4M-5 知識ベースシステムのための知識コンパイルについての考察

村上 昌己　横田 治夫*　西田 健次　大場 雅博　伊藤 英則

(財) 新世代コンピュータ技術開発機構　*(株) 富士通研究所

1. はじめに

第五世代コンピュータ計画では，知識を複数のユーザの多様なアプリケーションの間で共有し，知識の利用効率を向上させることを目的の一つとした，知識ベースシステムに関する研究を進めている。筆者らは，このような知識ベースシステムの一部として，知識コンパイラを検討している。この知識コンパイラは，様々な知識表現で記述された知識をある中間コードの上の表現に変換し，知識に対する検索・更新等の操作を中間コード上の演算に落とす機能をもつ。筆者らは，知識ベースシステムが知識表現，推論機構の高度化に対応できるよう，知識コンパイラに柔軟な拡張性を持たせるための方法について検討している。本稿では，知識コンパイラ自身を知識ベース化し，容易にその構築・拡張が出来るように構成する方法について述べる。

2. 準備

ここでは知識ベースシステムにおいて扱われる知識表現の記述言語は，その構文と意味が明確に定まった形式言語と仮定する。即ち意味ネットワーク，フレーム等を記述するために例えばHorn節，S式等の構文の部分集合が用いられ，それらはそれぞれ一階述語論理のHerbland空間，λ論理のD∞等の意味の領域を持っている。以下ではこれから対象とする知識表現の文面の集合を，ある字面の集合SYMの部分集合SYNTAXsで表わし，その知識表現の意味論を与える領域をDとする。Dは計算可能な部分を取り出したものであるとする。SYMからDへの意味写像をsemであらわす。

次に，コンパイラのターゲットとなる中間コードを，やはりあるデータ型の意味領域として扱う。例えば，Prologを中間コードとした場合，それは項の集合が値の集合となり，単一化をその上の演算とするデータ型から構成される領域と考える。このような領域をSymで表わす。

ここでSYNTAXsおよびDが定まったとき，そのSYNTAXsの元のDでの意味に対応するSymの元が存在するようSymを構成することが必要となる。即ち領域SymとDの間に次のような写像：

Sem : Sym → D

が存在し，以下の関係を充たすことが求められる。

∀P ∈ SYNTAXs, sem(P) ∈ [D* → D]

∀Q ∈ SYNTAXs, sem(Q) ∈ D* について

p ∈ [Sym→Sym], q∈ Symが存在し

Sem(p(q)) = Sem(p)(Sem(q))

= sem(P)(sem(Q))

ここでD*はDの有限カルテジアン積を表わす。

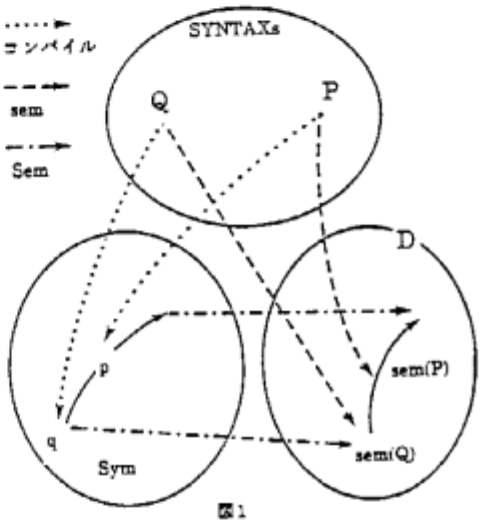

図1

以上の議論では，SYNTAXsの元とSymの元との対応は天下り的な物であった。知識コンパイラを実現するとは，上のPからpを求めるような関数を実現することとなる。コンパイラは知識表現及び中間コードの字面そのものをデータとして扱う。ここで，中間コードの字面の領域をSYNTAXc (⊂ SYM) で表わす。字面をデータとして扱うときは，それぞれSymc, Syms という領域の元とみなすことにする。即ち，

Ψ0s : SYNTAXs → Syms

Ψ0c : SYNTAXc → Symc

なる可逆な写像を用いて，コンパイラcomps: Syms → Symcは次のような写像として記述できる。即ちp, q, P, Qを上の通りとすると，

p = Ψ1(Ψ0c⁻¹(comps(Ψ0s(P))))

Compilation of Knowledge for Knowledge Base System

Masaki Murakami, Haruo Yokota*, Kenji Nishida, Masahiro Oba, Hidenori Itoh

ICOT, *Fujitsu Laboratories ltd.

q = Ψ1(Ψ0c⁻¹(comps(Ψ0s(Q))))

となる．ただしΨ1：

SYNTAXc → [Sym* → Sym]

は，中間コードの意味写像であり，x∈SYMに対して，

Sem(Ψ1(x)) = sem(x)

である．

3． コンパイラの知識ベース化

本節では，先に示したコンパイラcompsに対応するSymの上の演算について考察する．ここでは特に，コンパイラ自身を知識としてあつかう場合を考慮して，コンパイラを中間コードの構文によって記述する場合について述べる．まずcompsをSymからSymへの関数として実現したい．そのためにはSymは，

SymsUSymc ⊂ Sym

でなければならない．Ψ0：SYM→Symを次のように定義する．

Ψ0c(x) if x ∈ SYNTAXc

Ψ0(x) =

Ψ0s(x) if x ∈ SYNTAXs

あるSYNTAXsについて，次の式を成立させるようなSym上の関数psが存在すれば，ps は知識表現sのインタプリタとなる．

ps ∈ [Sym*→Sym] かつ

∀ P1, P2 ∈ SYNTAXs,

sem(P1) ∈ [D*→D]，sem(P2) ∈ D*に対し，

Sem(ps(Ψ0(P1), Ψ0(P2))) =

sem(P1)(sem(P2))

ここでIs ∈ SYNTAXc，Ψ1(Is) = ps ならば，インタプリタが記述できたことになる．以下ではこのようなIsをインタプリタという．ps の存在は，ある知識表現による記述を入力として受け取り，意味領域の上での動作をSym上でシミュレートできることを意味している．

[定義] Φ ∈ [Sym* → Sym] を次のように定める． ∀P，∀Q ∈ SYMsem(Q)∈D，

sem(P)∈[D → [D* → D]] について

Φ(Ψ0(P), Ψ0(Q)) = Ψ0(R) (R ∈ SYM)

とする．ただし，

Ψ1(R) = Ψ1(P)(Ψ0(Q))．

このような条件を充たすΦは，いわゆるcurry化された関数を評価していることが次の命題よりわかる．

[命題1] P ∈ SYM，Q ∈ SYM に対し，次に示す条件：

Ψ0(R) = Φ(Ψ0(P), Ψ0(Q))

を充たす R ∈ SYMが存在するとき，そのようなRについて

sem(R) = sem(P)(sem(Q))

が成立する．

[命題2] Is ∈ SYMがあるSYNTAXs ⊂ SYM のインタプリタであるとき，P ∈ SYNTAX に対してR ∈ SYMを次のように定める．（sem(P) ∈ [D*→D]）

Φ (Ψ0(Is), Ψ0(P)) = Ψ0(R)

そのとき，

sem(P)= sem(R)

すなわちRはPと等価なD上の関数を記述しているコンパイルコードとなる．ゆえに，Sym上の関数：

λX.Φ (Ψ0 (Is) ,X)

は先に挙げたcompsの性質を充たすコンパイラとなることが命題2よりわかる．このコンパイラ自身を表示する字面（SYMの元)は，もしΦを記述する字面が得られれば，次のようにして得ることができる．これによってコンパイラ自身を知識として字面データの形でSym上で扱うことが可能となる．即ち，

F ∈ SYM, Ψ1(F) = Φ

とおくと

Φ(Ψ0(F), Ψ0(Is)) ∈ Sym

Φ(Ψ0(F), Ψ0(Is)) = Ψ0(Ss) なるSs∈ SYMを考えると，このSsがコンパイラを記述してることが次の命題によりわかる．

[命題3] Ssを上記の通りとする．Q∈ SYNTAXs，sem(Q) ∈ [D* → D] について，

Ψ1(Ss)(Ψ0(Q)) = Ψ0(T)

となる T∈ SYM，sem(T)を考えると，任意の x ∈ SYM，sem(x) ∈ D* について，

sem(T)(sem(x)) = sem(Q)(sem(x))．

4．まとめ

知識ベース内で知識として取り扱える知識コンパイラの構成について述べた．この構成法では，知識表現のコンパイラはそのインタプリタから，中間コードの領域の上の演算Φを用いて構成される．これは，コンパイルを部分評価と見做す立場(2)に沿ったものである．今後，Prolog等の上にΦを実現し，それを用いた知識コンパイラの構築の実験を進めてゆきたい．

[謝辞] 有益な討論をして下さったKC会議メンバに感謝します．

参考文献

(1) Yokota et.al. "A Model and Architecture of Relational Knowledge Base" ICOT-TR, No.144 1985

(2) 二村：プログラムの部分計算法，電子通信学会誌，Vol.66, No.2, 1983