3G−6　アドホックな問合せにおけるグラフ統合演算の有効性

宝珍 輝尚

NTT情報通信網研究所

1　はじめに

データグラフ [3] は，グラフを直接利用するデータモデル [1,2等] の一つである．データグラフの演算は，他のグラフ演算 [1,2等] と同様に，集合を扱う演算であり，アドホックな問合せでの使用も念頭に置いている．統合演算は，宣言的に複数のグラフをマージするデータグラフ演算である．他のグラフ演算 [1,2等] には同等の演算は含まれておらず，統合演算がどの程度グラフ演算として必要か明確ではない．

そこで，本論文では，グラフ演算としての統合演算の必要性を示すことを目標とし，Welty と Stemple の手続き度 [4] を使用して，アドホック問合せの際の統合演算の有効性を評価する．

2　グラフ統合演算

2.1　データグラフ

ここでは拡張を施したデータグラフ [3] について述べる．

データグラフはデータを表現するラベル付き有向グラフである．ラベルは3つ組（d_{ID}, N, d）で表わされる．ここで，d_{ID} は一意識別子，Nは要素名，dはデータである．点および枝にはこのラベルが付与される．以降(図を含む)では，簡単化のために一意識別子 d_{ID} を省略し，「N: d」で要素を表現する．枝は始点集合と終点集合を持つ．点集合Vと枝集合Eからなる順序組（V, E）で表現される（連結または非連結な）グラフをデータ表現グラフという．さらに，データ表現グラフを成分にもつグラフをデータグラフという．データグラフの集合がデータベースである．データグラフの要素名のみをラベルとしたグラフを，そのデータグラフのスキーマグラフという．

スキーマグラフの例を図1(a)に示す．この例では2スキーマグラフ (Residence と CityInf) がある．Residence には2枝 (PF, CC) がある．枝 CC の始点は点 Address，終点は点 City と点 County である．図1(b) はデータグラフの例である．データグラフは一点鎖線で，データ表現グラフは点線で示した．データグラフ Residence には2データ表現グラフが含まれている．点 Address:a1 は点 Pref:Kanagawa, 点 County:Miura と連結している．

2.2　統合演算

統合演算は，一方のデータ表現グラフの点を他方のデータ表現グラフの点集合に加える演算である．統合結果のスキーマグラフは，2スキーマグラフの要素の和で構成されるグラフである．

評価後変化しないデータ表現グラフを統合結果に含めるか否かにより，3種類の統合演算 (制約統合，関係統合，外統合) がある．制約統合は評価後変化しないデータ表現グラフを全て残し，関係統合は評価後変化しないデータ表現グラフを全く残さない．外統合は，評価後変化しないデータ表現グラフの一方を残す．図1(b) の Residence と CityInf を制約して得られる Residence' と CityInf' に対して制約統合を施した例を図2に示す．

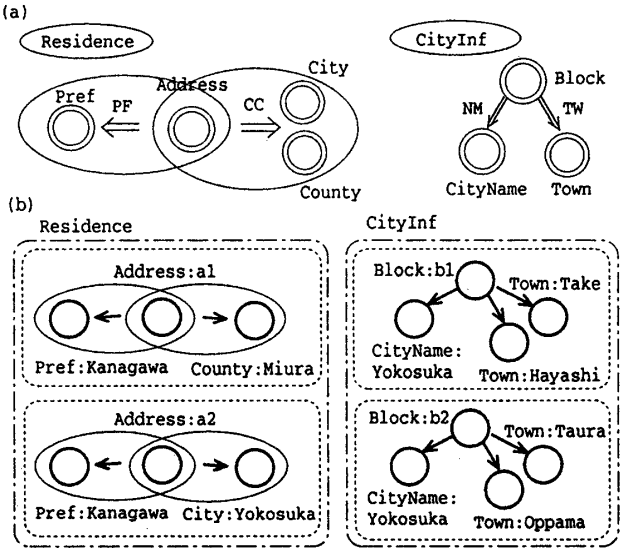

図1　スキーマグラフ(a)とデータグラフ(b)

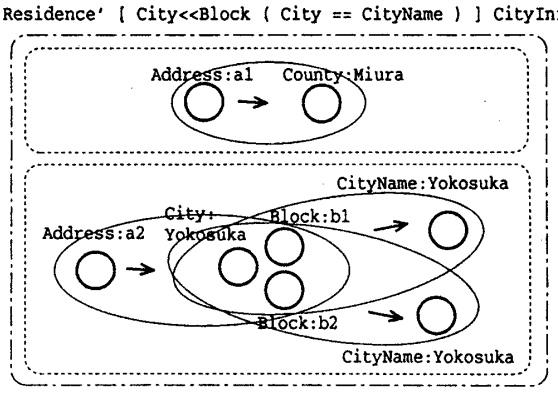

図2　統合の例 (制約統合)

Effectiveness of Graph Unite Operations in Making Ad-Hoc Queries
Teruhisa HOUCHIN
NTT Network Information Systems Laboratories

3 評価

簡単な問合せは宣言的に記述する方が手続き的に記述するよりも誤りにくいことが，Welty と Stemple によって実験的に示されている [4]．アドホックな問合せはほとんどが簡単な問合せであると考えられるので，アドホックな問合せは宣言的に記述できる方が良いことになる．そこで，統合演算のアドホック問合せとしての有効性を問合せの宣言性により評価する．

3.1 評価測度：手続き度

Welty と Stemple によって提案された手続き度は言語のステップバイステップの度合を示すための測度である．手続き度PMは下式によって求められる [4]．

$$PM = \frac{N_v}{N_{vo}} + \frac{N_o}{N_{oo}} \tag{1}$$

ここで，N_v は変数束縛の数，N_{vo} は許される変数束縛順序の数，N_o は演算の数，N_{oo} は許される演算順序の数である．許される順序とは評価される文で示された順序または言語の構文上および意味上評価される文と同じとみなされる順序である．

3.2 例による手続き度計算

統合演算を使用する場合 (Case U) と使用しない場合 (Case C) の手続き度を考える．ここでは例として，図1(b) のデータグラフから図2のデータグラフを得ることを考える．

Case U では，2制約演算 (Residence から Residence'，CityInf から CityInf') と統合演算を使用すれば実現できる．2制約演算の順序は変えられる．Residence' と CityInf' は一種の変数で，これらの順序は変えられない．従って，PM = 2/1 + 3/2 = 3.5 である．

Case C では，上記の2制約演算の後に，各枝に対して点を加える処理を行わなければならない．変数は，Residence' と CityInf' のほかに，City，Block とそれらに連結する枝 (CC, NM) を束縛する変数が必要である．これらの変数の順序は変えられない．枝 CC と枝 NM に対してマージする点を加える演算 (JN 演算) が上記の2制約演算のほかに必要であり，これらの順序は変えられる．従って，PM = 6/1 + 4/(2*2) = 7 である．

3.3 統合処理の手続き度

3.3.1 統合に関する処理の手続き度

ここでは，統合に関する処理のみの手続き度を求める．一般に，Case U では1演算，束縛変数なしで操作できる．Case C の場合，統合に関与するデータ表現グラフを束縛する変数が必要である．また，定義上の要素に対して複数の要素が存在し得るので，要素を束縛する変数が必要となる．これらの変数の順序は変えられない．統合点に連結している枝の総数を n とすると，n 個の JN 演算が必要となり，これらの順序は変えられる．これが，各統合点についてあてはまる．

以上より，Case U, Case C の PM (PM_u,PM_c) は，統合点の数を m，各統合において関与する枝の数を n_i とすると，下式で与えられる．

$$PM_u = \frac{0}{1} + \frac{1}{1} = 1 \tag{2}$$

$$PM_c = \frac{2+2m}{1} + \frac{\sum_{i=1}^{m} n_i}{(\prod_{i=1}^{m} n_i!)m!} \approx 2(1+m) \tag{3}$$

3.3.2 全体の手続き度

統合処理とそれ以前の処理全体の手続き度を考える．Case U, Case C の PM (PM_{ut},PM_{ct}) は，統合処理以前の処理の手続き度を $PM_p = n_v/n_{vo} + n_o/n_{oo}$ とすると，下式で与えられる．ただし，統合対象のデータグラフは統合処理以前の処理の中の変数で束縛されていると仮定している．

$$PM_{ut} = \frac{n_v}{n_{vo}} + \frac{n_o+1}{n_{oo}} = PM_p + \frac{1}{n_{oo}} \tag{4}$$

$$PM_{ct} = \frac{n_v+2m}{n_{vo}} + \frac{n_o+\sum_{i=1}^{m} n_i}{n_{oo}(\prod_{i=1}^{m} n_i!)m!} \tag{5}$$

3.4 考察

提案した統合演算を使用すると，統合点の数に関係なく手続き度を一定にできる．これは，アドホックな問合せ記述において望ましい特性と考えられる．また，統合点の数が多い場合には，統合演算を使用すると手続き度を低くでき，有利であることが分る．これは，演算数ではなく変数束縛数の影響が大きい．

ここでは制約統合を考えた．関係統合や外統合では順序の変えられない演算が必要なので，これらの PM_c は制約統合時の PM_c よりも高くなる．従って，統合演算を使用する方が有利である．

4 おわりに

データを表現する複数のグラフをマージする統合演算の，アドホックな問合せ記述における有効性を示した．

今後は，グラフの分離を行うグラフ演算の検討，データグラフの表現能力の評価や実装が課題である．

参考文献

[1] Gyssens M., et al: A Graph-Oriented Object Database Model, Proc. of 9th ACM PODS, pp.417-424 (1990).

[2] Guo M., et al: An Association Algebra For processing Object-Oriented Databases, Proc. of Int'l Conf. on Data Engineering, pp.23-32 (1991).

[3] 宝珍：拡張可能 DBMS:COMMON の格納構造と基本演算について，情処学データベース・システム研報 82-6 (1991).

[4] Welty C. and Stemple D. W. : Human Factors Comparison of a Procedural and a Nonprocedural Query Language, ACM TODS, Vol. 6, No. 4, pp.626-649 (1981).