制約式の伝播による オブジェクト指向計算モデル

横山 孝典

(財) 新世代コンピュータ技術開発機構

1. はじめに

近年，知識工学，ユーザインタフェース，シミュレーション，データベース等の分野で，オブジェクト指向への制約の導入が図られている[1〜6]。制約の導入によりオブジェクト間や属性間の関係を宣言的に記述できるという利点がある。

しかし，従来手続き的プログラミングの視点でとらえられてきたオブジェクト指向の枠組みに，単純に制約を導入しただけでは両者の概念がうまく融合しない。例えば，従来のオブジェクト指向言語ではオブジェクトの動作がメソッドという手続き的表現により記述され，メッセージ・パッシングはメソッドの起動という手続き的な意味を持つが，これらを宣言的プログラミングの枠組みでとらえ直す必要がある。

本研究は知識処理への応用を目的に，制約の導入により，宣言的プログラミングとしてのオブジェクト指向を確立することを目指している。本報告では，オブジェクトをデータと制約のまとまりと見なし，制約充足機能を有するオブジェクト間で制約式を伝播することにより問題解決を行う計算モデルを提案する。

2. オブジェクトと制約

一般にオブジェクトはひとまとまりのデータとそれに関する手続きをまとめたものと見なされる。データはオブジェクト変数，スロット等で表現されるオブジェクトの属性や構成要素などの情報である。手続きはメソッドで表現される。

これに対しここでは，オブジェクトをデータと制約をまとめたものと考える。そしてオブジェクトは常に制約充足状態を維持する機能を有するものとし，オブジェクトが満足すべき制約を宣言的に記述したり付加したりするのみで，オブジェクトの自律的な問題解決を実現する。すなわち，

オブジェクト =
データ + 制約の記述 + 制約充足機構

と表現できる。

なお，オブジェクト指向に制約を導入した従来システムの多くは，制約をオブジェクトとは独立の概念と見なしたり，制約オブジェクトという概念を導入する等の方式を採用しているが，複数のオブジェクト間に制約が発生するのは，それらのオブジェクトを含む上位のオブジェクト，あるいは場（これもオブジェクトで表現できる）が存在するためと考えるのが自然である。従って制約は上位のオブジェクトに記述すればよく，制約をオブジェクト外の独立した概念と考える必要はないと考える。

3. 情報隠蔽

オブジェクト指向の重要な考え方のひとつに情報隠蔽，あるいはデータのカプセル化と呼ばれるものがある。ところがオブジェクト指向に制約を導入し，異なるオブジェクトのデータ間の制約を記述可能とすることは，データのカプセル化に反する可能性がある。

しかし，一般のオブジェクト指向言語ではデータは隠蔽するものの，オブジェクトにメッセージを送信するためのメッセージの形式，すなわち，メソッド名、引数などのインタフェース情報を外部に公開する必要がある。これは、データ抽象が手続き的（操作的）プログラミングの立場であり、（外部から見える）オブジェクトの性質（機能）をそれがもつ「手続き」によって特徴づけるという考え方を採用しているためと考えられる。

これに対し，宣言的プログラミングの立場からは，（外部から見える）オブジェクトの性質は，手続きでなく，それがもつ代表的な「属性」データによって特徴づけられるという考え方を採用することができる。従って，必ずしも全てのデータを隠蔽する必要はなく，オブジェクトの抽象的性質を表す属性（データの一部）は公開してよい。

外部から見えるオブジェクトの性質を表わすデータを「公開属性」と呼ぶことにする。公開属性は外部から参照でき，公開属性間に制約を定義できる。もちろん，宣言したオブジェクト内であれば公開されてない属性間にも制約を定義できる。公開属性という考え方はデータ抽象の考え方とは異なるが，宣言的プログラミングや知識表現における抽象化の考え方としては違和感はないと考える。

4. オブジェクトの制約充足機能

一般のオブジェクト指向ではオブジェクト間のメッセージ・パッシングにより処理が進むのに対し，本方式ではオブジェクト間の制約伝播により処理が進む。制約伝播によって伝達される情報は，従来のシステムでは属性値のみであることが多かったが，本方式では述語や数式のような制約式そのものであるという特徴がある。制約には属性値に関する制約，複数の属性間の関係，オブジェクト間の関係，オブジェクトの型（クラス）に関する制約等がある。

制約には静的に宣言されるものと動的にオブジェクトに付加されるものがある。前者はクラスに記述され，変更したり，削除したりすることはできない。これを「恒久的制約」と呼ぶことにする。後者は処理の実行中に動的に付加され，不必要になれば削除することができる。これを「動的制約」と呼ぶ。通常の動的制約は付加された後，明示的に削除されない限りオブジェクトは常にそれを満足する必

Object-Oriented Computation Model with Constraint Propagation
Takanori YOKOYAMA
Institute for New Generation Computer Technology

要がある。これに対し，付加時に一時的に作用するものの，制約充足が終了した後は無効となる制約がある。これを「一時的制約」と呼ぶことにする。恒久的制約及び動的制約はオブジェクトに記憶されるが，一時的制約は記憶されない。

制約式を受け取ったオブジェクトは制約充足処理を実行し，制約を満足するように状態を変化する。オブジェクトの基本動作を図1により説明する。まず制約式が与えられたならば，制約充足機構は制約式の定義を参照しながら，与えられた制約式と記憶されている制約式を同時に評価する。制約評価の基本は制約論理プログラミング[7]の考え方に従っており，評価とは制約式を簡約化すること（標準形への変換）である。簡約化により，属性値を算出したり，他の特定のオブジェクトのみに関する制約式を抽出することができる。

制約式の定義は，例えば

```
dif_p_q(X) :- p - q = X;
```

のようなホーン節の形式で記述できる。この例で，属性 p の値が 5 で q の値が未定の時，制約 dif_p_q(3) が伝播されてきたとすると，q = 2 が得られる。

評価結果はオブジェクトの状態に反映される。すなわち，得られた属性値はデータ記憶に格納され，一時的制約でなければ，簡約化された制約式は制約式記憶に記憶される。そして他のオブジェクトに関する制約式が得られれば，それをそのオブジェクトに伝播する。与えられた制約が通常の動的制約であれば伝播する制約も動的制約，一時的制約であれば伝播する制約も一時的制約として扱う。

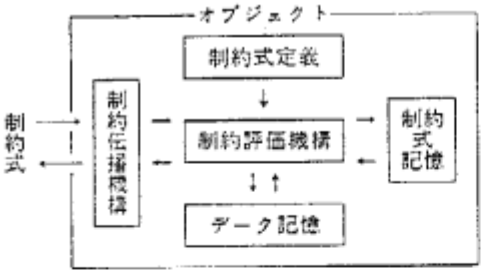

図1 オブジェクトの制約充足機構

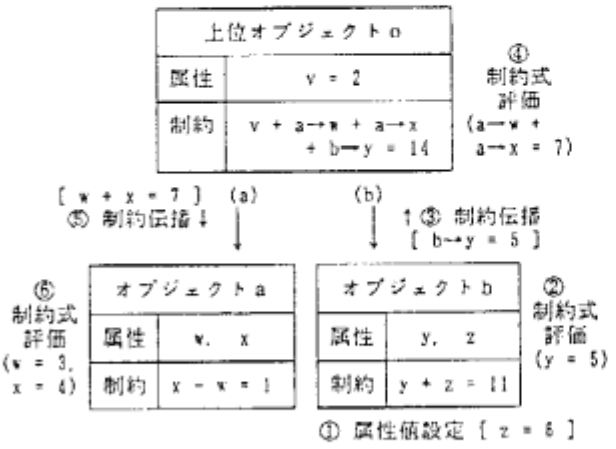

図2 オブジェクト間の制約伝播の例

5．制約伝播とメッセージ

制約伝播の例を図2に示す。ここでオブジェクトa，bは上位オブジェクトoの構成要素である。また制約式中の"a→x"は構成要素 a の属性 x を表わす。オブジェクトbに与えられた制約（属性値設定）により，①から⑥のように制約を評価しながら制約伝播がなされ，属性値が決定していく。

この例では伝播される制約は数式（標準形）であるが，ユーザ定義の制約式の場合もある。例えば上位オブジェクトに制約式 differ(p, a→r) が存在し，p = 2 が与えられたならば，differ(2, a→r) は構成要素 a のみに関する制約式なので，そのオブジェクトに伝播される。

ところで，制約式の定義に従って一般的手法により制約式を解くことはコストが大きいので，制約式の定義（解き方）をオブジェクト（クラス）毎にドメインに依存したヒューリスティクスを含んだ形で記述することが考えられる。この定義はそのオブジェクト（クラス）内でのみ有効なものとなる。さらに，あらかじめ制約式を伝播すべきオブジェクトがわかっている場合は，制約式の定義を例えば

```
dif_p_ar(X) :- Y = p - X, a:dif_r(Y);
```

のように記述し，制約 dif_r(Y) を構成要素 a に伝播することを明示的に指定することも考えられる。

以上のような考え方に立つと，オブジェクト指向におけるメッセージ・パッシングを制約伝播の一種と見なすことができる。ただし，通常のメッセージパッシングは副作用的にオブジェクトの状態を変更するが，それがその後も作用し続けることはないから，一時的制約の伝播である。また，一般のオブジェクト指向言語におけるメソッドは，ここでは制約式の定義（解釈）に対応する。

6．おわりに

以上オブジェクト指向と制約を融合する手法を提案した。本方式は制約式の伝播により問題解決を行うものである。そして，公開属性による抽象化という考え方を導入するとともに，制約を恒久的制約，動的制約，一時的制約に分類し，メッセージパッシングを一時的制約の伝播と見なせることを明らかにした。また，ここでは触れなかったが，オブジェクト指向の特徴のひとつであるクラスの継承機能はそのまま本方式に適用でき，オブジェクトが満足すべき制約や制約式の定義等を継承させることができる。

現在，本方式の知識表現システムへの適用を図っている。今後はいくつかの残された課題について検討を深めるとともに，並行処理への展開を図っていきたいと考えている。

参考文献

[1] Harris "A Hybrid Structured Object and Constraint Representation Language", Proc. AAAI-86 (1986)
[2] Borning & Duisberg "Constraint-Based Tools for Building User Interfaces", ACM Trans. on Graphics, vol.5, no.4 (1986)
[3] Shepherd & Kerschberg "Constraint Management in Expert Database Systems", in Kerschberg (ed.) "Expert Database Systems" (1986)
[4] 杉本ほか "オブジェクト指向言語VEGAMSの制約保持機能(1)制約保持機構", 第35回情処全大, 2R-3 (1987)
[5] 中島 "制約伝播機構を内蔵するオブジェクト指向言語: COOL", 情報処理学会論文誌, vol.30, no.1 (1989)
[6] 横山 "設計対象記述のための知識表現システムFREEDOMの提案", 情報処理学会研究会資料, 88-AI-60 (1988)
[7] Jaffer & Lassez "Constraint Logic Programming", Proc. 14th ACM POPL Conf. (1987)