図面認識と図面上の推論詳細検討　　2013/02/17（日）

図面を入力し、意味を解析して人の意図を得ることは、言語による計算機との対話と同じくらい重要な技術だと思います。言葉では表せないイメージの世界というものがあります。記号システムを作って、人がそれに従って計算機にデータを送り込むこともできますが、なんかヒューマン・セントリックで無い気がします。同じ事は、自然言語による計算機入力についてもいえるでしょう。人間の言葉で入力するから、親しみのある世界が広がるのです。記号システムにあわせて手で入力するのはしんどくあります。この自然言語入力を補助するのが、図面入力認識システムです。

１．図面認識

図面は基本的に単純な線とシンボルと文字と領域で成っている物とします。写真などは無いものとして、地図なども線画で書かれている情報が読み取れれば良いようなものとします。

処理として、行うことは、次のようなものになるでしょう。

（１）テクスチャや塗りつぶし領域候補を切り出す。これには、空間の大きさ、図面上の点や線の分布の様子を評価していくことによって行います。

【モジュール名】EstimateLineAndArea

（２）シンボルと文字領域を推定する。これはおおきくもなく、小さくもない、線分の固まり領域を抽出して行います。

【モジュール名】EstimateSymbolAndCharacter

（３）線候補を抽出する。図面は線が命です。これを抽出します。シンボルと文字候補も線候補のものは抽出します。そうして、細線化も実行します。

【モジュール名】EstimateLines

（４）Ｌｉｎｅトレースする。線の曲率、特異点、線長、線の交差、線の固まり具合を評価します。閉じ曲線の様子も評価して、領域かどうかの判断をします。

【モジュール名】EvaluateLines

（５）シンボル認識をする。パターンマッチングします。領域と線の配置、向き、大きさ、曲率をつきあわせるのですね。

【モジュール名】EvaluateSymbols

（６）文字認識をする。パターンマッチングします。領域と線の配置、向き、大きさ、曲率をつきあわせるのですね。

【モジュール名】EvaluateCharacters

（７）図面最終認識をする。意味トポロジーデータを返します。
【モジュール名】EvaluateTotalFigure

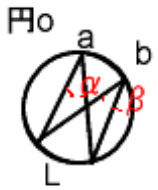
モジュールのパラメータは、
（１）解析窓
（２）２次元イメージのビットマップ
（３）マスクビットマップ
（４）解析結果の意味記号を集約する２次元マップ

です。

知識の持ち方は、知識要素の固まりを見だしの知識としていきたいので、ＲＤＢを想定しています。図面認識では文章による説明と図版による説明の両方が知識オブジェクト（モデル・・・ＲＤＢのテーブル）として記録されるようにします。文章は Prolog コマンド、図版は、ノードとアークのネットワークとして表現します。

２．図面上の推論
２次元から３次元の様子を推論する場合もありますが、基本的にノードと辺とのリンクで、図面を表します。これで十分表せるはずです。ノードも辺もオブジェクトです。データを持ちます。名前、格データ、意味データ、属性データです。今のところこれらは HashMap に（キー、バリュー）の２項関係を保持するということを考えています。

推論開始のコマンドは、基本的に Prolog の表記に則りたくあります。ＬＩＳＰでも良いのですが。
（コマンド例）次の図で、角度αとβは同じ。



【命題コマンド】EQUAL("agent_case","α","with_case","β")；

このコマンドの推論のイメージは次のようになります。

２．１　基本的なプロセス

（１）問題解決進捗管理・・・結果と現状の距離を評価、進捗状況を保持します。

進捗の目安としては、「変数の数と異なる数式の数が同じに近づいているか」、「パターンが同じ部分が大きいか」、「議論が全ての場合を尽くしているか」とかです。

（２）連想処理・・・知識要素をリンクをたどることと、知識要素内を検索します。

（３）パターン発見・・・数式やオブジェクトの並びなどのパターンは一致する部分を評価していく必要があります。

（４）オンデマンド解析・・・イメージ（意味ネットワーク）を解析します。曲率とか、多角形の様子、必要な図形を切り出せるかとか、合成できるかとかを解析します。

オンデマンド解析が無いと、Prolog は全ての解にいたる知識を持っていなくてはなりません。フレーム問題を引き起こします。実現不可能なのです。

（５）数式処理・・・変数を立て、数式を知識から合成して、そして、数式処理の進行を評価・管理していきます。

（６）数学の基本的なプロセス

「公理系」、「基本的な定理系」、「１対１対応」、「集合論」、「群論」、「数え上げ」などです。

２．２　推論の動き

【コマンド】：＝ PROOF("object_case",F);

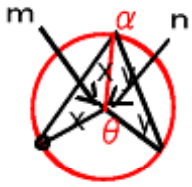
F ：＝ EQUAL("agent_case","α","with_case","β")；

このコマンドの動きを見てみましょう。

（１）角αと角βをただ見ているだけでは、問題が解けません。そこで、弧Ｌに注目して、このＬと中心Ｏ戸の成す扇の角度θを連想してみましょう。これは、円と円の幾何学の基本的な性質を保った知識要素を参照して得る連想です。そこで、θはαとβに共通する連想項目であることが重要です。

（２）θはαとβの共通の変数ですから、αとβがθと数値だけの式から表現できればαとβが同じ大きさになることが結論づけられます。これは、「共通の変数と数値から対象の変数を表現できれば値は同じ」という知識を連想することです。「等しい」という言葉から、この事実を連想する能力が必要ということです。

（３）２つの三角形と角度θ、αの関連する方程式を全部あげていきます。ここで、円の性質から２つの三角形は二等辺三角形であることから、それぞれの三角形の２つの底辺は同じということも連想で得ます。



α =x+y
180=x+x+m
180=y+y+n
360= θ +m+n

これらの方程式から、x と y,m,n を消去できますので、α＝θ/2 が求まります。

この図は、x がマイナスの場合も成り立ちますから、結局、証明は完了したことになります。

三角形の知識と円の知識、・・・この２つを巧く連想に使って、所定の目的を達成したのですが、何にもまして、「等しい」とはどうゆう事かを連想して、その方向で解答の見当を得ていくこと、その見当にどうして近づけていくかという・・・解答処理の制御・・・これも重要であることが分かります。それらの能力は、作り込まねば生まれないわけで、プリミティブな知識（オントロジー）の設計が重要に成るわけです。Prolog ランタイム処理系は、知識とプロセス要素を、問題解決の距離を測りながら呼び出しを制御していくことで、難局を乗り切っていくように動くのですね。

３．まとめ

連想の所が、設計上、一番難しい所でしょう。適切な知識を呼び起こさねばなりません。連想と同時に評価も行っている・・・そんな機序に成ると思えます。問題解決の方向に向かって連想したかということです。

知識オブジェクトの内容を検索していく。そのときに得るデータが問題解決との距離を縮める何かであること、・・・今、手持ちのデータと目標の解で必要となるデータを含むように拡大していっているということを把握する評価です。

あと重要なのが、知識を機械学習していく方策を練ることです。図面と、その図面の意味とを自然言語と数式で与えて、問題解決の為の知識を獲得していくことです。色々の知識間の連想を記憶していくことですので、図面認識と図面上の推論機構が完成すれば、比較的簡単に実現できるでしょう。オンデマンド解析のアルゴリズムもこの中で、実現できるでしょう。

おわり