XQuery 問い合わせプログラムの生成に関する研究

M2005MM015 水野 耕太
指導教員　野呂 昌満

1　はじめに

近年，XML 文書に対する問い合わせ言語として，XQuery[6] が注目されている．XQuery を扱うために，さまざまな XQuery 処理系が開発されている．既存の XQuery 処理系では，構文木を作成する方法や，ストリームとして処理する方法で，XML 文書を処理している．構文木を作成する XQuery 処理系には SAXON[5] がある．ストリームとして処理する XQuery 処理系には，石野らの研究 [3] がある．

既存の，構文木を作成する XQuery 処理系は，大規模な XML 文書をあつかうさいに，多量の処理時間，メモリ領域が必要となる [4]．ストリーム指向 XQuery 処理系を用いることで大規模な XML 文書を高速，かつ，少ないメモリ領域で処理できるが，結合処理のような扱えない問い合わせがある．実用性を考慮する場合，扱えない問い合わせがある処理系は十分ではない．

本研究では，効率のよい問い合わせプログラムを生成する XQuery 問い合わせプログラム生成系を設計，実現することを目的とする．前述の問題は，問い合わせ処理を木の探索問題ととらえていることに起因すると考える．すなわち，既存の XQuery 処理系では，対象の XML 文書全体に対応する抽象構文木を作成し，それを用いて問い合わせ処理をおこなっている．本研究では，問い合わせ処理を構文解析問題ととらえる．すなわち，処理の過程で，XML 文書の構文解析をおこない，処理に必要な要素 (以下，必要要素) だけからなる構文木 (以下，必要要素木) を作成し，問い合わせを処理する．

一般に，実用的な構文解析処理系の作成には直構文記述を用いるので，本研究では，問い合わせプログラムを直構文記述を用いて記述する．問い合わせプログラムは，DTD，XQuery から生成する．XML 文書を字句，構文解析するためのコードは，DTD から生成できる [2]．XQuery からは，必要要素木を作成する処理と，必要要素木を用いた問い合わせ処理を生成する．

生成した問い合わせプログラムを用いて問い合わせ処理をおこない，既存の XQuery 処理系 SAXON と比較した．SAXON と比較して，最大，約 1/5 倍の処理時間，最大，約 1/10 倍のメモリ使用量で処理できた．

2　関連研究

本研究の関連研究として，XML 文書問い合わせ言語 XQuery，XML 文書処理系の自動生成に関する研究について述べる．

2.1　XQuery

XQuery とは, W3C において定められている, XML 文書に対する問い合わせ言語である．XQuery では FLWR 式と呼ばれる形式で問い合わせを記述する．FLWR 式の記述方法を **表 1** 示す.

表 1　FLWR 式

記述	概要
for [変数] in [要素リスト]	[要素リスト] から要素をひとつずつ抽出して [変数] に代入
let [変数] := [要素リスト]	[変数] に [要素リスト] を代入
where [条件]	FOR 節に条件を付加
return [出力]	[出力] に指定された形式で出力

既存の XQuery 処理系は, FLWR 式のインタプリタに相当する．FLWR 式にしたがって XML 文書を解析し，問い合わせ結果を出力する．本研究の問い合わせプログラム生成系は，コンパイラに相当すると考える．XQuery から必要要素木を用いて問い合わせをおこなうプログラムを生成する．

2.2　xpgen

XML 文書の字句，構文解析処理を自動生成する研究として，xpgen[2] がある．xpgen では，XML 文書が文脈自由文法で定義できることに着目している．XML 文書の文法に相当する DTD から，XML 文書を字句解析，構文解析するためのコードを生成する．

xpgen における, DTD からの字句, 構文解析処理の生成について述べる．本研究では，xpgen における生成方法をもとに，XML 文書の解析処理を生成する．字句解析処理の生成では，タグと，要素のもつ値を字句としてあつかうパターンを生成する．構文解析処理の生成では，構文規則を要素ごとに生成する．要素と構文規則を対応させることで，直構文記述を用いて，要素に対する処理を記述できる．

DTD と構文規則の対応を **表 2** に示す．DTD では，連接，選択，繰り返しを用いて要素を表す．連接，選択，繰り返しは，BNF を用いて表現できる．DTD では，要素ごとに宣言をおこなう．構文規則を要素ごとに生成することで DTD と対応できる．

3　問い合わせプログラム生成系の設計，実現

問い合わせプログラム生成系を設計，実現する．問い合わせプログラム生成系を用いた問い合わせの概略を **図 1** に示す．DTD，XQuery を入力とし，問い合わせプログラムのソースコードを生成する．コンパイルした問い合わせプログラムに XML 文書を入力することで，問い合わせ結果を得る．

表 2　DTD と構文規則の対応

	DTD	構文規則
1	<!ELEMENT A #PCDATA>	<A>::=<BGN_A> <STR> <END_A> <STR>::=**任意の文字列**
2	<!ELEMENT A B>	<A>::=<BGN_A> <B> <END_A>
3	<!ELEMENT A (B,C)>	<A>::=<BGN_A> <A_CNT> <END_A> <A_CNT>::=<B> <C>
4	<!ELEMENT A (B|C)>	<A>::=<BGN_A> <A_CNT> <END_A> <A_CNT>::=<B> | <C>
5	<!ELEMENT A B?>	<A>::=<BGN_A> <B> <END_A> | <BGN_A> <END_A>
6	<!ELEMENT A B+>	<A>::=<BGN_A> <B_PLS> <END_A> <B_PLS>::=<B> | <B_PLS> <B>
7	<!ELEMENT A B*>	<A>::=<BGN_A> <B_AST> <END_A> <B_AST> ::= ε | <B_AST> <B>

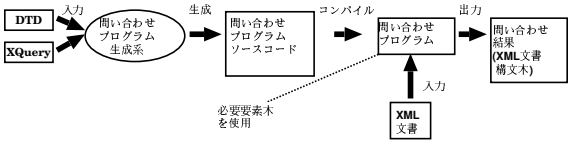

図 1　問い合わせプログラム生成系を用いた問い合わせの流れ

3.1　必要要素，必要要素木の定義

本研究における必要要素，必要要素木の定義について述べる．必要要素を **表 3** に示す．必要要素木とは，必要要素だけで構成される XML 文書構文木とする．問い合わせでは，FOR 節，LET 節で抽出した要素を用いて，WHERE 節による条件判別，RETURN 節による出力をおこなう．WHERE 節, RETURN 節に現れるパスに含まれる各要素は，抽出した要素から処理する要素を，パスにしたがって取得するさいに用いる．WHERE 節, RETURN 節に現れるパスの末端要素は，条件判別，出力処理に用いる．

表 3　問い合わせ処理に必要な要素

必要な要素	例
WHERE 節，RETURN 節に現れるパスに含まれる各要素	/person/address/city のとき person，address，city
WHERE 節，RETURN 節に現れるパスの末端要素の子孫要素	/person/address/city のとき city の子孫要素すべて

3.2　設計

問い合わせプログラム生成系を設計する．問い合わせプログラムを生成する手順を以下に示す．

1. DTD から字句，構文解析処理生成
2. XQuery から問い合わせ処理生成
3. XQuery から必要要素木作成処理生成

DTD からの字句，構文解析処理の生成

本研究では，xpgen における XML 文書処理生成方法をもとに，XML 文書の字句，構文解析処理を生成する．問い合わせプログラムでは，必要要素木を作成するために，XML 文書を字句，構文解析する．DTD からの字句，構文解析処理の生成方法は，2.2 節に示した．

XQuery からの問い合わせ処理生成

問い合わせ処理の生成は，以下の問い合わせごとに考える．

1. 1 重の FOR 節を用いた問い合わせ
2. FOR 節の入れ子を用いた問い合わせ

■1 重の FOR 節を用いた問い合わせ　1 重の FOR 節を用いた問い合わせと，問い合わせ処理との対応を **図 2** に示す．1 重の FOR 節を用いた問い合わせの処理は，XML 文書を 1 パスだけ解析することで処理できるので，構文解析中におこなう．1 パスしか解析しないので，一度使用した要素を格納しているメモリ領域は，直後に解放できる．

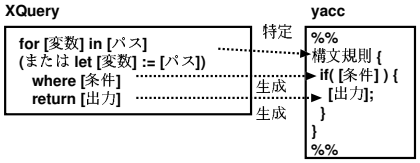

図 2　XQuery と問い合わせ処理の対応

■入れ子の FOR 節を用いた問い合わせ　入れ子の FOR 節を用いた問い合わせと，問い合わせ処理との対応を **表 4** に示す．構文解析中における問い合わせ処理では，結合処理のような FOR 節の入れ子を用いた問い合わせを扱えない．FOR 節の入れ子に対応するために，構文解析中に必要要素木を作成して，解析後，必要要素木を参照して処理をおこなう．作成した必要要素木を参照することで，複数回の解析をおこなうことができる．

表 4　FLWR 式とプログラムコードの対応

FLWR 式による記述	C 言語による記述
for [変数] in [要素リスト]	for([変数] = [要素リスト]; [変数] != NULL; [変数] = [変数] -> next)
let [変数] := [要素リスト]	[変数] = [要素リスト];
where [条件]	if([条件])
return [出力]	結果格納変数 = [出力] を作成;

XQuery からの必要要素木作成処理生成

XQuery から必要要素を判別する情報を取得し，必要要素木の作成処理を生成する． **表 3** に示した対応をもとに，必要要素に対応する構文規則のアクションに木を作成する処理を生成する．必要要素に対応する構文規則が適用されたときにだけ木を作成することで，必要要素木を作成できる．

3.3　実現

DTD，XQuery と問い合わせプログラムの対応をもとに，問い合わせプログラム生成系を実現する．実現には，C, lex, yacc を使用する．lex, yacc は，DTD，XQuery の解析に用いる．行数は約 5000 行である．

4　実験

本研究で実現した問い合わせプログラム生成系により生成した問い合わせプログラムの処理性能を評価するために XMark[1] を用いた実験をおこなう．実験結果をもとに，既存の XQuery 処理系の一つである SAXON と比較する．実験は，Intel Pentium 3.20GHz，メモリサイズ 1GB，OS は Vine Linux 4.0 でおこなった．以下の二つについて比較することで性能の評価とする．

- 処理時間
- メモリ使用量

XMark とは，XQuery 処理系の性能評価として広く用いられているベンチマークである．XMark では，20 個の問い合わせを用いて性能評価をおこなう．XMark で用意されている各問い合わせで評価できる処理を **表 5** に示す．本研究では，XMark のうち，Q18，Q19 は取り扱わない．Q18 は，ユーザが独自に定義した関数を用いた場合の処理性能を評価するための問い合わせである．Q19 は，整列性能を評価するための問い合わせである．本研究では，どちらも性能向上の対象としていない．

表 5　XMark の問い合わせ

No	評価
1	値による絞りこみ
2	位置指定
3	位置指定，算術演算
4	位置指定
5	要素数の取得
6	必要要素が多い処理，要素数の取得
7	必要要素が多い処理，要素数の取得，算術演算
8	結合処理
9	結合処理
10	結合処理，要素の作成
11	結合処理，算術演算
12	結合処理，算術演算
13	要素の作成
14	文字列の検索
15	子要素の取得
16	子要素の取得，子要素を有無
17	子要素の有無
18	ユーザ定義関数
19	整列
20	複数回の問い合わせ

4.1　処理時間

本研究で生成した問い合わせプログラムと SAXON における処理時間を比較する． **図 3** に問い合わせプログラムと SAXON の処理時間を示す．Q8 から Q12 の問い合わせは，それ以外の問い合わせと比較して，値が大きく異なるので，グラフを二つにわけて示す．Q8 から Q12 は，FOR 節の入れ子を用いた問い合わせである．それ以外の問い合わせは，1 重の FOR 節を用いた問い合わせである．

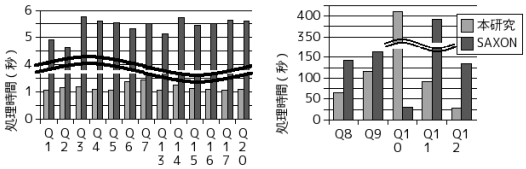

図 3　Xmark における処理時間

4.2　メモリ使用量

本研究で生成した問い合わせプログラムと SAXON におけるメモリ使用量を比較する． **図 4** に，問い合わせプログラムと SAXON のメモリ使用量を示す．

4.3　実験結果比較

XMark の実験結果を用い，既存の XQuery 処理系 SAXON と比較する．

処理時間

SAXON と比較して，Q10 以外の問い合わせを高速に処理できることを確認した．高速に処理できた原因は，

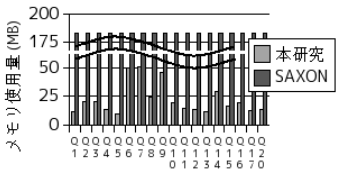

図 4　XMark におけるメモリ使用量

必要要素木を用いたことと考える．5.1 節で，必要要素木を用いたことによって削減できた処理時間を求める．5.2 節で，Q10 の処理を改善する方法を考察する．

メモリ使用量

SAXON と比較し，すべての問い合わせにおいて少ないメモリ領域で処理できることを確認した．原因は，必要要素木を用いたことと考える．5.1 節で，必要要素木を用いたことによって削減できたメモリ領域を求める．

5　考察

実験結果をもとに本研究の問い合わせプログラムの性能について考察する．

5.1　定量的評価

作成する必要要素数と，処理時間，メモリ使用量の関係から，必要要素木を用いることによって削減できる処理時間，メモリ領域を求める．

処理時間

削減時間は，XML 文書全体の要素数と必要要素数の差と，要素一つあたりの作成時間との積とする．求めた削減時間を **図 5** に示す．

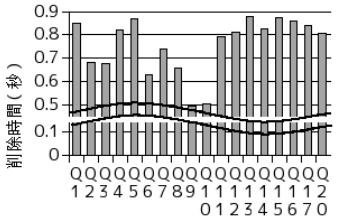

図 5　削減できた時間

図 3 ， **図 5** をもとに，処理時間について考察をおこなう．1 重の FOR 節を用いた Q8 から Q12 以外の問い合わせと，FOR 節の入れ子を用いた Q8 から Q12 では，大きく時間が異なるので，それぞれについて考察する．

■1 重の FOR 節　1 重の FOR 節を用いた問い合わせの処理では，全体の約 90 % を木の構築時間が占めているので，必要要素木を用いて，大きく処理時間を削減できた．Q3 を除き， **図 5** に示す削減時間が多いほど，処理時間が少ない．必要要素木を用いることで， **表 5** に示す 1 重の FOR 節を用いた問い合わせの処理を高速化できると考える．

Q3 は，算術演算を評価するための問い合わせであることから，本研究の問い合わせプログラムは，算術演算を高速に処理できると考える．SAXON は，インタプリタにより実行するのに対し，本研究では，ネイティブコードとして実行することが原因だと考える．実験により，問い合わせプログラムでは，SAXON の約 1/16000 倍

の時間で算術演算を処理できることを確認した．

■FOR **節の入れ子**　FOR 節の入れ子を用いた問い合わせの処理では，必要要素木による削減時間は，全体と比較すると非常に小さい．木の構築時間が，全体の約 10 % しかないことが原因と考える．

Q11，Q12 は，Q8，Q9 と比較して，SAXON より大きく処理時間が削減できた．Q11，Q12 は，算術演算を評価するための問い合わせであることから，Q3 と同様に，SAXON より大きく削減できたと考える．

Q8，Q9 の実験結果より，子要素の取得処理だけをおこなう FOR 節の入れ子を用いた問い合わせは大きく高速化できないと考える．Q8，Q9 では，要素の作成や，子要素の取得のような，SAXON で高速化されている処理だけを扱っていることによる．実験により，本研究の要素の作成処理は，SAXON の約 2 倍，子要素の取得処理は，約 4.0×10^7 倍かかることを確認した．

Q10 の処理には，SAXON よりも処理時間が必要なことを確認した．Q10 は，要素の作成処理を評価するための問い合わせであることから，本研究の問い合わせプログラムは，要素の作成処理に問題があると考える．

メモリ使用量

削減できたメモリ使用量は，XML 文書全体の要素数と必要要素数の差と，要素一つあたりのメモリ使用量との積とする．求めた削減メモリ使用量を **図 6** に示す．全ての問い合わせにおいて，削減できたメモリ使用量が大きい問い合わせほど，少ないメモリ使用量で処理できることを確認できた．必要要素木を用いることで，**表 5** に示す問い合わせ処理のメモリ使用量を削減できると考える．

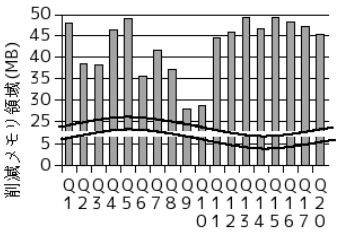

図 6　削減できたメモリ使用量

5.2　Q10 の処理時間に関する考察

本研究の問い合わせプログラムが，SAXON より処理時間を必要とする Q10 を改善する方法を考察する．Q10 では以下の処理がおこなわれている．

1. 要素の作成処理
2. 子要素の取得処理

1，2 を改善することで，問い合わせプログラムを高速化できると考える．1 は，一度に複数個分のメモリ領域を確保し，領域確保の回数を削減することで削減できると考える．本研究では，要素ひとつ作成するたびに，領域の確保をおこなっている．2 は，ハッシュ表を用いた要素の探索により，削減できると考える．子要素の取得処理において，要素の探索処理は，処理する要素数に比例して時間が増加する．探索時間を削減することで，子要素の取得処理を高速化できると考える．

本研究では，必要要素木を用いることによる，作成する要素数の削減を主眼としている．要素の作成，子要素取得の改善の実現は今後の課題とする．

6　おわりに

本研究では，必要要素木を用いることで効率のよい問い合わせをおこなうプログラムを生成する問い合わせプログラム生成系を設計，実現した．XMark を用いて生成した問い合わせプログラムの性能を評価した．既存の XQuery 処理系 SAXON と比較して，最大，処理時間を約 1/5 倍に，メモリ使用量を約 1/10 倍に削減できた．

今後の課題を以下に示す．

1. 木操作の効率化
2. 子孫要素の指定，任意の子要素指定への対応
3. 1 パスで処理できる FOR 節の入れ子を用いた問い合わせの発見

1 は，5.2 節で示した問題を改善するためにおこなう．2 は，本研究では，問い合わせ処理と比較して少ない時間，メモリ領域で処理できると考え，対象としていなかった．問い合わせプログラムを生成するさいに作成する DTD の構文木を参照することで対応できると考える．3 により，構文解析後の処理として対応している，FOR 節の入れ子を用いた問い合わせの処理を構文解析中におこなうことができる．例として，内側の FOR 節で，外側の FOR 節により抽出した要素を用いる問い合わせがある．

謝辞

本研究を進めるにあたり，熱心に御指導下さいました野呂昌満教授，有益なアドバイスを頂いた蜂巣吉成先生，張漢明助教授に深く感謝致します．

参考文献

[1] A. R. Schmidt, F. Waas, M. L. Kersten, D. Florescu, I. Manolescu, M. J. Carey, R. Busse, ”XMark : A Benchmark for XML Data Management,” in *Proc. of the VLDB Conference,* 2002, pp. 974-985.

[2] 蜂巣吉成, 野呂昌満, 張漢明, ”データ型を考慮した軽量な XML 文書処理系の自動生成,” コンピュータソフトウェア, Vol. 19, No. 5, pp. 333-342, 2002.

[3] 石野明, 竹田正幸, ”ストリーム指向の XQuery 処理システムについて,” 情報処理学会論文誌, Vol. 2005, No. 35, pp. 73-80, 2005.

[4] M. Nicola and J. John, ”XML parsing: a threat to database performance,” in *Proc. of the CIKM,* 2003, pp. 175-178.

[5] The SAXON XSLT and XQuery Processor, http://saxon.sourceforge.net/, 2007.

[6] XQuery, http://www.w3.org/TR/xquery/, 2006.