制約充足問題としてのビューの更新可能性と SQL による解法
―バッグ意味論の下での直積ビューの更新可能性―

増永 良文† 長田 悠吾‡ 石井 達夫‡

†お茶の水女子大学（名誉教授） 〒112-8610 東京都文京区大塚 2-1-1
‡SRA OSS, Inc. 日本支社 〒171-0022 東京都豊島区南池袋 2-32-8
E-mail: †masunaga.yoshifumi@ocha.ac.jp ‡{nagata, ishii}@sraoss.co.jp

あらまし リレーショナルデータベースにおけるビュー更新には,これまで理論的にも実践的にも強い制約が課せられてきた．しかしながら，我々の提案してきた「更新意図の外形的推測に基づくアプローチ」，略して「意図に基づくアプローチ」の下では，従来のアプローチでは更新不可とされてきた直積ビューや結合ビューが状況により更新可能となる．この新規アプローチは PostgreSQL 上でプロトタイプされて，所望の動作が確認されている．しかしながら，実装された更新可能性判定アルゴリズムは暴力法によっており，より洗練された判定アルゴリズムが求められてきた．我々は，この問題に検討を加え,一般にバッグ意味論の下での直積ビューの更新問題が非線形連立方程式を解く問題に帰着できることを明らかにしてる．本報告では，その問題にさらに検討を加えた結果，この問題は制約充足問題(CSP)として定式化でき，一般に CSP は NP 完全であるものの，CSP を解くアルゴリズムの一つとして提案された併合法（merge method）を用いて，ストレートに解けることを示す．さらに，併合法の特性に着目すると,バッグ意味論の下での直積ビューの更新判定アルゴリズムがリレーショナルデータモデルの自然結合演算で実現できるので，リレーショナル DBMS 上で意図に基づくアプローチを実装するうえで，暴力法に代わる新規実装法の提案に繋がると考えられる．

キーワード ビュー，ビュー更新問題，更新意図の外形的推測に基づくアプローチ，意図に基づくアプローチ，バッグ意味論，SQL，PostgreSQL，リレーショナルデータベース，非線形連立方程式，制約充足問題，併合法

1 はじめに

ビューはリレーショナルデータモデルの発案者である E. F. Codd により導入さたが[1]，ビューを更新しようとすると異状をきたす場合があり，どのような場合に異常が発生しないのかを問う「ビュー更新問題」はこれまで長年にわたり数多くの研究が報告されてきた．従来のアプローチは構文的[2], 意味的[3], そしてインタラクティブ[4] アプローチに大別されるが，いずれも更新可能とされるビューへの制約が強かった．

我々は，この原因が更新可能性を，従来のアプローチはデータベースの“スキーマレベル”で論じてきたことにあることを指摘し，そうではなく，それを“インスタンスレベル”で論じると，従来更新不可とされてきた直積ビューや結合ビューが状況によっては更新可能となる「更新意図の外形的推測に基づくアプローチ」，略して「意図に基づくアプローチ」を発案し，これまで更新不可とされてきた直積ビューや結合ビューへの更新可能性に道を拓いた[5,6].

当初，このアプローチはリレーショナル代数で定義されるビュー，つまり「集合意味論」の下でのビューを対象としていたが，表（table）に重複した行（row）の生起を許す SQL，つまり「バッグ意味論」（マルチ集合意味論ともいう）の下でのビュー更新可能性の判定にも適用できるように拡張されている[7,10]．また，意図に基づくアプローチをオープンソースのリレーショナル DBMS として広く流布している PostgreSQL にプロトタイピングし，意図した動きをすることを確認してきた[8,9,10,12].

しかしながら，このプロトタイピングで実装された直積ビューや結合ビューへの更新要求変換アルゴリズムは,「暴力法」（brute force，総当たり法ともいう）であったために，より明解なビュー更新判定アルゴリズムが模索されることとなった．この問題に対して，検討を加えた結果，そもそも，意図に基づくアプローチは，更新対象となったビューがその更新要求を受け付けることができるか否かを決定するために，ビューを“一時的にマテリアライズする(体現化する)”ことにその本質があるが,改めてそこに着眼することにより,一般にバッグ意味論の下での直積ビューとそれに対する更新操作が与えられたときに，その変換可能性を判

定する問題が「非線形連立方程式」が解を持つか否かという問題に帰着できることを明らかにした[11, 13].

本稿では，この問題は，一般に制約充足問題（CSP）[14]の一つとして知られている整合ラベリング問題(consistent labeling problem, CLP)として定式化でき，さらに CLP を解くアルゴリズムの一つである併合法（merging）[15]を適用することにより，問題解決できることを示す．本稿では，さらに，併合法を用いたバッグ意味論の下での直積ビュー更新判定アルゴリズムがリレーショナルデータモデルの自然結合演算で記述できることを確認する.これは,リレーショナル DBMS 上で意図に基づくアプローチを実装するうえで，暴力法に代わる新規実装法の提案に繋がると考えられる．

以下，2 章でビュー更新問題と意図に基づくアプローチを要約する．第 3 章でバッグ意味論の下での直積ビューの更新可能性判定が非線形連立方程式として表現できることを示す，第 4 章では，第 3 章で示した問題が整合ラベリング問題(CLP)として表現され,その解法の一つである併合法により，直積ビュー更新問題が SQL の自然結合演算を実行することで判定できることを示す．第 5 章はまとめである．

2 ビュー更新問題と意図に基づくアプローチの概要

2.1 バッグとビュー

バッグ(bag)とは，次のように定義さえる．

【定義 1】（バッグの定義）

$\boldsymbol{R}(A_1, A_2, \ldots, A_n)$をリレーションスキーマとする．$\Omega_{\boldsymbol{R}}=\{A_1, A_2, \ldots, A_n\}$，dom をドメイン関数とし，$dom(\boldsymbol{R})=dom(A_1)\times dom(A_2)\times\cdots\times dom(A_n)$とする．このとき，$\boldsymbol{R}$ のバッグ R を次のように定義する．

$$R=\{t_i(k_i) \mid t_i\in dom(\boldsymbol{R}) \ \wedge \ k_i\geqq 1\}$$

ここに，k_i は R 中に生起するタップル t_i の**重複度**（multiplicity）を表す．$k_i=1$ のとき，$t_i(1)$と書かずに，単に t_i と書くことも多い.バッグ R の濃度(cardinality)，$|R|_{bag}$ と表す，は有限とし，$|R|_{bag}=\sum_{i=1,..,p} k_i$ と定義する．また，$(\forall t_i(k_i), t_j(k_j)\in R)(t_i = t_j\Rightarrow k_i = k_j)$である．これは,重複タップルの識別不可能性を表すためである．ここに，⇒は含意（logical implication）を表す．

バッグ R は$(\forall i=1,..,p)(k_i = 1)$のときリレーション(＝集合)である.SQL で扱われるテーブルやビューはリレーションではなく，一般にバッグであることに注意する．本稿では，SQL という実用言語で定義されるビューの更新可能性を論じるために,「集合意味論」(set semantics)ではなく,「バッグ意味論」(bag semantics)の下でのビュー更新可能性を取り扱う．

SQL でビューが次のように定義される．

【例題 1】(SQL でのビュー定義例)

R(A)={1(2), 2}，S(B)={1, 2(2)}を実バッグ*とし，バッグ直積ビュー†V を次のように定義する．

```
CREATE VIEW V (A, B)
AS
SELECT * FROM R CROSS JOIN S
```

このビューは，バッグ代数[7, 10]では次のように定義される．

$$V=R\times_{bag}S$$

2.2 ビュー更新問題

ビュー更新問題を整理しておく．そのために，まずビューの更新可能性の定義を示す[2]．ビューは実バッグ群がなすデータベース状態(database state)からビュー状態(view state)への「関数」(function)であると定義することから始まる．

【定義 2】（ビューの更新可能性）

s_τをある時刻τにおけるデータベースの状態，V をビュー定義，$V(s_\tau)$は（その時刻τにおける)ビューの状態，$\boldsymbol{u}$ を $V(s_\tau)$に対して発行された更新操作とするとき，$\boldsymbol{u}$ が**変換可能**(translatable)であるとは，$\boldsymbol{u}$ を s_τ への更新操作に変換する副作用がなく(no side effects)かつ一意(unique)な変換(translation) T が存在するときである．

すなわち，図 1 の可換図式(commutative diagram)が成立するときをいう．なお，変換 T に副作用がないとは，$\boldsymbol{u}(V(s_\tau)) = V(T(\boldsymbol{u})(s_\tau))$ が成立することである．変換 T に一意性を課したことは議論の余地があるところだとしながらも，その理由は T に代替案があった時，その選択基準を設けられないためである．

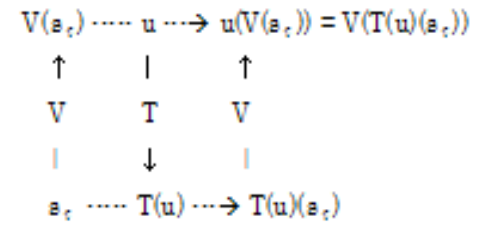

図 1 時刻τにおいてビュー更新操作 u が変換可能であることを示す可換図式

さて,「ビュー更新問題」とは，いかなる時に図 1 の可換図式が成立するのかを問う問題であるが，ここで，可換図式の成立について補足説明をする．上の定義から明らかなように，可換性は本来インスタンスベースである．つまり，あるデータベースの状態，あるビュー定義,そしてある更新操作が与えられたときに,

---

* データベースに格納されているバッグを実バッグと呼ぶ．実リレーションも同じ．

† バッグ直積ビューとはバッグ意味論の下での直積ビューという意味である．

その３つ組みに対して図１の可換図式が成立するか否かを問うて良い．

しかしながら，従来のアプローチでは，いずれもがスキーマレベルの可換性を問うている．たとえば，従来のアプローチでは「結合ビューは削除不可能である」とされているが，その意味は，いかなるデータベースの状態でも，ビュー定義が結合ビューであり，更新操作が削除操作であれば，それは受け付けないという意味である．これは図１の可換図式が成り立たないような,あるデータベースの状態,ある結合ビューの定義，そしてある削除操作の組合せの存在を示せるので，そうだと結論するからである（**反例の提示**）．

その理由としては，このようにスキーマレベルでビューの更新可能性を決定しておくと，リレーショナル DBMS はビューの更新可能性を管理しやすいからと考えられる（システムカタログで管理可能）．SQL では SQL-92 で初めて SQL で定義されるビューの更新可能性が規格化され，SQL:1999 で改訂が行われているが，いずれの場合にも，更新できるビューには強い制約が課せられているのは，スキーマレベルの考え方に立脚しているからである．

しかしながら，意図に基づくアプローチは，定義 2 に忠実に，インスタンスレベルのビュー更新可能性に立ち戻ってビュー更新問題を論じている．その結果，従来のアプローチや SQL では更新不可とされた直積ビューや結合ビューが状況によって更新可能となるのである．それを次に要約する．

2.3 意図に基づくアプローチ

我々の考案した「意図に基づくアプローチ」[5,6]は図 1 の可換図式の成立を定義 2 に忠実にインスタンスレベルで問う（**正例の提示**）．そのために，更新対象となったビューを“一時的にマテリアライズ（materialize, 体現化）する”ことが必要となる．そのために，パフォーマンスが劣化することが危惧されるが，この代償のもとに従来不可とされてきた直積ビューや結合ビューが状況によっては更新可能となる．このアプローチは SQL が立脚するバッグ意味論の下でも成立する[7,10]. この新規アプローチの下で, バッグ直積ビューが更新可能と判断される一例を示す．なお，ここで，挿入操作 $\boldsymbol{i}$ の考えられるすべての変換候補が列挙されて $\boldsymbol{i}$ の変換可能性が検証されているが，これが「暴力法」(brute force)である．

【例題 2】（更新可能なバッグ直積ビューと更新操作の一例）

実バッグ R(A)={1(2), 2}，S(B)={1, 2(2)}，バッグ直積ビューV を例題 1 の通りとする．V をマテリアライズすると，次の通りである．

$V^m = R \times_{bag} S = \{(1, 1)(2), (1, 2)(4), (2, 1), (2, 2)(2)\}$

このとき，V に対して次の挿入操作が発せられたとする．

$\boldsymbol{i}$= INSERT INTO V VALUES (3, 1), (3, 2)(2)‡

まず, 従来のアプローチや SQL では直積ビューへの挿入操作は受け付けられない．しかしながら，以下に示すように，意図に基づくアプローチではこの挿入操作は受け付けられる．

すなわち，挿入操作 $\boldsymbol{i}$ は VALUES 句に記されている通り, バッグ{(3, 1), (3, 2)(2)}を V に挿入したいという意味である．そうすると，暴力法の下では，$\boldsymbol{i}$ の可能な更新変換候補は，次に示す R への 4 つの挿入候補と S への 6 つの挿入候補のすべての組合せとなる．

$\boldsymbol{i}_{R1}$=no insert request
$\boldsymbol{i}_{R2}$=insert into R values (3)
$\boldsymbol{i}_{R3}$=insert into R values (3)(2)
$\boldsymbol{i}_{R4}$=insert into R values (3)(3)

一方，S に対しては次の 6 つの代替案を考えないといけない．

$\boldsymbol{i}_{S1}$=no insert request
$\boldsymbol{i}_{S2}$=insert into S values (1)
$\boldsymbol{i}_{S3}$=insert into S values (2)
$\boldsymbol{i}_{S4}$=insert into S values (2)(2)
$\boldsymbol{i}_{S5}$=insert into S values (1), (2)
$\boldsymbol{i}_{S6}$=insert into S values (1), (2)(2)

暴力法では基本的に 24（=4×6）個の組合せについて，ビューV をマテリアライズして，意図に基づくアプローチのもと実現可能性を検証することとなる．その結果，$\boldsymbol{i}_{R2}$ と $\boldsymbol{i}_{S1}$ の組合せだけがただ一つ副作用を引き起こすことなく，丁度 $\boldsymbol{i}$ を実現することが分かる．そして，一般に意図に基づくアプローチのもとで，バッグ直積ビューへの挿入操作は変換可能であると結論される．

3 バッグ直積ビューの更新可能性と非線形連立方程式

3.1 ビュー更新問題と非線形連立方程式

さて，暴力法は所望のビュー更新を実現してくれるかもしれない変換候補が見つかるか否かを検証するために，文字通りあらゆる代替案を列挙することが必要であった．そこで，この煩雑さを回避でき，より直截にビューへの更新操作の諾否を判定できる手法がないものか，模索することとなった．その結果，意図に基づくアプローチでは，ビューをマテリアライズしてビューへの更新要求の変換可能性を判定するのであるから，そのビューを定義している実バッグの行の**重複度**を変数として捉えて，そのビュー定義のもとでビュー更

‡ SQL に忠実に書けば，INSERT INTO V VALUES (3, 1), (3, 2), (3, 2)である．

新操作を満たし得る重複度の組は存在するのか，しないのかをダイレクトに問えばよいのではないか，という発想に至った．

実際にこの着想をバッグ直積ビューに対して定式化してみると，ビューの更新可能性は行の重複度を変数とする非線形連立方程式が解を有するか否かという問題に帰着されることを明らかにすることができた[11,13]．以下，それを具体的に要約する．

なお，本稿では，直積ビューの更新可能性に議論の焦点を当てているが，一般に“θ-結合演算”は直積演算を基に定義されるからである．例えば，バッグ R(A, B)とバッグ S(C, D)の B と C 上のθ-結合 $R[B\theta C]S = \sigma_{B\theta C}(R\times S)$，ここにσは選択演算，と定義され，θ-結合 R[BθC]S の更新可能性は，最終的に直積 R×S の更新可能性に帰着されるからである．

3.2 バッグ直積ビューの削除可能性と非線形連立方程式

意図に基づくアプローチのもと，バッグ直積ビューへの削除操作が発行されたとする．この操作が異状をきたすことなく実表に変換可能なのかを次のように判定する．例で示す．

【例題 3】実バッグ R(A)={1(2), 2}, S(B)={1, 2(2)}, バッグ直積ビューV を例題 1 の通りとする．このとき，V に対して次の削除操作 ***d1*** が発せられたとする．

d1: DELETE FROM V
　　WHERE A=2

●判定のための非線形連立方程式の導出

① V をマテリアライズする．その結果，$V^m = R\times_{bag}S$={(1, 1)(2), (1, 2)(4), (2, 1)(1), (2, 2)(2)}を得る．

② ***d1*** を変形して ***d1^m*** とし，それを実行する．

d1^m: DELETE FROM V^m
　　WHERE A=2

その結果，***d1^m*** は V^m から{(2, 1)(1), (2, 2)(2)}を削除したいという操作であることが分かる．そして，$W=V^m-_{bag}${(2, 1), (2, 2)(2)}= {(1, 1)(2), (1, 2)(4)}とする．

③ R(A)={1(2), 2}, S(B)={1, 2(2)}であるので，行の重複度を表す変数（＝未知数）x1, x2, x3, x4 を導入して，バッグ R^{d1}={1(x1), 2(x2)}, S^{d1}={1(x3), 2(x4)}を定義する．ここに，変数 x1, x2, x3, x4 は次を満たす．

$0\leqq x1\leqq 2,\ 0\leqq x2\leqq 1,\ 0\leqq x3\leqq 1,\ 0\leqq x4\leqq 2$

④ $R^{d1}\times_{bag}S^{d1}$ を計算する．その結果，バッグ直積演算結果の行の重複度計算の定義により次を得る．

$R^{d1}\times_{bag}S^{d1}$={(1, 1)(x1×x3), (1, 2)(x1×x4),
(2, 1)(x2×x3), (2, 2)(x2×x4)}

⑤ もし，V に対する ***d1*** が受理されるのであれば，$W= R^{d1}\times_{bag}S^{d1}$ が成立する x1, x2, x3, x4 が存在するはずである．そのためには x1, x2, x3, x4 は次の非線形連立方程式を満たさなければならない．

$0\leqq x1\leqq 2,\ 0\leqq x2\leqq 1,\ 0\leqq x3\leqq 1,\ 0\leqq x4\leqq 2$・・・(0)
$x1\times x3=2$　・・・(1)
$x1\times x4=4$　・・・(2)
$x2\times x3=0$　・・・(3)
$x2\times x4=0$　・・・(4)

つまり，ビューV への更新操作 ***d1*** の変換可能性は上記の非線形連立方程式が解けるか否かに帰着されたことになる．

3.3 整合ラベリング問題のための併合法を用いたバッグ直積ビューの更新変換可能性判定

3.3.1 整合ラベリング問題

非線形連立方程式を解く方法は多数知られているが，上述の問題は人工知能の研究分野で知られている整合ラベリング問題(consistent labeling problem, CLP)に帰着させることができる．一般に CLP は NP 完全な意思決定問題である[14]が，本稿では CLP の解法として提案されている併合法(merge method)[15]を用いて非線形連立方程式を解くことにする．併合法に着目したのは，それが非線形連立方程式の全解を求める手法であるからである．部分解ではなく全解を求めるという性質は我々の場合には必要である．なぜならば，求めた解がただ一つである場合に及びそのときのみバッグ直積ビューへの更新操作は変換可能であり，もし解がなければ変換不可能であると結論できるからである．

さて，前節で与えた(0)～(4)で与えられた非線形連立方程式を併合法で解いてみよう．まず，CLP を定義する．

【定義 3】(CLP の定義)

CLP は 4 項組(U, L, T, R)である．ここに，U，L，T，R はそれぞれ，ユニットの集合，ラベルの集合，ラベル間の制約の集合，そして局所解(local solution)の集合である．

例題 3 に照らし合わせれば，これらは具体的に次のようである：

U = {x1, x2, x3, x4}

L = {0, 1, 2}

T = {t1, t2, t3, t4}，ここに t1= (x1, x3), t2=(x1, x4), t3=(x2, x3), t4=(x2, x4)を表す．

R = {R_1, R_2, R_3, R_4}，ここに R_1, R_2, R_3, R_4 はそれぞれ次の通りである：

R_1 = {(1, 2), (2, 1)}

R_2 = {(2, 2)}

R_3 = {(0, 0), (0, 1), (1, 0)}

R_4 = {(0, 0), (0, 1), (0, 2), (1, 0)}

各 R_i の元の数を R_i のサイズといい，(t_i, R_i)の対を制約対という．V_i = (t_i, R_i)を頂点という．

3.3.2 制約ネットワークの導入

CLP として定式化された問題を併合法で解くために，制約ネットワーク(constraint network)を導入する．上で示した例題 3 の U，L，T，R を具体例として，制約ネットワークを描くと図 2 に示す通りとなる．

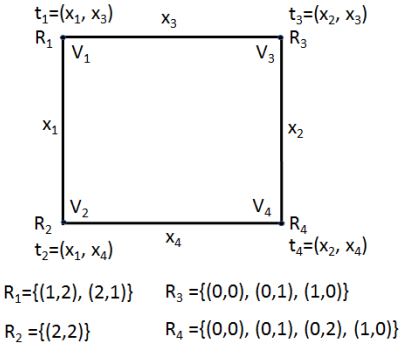

図 2　例題 3 に等価な制約ネットワーク

3.3.3 併合操作の遂行

併合法のストラテジに基づき，図 2 に示された制約ネットワークの頂点を“併合”していく．この場合，4 通りの併合が可能である(コスト最小選択は 3.3.5 節)：

① V_1 と V_3 を x3 を介して併合する
② V_1 と V_2 を x1 を介して併合する
③ V_3 と V_4 を x2 を介して併合する
④ V_2 と V_4 を x4 を介して併合する

例えば，①で図 2 のネットワークに併合操作を適用すると，図 3 のように変形される．

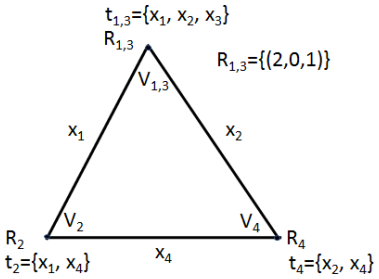

図 3　図 2 の制約ネットワークの併合結果

更に併合が可能で，例えば，図 3 の頂点 $V_{1,3}$ と V_2 を x1 を介して併合すると，図 4 を得る．

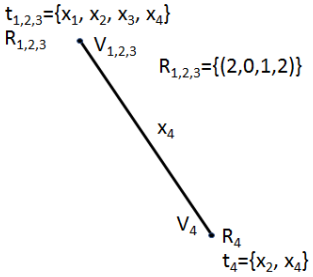

図 4　図 3 の制約ネットワークの併合結果

最終的に，$V_{1,2,3}$ と V_4 が x4 を介して併合可能で，その結果，図 5 を得る．

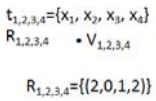

図 5　図 4 の制約ネットワークの併合結果

3.3.4 併合結果の解釈―変換可能性―

併合法を適用することにより，この例題では，図 2 に示された 4 頂点のグラフは単一頂点 $V_{1,2,3,4}$ のグラフに縮退し，$R_{1,2,3,4}$={(2, 0, 1, 2)}となった．

まず，着目するべき点は，$R_{1,2,3,4}$ がただ一つの元から成り立ってることである．これは例題 3 で示したビューV = R×S に対する削除操作 ***d1*** が変換可能であるということである．

では，具体的にビューV に対する削除操作 ***d1*** は実バッグ R と S へのどのような更新操作に変換されるのであろうか，それを算出する方法を次に示す．

(a) (2, 0, 1, 2)は x1=2, x2=0, x3=1, x4=2 を意味する．
(b) 一方，x1, x2, x3, x4 の初期値は x1=2, x2=1, x3=1, x4=2 であった．
(c) 両者を比較すると，x2 の値が初期値 1 から 0 に変化していることが分かる．
(d) ところで，x2 はテーブル R(A)の値が 2 の行数を表していた．つまり，最初は R に値が 2 の行が 1 本あったのであるが，***d1*** を実現するためにはそれを削除して 0 本にすればよいことをいっている．
(e) したがって，***d1*** は次の操作に変換可能である：

T(***d1***): DELETE FROM R
WHERE A=2

実際，T(***d1***)を R に適用して，その結果と S を CROSS JOIN すれば，所望の結果{(1, 1)(2), (1, 2)(4)}を得ることが確認できる．

3.3.5 併合法のプログラム概略

併合法のプログラムの概略を示す．ここに，Vi は制約対を，min($k_{i,j}$)は V_i-V_j という併合の対象となった頂点対を推定サイズが最小で選択する関数である．この関数を計算することによって，この例題では，3.3.3 節冒

頭で示した 4 つの併合の可能性のどれから始めると併合法の計算コストが小さくなるかという判定に使われる．min($k_{i,j}$)を計算するにあたっては，貪欲法(greedy method)を適用すると，併合法のプログラムの概略は次のようになる．

```
begin
    while (two or more vertices remain)
        merge (Vi, Vj) for min(ki,j)
    compute and output all solutions
end
```

4 併合法の SQL による実現

4.1 制約のテーブル表現

バッグ直積ビューへの更新操作の変換可能性が非線形連立方程式の解の存在に帰着され，それは整合ラベリング問題の一種で，そのための解法の一つとして知られている併合法を用いると，それは上記のように解けることを示した．

ところで，併合法の行っているグラフの併合操作はリレーショナルデータベースでよく知られている“自然結合演算”に該当する操作である．このことを，つぶさに見ていく．そうすることによって，結合ビューの更新可能性の判定アルゴリズムをリレーショナル代数演算の世界で実現できるということになる．ここでは，これまでと同じ，例題 3 で示した，一般にはバッグ意味論の下で定義されるテーブル R(A)={1(2), 2}と S(B)={1, 2(2)}の CROSS JOIN ビューV(A, B) = R×S に削除操作 ***d1*** が発行されたとしてそのことを見ていく．

まず，変数間の制約は次の通りであった：

$x1 \times x3 = 2$ ・・・(1)
$x1 \times x4 = 4$ ・・・(2)
$x2 \times x3 = 0$ ・・・(3)
$x2 \times x4 = 0$ ・・・(4)

このとき，各変数が採り得る値の制約は次の通りであった：

$0 \leqq x1 \leqq 2,\ 0 \leqq x2 \leqq 1,\ 0 \leqq x3 \leqq 1,\ 0 \leqq x4 \leqq 2$ ・・・(0)

したがって，各制約 R_1, R_2, R_3, R_4 の局所解は次の通りであった：

$R_1 = \{(1, 2), (2, 1)\}$
$R_2 = \{(2, 2)\}$
$R_3 = \{(0, 0), (0, 1), (1, 0)\}$
$R_4 = \{(0, 0), (0, 1), (0, 2), (1, 0)\}$

そこで，R_1, R_2, R_3, R_4 をテーブル表現することを考える．それらの結果を図 6 に示す．

R_1

x1	x3
1	2
2	1

R_2

x1	x4
2	2

R_3

x2	x3
0	0
0	1
1	0

R_4

x2	x4
0	0
0	1
0	2
1	0

図 6 例題 3 の制約のテーブル表現

4.2 併合のために自然結合演算の適用

先に，図 2 に例題 3 に等価な制約ネットワークを示し，頂点 V_1 と V_3 へ併合して，頂点 $V_{1,3}$ を得ている．このとき行われた操作は，t_1= (x1, x3), t_3=(x2, x3)であり，$R_1 = \{(1, 2), (2, 1)\}$，$R_3 = \{(0, 0), (0, 1), (1, 0)\}$であるから，x3 を介して両者が併合されるとすれば，(2, 1)∈R_1と(0, 1)∈R_2の組合せのみが可能で，その結果，x1=2, x2=0, x3=1, つまり $R_{1,3}=\{(2, 0, 1)\}$となったのであった．

しかるに，この演算はテーブル R_1 と R_3 の自然結合演算と同じである．

$R_{1,3}=R_1*R_3$

x1	x2	x3
2	0	1

この結果が，図 2 から図 3 を得ることに相当している．続けて，図 3 から図 4 を得る操作は次の自然結合演算を行うことに等しい．

$R_{1,2,3}=R_{1,3}*R_2$

x1	x2	x3	x4
2	0	1	2

さらに，図 4 から図 5 を得る操作は次の自然結合演算を行うことに等しい．

$R_{1,2,3,4}=R_{1,2,3}*R_4$

x1	x2	x3	x4
2	0	1	2

上記の議論をまとめる．例題 3 の場合を例にとると，次のようになる．

【命題 1】意図に基づくアプローチの下で，バッグ直積ビューV への削除操作 ***d1*** が変換可能か否かは，R_1, R_2, R_3, R_4 の自然結合演算の結果 $R_1*R_2*\ R_3*\ R_4$ が単集合(singleton)になっているとき及びそのときのみ可能である．

4.3 併合法を用いた削除可能性，挿入可能性の判定例

ここまで一貫して例題 3 で示したバッグ直積ビューV(A, B)=R(A)×S(B)への削除要求 ***d1*** の変換可能性を論じ，それが更新意図の外形的推測アプローチの下で，変換可能であることを，併合法，並びにその自然結合

演算処理に基づく方法で示してきた.

以下，削除操作の変換可能性を補うと共に，併合法の下でと挿入操作の変換可能性がどのように扱われるのか示す.

4.3.1 削除操作の変換―補遺―

【例題 4】実バッグ R(A)と S(B)，およびバッグ直積ビューV は例題 3 で与えた通りとする．このとき，V に対して次の削除操作が発せられたとする.

d2: DELETE FROM V
WHERE A=2 AND B=2;

d2 が変換可能か否か併合法で判定する．まず，判定のための非線形連立方程式は次の通りである.

$0 \leqq x1 \leqq 2,\ 0 \leqq x2 \leqq 1,\ 0 \leqq x3 \leqq 1,\ 0 \leqq x4 \leqq 2$ ・・・(0)

$x1 \times x3 = 2$ ・・・(1)

$x1 \times x4 = 4$ ・・・(2)

$x2 \times x3 = 1$ ・・・(3)

$x2 \times x4 = 0$ ・・・(4)

このとき，制約のテーブル表現は次の通りとなる.

R_1

x1	x3
1	2
2	1

R_2

x1	x4
2	2

R_3

x2	x3
1	1

R_4

x2	x4
0	0
0	1
0	2
1	0

図 7 例題 4 の制約のテーブル表現

すると，$R_1*R_2*R_3*R_4=\Phi$(空)となり，この連立方程式に解はない．つまり，V に対する削除操作 ***d2*** は変換不可能である．つまり，受理できない.

【例題 5】実バッグ R(A)と S(B)，およびバッグ直積ビューV は例題 3 で与えた通りとする．このとき，V に対して次の削除操作が発せられたとする.

d3: DELETE FROM V
WHERE A=1 OR (A=2 AND B=2);

判定のための非線形連立方程式は次の通りである.

$0 \leqq x1 \leqq 2,\ 0 \leqq x2 \leqq 1,\ 0 \leqq x3 \leqq 1,\ 0 \leqq x4 \leqq 2$ ・・・(0)

$x1 \times x3 = 0$ ・・・(1)

$x1 \times x4 = 0$ ・・・(2)

$x2 \times x3 = 1$ ・・・(3)

$x2 \times x4 = 0$ ・・・(4)

すると，この例題の制約のテーブル表現は次の通りとなる.

R_1

x1	x3
0	0
0	1
0	0
1	0
2	0

R_2

x1	x4
0	0
0	1
0	2
1	0
2	0

R_3

x2	x3
1	1

R_4

x2	x4
0	0
0	1
0	2
1	0

図 8 例題 5 の制約のテーブル表現

$R_1*R_2*R_3*R_4=\{(0, 1, 1, 0)\}$．つまり，***d3*** を次に示す R と S に対する削除要求に変換すれば所望の更新を実現できることが示された.

***d3**R*：DELETE FROM R
WHERE A=1

***d3**S*：DELETE FROM S
WHERE B=2

4.3.2 挿入操作の変換例

【例題 6】実バッグ R(A)と S(B)，およびバッグ直積ビューV は例題 3 で与えた通りとする．このとき，V に対して次の挿入操作が発行されたとする.

i：INSERT INTO V(A, B)
VALUES (3, 1) , (3, 2)(2)

判定のための非線形連立方程式は次の通りとなる．挿入を実現するためには，テーブル R に値が 3 の行が何本か挿入されねばならない．そこで，***i*** を実現するために R と S がそれぞれ，$R^i=\{1(x1), 2(x2), 3(x3)\}$, $S^i=\{1(x4), 2(x5)\}$となったとする．そうすると，$R^i \times_{bag} S^i=\{(1, 1)(x1 \times x4), (1, 2)(x1 \times x5), (2, 1)(x2 \times x4), (2, 2)(x2 \times x5), (3, 1)(x3 \times x4), (3, 2)(x3 \times x5)\}$である.

もし，V に対する ***i*** が受理されるのであれば，$V \cup_{bag} \{(3, 1) , (3, 2)(2)\} = R^i \times_{bag} S^i$が成立する x1, x2, x3, x4, x5 が存在するはずであるから，次が成立しなければならない.

$x1 \times x4 = 2$ ・・・(1)

$x1 \times x5 = 4$ ・・・(2)

$x2 \times x4 = 1$ ・・・(3)

$x2 \times x5 = 2$ ・・・(4)

x3×x4=1　・・・(5)

x3×x5=2　・・・(6)

一方，今回は削除操作ではなく，挿入操作の変換可能性を考えている．従って，次が成りたたなくてはいけない．ここに，x4, x5 の上限が 2 と 4 に設定されていることに注意する．

x1=2, x2=1, 1≦x3≦3, 1≦x4≦2, 2≦x5≦4　・・・(0)

すると，この例題の制約のテーブル表現は次の通りとなる．

R_1

x1	x4
2	1

R_2

x1	x5
2	2

R_3

x2	x4
1	1

R_4

x2	x5
1	2

R_5

x3	x4
1	1

R_6

x3	x5
1	2

図 9 例題 6 の制約のテーブル表現

R_1*R_2*R_3*R_4*R_5*R_6 = {(2, 1, 1, 1, 2)}となり，$\boldsymbol{i}$ を次に示す挿入要求に変換すれば所望の更新を実現できることが保証される．

$\boldsymbol{i}^R$：INSERT INTO R(A)
VALUES 3

5 おわりに

筆者らは，意図に基づくアプローチの下で，バッグ意味論の下での直積ビューの更新問題が「非線形連立方程式が解を有するか否か」という問題に帰着できることを示した[11]．本稿はその結果を受けて，その問題が整合ラベリング問題(CLP)のための併合法(merge method)を用いて解けることを示すと共に，併合法の特性に着目すると，バッグ意味論の下での直積ビューの更新判定アルゴリズムがリレーショナルデータモデルの自然結合演算で表現できることを示した．直積ビューの更新可能性は一般に結合ビューの更新可能性の根幹をなすものであり，本稿で示しえた結果は，今後，意図に基づくアプローチをリレーショナルDBMS上で暴力法に代わりよりスマートに実装するための指針を与えるものである．今後，これらの結果にさらに検討を加えて，極限にまで更新可能なビューサポートメカニズムの実装に取り組み，リレーショナルデータベースユーザに資することとする．

【謝辞】 本研究は JSPS 科研費 JP16K00152 の助成を受けたものである．

参 考 文 献

[1] E. F. Codd, “Recent Investigations in a Relational Database System,” Information Processing 74, pp. 1017-1021, North-Holland, 1974.

[2] U. Dayal and P. Bernstein, “On the Updatability of Relational Views,” Proc. 4th VLDB, pp.368-377, 1978.

[3] Y. Masunaga, “A Relational Database View Update Translation Mechanism,” Proc. 10th VLDB, pp.309-320, 1984.

[4] A. Sheth, J. Larson and E. Watkins, “TAILOR, A Tool for Updating Views,” LNCS, Vol. 303, pp.190-213, Springer, 1988.

[5] 増永良文, “更新意図の外形的推測に基づくリレーショナルデータベースビューの更新可能性,” 8p., WebDB Forum 2015 会議録, 2015 年 11 月.

[6] Y. Masunaga, “An Intention-based Approach to the Updatability of Views in Relational Databases,” Proceedings of the 11th International Conference on Ubiquitous Information Management and Communication (ACM IMCOM 2017), P5-8, Beppu, Japan, January 5-7, 2017.

[7] 増永 良文, 長田 悠吾, 石井 達夫, “バッグ意味論のもとでのビュー更新問題の検討－更新意図の外形的推測に基づくアプローチの適用可能性－,” DEIM Forum 2017 会議録, 3-5, 2017 年 3 月.

[8] Y. Nagata and Y. Masunaga, “Extending View Updatability by a Novel Theory -Prototype Implementation on PostgreSQL-,” PGCon 2017-The PostgreSQL Conference, Ottawa, Canada, May 25-26, 2017.

[9] 長田悠吾, 石井達夫, 増永良文, “意図に基づくアプローチによる更新可能ビューの拡張－ PostgreSQL におけるプロトタイピング －,” WebDB Forum 2017 会議録, DBS-165, No.14, 2017 年 9 月.

[10] Y. Masunaga, Y. Nagata and T. Ishii, “Extending the View Updatability of Relational Databases from Set Semantics to Bag Semantics and Its Implementation on PostgreSQL,” Proceedings of the 12th International Conference on Ubiquitous Information Management and Communication (ACM IMCOM 2018), 8-3, Langkawi, Malaysia, January 5-7, 2018.

[11] 増永 良文, 長田 悠吾, 石井 達夫, “意図に基づくアプローチのもとでの SQL ビューの更新可能性－非線形連立方程式問題への帰着－,” DEIM Forum 2018 会議録, G1-3, 2018 年 3 月.

[12] 長田悠吾, 増永良文, 石井達夫, “意図に基づくアプローチのもとでの SQL ビューの更新可能性－ビュー更新判定問題の定式化と実問題への適用に向けた計算量削減の試み－,” DEIM Forum 2018 会議録, G1-4, 2018 年 3 月.

[13] Y. Masunaga, Y. Nagata and T. Ishii, “Making Join Views Updatable on Relational Database Systems in Theory and in Practice,” Proceedings of the 13th International Conference on Ubiquitous Information Management and Communication (IMCOM 2019), 9-1, Phuket, Thailand, January 4-6, Phuket, Thailand, 2019.

[14] Tsang, E., “Foundations of Constraint Satisfaction,” Herstellung und Verlag: BoD (1996).

[15] 西原清一, “整合ラベリング問題と応用,” 情報処理, Vol.31, No.4, pp.500-507 (1990).