1M−6

E-R モデルと制約に基づく仕様部品の合成と制約論理プログラムへの変換

佐藤 正和　橋本 正明　竹中 豊文
ATR 通信システム研究所

1 はじめに

筆者らは、E-R モデルに基づく宣言的なプログラム仕様の記述言語 (PSDL) およびそれからのプログラム合成について研究を行なってきた [1, 3]。PSDL で記述される属性間の制約は従来、計算式として与えられていたが、仕様の部品化とその合成の柔軟性を達成するためには制約指向プログラミングで使われている双方向制約伝搬を可能とする制約式の形でこれを記述できることが望ましい。本論文では、PSDL に制約式の記述を可能とする PSDL-GR (Program Specification Description Language Graffic Representation) についてそれを用いた仕様部品の合成と実行方法を述べる。

2 E-R モデルと制約

PSDL-GR では、PSDL で定式化された 3 階層のうち、情報層およびデータ層のモデルを用いている。情報層では、E-R モデルに従って対象世界をエンティティ・タイプ (Entity type) とリレーションシップ・タイプ (Relationship type) で表現する (図 1(上))。エンティティ・タイプはその属性を表現するための一つの主キー・アトリビュート (Primarly key attribute) および複数のアトリビュート (Attribute) をフィールドに持ち、同種のアトリビュートを持つ個々のエンティティの集合である。また、リレーションシップ・タイプは、エンティティ間を関係付けるための主キー・アトリビュートの組として表現されるリレーションシップの集合である (図 1(下))。

情報層では以下の 3 種の制約を記述することによって、宣言的なプログラム仕様を与えることを可能とする。

- **エンティティ存在従属性制約**
  あるエンティティの属性値に基づいて、他のエンティティとそれらを対応づけるリレーションシップが存在するための条件式。
- **リレーションシップ存在従属性制約**
  複数のエンティティの属性値に基づいて、エンティティ間にリレーションシップが存在するための条件式。
- **属性値従属性制約**
  リレーションシップで関連づけられたエンティティ間で、あるエンティティの属性値と他のエンティティの属性値が満たすべき制約式。

以後、これらの制約を順に *ed, rd, ad* と書く。PSDL では *ad* が通常の計算式で書かれていたが、PSDL-GR では制約式を用いている点が異なっている。また、*ed, rd* も制約式として解釈される。

一方、データ層はエンティティやリレーションの値を実際に与えるものである。データ層から変数として与えられるアトリビュートおよびデータ層からは与えられないエンティティに属するアトリビュートの値は上述の制約を充足するようにシステム側で自動的に決定される。

3 部品の合成

PSDL-GR で書かれた個々のプログラム仕様を部品として相互に結合し、新たなプログラム仕様を作成する場合、結合の方法として以下の 3 種の基本操作を定義する (図 2)。

- 直列結合
  1 つのエンティティ・タイプを介してリレーションシップ・タイプが直列に接続する。

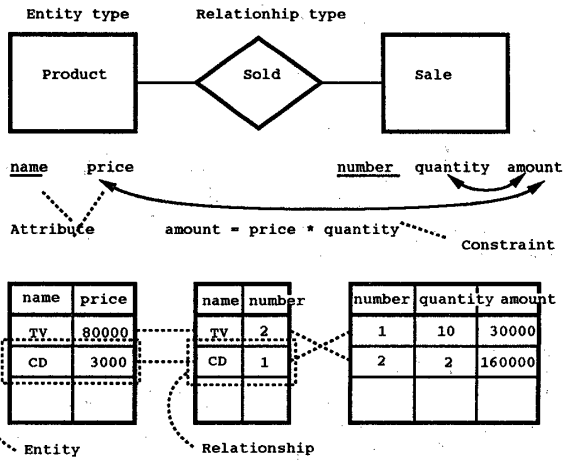

図 1: PSDL-GR の情報層

- 並列結合
  リレーションシップ・タイプの両端のエンティティ・タイプを介して並列に接続する。
- 融合
  並列結合においてリレーションシップ・タイプおよび *rd, ed* が等しく *ad* が異なる場合は両者は融合され、*ad* は AND で結ばれるものとする。

このとき *ad* を制約式として解釈することにより、以下のようなメリットが得られる。

- データフローの双方向性
  制約の伝搬は双方向に進むので、部品を結合する際にデータフローの方向性を気にする必要がない。従って、一つのプログラム仕様を結合箇所を替えることにより多様な目的に再利用できる。
- 制約式の簡約化
  複数の *ad* が融合によって結ばれた場合、それらを満たすような解を自動的に求める機構が実現できる。

4 制約論理プログラム言語による PSDL-GR の実行

プログラム仕様は、プロトタイピング等を考慮すると直接実行可能であることが望ましい。筆者は、PSDL-GR の実行系として制約論理プログラミング言語 [4] (以下、CLP と記す) を用いることを計画している。ここでは、簡単な操作により PSDL-GR が CLP に変換可能であることを示す。

4.1 拡張ホーン節への変換

PSDL-GR の情報層におけるエンティティ・タイプやリレーションシップ・タイプはリレーショナルデータベースにおけるリレーションと同様の表構造を持っている。ただし、リレーショナルデータベースではデータ構造の外側で定義される問い合わせ言語 (Query language) によって計算が規定されるのに対し、PSDL-GR ではデータ構造と一体となって定義される制約に従って計算が行なわれる点に違いがある。従来、リレーショナルデータベースと Prolog との

Specification Synthesis and Implementation Based on E-R Model and Constraint
Masakazu Sato, Masaaki Hashimoto, Toyofumi Takenaka
ATR Communication Systems Research Laboratories

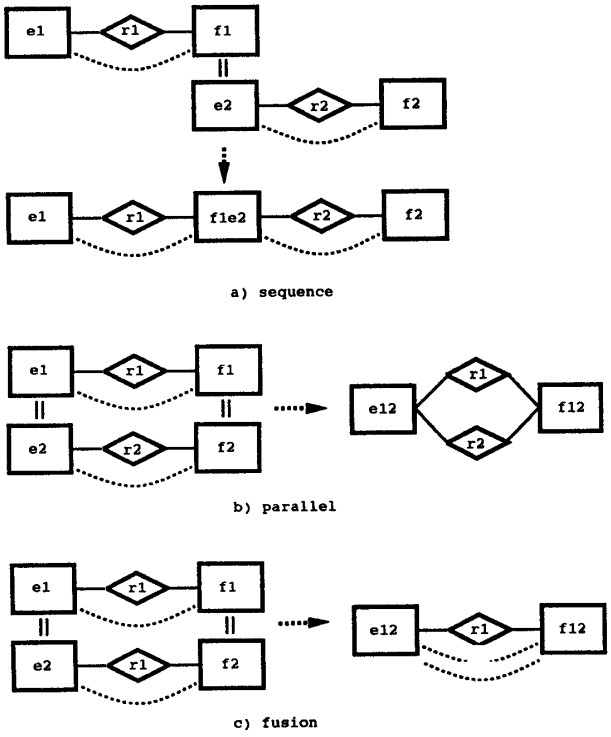

図 2: 部品結合の各種形態

間に深いつながりがあることが知られている [2]。同様な議論によって PSDL-GR と CLP の関係を定式化する。

まず、2 つのエンティティ・タイプと 1 つのリレーションシップ・タイプで構成される PSDL-GR について、エンティティ間がリレーションシップで対応付けられる際の対応関係が 1 対 1 に限られる場合を考える。

エンティティ・タイプ e およびリレーションシップ・タイプ r を以下の述語として定義する。

$$e(Xk, X_1, \ldots, X_n).$$

$$r(Xk, Yk).$$

この時、エンティティタイプ e,f、リレーションシップタイプ r および制約 *er, rd, ad* 間の関係を、ホーン節を制約論理に拡張した拡張ホーン節で表現する。

$$e\_r(Xk, X_1, \ldots, X_n, Yk, Y_1, \ldots, Y_m) :-$$
$$\quad e(Xk, X_1, \ldots, X_n), r(Xk, Yk), f(Yk, Y_1, \ldots, Y_m),$$
$$\quad ad, rd, ed.$$

ここで、ad は $Xk, X_1, \ldots, X_n, Yk, Y_1, \ldots, Y_m$ の部分集合を含む制約式、*rd, ed* は条件式である。一方、データ層は e, r, f の変数にアトムを代入した述語の集合として与えられる。

これらを CLP に与えた状態で、

$$?- \; e\_r(Xk, X_1, \ldots, X_n, Yk, Y_1, \ldots, Y_m).$$

の問い合わせにより、PSDL-GR を実行する。

4.2 部品の結合操作

前節で述べた部品の結合操作は、結合前の個々の部品を e_r1, e_r2 とすると以下に示すような拡張ホーン節に対する操作に置き換えられる。ここで、$\{Xk, X_1, \ldots, X_n\}$ は $Xk, X_1, \ldots, X_n$ の変数名の集合とする。

- 直列結合

$$e\_r(X1k, X1_1, \ldots, X1_n,$$
$$\quad Zk, Z_1, \ldots, Z_l, Y2k, Y2_1, \ldots, Y2_m) :-$$
$$\quad e\_r1(X1k, X1_1, \ldots, X1_n, Y1k, Y1_1, \ldots, Y1_j),$$
$$\quad e\_r2(X2k, X2_1, \ldots, X2_k, Y2k, Y2_1, \ldots, Y2_m).$$

ただし、

$$\{Zk, Z_1, \ldots, Z_l\} =$$
$$\quad \{Y1k, Y1_1, \ldots, Y1_j\} \cup \{X2k, X2_1, \ldots, X2_k\}$$

- 並列結合

$$e\_r(Zk, Z_1, \ldots, Z_n, Wk, W_1, \ldots, W_m) :-$$
$$\quad e\_r1(X1k, X1_1, \ldots, X1_i, Y1k, Y1_1, \ldots, Y1_j).$$

$$e\_r(Zk, Z_1, \ldots, Z_n, Wk, W_1, \ldots, W_m) :-$$
$$\quad e\_r2(X2k, X2_1, \ldots, X2_k, Y2k, Y2_1, \ldots, Y2_l).$$

ただし、

$$\{Zk, Z_1, \ldots, Z_n\} =$$
$$\quad \{X1k, X1_1, \ldots, X1_i\} \cup \{X2k, X2_1, \ldots, X2_k\}$$
$$\{Wk, W_1, \ldots, W_m\} =$$
$$\quad \{Y1k, Y1_1, \ldots, Y1_j\} \cup \{Y2k, Y2_1, \ldots, Y2_l\}$$

- 融合

$$e\_r(Zk, Z_1, \ldots, Z_n, Wk, W_1, \ldots, W_m) :-$$
$$\quad e\_r1(X1k, X1_1, \ldots, X1_i, Y1k, Y1_1, \ldots, Y1_j),$$
$$\quad e\_r2(X2k, X2_1, \ldots, X2_k, Y2k, Y2_1, \ldots, Y2_l).$$

ただし、

$$\{Zk, Z_1, \ldots, Z_n\} =$$
$$\quad \{X1k, X1_1, \ldots, X1_i\} \cup \{X2k, X2_1, \ldots, X2_k\}$$
$$\{Wk, W_1, \ldots, W_m\} =$$
$$\quad \{Y1k, Y1_1, \ldots, Y1_j\} \cup \{Y2k, Y2_1, \ldots, Y2_l\}$$

4.3 エンティティの対応が非 1 対 1 の場合

PSDL においてエンティティの対応が非 1 対 1 になる場合としては、1) 同一エンティティが *ad* によって複数回参照される、2) 一つのエンティティの対して複数のエンティティが参照される（集合関数）、3) 上記の複合、が存在する。集合関数は PSDL の述語的な解釈では 2 階の述語として扱われるが、制約式として解釈することは難しい。自己参照リレーションシップタイプについても同様の問題が存在する。

5 おわりに

本稿では、制約指向のプログラム仕様記述法 PSDL-GR の概要とこれを用いた部品の複合および実行方法について述べた。現在の検討では、制約式がエンティティ間の対応が 1 対 1 の場合に制限されているが、1 対多、多対 1、多対多を許す一般的な場合についての検討が今後の課題である。

参考文献

[1] 橋本正明. プログラム仕様記述のための計算指向 EAR モデル. 情報処理学会論文誌; Vol. 27, No. 3, pp. 330–338, 1986.

[2] D Li. *A Prolog Database System*. Research Studies Press,Ltd., England, 1982. 安部憲広訳 (1985). PROLOG データベースシステム. 近代科 学社.

[3] 岡本克己, 橋本正明. 制約指向の概念モデルを用いた高次部品化によるプログラ ム合成. 信学技報, SS89-18, 1989.

[4] 溝口 文雄, 古川康一, J-L. Lassez(編). 制約論理プログラミング. 共立出版, 1989.