考察：「オブジェクト指向自然言語処理」

　オブジェクト指向によって、どれだけ強力で柔軟な表現を自然言語にもたらすか、具体的に考えていきたい。
　オブジェクト指向が有効なのは、自然言語処理の対象は細かなデータのまとまりとして捉えることができて、そのデータとその基本操作を一つの塊として表現することができる柔軟な方法であるからだ。そのデータと基本操作のまとまりはカテゴリー（クライテリア/タクソン）の体系をなす。タクソンはオブジェクト指向的である。クライテリアもタクソン生成のアルゴリズムであるが、状況に依存するデータを保存し、データと操作の塊と捉えればタクソンと併せて、オブジェクト指向ということになる。実際、タクソンが学習によって順次生まれていくように、クライテリア自身もそれにともなって生成されて行かなくてはならない。それは、タクソンとクライテリアは双対しているからである。簡単のためにこんなオブジェクトを作業品詞と呼ぶことにして、以下、議論を進めていく。

　議論の対象は次の技術である。
（１）オブジェクト指向文解析/文生成/文脈処理
（２）オブジェクト指向カテゴリーシステム
　・コーパスと学習
　・カテゴリーの表現
　・相同、相異を分かること
　・パターンマッチング
（３）オブジェクト指向知識ベースシステム
　・知識の論理表現
　・知識のインコア表現
　・知識のファイル化
　・内容検索

第１章　オブジェクト指向自然言語処理
　文解析、文生成、文脈処理を考えて行きたい。
　基本的に、オブジェクトは幾つかの文章要素を構成する。例えば、
（１）単語
（２）情報の単位
（３）句、節
（４）文
（５）文章
　それぞれが、それぞれのレベルで意味の要素と文法の要素、文脈の要素、文体の要素を持っている。意味というものは、３次元の空間要素と１次元の時間要素を反映したもので、物体、運動、抽象概念、場というものを表現している。そのため構造を持つので、計算機

的データ表現としてフレームとして表現されるものである。文法も文脈も意味と深く関わるから、この３者はフレームとして表現していくべきものであるといえる。
　フレームをフィールドに、各種処理手順をメソッドとしたオブジェクトが作業品詞である。

１．１　文解析

　簡単な文生成手順を先ずは追ってみたい。名詞、動詞、形容詞、副詞、助詞、助動詞、接続詞、それに単純な意味記号を扱うのみとする。
　文解析は、「てにをは」などの表層の係り受け解析し表層のフレーム表現を得るステップ（表層解析）と表層フレームを意味レベルの表現に深めるステップ（深層解析）でできている。

１．１．１　表層解析ステップ

　日本語の文解析は次のようになるだろう。文法はタップルで示すこととする。タップルの先頭がキーでタップルの終了がキーの修飾先を示す。語と意味はスラッシュで区切って示す。つまり、
　（キー　＊　キーの修飾先）　　*は間に語が入りうることを示す。
　キー：語/意味記号　意味記号と品詞は英字で表現する。

　表層解析には各単語にこのような修飾先を示すフレームを対応させる辞書（テーブル）が必要なのである。

【入力文１】私は大いなる雲間に浮かぶ飛行船を長い時間見ていた。
<step 01>(私は　＊　VB)　　；ND->(ND * VB)
<step 02>(私は　(おおいなる　＊　ND) ＊　VB)　　；AN->(AN * ND)
<step 03>(私は　(おおいなる　雲間に)　＊　VB)
　　(私は　((大いなる雲間に)　＊　VB) ＊　VB)；ND->(ND * VB)
　　(私は　(大いなる　((雲間に　＊ VB) ＊ ND)) ＊ VB)；ND->(ND * VB)

<step 04>(私は　(大いなる　雲間に)　浮かぶ)
　　((私は　(大いなる雲間に)　浮かぶ) * ND)　；VB->(VB *　ND)
　　(私は　((大いなる雲間に)浮かぶ) * VB)
　　(私は　(((大いなる雲間に)浮かぶ) * ND) * VB)；VB->(VB * ND)
　　(私は　(大いなる　((雲間に　浮かぶ) * ND)) * VB)

<step 05>(私は　(大いなる雲間に)　浮かぶ)->死滅
　　((私は　(大いなる雲間に)　浮かぶ)飛行船)->意味的に死滅
　　(私は　((大いなる雲間に)浮かぶ) * VB))->死滅
　　(私は　(((大いなる雲間に)浮かぶ)飛行船を) * VB)

(私は　（(((大いなる雲間に)浮かぶ)飛行船を）＊VB）*VB)；ND->(ND＊VB)
(私は　（大いなる((雲間に浮かぶ)飛行船を))＊VB)
(私は　((大いなる((雲間に浮かぶ)飛行船を))＊VB）＊VB)
(私は　((大いなる（((雲間に浮かぶ)飛行船を）＊VB）　＊ND）＊VB)

<step 06>(私は　(((大いなる雲間に)浮かぶ)飛行船を）（長い ＊ND）＊　VB)
(私は　((((大いなる雲間に)浮かぶ)飛行船を）(長い ＊ND）＊VB)*VB)
(私は（大いなる((雲間に浮かぶ)飛行船を))(長い*ND）*VB)
(私は（((大いなる((雲間に浮かぶ)飛行船を))(長い*ND)*VB)*VB)
(私は（((大いなる(((雲間に浮かぶ)飛行船を)(長い*ND)*VB)*ND)*VB)

<step 07>(私は　(((大いなる雲間に)浮かぶ)飛行船を）(長い時間)*VB)
(私は（(((大いなる雲間に)浮かぶ)飛行船を)((長い時間)*VB)*VB)
(私は（((((大いなる雲間に)浮かぶ)飛行船を)(長い時間)*VB)*VB)
(私は（((((大いなる雲間に)浮かぶ)飛行船を)((長い時間)*VB)*VB)*VB)
(私は（((大いなる((雲間に浮かぶ)飛行船を))(長い時間)*VB)*VB)
(私は（((大いなる(((雲間に浮かぶ)飛行船を)(長い時間)*VB)*ND)*VB)

<step 08>（私は（(((大いなる雲間に)浮かぶ）飛行船を)(長い時間)見ていた)
（私は（((大いなる(雲間に浮かぶ)飛行船を))（長い時間)見ていた）

【入力文２】私の登った山は大きくて、小さい花がいっぱい咲いていた。
<step 01>(私の ＊ND)
((私の　＊VB)ND)

<step 02>(私の（登った）*ND)
((私の登った)ND)

<step 03>((私の(登った)山は)*VB)
(((私の登った)山は)*VB)

<step 04>((私の(登った)山は)大きく/junction)(*VB)
((私の(登った)山は)(大きく　*ND)*VB)
(((私の登った)山は)大きく/junction)(*VB)
(((私の登った)山は)(大きく　*ND)*VB)

<step 05>((私の(登った)山は)大きく/junction)((小さい *ND)*VB)
((私の(登った)山は)(大きく　小さい*ND)*VB)
((私の(登った)山は)(大きく　(小さい *ND)*ND)*VB)

(((私の登った)山は)大きく/junction)((小さい *ND)*VB)
(((私の登った)山は)(大きく　小さい*ND)*VB)
(((私の登った)山は)(大きく　(小さい*ND)*ND)*VB)

<step 06>((私の(登った)山は)大きく/junction)((小さい 花が)*VB)
((私の(登った)山は)(大きく　小さい花が)*VB)
((私の(登った)山は)(大きく　(小さい 花が)*ND)*VB)
(((私の登った)山は)大きく/junction)((小さい 花が)*VB)
(((私の登った)山は)(大きく　小さい花が)*VB)
(((私の登った)山は)(大きく　(小さい花が)*ND)*VB)

<step 07>((私の(登った)山は)大きく/junction)((小さい 花が)いっぱい*VB)
((私の(登った)山は)(大きく　小さい花が)いっぱい*VB)
((私の(登った)山は)(大きく　(小さい 花が)*ND)*VB)->死滅
(((私の登った)山は)大きく/junction)((小さい 花が)いっぱい*VB)
(((私の登った)山は)(大きく　小さい花が)いっぱい*VB)
(((私の登った)山は)(大きく　(小さい花が)*ND)*VB)->死滅

<step 08>((私の(登った)山は)大きく/junction)((小さい 花が)いっぱい咲いていた)
((私の(登った)山は)(大きく　小さい花が)いっぱい咲いていた)
(((私の登った)山は)大きく/junction)((小さい 花が)いっぱい咲いていた)
(((私の登った)山は)(大きく　小さい花が)いっぱい咲いていた)

以上なような手順で解析が進む（この手法は ATN と数学的には同等かも知れない)。タップルは一つのオブジェクトである。次のような予測規則を用いた。
（１）形容詞には係先として名詞を予測、あるいは自分を動詞（BE 動詞）と予測する。
（２）副詞には動詞、形容詞を係先として予測する。
（３）名詞には助詞を伴って動詞を修飾することを予測する。
（４）動詞は名詞を修飾することを予測する。
（５）接続詞には文をまとめるのと、同格の品詞の並置を予測することを行う。

【API】
SurfaceAnalyser sa=new SurfaceAnalizer(KnowledgeBase kb);//テキストの表層解析準備
Frame[] frame=sa.analize(String text);//解析の実行
kb.setCopus(String text);//コーパスへのテキストの登録
kb.setCopus(Frame frame);//コーパスへのフレームの登録

１．１．２　深層解析ステップ

表層フレームと深層フレームの対応表を知識ベースとして持っていることで、実現される処理である。対訳していくだけのものであるが、文脈によりどの表層フレームを有効にするかはフィルターされることになる。

また複数の意味表現候補（深層フレーム）があるので、その曖昧性を文章の進展や対話の枠組みから無くしていき、最終的に一つに絞っていくという処理も必要である。これは、大きな意味の監視装置によって経時的に行われるものであるのである。監視装置はデーモンとして実現される。

この深層解析ステップには、表層フレーム対深層フレームを与える辞書（テーブル）が必要である。

【API】

```
DeepAnalyser da=new DeepAnalyser(KnowlegeBase kb);//深層解析の準備
Frame[] frame=da.analize(Frame surface_frame);//フレームの対訳
kb.setMeanCopus(Frame frame);//コーパスへの意味の登録
```

１．２　文生成

文生成は知識文脈情報から文や文章を生成することである。基本的に動画システムを想定している。

文生成は自然言語処理を超えた部署からの働きかけで起動し、制御される。ここが重要である。何か重要なことかを決めることは自然言語処理系のすることではない。外部から自然言語処理に対して働きかけるインターフェースを先ず持って定義しなくてはならない。

動画システムは知識や文脈を次のフレームで表現するものである。その要素は、次の４つである。

（１）時間枠
（２）空間枠
（３）アクター
（４）アクターの行動と事物の定義

これらは入れ子になっている。

動画システムは知識ベースもしくは文脈情報から取り出す。知識のどの部分を発話していくかを先ず決定していかねばならない。その知識ベースの文に表現する情報を決定することをフォーカスを当てると呼ぶことにする。

発話はフォーカスを次々に設定し、動画システムを取り出していくことで文生成の種を形作り、しかる後、意味記号や文脈情報から情報の単位を、そして単文を生成し、最後に文章にするのである。

文や文章は３ステップで生成する。
（１）第１ステップは単文を生成することで、動画システムの意味体系から、情報の単位を考慮して単語を選んでいくことである。
表現としては、（時間枠、空間枠、副詞群、主語などの格の項群、動詞）フレームとなる。
（２）第２ステップは文体の選択、強意点とフォーカスの決定、疑問形や複文構造にする範囲を決める事である。
これらの情報を指示フレームとして作る。指示フレームもタップルで表現される。
（３）第３ステップは文の変形を行い、最終文を決定することである。
表現としては、単語や情報の単位を変更し（活用形の決定やより適切な情報の単位に変える)、新しいタップルを組み立てることを行う。最後にそのタップルから文を生成していく。
ここでも、タップルは一つのオブジェクトとして把握していく。

逆に、生成した文はコーパス化する。そうして解析して、再起的に、意味や文法、文脈に登録して、曖昧性管理に利用する。曖昧性の把握は異なる意味の表現の衝突からくみ取らねばならない事柄であるからだ。

【文生成の例】
フレームとは次のような情報をもっている。秋の日の体験を綴ったものである。
<　時間枠：秋、日曜日
空間枠：長野、長池、公民館（館内、館外、室内）
アクター：私、人々（来客、開催者)、展示物（書画、盆栽)、靴、飲み物
時系列：・私は公民館で靴を脱いだ。
・私は室内に入った。
・私は書画を見た。
・部屋は人々でいっぱいだった。
・開催者の人が、飲み物でもてなしてくれた。
・盆栽が圧巻だった。
・私は感心した。
・私は部屋の外に出た。
・私は靴を履いた。
・私は公民館を去った。

>

このような状況で、自然言語利用システムから、重要事項として、「公民館ではなにが良かったか発話せよ。」という指令（重要事項のリストの提示）を得たとき、「盆栽が圧巻だった」と「私は感心した」という、感情表現を選択して、発話することになる。
文を合成すると、「盆栽が圧巻で、私は感心した。」となるはずである。

【API】

KnowledgeBase kb=new KnowledgeBase(KnowlegeControl kl);//知識ベースの読み込み
Context context=new Context();//文脈の生成
kb.setFocus(Node node);//知識のノード（作業品詞）にフォーカスを当てる
Frame[] frames=kb.getFrames(Node node);//作業品詞のフレームを取り出す
Scenario s=new Scenario(Frame[] frames);//フレームからシナリオを生成する
Sentence[] sentences=s.getSentences(KnowledgeBase kb);//シナリオから文要素を生成する
String text=s.makePhrase(Expression e,Sentences[] sentences);//文体を指定して、テキストを生成する

１．３　文脈

文脈は文章を解析して得た情報と常識という知識を統合した、発話認識の結果を表現したフレームである。文脈を参照しながら文章は解析され、結果はまた文脈に組み入れられていく。文脈フレームは発話の順序による時系列の時間枠、空間枠、アクター、アクターの行動と事物の定義といった情報を保持するものである。動画システムとして表現するのがよろしい。動画システムのエントリはオブジェクトとして管理するのがよい。なぜなら、常識の裏打ちをされた情報の塊であるからである。単なる記号ではない。各エントリーはシステムなのである。

文章の解析結果はタップルとして表すとしたが、これは最終的にフレームとして、一つのオブジェクトとして文脈の動画システムのエントリとなる。

Context c=knowledge_base.getContext();//知識ベースから文脈を得る
c.setValue(Node key,Node value);//文脈に要素を設定する
c.setValues(Node key,Frame frame);//文脈にフレーム要素を設定する
Node value=c.getValue(Node key);//文脈からノード（作業品詞）を得る
Frames[] values=c.getValues(Node key);//文脈からフレーム要素を得る

文脈には次の重要な作業がある。
（１）曖昧性を時間を掛けて（何ステップにも渡る入力文により）取り除いていく。
（２）知識ベースと入力文章をマージして、予測制御したり、推定処理をしたりする。
（３）確定事項と推測事項を分けて管理する。

以前の記事で提唱した動画システムには不備がある。アクターの扱いが貧弱すぎて、文脈処理が満足にできないのだ。アクターは枠組みとして大きな位置を占めるべきである。例えば、

（例文）「純は山を見ていた。山は美しかった。それに比べて、川には魚がいず、残念だった。」

この例文への次の問いかけに人は答えられるだろう。

（問いの例）「純は山が好きか？」とか「純が好感を持ったのはなに？」

アクターにはこのように、アクター間の関係属性が付随するし、それはまたアクター毎のコーパスを収集とその解析が必要と言うことである。また、アクターの好みとかの性向（人格）とかの情報も必要になる。またアクターにはさまざまな修飾がなされるわけで、それらの情報を簡単に見つけられるフレーム構成になっているべきである。このように考えてくると、アクターは枠組みとして管理し、そしてフレーム構造で情報を管理していくべきという結論になる。

さらに、アクターの運動とか操作なんかも事象と同じ扱いが成されなくてはならない。理由を例であげよう。

（例文）「私は学校へ歩いた。その歩幅は１メートルセンチだった。」

このように「その」というような指示詞で、「歩く」という動作が参照されたりすることだ。動作などの動詞も次々に修飾を受けるのである。だから、アクター枠に対して、その時系列サブフレームとして動詞の名詞化（アクター化）のもとで、動作フレームを持つべしということになる。

第２章　カテゴリーシステム

文解析、文生成で必要とする情報はオブジェクトとした。情報をオブジェクトとして管理するには基本としてカテゴリーというもので整理していく必要がある。カテゴリーは自然言語システムが自己組織化の一環として自ら生成、消滅させていく事が求められる。人手では対応できないからである。カテゴリーは無限に有るものだからである。

カテゴリーはクライテリアを基にタクソンを生成していくという方略をとる。クライテリアもタクソンも基盤となるプリミティブなクライテリア、タクソンがあって、その組み合わせから成長がなされていく。プリミティブは人手で作り上げる。なぜなら、プリミティブはクオリアであって、学習によってどうにかできるものではないからである。また、パターン認識により取り込むべき、学習要素もある。

カテゴリーシステムとして構築しなくてはいけない情報は、次の４つである。

・文法
・意味
・文脈
・文体

あとは文章表現を考えて、

・単語
・情報の単位
・句・節
・文
・文章

これらは、単語とか情報の単位、句/節、文、文章という記号体系に対応して作られるものである。単語はどんなカテゴリーのものかというような問いがある一方で、このカテゴリー要素はどんな単語かという問いに、自然言語で答える局面もある。

単語や情報の単位、句/節、文、文章といったものは、文法とか意味、文脈のオカランスの一つとして特徴づけられるが、意味とか文脈の前後の変化としても特徴づけられる。そして、現象して観測されるのは文法とか意味、文脈のカテゴリーシステムではなくて、単語や情報の単位、句/節、文、文章といった自然言語の要素である。従って、カテゴリーシステムは自然言語表現を基盤に構築していかねばならない。カテゴリーシステムのプリミティブ群のみを人手で作り込みあとは自然言語表現を基盤にした自己組織化学習にゆだねるのである。

文が知識ベースに蓄えられるべしという発想は、文のレベルで新しい意味が生じることからも得られることである。

例えば、「山が動いた」という表現は、政治の世界（文脈）では、「不可能と思われていたことが突然可能となった」という意味をはらんで使われる。

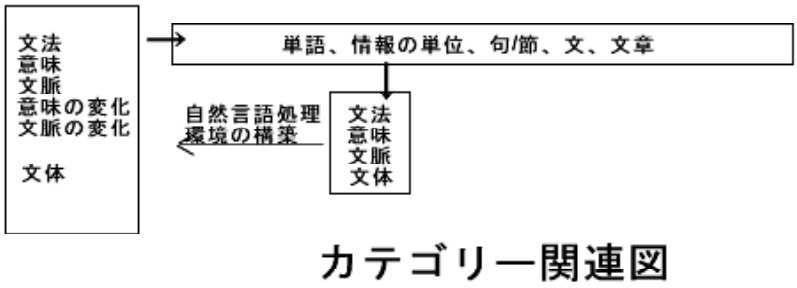
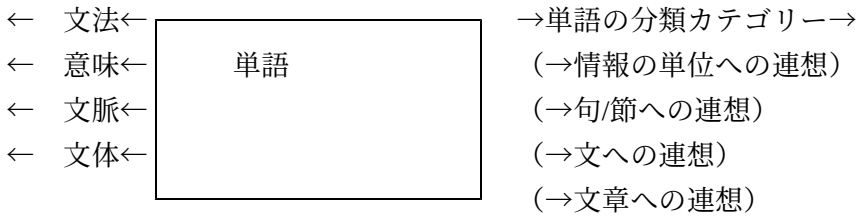
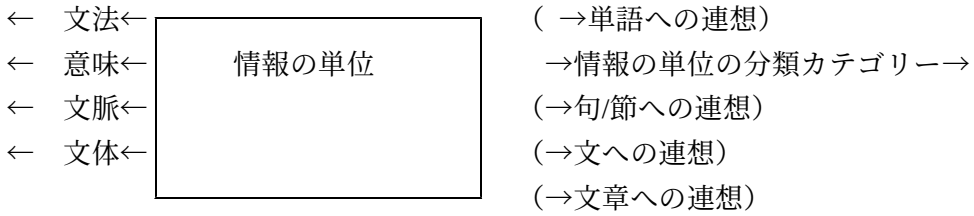


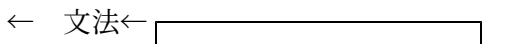
図２．１　カテゴリーと言語表現の関係

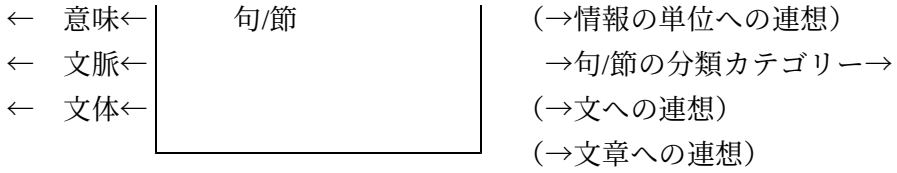


a. 単語からの成長ネットワーク



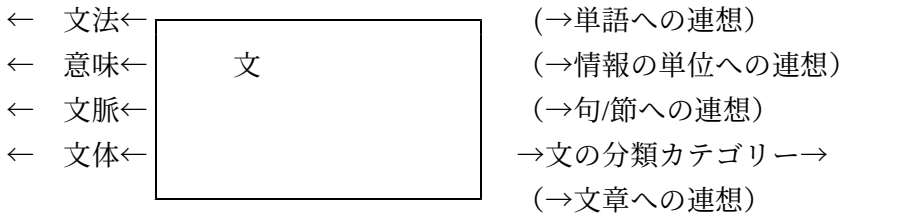
b. 情報の単位からの成長ネットワーク





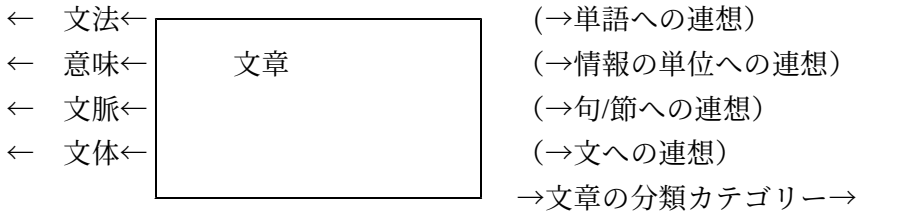
c.句/節からの成長ネットワーク

（注）句/節のネットワークは特にリーズニング（エキスパートシステム）を実現するのに使われる。



d.文からの成長ネットワーク

（注）文のネットワークは特にリーズニング（エキスパートシステム）を実現するのに使われる。



e.文章からの成長ネットワーク

（注）文章のネットワークは文脈、文体枠を構築するのに重要である。

図２．２　カテゴリーの成長ネットワーク
（図の最左の「←」はカテゴリーとしての成長を示す。）

タクソンとクライテリアの成長の例をあげる。

(step01)入力文：「雀は空を飛ぶ」　　→雀の作業品詞：タクソン：すずめ
クライテリア：fly-able

(step02)入力文：「ツバメは空を飛ぶ」→ツバメの作業品詞：タクソン：ツバメ
クライテリア：fly-able

(step03) step01 と step02 から

work01 作業品詞（カテゴリー生成）:タクソン：fly-able
クライテリア：fly-able
要素：雀、ツバメ

(step04)入力文：「雀は鳥である」　→鳥の作業品詞を起こして
タクソン：鳥
クライテリア：雀の要素（サブセット）

(step05)step04 と step03 から
asume-work01 作業品詞を起こして、鳥の作業品詞と関連づけする
タクソン：鳥
クライテリア：fly-able

２．１　コーパスと学習

言語現象とカテゴリーシステムとの関係を構築していくことは、先ずは言語現象を蓄えて行かなくてはならない。言語現象をコーパスとして蓄積していくときには、カテゴリーシステムとの関係も明確になっていなければならない。言語現象が文法とか意味、文脈を作ることを考えると、こうしたカテゴリーが既にあることを仮定するのは矛盾するようであるが、ロボットのように生活していくオブジェクトならば、イメージとか体験とかを通して得られたカテゴリーシステムを基盤として選択できる。言語現象と体験とから構築するカテゴリーを徐々に精密化していけば学習システムが成り立つのである。

オーソライズされたカテゴリーと生カテゴリーは似て非なるものであることに注意しよう。オーソライズされたカテゴリーは抽象であり、記号化されたもので、プロトタイプへのポインタとか、プリミティブ記号のセットである。生カテゴリーは状況に応じた変形があって、記号化で捨てられる情報をも持っている。この生カテゴリーの成長によって、オーソライズカテゴリーも精密化が成されていくのである。

インコアなコーパスはこの生カテゴリーと相関していて、ファイル化されたコーパスはオーソライズされたカテゴリーの記号システムと相関を持つようにすべきである。オーソライズはそのカテゴリーとカテゴリーへのネットワークパスの使用頻度の高さによって行う。

学習はデータマイニング技術を応用して実現する。相同性の解析と相異性の解析をしていくのである。超並列計算機の世界である。

基本的に「成長戦略」でいく。成長戦略とは、曖昧性も抽象化も汎化、分離も状況に要求されたらひたすらそのまま作業品詞を作り、パスを張ってネットワークを成長させていき、刈り込みや吟味などの処理を施さないという意味である。重要ならば、作業品詞とそれへのパスの利用頻度が高くなるので、それを持って評価し、オーソライズしていくのである。厳密な管理はしない。ということである。

作業用ネットワークとオーソライズされたネットワークを別立てで保持することが重要である。

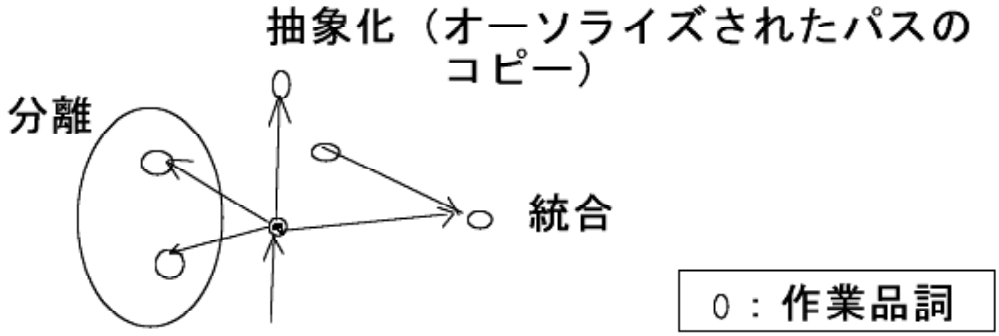


図２．３　作業品詞の成長

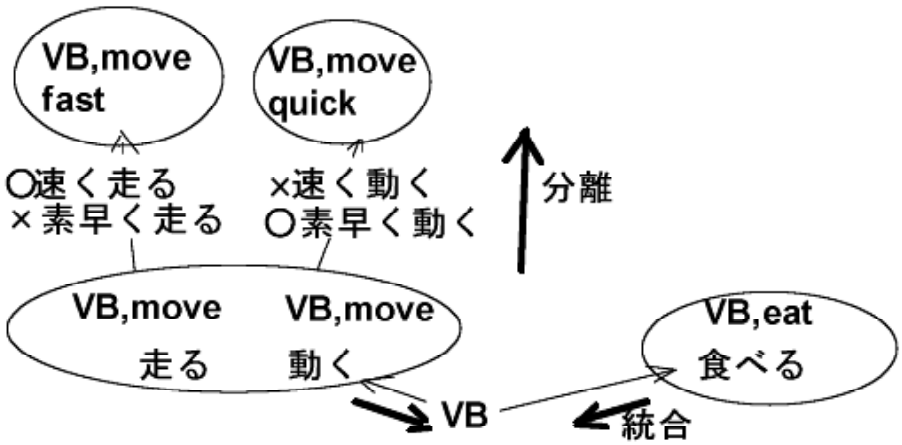


図２．４　オーソライズされた作業品詞のネットワークの成長

２．２　カテゴリーの表現

文法は、単語配置と修飾関係、機能を表現するプリミティブのタップルで表現できる。もっとも単純なカテゴリーシステムを成す。

意味と文脈もフレーム構造を成す。イメージも持ち、イメージはオンデマンドで記号化されなくてはならない。こうして、オブジェクトとして管理、利用していくことが適切なシステムであるといえるのだ。まとまりとしては単純であるが、その内部は複雑であるからだ。処理機構も内蔵されなくてはならないことも理由にあげられる。

単語、情報の単位、句/節、文、文章も対応する意味/文脈カテゴリーシステムに対応して、カテゴリー化していく必要がある。自然言語表現から見たカテゴリーというものが有用になるからである。文体とか、適切な表現とかは意味/文脈カテゴリーではない。この文体カテゴリーもオブジェクトとして表現していく。

そしてオブジェクト（作業品詞）の中身はフレームである。時間枠、空間枠、アクター、アクターの行動、定義、そして、言語の前後の状態の変異をフレームで表現する。

２．３　相同と相異が分かること

カテゴリーシステムの構築には事象（オブジェクト）の相同と相異を解析し、認知できなくてはならない。

基本的に共通の意味記号とか単語があれば、そのクライテリアの下に相同といえる。一方で、相異なる意味記号とか単語があれば、そのクライテリアの下に相異といえる。相同相異とはそうしたものであるが、もっと高い見地で、共通性の測度を導入して、相同とか相異とかを解析すべきかもしれない。その時には、その測度がクライテリアになるのである。

相同相異ともパターンマッチング処理が基盤技術になるが、フレームの項のマッチング処理になるはずである。

【API】

```
Frame frame=CategoryProcess.differrence(Category orgcat,Frame f1,Frame f2);
        //２つのフレームの内容の違いを取り出す
Frame framet=CategoryProcess.same(Category orgcat,Frame f1,Frame f2);
        //２つのフレームの内容の相同文を取り出す
```

【API】

```
Frame   f=knowledge_base.getInformationOnDifference(Node node1,Node node2);
                    //相違点の検出
Frame   f=knowledge_base.getInformationOnSameness(Node node1,Node node2);
                  //相同点の検出
Frame   f=knowledge_base.getDeepInformationOnDifference(Node node1,Node node2);
                    //相違点の検出
Frame   f=knowledge_base.getDeepInformationOnSameness(Node node1,Node node2);
                  //相同点の検出
```

文の特徴としては、条件結果表現文とか、動作記述文とか、内容の深さのレベルとか、記述の枠組みとかいろいろなものが考えられる。こうゆうものを、上記の相同、相異検出 API で捉えて、文の構造を解析していかねばならない。

カテゴリーやコーパスはネットワークとして成長していく。ここにも、相同/相異を判定する局面がある。単語や情報の単位毎に意味や文脈、文法がつながるが、同じ意味、文脈、文法であっても単語や情報の単位が異なれば、別のノードとして成長させなくてはならない。途中で別のパスを生成しなくてはならなくなることがあるからである。ここが大切なことである。

ネットワークパスは、
（１）抽象化
（２）分離
（３）統合
の３種がある。抽象化は既存のオーソライズされたパスをコピーすること（推論）のパスであり、分離は相異なる要素によりノードが分離するパスであり、統合は相同なる要素によりノードが結合するパスである。

また、統合には、属性を単純に結合して作る、コレクションとしての作用もある。例えば、ペンギンは空を飛ばないが「鳥」に加えられる。飛ぶものとしての「鳥」の例外となる。例外として、「鳥」に飛べない種族があることを付加する必要がでてくるのである。

【API】

```
Category[] cats=CategoryProcess.separateTo(Category orgcat,Frame f1,Frame f2);
        //２つのフレームの内容の違いをカテゴリフレームに取り出す
Category cat=CategoryProcess.mergeTo(Category cat1,Category cat2);
        //２つのカテゴリーの内容をマージして一つのカテゴリーフレームにする
```

２．４　タクソンとクライテリア

タクソンは分類で、クライテリアは分類基準である。双対している。ともにカテゴリーとして成長させていく必要がある。実は、タクソンもクライテリアも記述表現は同じである。利用の仕方が異なるだけである。

+分類記号　：「分類記号」を持つことを示す。　例）+VB +move
-分類記号　：「分類記号」を持たないことを示す。例）-flyable
and　　：積集合を表す。　　例）(+eatable and +flyable)
or　　：和集合を表す。　　例）(+stay or +change)
()　　：論理の優先順位

第３章　知識ベースシステム

知識の基盤はオブジェクト表現である。ファイル化するときには連想（ポインタ）とかコレクションとかは特別な工夫をもって対応していかなければならない。

知識ベースとしては、カテゴリーとコーパスを基盤として作られるものである。

３．１　知識のインコア表現

知識はカテゴリーのネットワークである。情報はハッシュマップとする。キーとデータの組のテーブルを複数持つという方式を貫く。オブジェクトのコネクションはベクターコレクションとして複数持つ。

オブジェクトは識別子を持ち、ポインタをファイル化する助けとする。このため、知識オブジェクトは Factory パターンで生成するようにし、ファクトリーの中で識別子を自動で振り付け、またポインタの管理テーブルをシステムで持てるようにする。

３．２　知識のファイル表現

リレーショナルデータベースを用いて、オブジェクトをファイル化する。一つのオブジェクトを格納するのは基本的に３種のテーブルで、必要に応じて、拡張テーブルを持つものとする。拡張テーブルも３種のテーブルで構成される。

一つのテーブルの列はキーを保存する。もう一つのテーブルの列は、キーテーブルの対応する列に格納するデータの種類を示す。そして、最後のテーブルの列にデータを text 形式で保存する。ポインタとか、テーブルの拡張識別記号もデータ種類テーブルの特定のコード（"pointer"、"extension"）で示すこととする。これで、ハッシュマップやベクターコレクションも、またオブジェクトの連想(コネクション)もファイルで表現できることが保証される。

３．３　フレーム構造

オブジェクトのデータをフレームで表す。基本的にキー付きのテーブルである。主な種類としては次のものがある。

（１）共起フレーム
（２）意味フレーム
（３）文脈フレーム
（４）文法フレーム
（５）文体フレーム

３．３．１　共起フレーム

単語や意味の共起関係を保持する。各単語や意味がテーブルになる。テーブルのエントリは他の共起関係にある単語や意味記号とその共起の重要度である。

単語「リンゴ」がテーブル名

1	みかん	重要度：１０
2	なし	重要度：２０
3	くり	重要度：５

図３．１　共起フレームの例

３．３．２　意味フレーム

意味記号を保持する。記号列と意味定義文書の２種類がある。それぞれテーブルである。記号列はプリミティブ記号とか、既に定義済みの記号をエントリに持つテーブルとする。テーブルの形は、共起フレームに同じである。

意味定義文書は、疑似自然言語で意味を定義する。構文を持つことで、より精密な記号定義を可能とする。疑似自然言語は、品詞とか意味記号で定義された単語列である。ファイル形式は記号列テーブルに同じ。プレフィックスによって、記号列テーブルか意味定義文書テーブルかを弁別できるようにする。

３．３．３　文脈フレーム

動画システムを入れ子のオブジェクトで表現し､各オブジェクトをキーと値の対にして、テーブルで表現する。構造はポインタを持つので、ポインタを識別子で表現し、ファイル化できるようにする。ポインタをテーブルで管理し、構造化データとして保存する。基盤はテーブルである。

３．３．４　文法フレーム

次の事柄を、テーブルの列要素として管理する。

・修飾関係
・共起関係
・語順
・品詞

．３．５　文体フレーム

文体は次の事柄に集約される。

・共起関係
・意味への関係

これらは、共起フレームや意味フレームへのポインタを保持することで実現される。

３．３．６　曖昧性フレーム

各フレームのエントリに曖昧性があれば、複数の候補フレームを多重に持たねばならないので、エントリに曖昧性記号を付加して、この曖昧性フレームを参照できるようにする。曖昧性フレームは一意の共起フレーム、意味フレーム、文法フレーム、文脈フレーム、文体フレームをポイントしている。

３．４　メトリック

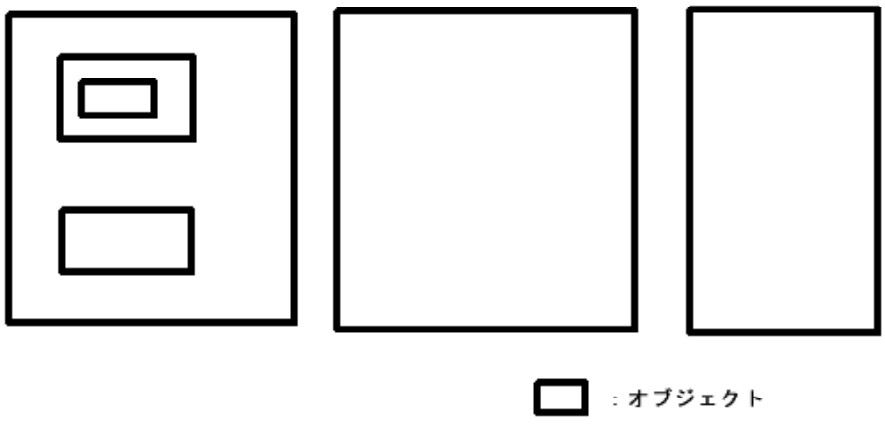
各フレームの内容には重要度で評価されなくてはならない。その評価指標としてつぎのものが可能であるべきである。

- 頻度　　：同じエントリーが起きた回数
- 重要度　　：エントリーの注目度（システムの性格を反映する）
- 確信度　　：エントリーの起きる予想確率
- 強さ　　：感情などの量的属性の値

各エントリーにはこの４つの値を付加して、システムの成長を管理できるようにしなくてはならない。

３．５　内容検索

データの検索は各フレームのエントリの値によって検索される場合が殆どである。是非、内容検索を実現したい。



オブジェクトは入れ子になっていて、その中に検索キーとなる記号が保存されている。

図３．１　内容検索の環境

各オブジェクトにインバースインデックスを配置して、内容検索を実現する。オブジェクトのインバースインデックスの合成はプライミングとして、最新に想起された記号と共起関係にあるオブジェクトのものを優先するように制御する。

３．６　曖昧性管理（文脈依存知識）

曖昧性管理は、枠組み指向で行くべきである。すなわち、時間枠、空間枠、アクター枠、行動/事象枠というものにオブジェクト（作業品詞）を立てて、そこに属性を登録していき、その属性のメトリックを管理することにより、曖昧性を除去していくのである。確信度とか、強さといった値が重要である。属性には枠固有の属性と、枠間の関係で決まる属性とがある。

（例題）「明雄は信子が好き」
→（解釈１）明雄は信子のことが好き
→（解釈２）信子は明雄のことが好き

この曖昧性は前後の文章の解析で、「信子」の「明雄」に対するフェボライト値が高くなっていれば、解釈２が取られる・・・というふうに、曖昧性傾向が除去できる。結局両方であるという解釈も成り立つので、文章解析としては、曖昧性を残しておくのが、最終的には精確な解釈になることを保証すると言える。

３．７　知識の整理

知識はオブジェクトのカテゴリー別に（各オーソライズされた作業品詞毎に）作られていくべきである。つまり、「長野は冬寒い」とか、「人はお金を好む」、「人は他人に好かれたいものである」とか、知識を分析していくと、「長野」とか「人」、「金」、「他人」というような特徴的オブジェクトがあって、その属性、行動としてなにがあるかというような形になっていることに気づく。知識とはそういうものであるらしい。

文法も、意味体系も、文脈もすべからく、このような知識体系である。このことをしっかり捉えて知識ベースを構築していくことが、あとあとでの推論処理などに有効に利用できるようにするために重要なことである。

とくにアクターは重要である。文法も文脈も文体も属人的（属物的）である。「うちは・・・だっちゃ」という人がいれば、それは「うる星やつら」のラムだと分かる。

第４章　おわりに

人工知能のプログラムを組んでいると、設計のやり直しをしたくなる局面に多々遭遇する。始めの設計が想定していた状況が狭すぎるのに気づくのである。こうしなければならないのに、この機構では無理だ・・・。どうしよう・・・。の果てに挫折する。

特に学習システムはあらゆる情報を扱えないといけないので、この手戻りが大きい。かといって、始めから全ての場合を想定した機構を案出するのは不可能なこと。

人工知能システムは柔軟でなくてはならない。柔軟さはシンプルさを要請する。しかも手直しは局限した手段で実現したい。ならば、あらゆるデータとプロセスを隠蔽し、インターフェースだけが不変（普遍）なオブジェクト指向があっているというものである。

そして知識はネットワークで表現するのがもっとも柔軟なものとなるので、オブジェクトがネットワークでつながったシステムとするのがよろしい。ネットワークは構造を持つが、構造はベースとは独立なグループとして、これもまたネットワークで表現する。

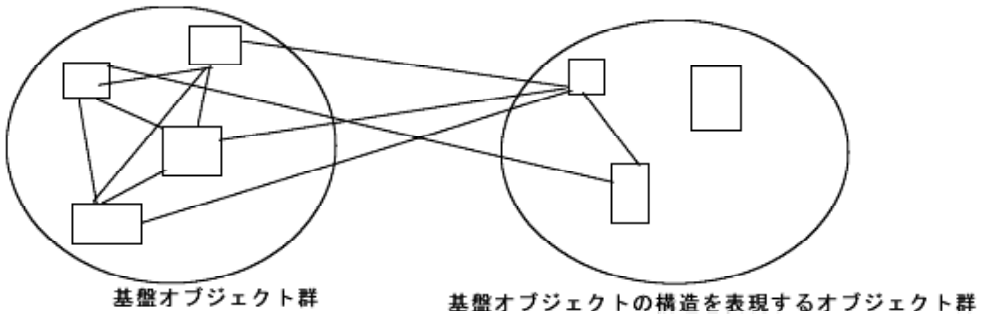


図４．１　知識のネットワーク

学習は試行錯誤が行われる世界である。したがって、試行錯誤を自在に行うために、作業用の知識ネットワークを設け、自身を持った知識体系はオーソライズされた知識として分けた方が良いと考える。実際の所どうなのかは、これからインプリメントしながら体験することになるだろう。

終わり