

TFE 処理系におけるマルチセッション高速化

2D-6

実政宏幸[†] 遠山元道[‡][†] 慶應義塾大学 理工学研究科 管理工学専攻[‡] 慶應義塾大学 理工学部 管理工学科

1 はじめに

従来、RDB からの出力はフラットな二次元の表であり、表現力の面では貧弱さを拭い得なかった。RDB からのそのような出力に対して、三次元の構造化レイアウトを行なうことにより表現力を向上させたのが TFE (Target Form Expression) である。

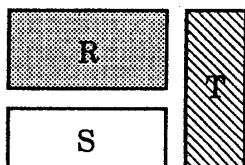
本研究ではこの TFE 処理系をより実用的なものとするため、TFE 質問文特有の質問形態に注目し、1つの質問文を複数の質問文に分割して処理するマルチセッション実行を提案し、それによって処理系全体の高速化を図った。

2 従来の TFE 処理系の問題点

従来の TFE では表現力の豊かさゆえ、RDB に対して一般には記述しないような問い合わせ文を記述することが多々ある。

例えば、全く独立な、タプル数 l の関係 R とタプル数 m の関係 S 、タプル数 n の関係 T を以下のようなレイアウトで出力したいとしよう。その時の TFE 質問文は以下のような形をとる：

- レイアウト



- TFE 質問文

```
GENERATE <媒体指定> {R.*!S.*},T.*
<媒体指定> ::= <空> | plain | latex |
tcltk | HTML
```

ここで、従来の TFE 処理系では、この TFE 質問文が次のような中間質問文に変換されて RDB へ渡される：

- 中間質問文 1

```
SELECT R.*,S.*,T.*
```

RDB から得られるこの中間質問文の結果 (中間形) は、属性集合間に多値従属性が成立しており、 $R.*, S.*, T.*$ の直積として表される。その大きさは

$$l \times m \times n$$

という積の形で表すことができる。このように、関係 R, S, T の個々は小さなものであったとしても、中間形としては非常に大きな表が生成されるケースは珍しくない。

この巨大な中間形は、RDB からのデータの読み出し、及び TFE 処理系内部で表の構造化を行なうのに必要とされる時間的コストを増加させるだけでなく、中間形を一旦ファイルとして保存しておかなければならないことから、そのための領域的なコストも増加させる。

ゆえに、TFE をより実用的にするためには、上記のコストを削減することによる処理系自体の効率化を行ない、高速化を図る拡張が必要とされる。

3 マルチセッション実行

先の例のような直積 (あるいは、場合によっては部分的な直積) が生じるのを防ぐためには、多値従属性が成立する属性集合同士が同一の中間質問文内に並べられることを抑制せねばならない。そのためには、1つの TFE 質問文を複数の中間質問文に変換し、RDB から属性集合 $R.*, S.*, T.*$ のそれぞれを別々に読み出してやり、得られた複数の中間形を加工して本来ユーザが要求している構造を作り上げていく、という一連のプロセスが必要となる。つまり、先に示した TFE 質問文は次のような複数の純粹クエリーとして処理されることになる：

- 中間質問文 2

```
SELECT R.*
SELECT S.*
SELECT T.*
```

従来の TFE 処理系がユーザからの1つの問い合わせを、1つの質問文として RDB に渡していたのに対して、この処理法は1つの問い合わせを複数の質問文に分割して RDB に渡すものである。ゆえに、従来をシングルセッション実行とすれば、この処理法は「マルチセッション」実行と呼ぶことができる。

この処理法を用いた場合、中間形として得られる表の大きさは

$$l + m + n$$

という形で表すことができ、たとえ l, m, n の各要素のサイズが大きな場合であっても、中間形の大きさ自体は高々個々の集合の大きさの和に抑えることが可能となる。

つまり、マルチセッション実行は得られる中間形の大きさを従来よりも小さなサイズに抑えることにより、RDB からの読み出しとデータの構造化に必要とされる処理時間を削減するものであり、最終的には処理系の高速化に結び付く。

4 評価・検討

従来の処理系とマルチセッション実行を行なう処理系とで、処理時間にどの程度の差が生じるか、最も単純なパターンである 2 つの集合の直積が生じる場合と、TFE 特有のグルーピングを用いた場合の 2 つのケースで実験を行ない、その結果をまとめたのが次の表 1, 2 である。ここで、指定される属性集合のデータタイプは char 30 であり、Sparc Station ELC において測定を行なった。

表 1: 処理時間の比較 (グルーピング無)

タプル数	10×10	50×100	50×200	50×300
改善前	8.35	68.35	128.35	188.35
改善後	10.03	19.13	28.23	37.33

(単位: 秒)

表 2: 処理時間の比較 (グルーピング有)

* グルーピング要素のタプル数: 25 *

タプル数	5×10	5×50	5×100	5×150
改善前	172.22	1,014.20	1,983.25	3,103.5
改善後	14.66	20.47	28.82	37.27

(単位: 秒)

ちなみに、グルーピングのある例としては、あるデパートの情報をもつ RDB から売場名の横に従業員名の一覧と商品名の一覧をレイアウトしたい、などが挙げられる。その時の TFE 質問文は以下のように記述できる:

● TFE 質問文

```
GENERATE [floor, [employee]!, [item]! ]%
```

グルーピングが存在する場合、例えばこのデパートの例では、マルチセッション実行として「売場名と従業員名」、「売場名と商品名」をジョインしたものを別々に取り出し、それぞれを構造化してから売場名を基準に合成していく方法と、まず「売場名」を取り出しておき、そのデータを利用して各売場名ごとに「従業員名」と「商品名」それぞれを取り出し、それら全てを合成する方法とが考えられる。先の表 2 の結果は後者によるものであり、先の売場名がグルーピング要素に相当する。

さて、表 1 で示されていることだが、処理時間の優劣がタプル数によって入れ替わっていることに注意して欲

しい。つまり、得られる出力が極端に小さなケースではマルチセッション実行が本来の目的である高速化として機能しておらず、シングルセッション実行の方が処理時間が短くなっているが、これは質問文を分割処理することによって RDB でのオーバーヘッドが従来の処理系においてよりも増大するためである。

しかし、表 1, 2 ともにタプル数が大きくなるにつれてマルチセッション実行は高速化として機能するようになり、やがてシングルセッション実行よりも圧倒的に速くなっていく。また、タプル数の増加に対して、従来の処理系での処理時間は急激に増加しているのに対して、マルチセッション実行での処理時間は緩やかに増加している。このことから、タプル数が多いほどマルチセッション実行による効果も大きくなるのが分かる。

5 おわりに

本研究では 1 つの TFE 質問文を複数の中間質問文に分割して処理するマルチセッション実行による処理系の高速化を試みると共に、実験結果から多くのケースでその有効性を確認した。

今後の課題として、現時点ではマルチセッション実行を行なう基準を、TFE における階層構造の一番上の階層で判断しているが、それ以下の階層においても分割が可能となるケースが多々考えられる。そこで、再帰的定義を利用した階層の深さを問わないアルゴリズムの実装が望まれる。

また、実験評価の欄でも述べているが、マルチセッション実行では複数に分割された質問文それぞれの処理に対してオーバーヘッドが付加することから、出力があまりに小さい時にはかえって従来の処理系より処理時間が増加してしまう。今回の拡張でも評価関数を導入してはいるが、実用という観点に立てば、より有効な評価関数の導入が必要となる。

参考文献

- [1] Toyama, M.: Three Dimensional Generalization of Target List for Simple Database Publishing and Browsing, in proc. 3rd Australian Database Conference, Research and Practical Issues in Databases, World Scientific Pub. Co.(1992), pp. 139-153
- [2] 遠山元道他: レイアウト式 TFE の拡張, 情報処理学会データベースシステム研究会資料, 95-DBS-104(1993) pp. 217-224
- [3] 増永良文: リレーショナルデータベースの基礎-データモデル編-, オーム社, pp. 70-76, 127-129, 139-149