考察：「未夢の現在とこれから」

　未夢プロジェクトは今、テストのし残しはまだありますが、予定していた機能は大体、なんとなく動いているというまでに進捗しました。これで、日本語研究の為の強力な道具ができたといえるでしょう。当面は日本語研究をしていくつもりです。
　まだまだ、考えなくてはならない文章があります。
（例文１）山行った。
（例文２）リンゴ食った。
などの省略文とか、
（例文３）花が秀麗の洋画を買った。
などの「の」の持つ特殊構文とか。
　助詞や助動詞の処理の追求というものがあります。「中学国文法　青木一男著　評論社」に納められている事は対応可能というレベルを目指したく思っています。

　今の想定は機械翻訳に使えるものということで、意味処理を考えていく予定です。正確な対訳語、対訳文を生成できるだけの情報を日本語文章から抽出していく・・・、そこを狙った日本語解析のための日本語研究です。それをやっていく。
　それは問い合わせ応答システムですと、シミュレーターを作らねばならないので、シミュレーターを作っているときに、未夢のプログラムのコードを忘れてしまうのではないかと恐れるからです。本当は、今すぐにでも Prolog ベースのシミュレータを作りに掛かりたいわけです。そのプロジェクト名も作りました。「香澄プロジェクト」です。もう、正式な研究課題にしたということです。未夢の日本語研究が十分にいったら、その情報は辞書のなかに蓄えられますので、コードを忘れてもなんとか思い出せるようになって、夏樹プロジェクトのための分析ができると判断でき、新しいプログラム作りに掛かれる体制になるということです。日本語研究は最低３ヶ月を予定してます。日本語力を未夢に付けさせねば。
　香澄プロジェクトがなれば、日本語意味理解システムというものの全貌が分かって来るでしょう。それを今年中には成し遂げたい。遅くとも来年春までに・・・。夏樹プロジェクトの最盛期は夏が相応しいから。

　ということで、今回は未夢の現状のデータ構造を紹介して、香澄プロジェクトを含めた問い合わせ応答システムの構想を述べたいと思います。

第１章　未夢の構造

未夢のデータ構造は今のところ下図のようになっています。これは、文解析の結果を保持する、重要な部分です。この上にステージ、シーン、そしてその上に動画システムが構築されています。

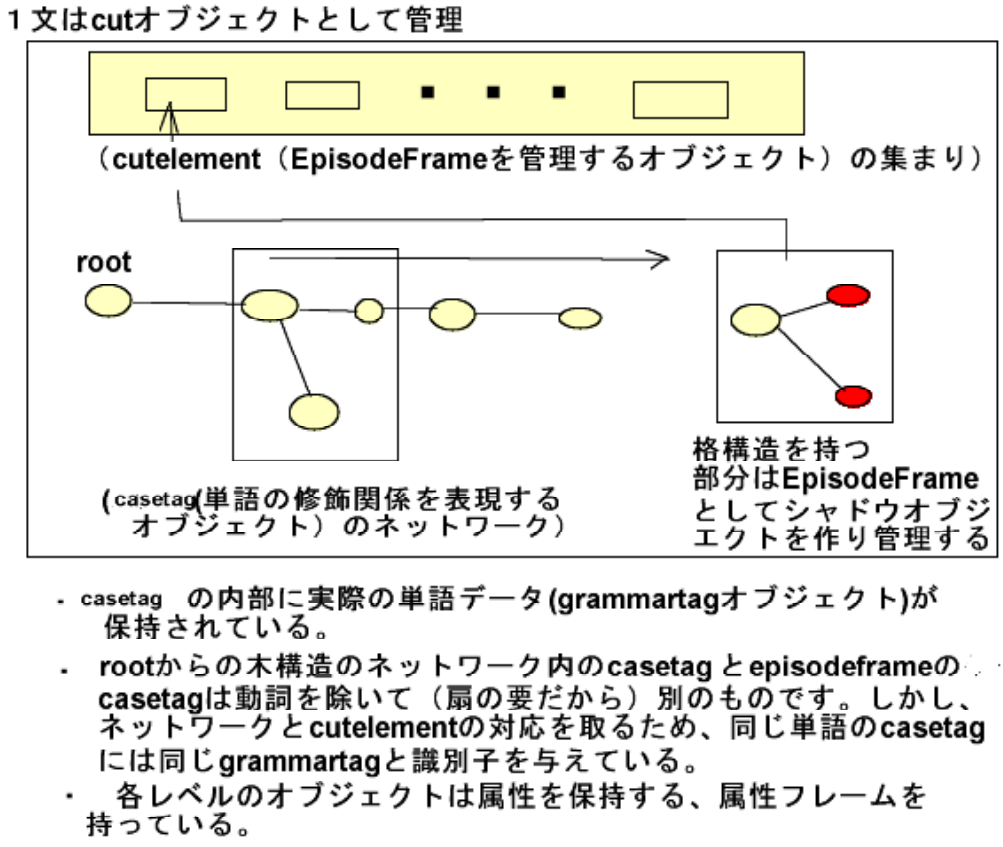

図１．未夢の文解析結果部分の構造

この構造を取るときに問題になったのは、使役構文です。

（例文１．１）信子は猫に魚を食べさせた。

構造的には「食べる」と「させる」が融合した構造になっています。それは工夫して２つの EpisodeFrame 構成にして解決しました。つまり、

「　（信子,that,make)EpisodeFrame の「that」casetag を（猫,魚,食べる）EpisodeFrame が修飾する・・・・・　」

という形にしたのです。

あとは特に問題になってません。格構造の曖昧性は EpisodeFrame を複数対応させ、動詞 casetag に割付け、一つの cutelement で管理するということで吸収しました。これで、曖昧性はあとから文脈情報によって無くしていくことができます。曖昧性をなくすことなく、利用に応じて情報全体を利用できるように、デフォルト EpisodeFrame というものも設けました。絞り込んだものをこれに割り当てればいいのです。全体の情報は捨てないことが肝心と判断してのことです。

第２章　機械翻訳システムからの利用

未夢は基本的に、辞書を興しさえすれば、どんな情報も、単語から文章の各解析結果に付加することができます。指示代名詞や欠落語の処理もしています。意味は基本的に英語をベースとしたプリミティブで表現していますので、核となるプリミティブ意味記号に、各レベルで保持している情報をもとに適切な単語を選ぶことができます。

用意するものは、各言語に対応した生成規則だけです。意味情報と各情報がしっかりあれば、生成変形文法をつくることで、文章が生成できることでしょう。

これから未夢で追求することは、どんな日本語文章も解析できて、意味構造を構築し、曖昧性を除去した意味ストリームを推奨として生成することです。未夢は曖昧性を内包しているので、このままでは、生成変形文法の設計者に大きな負担となるから、この曖昧性を除去した意味体系を生成する能力を作り込む事は必須です。

この意味で、文章解析の結果の各部分で、複数の選択項目を持つものはデフォルト値を持てる機構にしておく事は必須ということになります。

後は、保持している情報を、意味記号をキーにしてまとめて取り出せる機能を作り込むことですね。動詞のプリミティブ「say」を持つ casetag が指示されたら、連想できる範囲の情報をフレームとして取り出していく。個人への話かけなら、「say」でしょうが、群衆への話しかけなら「speak」でしょう。そんな、単語選択付加情報を用意するのも、生成変形文法を用意するように、文生成部分の担当になるでしょう。要は意味記号の標準化を図ることです。付加情報の８０％は単語の共起性から得られるものになると踏んでいます。さて、どうでしょうか。

まとめると、機械翻訳用に未夢を仕立てるのに必要な作業は次の通りになります。

・デフォルト項目の設定の徹底
・デフォルト項目選択のプロセスモジュールの充実
・記号から関連情報を検索し、フレームにまとめて提示する機能の作り込み
・関連情報の充実（機械翻訳を意識して）

この機械翻訳に当たっては大意を間違えないようにしていくことが肝要です。逐語訳も良いですが、未夢を使うなら意訳に踏み込みたいもの。そのためにも、大局的にフォーカスされたアクターは何か（誰か）、行動の大意は何か（トピックス）を動画システムの意味管理フレームに保持して、それをブレークダウンするようにカットを検索していって、適切な文脈の単語を選択し、美しい対訳を生成していくという方略を採りたいものです。

日本語は頭でっかちなのが読みやすいですが、英語では頭でっかちはまずいわけで、修飾語を後ろに展開したいわけです。意訳的に記述の等価変換をしてこのことを可能とできるでしょう。つまり、頭でっかちとなるような単語は後ろにもっていけるように文章を作っていくのです。あるいは、文章の分断も行ったり、組み替えたりも必要でしょう。未夢のデータを利用すればそれができるはずです。　多義語の本格的な処理を目指せるでしょう。特に、適切に代名詞を生成できるというのが強みです。

第３章　問い合わせ回答システムのための作業

ここは何と言っても、香澄プロジェクトを起こすことです。そして、未夢の中に組み込む機構を設けることです。

その議論の前に香澄プロジェクトのことをもう一度振り返ってみましょう。Prolog ベースのプログラムとして文章のシミュレータを作るのでした。

（例文３．１）　長野で買い物をした。猫がいた。あの猫、三毛といったなと思い出した。

また、子供がキャッチボールしているのをみた。

このシミュレータは次のような Prolog 文になるでしょう。

```
Scene(Start):-Cut(Delta Time).
Cut(Delta Time):-do(買い物),Case（agent 私）,Case（location 長野）.
Cut(DeltaTime):-いる（猫）,Case(agent 猫).
Cut(Delta Time):-思い出す(x),Case(object x),Case(agent 私).
Case(object):-named(猫　三毛).
Cut(Delta Time):-見る(y),Case(object y),Case(agent 私).
Case(object y):-do(キャッチボール),Case(agent 子供)
Scene(End).
```

長野で見た猫の名前は？という問いに対して、トリガーを起こす事をデータベースに通知する命令を設定します。例えば、

Torigger(x),Case(Location 長野）,Object(猫)

取り出す情報を指示します。例えば、

Get(Attribute),Attribute(name),Object(猫)

という風に指定します。

するとシミュレータは Prolog プログラムを実行していき、データベースに場所とかアクターの実行状況を書き込んでいきます。で、Case(object):-named(猫　三毛)でトリガーが発生して、答えを返すのです。

また、子供が掴んでいたのはなに？という問いかけには、連想辞書の助けを借りて、同じように実行していきます。つまり、

「掴む」のプライミングとして、「ノブ」とか、「鳥」とかあると思うのです。その中に「キャッチボール」がある。それは、Prolog 文の単語をなめながら、単語の素材集合のなかに次の文があるのを発見することから事が始まります。

・「キャッチボール：２人以上の人間が、ボールを掴み、ボールを投げる、ボールを受け取る；遊技」

・「受け取る：飛んでくる何かを掴む、あいてから何かを渡されて、それを掴む」

辞書を Prolog で記述していれば、

```
do(キャッチボール):-掴む(ボール),Case(object ボール）,Cut(DeltaTime),
                    投げる(ボール),Case(object ボール),Cut(DeltaTime),
                    受ける（ボール）,Case(object ボール),Cut repeat).
```

これで、元のプログラムからこの命令をも実行するように、項オブジェクトプールを書き換えるのです。そして、トリガーが発して、答えのボールを回答することでしょう。

ということを踏まえて、未夢のデータ構造検索と香澄の文章シミュレーションの２本立てで、問い合わせ回答システムを構築していくことになります。無論、未夢のデータからシミュレーション用の Prolog プログラムを作っていくことになります。それを、香澄システムに渡すのです。

協調的な問題解決技術が必要です。基本的に並列処理でないとスピードが出ないでしょう。ただでさえ、未夢システムの動作は遅くて困ってます。並列処理には大幅な手が入るでしょうし、苦しいところです。資金があれば、始めから並列処理向きにプログラムを作ったのですが、是非もないです。

問い合わせ回答システムとしては、香澄システムがメインですが、未夢も検索機能を問い合わせ回答システム向けにつくっていかねばなりません。楽しいところです。

基本は、トップダウンな階層事に情報を付加して、検索の便を図っていくことでしょう。場所、時間、アクターとそれらの属性が基本的な情報でしょう。必要な情報を必要に応じて取り出せるように辞書検索のキーも属性として保持することになるでしょう。この辺は規格化しておいて、どこでも同じインターフェースで同じ形の情報（エンティティ）オブジェクトを得るようにします。

内的な問い合わせは問い合わせ項目フレームを用います。フレームのスロットを埋めていくのが検索システムの基本です。

第４章　まとめ

問い合わせ応答システムができたとき、意味理解の基盤技術がなったと言えると思います。無論、文章解析の内容だけでなく、業務知識もデータベースをアクセスして取り出せるようにすべきです。それも、文章理解の一部でしょう。でも、単純なデータベース問い合わせなら、たいした意味もない世界の話になってしまいますが、クライアントと会話しながらクライアントの意図を理解して、それをデータベースアクセスに持って行くというレベルなら、意味理解の一つの分野になるでしょう。問い合わせフレームの自動生成として挑戦的な課題ですね。

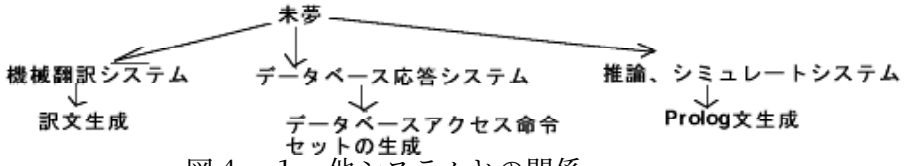

図４．１　他システムとの関係

おわり