a_score){
  if ($_ >= 100){
    $_ -= 100;
  }
}
```

```
foreach (@r_score){
  if ($_ >= 100){
    $_ -= 100;
  }
}

foreach (@T_score){
  if ($_ >= 100){
    $_ -= 100;
  }
}
```

knp 範囲外の最高スコア探し

```
while (($A_score[0] > 0 && $A_score[max_hai(@A_score)] >= 10)
|| ($O_score[0] > 0 && $O_score[max_hai(@O_score)] >= 10)
|| ($R_score[0] > 0 && $R_score[max_hai(@R_score)] >= 10)
|| ($L_score[0] > 0 && $L_score[max_hai(@L_score)] >= 10)
|| ($P_score[0] > 0 && $P_score[max_hai(@P_score)] >= 10)
|| ($W_score[0] > 0 && $W_score[max_hai(@W_score)] >= 10)
|| ($a_score[0] > 0 && $a_score[max_hai(@a_score)] >= 10)
|| ($r_score[0] > 0 && $r_score[max_hai(@r_score)] >= 10)
|| ($T_score[0] > 0 && $T_score[max_hai(@T_score)] >= 10)){

  if (max_case($A_score[max_hai(@A_score)],$O_score[max_hai(@O_score)],
  $R_score[max_hai(@R_score)],$L_score[max_hai(@L_score)],$P_score[max_hai(@P_score)],
  $W_score[max_hai(@W_score)],$a_score[max_hai(@a_score)],$r_score[max_hai(@r_score)],
  $T_score[max_hai(@T_score)]) == 1){
    $A = max_hai(@A_score);  $A_s = $A_score[$A];
    @A_score = ();           $A_score[0] = -1;
    $O_score[$A] = -1;       $R_score[$A] = -1;
    $L_score[$A] = -1;       $P_score[$A] = -1;
    $W_score[$A] = -1;       $a_score[$A] = -1;       $r_score[$A] = -1;
    $T_score[$A] = -1;
    kakarimoto_kesi($A) if($A < $m_end);

  }elsif (max_case($O_score[max_hai(@O_score)],$A_score[max_hai(@A_score)],
  $R_score[max_hai(@R_score)],$L_score[max_hai(@L_score)],$P_score[max_hai(@P_score)],
  $W_score[max_hai(@W_score)],$a_score[max_hai(@a_score)],$r_score[max_hai(@r_score)],
  $T_score[max_hai(@T_score)]) == 1){
    $O = max_hai(@O_score);  $O_s = $O_score[$O];
    @O_score = ();  $O_score[0] = -1;
    $A_score[$O] = -1;       $R_score[$O] = -1;
    $L_score[$O] = -1;       $P_score[$O] = -1;
    $W_score[$O] = -1;       $a_score[$O] = -1;       $r_score[$O] = -1;
    $T_score[$O] = -1;
    kakarimoto_kesi($O) if($O < $m_end);
```

```
}elsif (max_case($R_score[max_hai(@R_score)],$O_score[max_hai(@O_score)],
$A_score[max_hai(@A_score)],$L_score[max_hai(@L_score)],$P_score[max_hai(@P_score)],
$W_score[max_hai(@W_score)],$a_score[max_hai(@a_score)],$r_score[max_hai(@r_score)],
$T_score[max_hai(@T_score)]) == 1){
  $R = max_hai(@R_score);  $R_s = $R_score[$R];
  @R_score = ();  $R_score[0] = -1;
  $O_score[$R] = -1;       $A_score[$R] = -1;
  $L_score[$R] = -1;       $P_score[$R] = -1;
  $W_score[$R] = -1;       $a_score[$R] = -1;       $r_score[$R] = -1;
  $T_score[$R] = -1;
  kakarimoto_kesi($R) if($R < $m_end);

}elsif (max_case($L_score[max_hai(@L_score)],$O_score[max_hai(@O_score)],
$A_score[max_hai(@A_score)],$R_score[max_hai(@R_score)],$P_score[max_hai(@P_score)],
$W_score[max_hai(@W_score)],$a_score[max_hai(@a_score)],$r_score[max_hai(@r_score)],
$T_score[max_hai(@T_score)]) == 1){
  $L = max_hai(@L_score);  $L_s = $L_score[$L];
  @L_score = ();  $L_score[0] = -1;
  $O_score[$L] = -1;       $R_score[$L] = -1;
  $A_score[$L] = -1;       $P_score[$L] = -1;
  $W_score[$L] = -1;       $a_score[$L] = -1;       $r_score[$L] = -1;
  $T_score[$L] = -1;
  kakarimoto_kesi($L) if($L < $m_end);

}elsif (max_case($P_score[max_hai(@P_score)],$O_score[max_hai(@O_score)],
$A_score[max_hai(@A_score)],$R_score[max_hai(@R_score)],$L_score[max_hai(@L_score)],
$W_score[max_hai(@W_score)],$a_score[max_hai(@a_score)],$r_score[max_hai(@r_score)],
$T_score[max_hai(@T_score)]) == 1){
  $P = max_hai(@P_score);  $P_s = $P_score[$P];
  @P_score = ();  $P_score[0] = -1;
  $O_score[$P] = -1;       $R_score[$P] = -1;
  $L_score[$P] = -1;       $A_score[$P] = -1;
  $W_score[$P] = -1;       $a_score[$P] = -1;       $r_score[$P] = -1;
  $T_score[$P] = -1;
  kakarimoto_kesi($P) if($P < $m_end);

}elsif (max_case($W_score[max_hai(@W_score)],$A_score[max_hai(@A_score)],
$R_score[max_hai(@R_score)],$L_score[max_hai(@L_score)],$P_score[max_hai(@P_score)],
$O_score[max_hai(@O_score)],$a_score[max_hai(@a_score)],$r_score[max_hai(@r_score)],
$T_score[max_hai(@T_score)]) == 1){
  $W = max_hai(@W_score);  $W_s = $W_score[$W];
  @W_score = ();  $W_score[0] = -1;
  $A_score[$W] = -1;       $R_score[$W] = -1;
  $L_score[$W] = -1;       $P_score[$W] = -1;
  $O_score[$W] = -1;       $a_score[$W] = -1;       $r_score[$W] = -1;
  $T_score[$W] = -1;
  kakarimoto_kesi($W) if($W < $m_end);

}elsif (max_case($a_score[max_hai(@a_score)],$A_score[max_hai(@A_score)],
$R_score[max_hai(@R_score)],$L_score[max_hai(@L_score)],$P_score[max_hai(@P_score)],
$W_score[max_hai(@W_score)],$O_score[max_hai(@O_score)],$r_score[max_hai(@r_score)],
$T_score[max_hai(@T_score)]) == 1){
  $a = max_hai(@a_score);  $a_s = $a_score[$a];
```

```
    @a_score = ();  $a_score[0] = -1;
    $A_score[$a] = -1;       $R_score[$a] = -1;
    $L_score[$a] = -1;       $P_score[$a] = -1;
    $W_score[$a] = -1;       $O_score[$a] = -1;       $r_score[$a] = -1;
    $T_score[$a] = -1;
    kakarimoto_kesi($a) if($a < $m_end);

  }elsif (max_case($r_score[max_hai(@r_score)],$A_score[max_hai(@A_score)],
  $R_score[max_hai(@R_score)],$L_score[max_hai(@L_score)],$P_score[max_hai(@P_score)],
  $W_score[max_hai(@W_score)],$a_score[max_hai(@a_score)],$O_score[max_hai(@O_score)],
  $T_score[max_hai(@T_score)]) == 1){
    $r = max_hai(@r_score);  $r_s = $r_score[$r];
    @r_score = ();  $r_score[0] = -1;
    $A_score[$r] = -1;       $R_score[$r] = -1;
    $L_score[$r] = -1;       $P_score[$r] = -1;
    $W_score[$r] = -1;       $a_score[$r] = -1;       $O_score[$r] = -1;
    $T_score[$r] = -1;
    kakarimoto_kesi($r) if($r < $m_end);

  }elsif (max_case($T_score[max_hai(@T_score)],$A_score[max_hai(@A_score)],
    $R_score[max_hai(@R_score)],$L_score[max_hai(@L_score)],$P_score[max_hai(@P_score)],
    $W_score[max_hai(@W_score)],$a_score[max_hai(@a_score)],$r_score[max_hai(@r_score)],
    $O_score[max_hai(@O_score)]) == 1){
      $T = max_hai(@T_score);  $T_s = $T_score[$T];
      @T_score = ();  $T_score[0] = -1;
      $A_score[$T] = -1;       $R_score[$T] = -1;
      $L_score[$T] = -1;       $P_score[$T] = -1;
      $W_score[$T] = -1;       $a_score[$T] = -1;       $r_score[$T] = -1;
      $O_score[$T] = -1;
      kakarimoto_kesi($T) if($T < $m_end);
  }
}



print "     ～格解析結果～\n" if($pri == 1);
if (exists $A{$m_head} && $A_op{$m_head} == 0) {
  if ($A_s <= 0 || $A == 0){

    # $cases[$m_end] .= "A:* ";    #格の文節ナンバーをしまっておきたかったけど、見つか
    # らないなら「＊」を入れとく。
    $alig_den += Yorei_sum($A{$m_head});

    printf "動作主格(A) :  not found\n" if($pri == 1);
  }else{

    $cases[$m_end] .= "A:$A " if($pri == 1);    #あとで原子文作るために、格の文節ナンバ
    # ーをしまっておく。
    $hissu_reserved[$A] = 1;       #任意格を探す時に、すでに必須格となっているかどうか
    # 調べる配列

    $alig_num += Yorei_sum($A{$m_head});
    $alig_den += Yorei_sum($A{$m_head});
```

```
    $m_sum += M_sim($A, $A{$m_head});

    $A_head_no = $knp->get_bunsetu_head($A);
    $A_head = $knp->base($A_head_no);
    if($A_s >= 100){
      printf "動作主格(A) :  %s    スコア=%5.2f点  文節=%d  主辞=%d\n",
      $A_head, $A_s-100, $A, $A_head_no if($pri == 1);
      scorer_cause($A, $A{$m_head}, 2, $m_end) if($pri == 1);
    }else{
      printf "動作主格(A)   :  %s    スコア=%5.2f点  文節=%d  主辞=%d
      (KNPスコープ外)\n",
      $A_head, $A_s, $A, $A_head_no if($pri == 1);
      scorer_cause($A, $A{$m_head}, 2, $m_end) if($pri == 1);
    }
  }
}


if (exists $O{$m_head} && $O_op{$m_head} == 0) {
  if ($O_s <= 0 || $O == 0){

    # $cases[$m_end] .= "O:* ";
    $alig_den += Yorei_sum($O{$m_head});
    printf "対象格(O)   :  not found\n" if($pri == 1);
  }else{

    $cases[$m_end] .= "O:$O " if($pri == 1);
    $hissu_reserved[$O] = 1;

    $alig_num += Yorei_sum($O{$m_head});
    $alig_den += Yorei_sum($O{$m_head});
    $m_sum += M_sim($O, $O{$m_head});

    $O_head_no = $knp->get_bunsetu_head($O);
    $O_head = $knp->base($O_head_no);

    if($O_s >= 100){
      printf "対象格(O)   :  %s    スコア=%5.2f点  文節=%d  主辞=%d\n",
      $O_head, $O_s-100, $O, $O_head_no if($pri == 1);
      scorer_cause($O, $O{$m_head}, 1, $m_end) if($pri == 1);
    }else{
      printf "対象格(O)   :  %s    スコア=%5.2f点  文節=%d  主辞=%d
      (KNPスコープ外)\n",
      $O_head, $O_s, $O, $O_head_no if($pri == 1);
    scorer_cause($O, $O{$m_head}, 1, $m_end) if($pri == 1);
    }
  }
}


if (exists $R{$m_head} && $R_op{$m_head} == 0) {
  if ($R_s <= 0 || $R == 0){
```

```
    # $cases[$m_end] .= "R:* ";
    $alig_den += Yorei_sum($R{$m_head});
    printf "受け手格 (R)   :  not found\n" if($pri == 1);
  }else{

    $cases[$m_end] .= "R:$R " if($pri == 1);
    $hissu_reserved[$R] = 1;

    $alig_num += Yorei_sum($R{$m_head});
    $alig_den += Yorei_sum($R{$m_head});
    $m_sum += M_sim($R, $R{$m_head});

    $R_head_no = $knp->get_bunsetu_head($R);
    $R_head = $knp->base($R_head_no);

    if($R_s >= 100){
      printf "受け手格 (R)   :  %s    スコア=%5.2f 点  文節=%d  主辞=%d\n",
      $R_head, $R_s-100, $R, $R_head_no if($pri == 1);
      scorer_cause($R, $R{$m_head}, 0, $m_end) if($pri == 1);
    }else{
      printf "受け手格 (R)   :  %s    スコア=%5.2f 点  文節=%d  主辞=%d
      (KNP スコープ外)\n",
      $R_head, $R_s, $R, $R_head_no if($pri == 1);
      scorer_cause($R, $R{$m_head}, 0, $m_end) if($pri == 1);
    }
  }
}


if (exists $L{$m_head} && $L_op{$m_head} == 0) {
  if ($L_s <= 0 || $L == 0){

    # $cases[$m_end] .= "L:* ";
    $alig_den += Yorei_sum($L{$m_head});
    printf "場所格 (L)   :  not found\n" if($pri == 1);
  }else{

    $cases[$m_end] .= "L:$L " if($pri == 1);
    $hissu_reserved[$L] = 1;

    $alig_num += Yorei_sum($L{$m_head});
    $alig_den += Yorei_sum($L{$m_head});
    $m_sum += M_sim($L, $L{$m_head});

    $L_head_no = $knp->get_bunsetu_head($L);
    $L_head = $knp->base($L_head_no);

    if($L_s >= 100){
      printf "場所格 (L)   :  %s    スコア=%5.2f 点  文節=%d  主辞=%d\n",
      $L_head, $L_s-100, $L, $L_head_no if($pri == 1);
      scorer_cause($L, $L{$m_head}, 0, $m_end) if($pri == 1);
    }else{
      printf "場所格 (L)   :  %s    スコア=%5.2f 点  文節=%d  主辞=%d
```

```
      (KNP スコープ外)\n",
      $L_head, $L_s, $L, $L_head_no if($pri == 1);
      scorer_cause($L, $L{$m_head}, 0, $m_end) if($pri == 1);
    }
  }
}


if (exists $P{$m_head} && $P_op{$m_head} == 0) {
  if ($P_s <= 0 || $P == 0){

    # $cases[$m_end] .= "P:* ";
    $alig_den += Yorei_sum($P{$m_head});
    printf "相手格 (P)   :  not found\n" if($pri == 1);
  }else{

    $cases[$m_end] .= "P:$P " if($pri == 1);
    $hissu_reserved[$P] = 1;

    $alig_num += Yorei_sum($P{$m_head});
    $alig_den += Yorei_sum($P{$m_head});
    $m_sum += M_sim($P, $P{$m_head});

    $P_head_no = $knp->get_bunsetu_head($P);
    $P_head = $knp->base($P_head_no);

    if($P_s >= 100){
      printf "相手格 (P)   :  %s    スコア=%5.2f 点  文節=%d  主辞=%d\n",
      $P_head, $P_s-100, $P, $P_head_no if($pri == 1);
      scorer_cause($P, $P{$m_head}, 0, $m_end) if($pri == 1);
  }else{
      printf "相手格 (P)   :  %s    スコア=%5.2f 点  文節=%d  主辞=%d
      (KNP スコープ外)\n",
      $P_head, $P_s, $P, $P_head_no if($pri == 1);
      scorer_cause($P, $P{$m_head}, 0, $m_end) if($pri == 1);
    }
  }
}


if (exists $W{$m_head} && $W_op{$m_head} == 0) {
  if ($W_s <= 0 || $W == 0){

    $alig_den += Yorei_sum($W{$m_head});
    printf "方法格 (W)   :  not found\n" if($pri == 1);
  }else{

    $cases[$m_end] .= "W:$W " if($pri == 1);
    $hissu_reserved[$W] = 1;

    $alig_num += Yorei_sum($W{$m_head});
    $alig_den += Yorei_sum($W{$m_head});
    $m_sum += M_sim($W, $W{$m_head});
```

```
    $W_head_no = $knp->get_bunsetu_head($W);
    $W_head = $knp->base($W_head_no);

    if($W_s >= 100){
      printf "方法格(W)   :  %s    スコア=%5.2f点  文節=%d  主辞=%d\n",
      $W_head, $W_s-100, $W, $W_head_no if($pri == 1);
      scorer_cause($W, $W{$m_head}, 0, $m_end) if($pri == 1);
    }else{
      printf "方法格(W)   :  %s    スコア=%5.2f点  文節=%d  主辞=%d
      (KNPスコープ外)\n",
      $W_head, $W_s, $W, $W_head_no if($pri == 1);
    scorer_cause($W, $W{$m_head}, 0, $m_end) if($pri == 1);
    }
  }
}


if (exists $a{$m_head} && $a_op{$m_head} == 0) {
  if ($a_s <= 0 || $a == 0){

    $alig_den += Yorei_sum($a{$m_head});
    printf "範囲格(a)   :  not found\n" if($pri == 1);
  }else{

    $cases[$m_end] .= "a:$a " if($pri == 1);
    $hissu_reserved[$a] = 1;

    $alig_num += Yorei_sum($a{$m_head});
    $alig_den += Yorei_sum($a{$m_head});
    $m_sum += M_sim($a, $a{$m_head});

    $a_head_no = $knp->get_bunsetu_head($a);
    $a_head = $knp->base($a_head_no);

    if($a_s >= 100){
      printf "範囲格(a)   :  %s    スコア=%5.2f点  文節=%d  主辞=%d\n",
      $a_head, $a_s-100, $a, $a_head_no if($pri == 1);
      scorer_cause($a, $a{$m_head}, 0, $m_end) if($pri == 1);
    }else{
      printf "範囲格(a)   :  %s    スコア=%5.2f点  文節=%d  主辞=%d
      (KNPスコープ外)\n",
      $a_head, $a_s, $a, $a_head_no if($pri == 1);
    scorer_cause($a, $a{$m_head}, 0, $m_end) if($pri == 1);
    }
  }
}


if (exists $r{$m_head} && $r_op{$m_head} == 0) {
  if ($r_s <= 0 || $r == 0){

    $alig_den += Yorei_sum($r{$m_head});
```

```
    printf "原因格 (r)   :  not found\n" if($pri == 1);
  }else{

    $cases[$m_end] .= "r:$r " if($pri == 1);
    $hissu_reserved[$r] = 1;

    $alig_num += Yorei_sum($r{$m_head});
    $alig_den += Yorei_sum($r{$m_head});
    $m_sum += M_sim($r, $r{$m_head});

    $r_head_no = $knp->get_bunsetu_head($r);
    $r_head = $knp->base($r_head_no);

    if($r_s >= 100){
      printf "原因格 (r)   :  %s    スコア=%5.2f 点  文節=%d  主辞=%d\n",
      $r_head, $r_s-100, $r, $r_head_no if($pri == 1);
      scorer_cause($r, $r{$m_head}, 0, $m_end) if($pri == 1);
    }else{
      printf "原因格 (r)   :  %s    スコア=%5.2f 点  文節=%d  主辞=%d
      (KNP スコープ外)\n",
      $r_head, $r_s, $r, $r_head_no if($pri == 1);
    scorer_cause($r, $r{$m_head}, 0, $m_end) if($pri == 1);
    }
  }
}


if (exists $T{$m_head} && $T_op{$m_head} == 0) {
  if ($T_s <= 0 || $T == 0){

    $alig_den += Yorei_sum($T{$m_head});
    printf "時格 (T)   :  not found\n" if($pri == 1);
  }else{

    $cases[$m_end] .= "T:$T " if($pri == 1);
    $hissu_reserved[$T] = 1;

    $alig_num += Yorei_sum($T{$m_head});
    $alig_den += Yorei_sum($T{$m_head});
    $m_sum += M_sim($T, $T{$m_head});

    $T_head_no = $knp->get_bunsetu_head($T);
    $T_head = $knp->base($T_head_no);

    if($T_s >= 100){
      printf "時格 (T)   :  %s    スコア=%5.2f 点  文節=%d  主辞=%d\n",
      $T_head, $T_s-100, $T, $T_head_no if($pri == 1);
      scorer_cause($T, $T{$m_head}, 0, $m_end) if($pri == 1);
    }else{
      printf "時格 (T)   :  %s    スコア=%5.2f 点  文節=%d  主辞=%d
      (KNP スコープ外)\n",
      $T_head, $T_s, $T, $T_head_no if($pri == 1);
    scorer_cause($T, $T{$m_head}, 0, $m_end) if($pri == 1);
```

```
      }
    }
  }



#for 多義
$alig_den = 1 if ($alig_den == 0);
$alig_num = 1 if ($alig_num == 0);

$case_alig = $alig_num / $alig_den;
$mean_sim = $m_sum / $alig_num;

$frame_alig = $case_alig * $mean_sim;




    #チェック scorer_cause($O, $O{$m_head}, 0, $m_end);
    #引数の意味（1:格として対象にしてる文節 2:辞書の中身 3:1 は主格 2 は対象格 0 はそれ以外
    #4:対象としてる述語）
    #print "\n 要チェック\n" if($pri == 1);
    #print "W\n" if($pri == 1);
    #scorer_cause(6, $W{$m_head}, 0, $m_end) if($pri == 1);
    #print "O\n" if($pri == 1);
    #scorer_cause(4, $O{$m_head}, 1, $m_end) if($pri == 1);



#新任意格解析
  $location = 0;
  $time = 0;
  $loc_check = 0;
  $time_check = 0;
  $kakari_moto = $knp->kakarimoto_bunsetu($m_end);
  $knp_type = ();


  @d_kakari = split(/\s+/,$kakari_moto);
    foreach $d_ka (@d_kakari) {
      $location = 0;
      $time = 0;
      unless($hissu_reserved[$d_ka]){
        $ka_head_no = $knp->get_bunsetu_head($d_ka);

        $ka_hyouki = $knp->hyoki($ka_head_no);
        $ka_yomi = $knp->yomi($ka_head_no);
        $ka_yomi =~ tr/ア-ン/あ-ん/;

        $ka_category = $NTTTH::Word2Cat{"$ka_hyouki,$ka_yomi"};
        next if($ka_category eq '');
```

```
        @ka_cat_no = split(/:/,$ka_category);

        foreach $ka_ca (@ka_cat_no) {
          unless($location == 1){
          $location = &NTTTH::Orenokonanoka("0388", "$ka_ca"); #0388 は語彙体系の「場所」
          }
          unless($time == 1){
          $time = &NTTTH::Orenokonanoka("2670", "$ka_ca");     #2670 は語彙体系の「時間」
          }
        }

        $knp_type = $knp->bunsetu_type($d_ka);


        if($location == 1 && ($knp_type =~ /C:(\d)+:デ/ || $knp_type =~ /C:(\d)+:ニ/)){
            $cases[$m_end] .= "Lo:$d_ka ";
            $hissu_reserved[$d_ka];
            $loc_check = $d_ka;
          }elsif($time ==1 && $knp_type =~ /C:(\d)+:ニ/){
            $cases[$m_end] .= "To:$d_ka ";
            $hissu_reserved[$d_ka];
            $time_check = $d_ka;
          }else{
            next;
          }
      }
    }

#〜一応 任意格の表示 〜
  if($loc_check){
    print "場所格 (任意格) あり $loc_check\n" if($pri == 1);
  }
  if($time_check){
    print "時格あり (任意格) $time_check\n" if($pri == 1);
  }



#サ変名詞の解析 case_analysis の再帰呼び出し (最初、@cases にいれるとこでやってたけど、
#結果を出力する順番がおかしなったからここへ)


  if (exists $A{$m_head} && $A_op{$m_head} == 0) {
    if ($A_s > 0 && $A != 0){
      $case_no = $knp->bp_first($A);  #格として採用した節の先頭の単語が辞書にあり、なおか
      #つ文節のタイプが C であるなら、
        $case_head = $knp->base($case_no); #(すなわちサ変が名詞扱いされているなら) その節に
        #対して格解析を行う
        if ($ex{$case_head} == 1 && $mou_yatta[$A] != 1 && $knp->bunsetu_type($A) =~ /C/) {
          print "\n\n\n" if($pri == 1);
          $mou_yatta[$A] = 1;
          case_analysis(1, $A, 1) if($pri == 1);
```

```
      }

      $moto_meishi = $knp->kakarimoto_bunsetu($A);
      @d_m_m = split(/\s+/,$moto_meishi);
      foreach $d_mm (@d_m_m) {
        $case_no2 = $knp->bp_first($d_mm);
        $case_head2 = $knp->base($case_no2);
        if($ex{$case_head2} == 1 && $knp->bunsetu_type($A) =~ /C/
        && $mou_yatta[$d_mm] != 1 && $knp->kakari_type($d_mm) eq "D"){
          $mou_yatta[$d_mm] = 1;    #格として選ばれた語に被連体修飾語があるなら、
          #case_analysis(1, $d_mm);  #連体修飾語に格解析を繰り返す
        }
      }

    }
  }

if (exists $O{$m_head} && $O_op{$m_head} == 0) {
  if ($O_s > 0 && $O != 0){
  $case_no = $knp->bp_first($O);
    $case_head = $knp->base($case_no);
    if ($ex{$case_head} == 1  && $mou_yatta[$O] != 1 && $knp->bunsetu_type($O) =~ /C/) {
      print "\n\n\n" if($pri == 1);
      $mou_yatta[$O] = 1;
      case_analysis(1, $O, 1) if($pri == 1);
    }

      $moto_meishi = $knp->kakarimoto_bunsetu($O);
      @d_m_m = split(/\s+/,$moto_meishi);
      foreach $d_mm (@d_m_m) {
        $case_no2 = $knp->bp_first($d_mm);
        $case_head2 = $knp->base($case_no2);
        if($ex{$case_head2} == 1 && $knp->bunsetu_type($O) =~ /C/ && $mou_yatta[$d_mm] != 1
        && $knp->kakari_type($d_mm) eq "D"){
          $mou_yatta[$d_mm] = 1;
          #case_analysis(1, $d_mm);
        }
      }

  }
}

if (exists $R{$m_head} && $R_op{$m_head} == 0) {
  if ($R_s > 0 && $R != 0){
  $case_no = $knp->bp_first($R);
    $case_head = $knp->base($case_no);
    if ($ex{$case_head} == 1 && $mou_yatta[$R] != 1 && $knp->bunsetu_type($R) =~ /C/) {
      print "\n\n\n" if($pri == 1);
      $mou_yatta[$R] = 1;
      case_analysis(1, $R, 1) if($pri == 1);
    }

      $moto_meishi = $knp->kakarimoto_bunsetu($R);
```

```
      @d_m_m = split(/\s+/,$moto_meishi);
      foreach $d_mm (@d_m_m) {
        $case_no2 = $knp->bp_first($d_mm);
        $case_head2 = $knp->base($case_no2);
        if($ex{$case_head2} == 1 && $knp->bunsetu_type($R) =~ /C/
        && $mou_yatta[$d_mm] != 1 && $knp->kakari_type($d_mm) eq "D"){
          $mou_yatta[$d_mm] = 1;
          #case_analysis(1, $d_mm);
        }
      }

  }
}

if (exists $L{$m_head} && $L_op{$m_head} == 0) {
  if ($L_s > 0 && $L != 0){
  $case_no = $knp->bp_first($L);
    $case_head = $knp->base($case_no);
    if ($ex{$case_head} == 1 && $mou_yatta[$L] != 1 && $knp->bunsetu_type($L) =~ /C/) {
      print "\n\n\n" if($pri == 1);
      $mou_yatta[$L] = 1;
      case_analysis(1, $L, 1) if($pri == 1);
    }

      $moto_meishi = $knp->kakarimoto_bunsetu($L);
      @d_m_m = split(/\s+/,$moto_meishi);
      foreach $d_mm (@d_m_m) {
        $case_no2 = $knp->bp_first($d_mm);
        $case_head2 = $knp->base($case_no2);
        if($ex{$case_head2} == 1 && $knp->bunsetu_type($L) =~ /C/
        && $mou_yatta[$d_mm] != 1 && $knp->kakari_type($d_mm) eq "D"){
          $mou_yatta[$d_mm] = 1;
          #case_analysis(1, $d_mm);
        }
      }

  }
}

if (exists $P{$m_head} && $P_op{$m_head} == 0) {
  if ($P_s > 0 && $P != 0){
  $case_no = $knp->bp_first($P);
    $case_head = $knp->base($case_no);
    if ($ex{$case_head} == 1 && $mou_yatta[$P] != 1 && $knp->bunsetu_type($P) =~ /C/) {
      print "\n\n\n" if($pri == 1);
      $mou_yatta[$P] = 1;
      case_analysis(1, $P, 1) if($pri == 1);
    }

      $moto_meishi = $knp->kakarimoto_bunsetu($P);
      @d_m_m = split(/\s+/,$moto_meishi);
      foreach $d_mm (@d_m_m) {
        $case_no2 = $knp->bp_first($d_mm);
```

```
        $case_head2 = $knp->base($case_no2);
        if($ex{$case_head2} == 1 && $knp->bunsetu_type($P) =~ /C/
        && $mou_yatta[$d_mm] != 1 && $knp->kakari_type($d_mm) eq "D"){
          $mou_yatta[$d_mm] = 1;
          #case_analysis(1, $d_mm);
        }
      }

  }
}

if (exists $W{$m_head} && $W_op{$m_head} == 0) {
  if ($W_s > 0 && $W != 0){
  $case_no = $knp->bp_first($W);
    $case_head = $knp->base($case_no);
    if ($ex{$case_head} == 1 && $mou_yatta[$W] != 1 && $knp->bunsetu_type($W) =~ /C/) {
      print "\n\n\n" if($pri == 1);
      $mou_yatta[$W] = 1;
      case_analysis(1, $W, 1) if($pri == 1);
    }

      $moto_meishi = $knp->kakarimoto_bunsetu($W);
      @d_m_m = split(/\s+/,$moto_meishi);
      foreach $d_mm (@d_m_m) {
        $case_no2 = $knp->bp_first($d_mm);
        $case_head2 = $knp->base($case_no2);
        if($ex{$case_head2} == 1 && $knp->bunsetu_type($W) =~ /C/
        && $mou_yatta[$d_mm] != 1 && $knp->kakari_type($d_mm) eq "D"){
          $mou_yatta[$d_mm] = 1;
          #case_analysis(1, $d_mm);
        }
      }

  }
}

if (exists $a{$m_head} && $a_op{$m_head} == 0) {
  if ($a_s > 0 && $a != 0){
  $case_no = $knp->bp_first($a);
    $case_head = $knp->base($case_no);
    if ($ex{$case_head} == 1 && $mou_yatta[$a] != 1 && $knp->bunsetu_type($a) =~ /C/) {
      print "\n\n\n" if($pri == 1);
      $mou_yatta[$a] = 1;
      case_analysis(1, $a, 1) if($pri == 1);
    }

      $moto_meishi = $knp->kakarimoto_bunsetu($a);
      @d_m_m = split(/\s+/,$moto_meishi);
      foreach $d_mm (@d_m_m) {
        $case_no2 = $knp->bp_first($d_mm);
        $case_head2 = $knp->base($case_no2);
        if($ex{$case_head2} == 1 && $knp->bunsetu_type($a) =~ /C/
        && $mou_yatta[$d_mm] != 1 && $knp->kakari_type($d_mm) eq "D"){
```

```
          $mou_yatta[$d_mm] = 1;
          #case_analysis(1, $d_mm);
        }
      }

  }
}

if (exists $r{$m_head} && $r_op{$m_head} == 0) {
  if ($r_s > 0 && $r != 0){
  $case_no = $knp->bp_first($r);
    $case_head = $knp->base($case_no);
    if ($ex{$case_head} == 1 && $mou_yatta[$r] != 1 && $knp->bunsetu_type($r) =~ /C/) {
      print "\n\n\n" if($pri == 1);
      $mou_yatta[$r] = 1;
      case_analysis(1, $r, 1) if($pri == 1);
    }

      $moto_meishi = $knp->kakarimoto_bunsetu($r);
      @d_m_m = split(/\s+/,$moto_meishi);
      foreach $d_mm (@d_m_m) {
        $case_no2 = $knp->bp_first($d_mm);
        $case_head2 = $knp->base($case_no2);
        if($ex{$case_head2} == 1 && $knp->bunsetu_type($r) =~ /C/
        && $mou_yatta[$d_mm] != 1 && $knp->kakari_type($d_mm) eq "D"){
          $mou_yatta[$d_mm] = 1;
          #case_analysis(1, $d_mm);
        }
      }

  }
}

if (exists $T{$m_head} && $T_op{$m_head} == 0) {
  if ($T_s > 0 && $T != 0){
  $case_no = $knp->bp_first($T);
    $case_head = $knp->base($case_no);
    if ($ex{$case_head} == 1 && $mou_yatta[$T] != 1 && $knp->bunsetu_type($T) =~ /C/) {
      print "\n\n\n" if($pri == 1);
      $mou_yatta[$T] = 1;
      case_analysis(1, $T, 1) if($pri == 1);
    }

      $moto_meishi = $knp->kakarimoto_bunsetu($T);
      @d_m_m = split(/\s+/,$moto_meishi);
      foreach $d_mm (@d_m_m) {
        $case_no2 = $knp->bp_first($d_mm);
        $case_head2 = $knp->base($case_no2);
        if($ex{$case_head2} == 1 && $knp->bunsetu_type($T) =~ /C/
        && $mou_yatta[$d_mm] != 1 && $knp->kakari_type($d_mm) eq "D"){
          $mou_yatta[$d_mm] = 1;
          #case_analysis(1, $d_mm);
        }
```

```
        }

    }
  }

#$tetest = $Gv{$m_head};
#print "m_head $m_head:$m_end Gv{m_head} 1 $tetest 1\n";
#用言である対象格の解析 (case_analysis の再帰呼び出し)
if ($Gv{$m_head} == 1){
$moto_Gv = $knp->kakarimoto_bunsetu($m_end);
@dG = split(/\s+/,$moto_Gv);
  foreach $ddG (@dG) {
    $Ghead_no = $knp->get_bunsetu_head($ddG);
    $Ghead = $knp->base($Ghead_no);
    if ($Ghead =~ /よう/){
      $Gv = $ddG-1;
      $cases[$m_end] .= "Gv:$Gv ";
    }
  }
}
if ($Gv > 0){
  $GGG = $knp->get_bunsetu_head($Gv);
  $GGGG = $knp->base($GGG);
}


if ($Ev{$m_head} == 1){
$moto_Ev = $knp->kakarimoto_bunsetu($m_end);
@dE = split(/\s+/,$moto_Ev);
  foreach $ddE (@dE) {
    $Elast_no = $knp->bp_last($ddE);
    $Elast = $knp->base($Elast_no);
    $Elast_list = $knp->pos_list($Elast_no);
    if ($Elast eq "と" && $Elast_list =~ /格助詞/){
      $Ev = $ddE;
      $cases[$m_end] .= "Ev:$Ev ";
    }
  }
}
if ($Ev > 0){
  $EEE = $knp->get_bunsetu_head($Ev);
  $EEEE = $knp->base($EEE);
}


 return ($frame_alig);
}


          ###     〜メイン〜     ###
```

```
while(1) {
  # print "どの位置から？ (RET to exit)";
  # chomp($m_begin = <STDIN>);
  # last if $m_begin eq '';

  $m_begin = 1;
  print "述語動詞の位置を入れて下さい。 (RET to exit)";
  chomp($m_end = <STDIN>);
  last if $m_end eq '';
  # print "文頭は？";
  # chomp($m_buntou = <STDIN>);

  $mou_yatta[$m_end] = 1;
  case_analysis($m_begin, $m_end, 1);

  print "\n        ------------------------------------------------------------\n\n";

  $i = 100;
  while(1 <= $i){
    if ($cases[$i]){
      print "cases$i $cases[$i]\n";
    }
    $i -= 1;
  }


 ##     新～原子文生成部分～      ##

  $event_no = 1;
  $noun_no = 1;


  $ic = 100;      #文節数は 100 以内とする
  while(1 <= $ic){

    if($cases[$ic]){

      unless($reserved[$ic]){
        $event_first_no = $knp->bp_first($ic);
        $event_head = $knp->base($event_first_no);
        $event_sentence = "$event_head(e$event_no)";
        push(@atom, $event_sentence);   #イベント文を収納
        $reserved[$ic] = "e$event_no";
      }
      $main_e = $event_no;
      $event_no += 1;

      @arg_box = split(/[\s]+/,$cases[$ic]);

      foreach $arg (@arg_box) {
        @tmp = split(/:+/,$arg);
        unless ($reserved[$tmp[1]]){
```

```
$event_fir = $knp->bp_first($tmp[1]);
$event_hea = $knp->base($event_fir);
unless($ex{$event_hea}){
  $hei1 = 0;
  $hei2 = 0;
  $moto_noun = $knp->kakarimoto_bunsetu($tmp[1]);
  @d_m_n = split(/\s+/,$moto_noun);
  foreach $d_m (@d_m_n) {
    if($knp->kakarisaki_bunsetu($d_m) == $tmp[1] && $knp->kakari_type($d_m) =~ /P/){
      $heiretu_ari = 1;
      $hei1 = $d_m;
      $moto_noun2 = $knp->kakarimoto_bunsetu($hei1);
      @d_m_n2 = split(/\s+/,$moto_noun2);
      foreach $d_m2 (@d_m_n2) {
        if($knp->kakarisaki_bunsetu($d_m2) == $hei1
        && $knp->kakari_type($d_m2) =~ /P/){
          $hei2 = $d_m2;
        }
      }
    }
  }

  $noun_head2 = ();
  $noun_head1 = ();
  $noun_head = ();
  if($hei2){
    $tn = $knp->bp_first($hei2);
    while($tn <= $knp->bp_last($hei2)){
      $noun_head2 .= $knp->base($tn) if($knp->zirituqo_flag($tn) == 1);
      $tn += 1;
    }

    $tn = $knp->bp_first($hei1);
    while($tn <= $knp->bp_last($hei1)){
      $noun_head1 .= $knp->base($tn) if($knp->zirituqo_flag($tn) == 1);
      $tn += 1;
    }

    $tn = $knp->bp_first($tmp[1]);
    while($tn <= $knp->bp_last($tmp[1])){
      $noun_head .= $knp->base($tn) if($knp->zirituqo_flag($tn) == 1);
      $tn += 1;
    }

    $noun_sentence = "( $noun_head2(x$noun_no) ∨ $noun_head1(x$noun_no)
    ∨ $noun_head(x$noun_no) )";
  }elsif($hei1){
    $tn = $knp->bp_first($hei1);
    while($tn <= $knp->bp_last($hei1)){
      $noun_head1 .= $knp->base($tn) if($knp->zirituqo_flag($tn) == 1);
      $tn += 1;
    }
```

```
    $tn = $knp->bp_first($tmp[1]);
    while($tn <= $knp->bp_last($tmp[1])){
      $noun_head .= $knp->base($tn) if($knp->zirituqo_flag($tn) == 1);
      $tn += 1;
    }

    $noun_sentence = "( $noun_head1(x$noun_no) ∨ $noun_head(x$noun_no) )";
  }else{

    $tn = $knp->bp_first($tmp[1]);
    while($tn <= $knp->bp_last($tmp[1])){
      $noun_head .= $knp->base($tn) if($knp->zirituqo_flag($tn) == 1);
      $tn += 1;
    }

    $noun_sentence = "$noun_head(x$noun_no)";
  }
  push(@atom, $noun_sentence);   #名詞文を収納
  $reserved[$tmp[1]] = "x$noun_no";
  $noun_no += 1;


}else{
  $hei1 = 0;
  $hei2 = 0;
  $moto_noun = $knp->kakarimoto_bunsetu($tmp[1]);
  @d_m_n = split(/\s+/,$moto_noun);
  foreach $d_m (@d_m_n) {
    if($knp->kakarisaki_bunsetu($d_m) == $tmp[1] && $knp->kakari_type($d_m) =~ /P/){
      $heiretu_ari = 1;
      $hei1 = $d_m;
      $moto_noun2 = $knp->kakarimoto_bunsetu($hei1);
      @d_m_n2 = split(/\s+/,$moto_noun2);
      foreach $d_m2 (@d_m_n2) {
        if($knp->kakarisaki_bunsetu($d_m2) == $hei1
        && $knp->kakari_type($d_m2) =~ /P/){
          $hei2 = $d_m2;
        }
      }
    }
  }

  $noun_head2 = ();
  $noun_head1 = ();
  $noun_head = ();
  if($hei2){
    $tn = $knp->bp_first($hei2);
    while($tn <= $knp->bp_last($hei2)){
      $noun_head2 .= $knp->base($tn) if($knp->zirituqo_flag($tn) == 1);
      $tn += 1;
    }

    $tn = $knp->bp_first($hei1);
```

```
      while($tn <= $knp->bp_last($hei1)){
        $noun_head1 .= $knp->base($tn) if($knp->zirituqo_flag($tn) == 1);
        $tn += 1;
      }

      $tn = $knp->bp_first($tmp[1]);
      while($tn <= $knp->bp_last($tmp[1])){
        $noun_head .= $knp->base($tn) if($knp->zirituqo_flag($tn) == 1);
        $tn += 1;
      }

      $event_sentence = "( $noun_head2(e$event_no) ∨ $noun_head1(e$event_no)
      ∨ $noun_head(e$event_no) )";
    }elsif($hei1){
      $tn = $knp->bp_first($hei1);
      while($tn <= $knp->bp_last($hei1)){
        $noun_head1 .= $knp->base($tn) if($knp->zirituqo_flag($tn) == 1);
        $tn += 1;
      }

      $tn = $knp->bp_first($tmp[1]);
      while($tn <= $knp->bp_last($tmp[1])){
        $noun_head .= $knp->base($tn) if($knp->zirituqo_flag($tn) == 1);
        $tn += 1;
      }

      $event_sentence = "( $noun_head1(e$event_no) ∨ $noun_head(e$event_no) )";
    }else{


      $tn = $knp->bp_first($tmp[1]);
      while($tn <= $knp->bp_last($tmp[1])){
        $noun_head .= $knp->base($tn) if($knp->zirituqo_flag($tn) == 1);
        $tn += 1;
      }

      $event_sentence = "$noun_head(e$event_no)";
    }


    push(@atom, $event_sentence);   #イベントを収納
    $reserved[$tmp[1]] = "e$event_no";
    $noun_no += 1;
  }
}



if($tmp[0] eq "A"){
  $A_ex = "agt(e$main_e, $reserved[$tmp[1]])";
  push(@atom, $A_ex);
}elsif($tmp[0] eq "O"){
  $O_ex = "obj(e$main_e, $reserved[$tmp[1]])";
```

```
        push(@atom, $O_ex);
      }elsif($tmp[0] eq "R"){
        $R_ex = "rec(e$main_e, $reserved[$tmp[1]])";
        push(@atom, $R_ex);
      }elsif($tmp[0] eq "L"){
        $L_ex = "loc(e$main_e, $reserved[$tmp[1]])";
        push(@atom, $L_ex);
      }elsif($tmp[0] eq "P"){
        $P_ex = "par(e$main_e, $reserved[$tmp[1]])";
        push(@atom, $P_ex);
      }elsif($tmp[0] eq "W"){
        $W_ex = "way(e$main_e, $reserved[$tmp[1]])";
        push(@atom, $W_ex);
      }elsif($tmp[0] eq "a"){
        $a_ex = "abo(e$main_e, $reserved[$tmp[1]])";
        push(@atom, $a_ex);
      }elsif($tmp[0] eq "r"){
        $r_ex = "res(e$main_e, $reserved[$tmp[1]])";
        push(@atom, $r_ex);
      }elsif($tmp[0] eq "T"){
        $T_ex = "tim(e$main_e, $reserved[$tmp[1]])";
        push(@atom, $T_ex);
      }elsif($tmp[0] eq "Ev"){
        $Ev_ex = "inc(e$main_e, $reserved[$tmp[1]])";
        push(@atom, $Ev_ex);
      }elsif($tmp[0] eq "Gv"){
        $Gv_ex = "goa(e$main_e, $reserved[$tmp[1]])";
        push(@atom, $Gv_ex);
      }elsif($tmp[0] eq "Lo"){
        $L_o_ex = "loc_o(e$main_e, $reserved[$tmp[1]])";
        push(@atom, $L_o_ex);
      }elsif($tmp[0] eq "To"){
        $T_o_ex = "tim_o(e$main_e, $reserved[$tmp[1]])";
        push(@atom, $T_o_ex);
      }

    }

  }
  $ic -= 1;
}

  #結合 & 表示
$sentence = join(" ∧ ", @atom);
print "     〜論理式〜\n";
print "$sentence\n\n";



$cb = 1;                #解析結果の初期化
while($cb <= $m_end) {
  $cases[$cb] = ();
  $cb += 1;
```

```
  }


@event_parts = ();
@atom = ();


   ###変数の初期化

@A_score = ();
@O_score = ();
@R_score = ();
@L_score = ();
@P_score = ();
@W_score = ();
@a_score = ();
@r_score = ();
@T_score = ();

@reserved = ();
@mou_yatta = ();

}


          ###    〜メイン終了〜    ###




# 分類語彙表のクローズ
NTTTH::CloseFiles;

sub show_category {
  local($cat) = @_;

  printf "%s[%s, 親%s, 子%s, 深さ%d]"
    , $cat,
    $NTTTH::Label{$cat},
    $NTTTH::Parent{$cat},
    $NTTTH::Child{$cat},
    $NTTTH::Depth{$cat};
}
```