実行場所の切り替えによるXSLアプリケーション応答時間の削減

山田, 信太郎
九州大学工学部電気情報工学科

岩井原, 瑞穂
九州大学大学院システム情報科学研究院情報工学部門

出版情報：SLRC 論文データベース, 2001-03
バージョン：
権利関係：

実行場所の切り替えによるXSLアプリケーション応答時間の削減

山田 信太郎 †, 岩井原 瑞穂 ‡
† 九州大学 工学部 電気情報工学科, ‡ 九州大学大学院 システム情報科学研究院 情報工学部門

概要

汎用的なデータフォーマット言語であるXMLとそのスタイルシート言語であるXSLの組み合わせは, 表示のためにHTMLへ変換する処理を行う場所をクライアントとサーバで自由に切り替えて実行すること可能とする. サーバやクライアント, 通信路の状態等によって最適な実行場所を動的に選ぶことができれば, アプリケーションの応答時間を短縮することが可能である. そこで本稿では, サーバ側とクライアント側のそれぞれで処理を行った場合に要する時間を実験で測定し, 時間が逆転する条件を調べ, 最適な場所を選ぶための判断基準を考える.

1 はじめに

近年, 計算機ネットワークの発展により, ネットワークを用いたデータのやりとりが一般的となっている. ネットワークを用いると, 膨大な情報を利用することが可能となるが, アクセス集中によるサーバへの負荷増大やネットワークによる遅延の問題がある.

XMLは汎用的なデータフォーマット言語であり, そのスタイルシート記述言語であるXSLを用いてHTMLへ変換できる. このXSLを用いた変換はサーバとクライアントのどちらでも行うことができる. サーバで変換を行う場合とクライアントで変換を行う場合の特徴を考えると, サーバで変換を行う場合の特徴としては次のようなことが挙げられる.

- 送信するデータ量を減らすことができる可能性が高い.
- 毎回通信を行う必要がある.
- アクセスが多いと処理が遅くなる可能性がある.
- セキュリティの管理がしやすい.
- データの最新性を保つのが容易である.

クライアントで変換を行う場合の特徴としては次のようなことが挙げられる.

- 必要以上のデータを受信する可能性がある.
- 同じデータに対して処理を行う場合, 二回目以降は通信が起こらない.
- ハードウェアを占有できる.
- セキュリティの管理が複雑になる.
- データがサーバで更新されても手元のデータには直ちに反映されない.

ハードウェア性能や取り扱うデータ量, 通信路の状態によって最適な実行場所を動的に選択すれば, アプリケーションの応答時間を短縮する事が可能である. そのためには, サーバでの実行時間とクライアントでの実行時間が変換を行う前に予測できる必要がある. 本稿では実行時間を予測する手法を提案し, その手法を検証するための実験を行う.

本稿では同一のDTDを持ち, 大きさおよび要素数が一定のばらつきを持つXML文書の集合を対象とし, いくつかのXSLプログラムで変換を行う状況を想定する. そして過去に実行したプログラムの実行時間の構成から, 新たに実行するプログラムの実行時間を予測する. 実行時間の予測式を求めるために利用可能な情報は以下が考えられる.

- 変換前後のデータ量と実行時間
- ハードウェアの性能差
- ハードウェアの負荷状況
- ネットワークの混雑状況
- 処理回数

上記の測定値の関連を調べるため, 変換前及び変換後のデータ量を変化させ, 上記項目を測定した. この実験結果をもとに, 実行時間を予測する理論式を導出した. 予測式から求めた実行時間と実際の実行時間の誤差は5%程度であることを確認した. この手法を実際の環境で応用するには, 実行時間を記憶して, 予測式の係数を随時更新していけばよい. この予測式を用いて効率の良い実行サイトを決定することができる.

以下, 2節ではXSLを用いた変換の実行時間モデルを求める. 3節では予測式の決定のために行った実験の内容と結果を示す. 4節では3節の結果より予測式をもとめ, 実行時間を予測した結果を示す. 5節で

は最適な実行場所を選ぶ判断基準を求める．6 節では全体のまとめと今後の課題についてふれる．

2 実行時間モデル

2.1 XSL 変換の流れ

典型的な XSL を用いた変換の処理の流れは以下の通りである．

1. XML 文書を構文解析して XML 文書の構造を表す DOM 木を得る
2. DOM 木からテンプレートに合う要素を探し出す
3. テンプレートに従って文書を変換し，結果を出力する

本稿では，この XSL を用いた変換処理を XSL 変換と呼ぶことにする．図 1 にサーバ側とクライアント側それぞれで XSL 変換を行う場合の処理の流れを示す．同

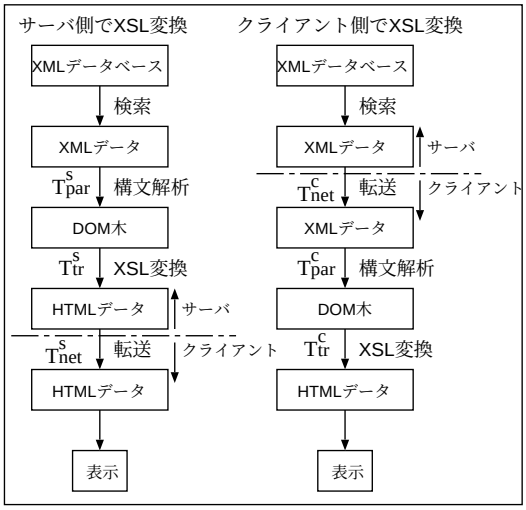

図 1: XSL 変換の流れ

じデータに処理を繰り返す場合，XSL 変換以降が繰り返されることになる．図 1 に示したとおり，サーバ側で XSL 変換処理を行う場合では転送が繰り返されるが，クライアント側で XSL 変換処理を行った場合では二回目以降転送は起こらない．

2.2 XSL 変換の実行時間モデル

通信時間を T_{net}，構文解析時間を T_{par},HTML への変換時間を T_{tr}，表示時間を T_{pr}，総実行時間を T_{total}，変換前データの大きさを D_{xml}，変換後データの大きさを D_{html} とおく．図 1 に従った処理時間はサーバ側とクライアント側のどちらで変換を行う場合でも

$T_{total} = T_{net} + T_{par} + T_{tr} + T_{pr}$

となると考えられる．同じデータに n 回処理を繰り返した場合は以下のようになる．

a) サーバ側で XSL 変換した場合

$T_{total} = T_{par} + n(T_{net} + T_{tr} + T_{pr})$

b) クライアント側で XSL 変換した場合

$T_{total} = T_{net} + T_{par} + n(T_{tr} + T_{pr})$

ネットワークやハードウェアの負荷を一定としてデータの大きさ以外のパラメータを一定とおき，T_{net} と T_{par}，T_{tr} を求めると

サーバ側で XSL 変換した場合は以下で表される．

$T_{net} = f_0(D_{html})$,
$T_{par} = F_0(D_{xml})$,
$T_{tr} = f_1(D_{xml})$,

クライアント側で XSL 変換した場合は以下で表される．

$T_{net} = f_2(D_{xml})$,
$T_{par} = F_1(D_{xml})$,
$T_{tr} = f_3(D_{xml})$,

ここで，$f_k(x)(k = 0, 1, 2, 3)$ は一次関数として近似する．F_0と F_1 は次節の実験で特性を求める．サーバ側で XSL 変換を行った場合とクライアント側で XSL 変換を行った場合では主に T_{net} と T_{par},T_{tr} について差が現れると考えられる．T_{par} と T_{tr} はハードウェアの性能によって決まると思われるため，ハードウェアの性能差が小さい場合は，両者の差は T_{net} によると考えられる．サーバ側の T_{net} を T^s_{net}，クライアント側の T_{net} を T^c_{net} とすると，多くの場合

$T^s_{net} < T^c_{net}$

となると思われるため，サーバ側で XSL 変換を行った方が実行時間が短いと考えられる．同じデータに対して複数回処理を行った場合には，ある時点で

$nT^s_{net} > T^c_{net}$

となり，クライアント側で XSL 変換を行う方が実行時間が短くなる場合が現れると考えられる．またハードウェアの性能の差が大きい場合，サーバ側の T_{par},T_{tr} とクライアント側の T_{par},T_{tr} の差は処理回数が増えると広がっていく．このため，T_{net} の差と T_{par},T_{tr} の差の関係によって実行時間の関係も決まると考えられる．

上記でデータ量の一次関数として表現した実行時間は，ハードウェアの性能及び負荷やネットワークの混雑状況によりこれらの一次関数の係数が変化するものとしてモデル化する．状況に応じてサーバ側変換とクライアント側変換で有利な方が逆転することが予想される．

3 XSL 変換実行時間の測定

実行時間を決定する要因を調べるため，変換前のデータ量を変化させた場合と変換後のデータ量を変化させた場合について，サーバで XSL 変換を行った場合にかかる時間とクライアントで XSL 変換を行った場合にかかる時間を測定し，実行時間の内訳と両者の時間の比較を行った．

3.1 実験環境

変換元のデータとして使用した XML 文書は論文査読システムに用いられているもので，論文の名前や PDF ファイルの場所，査読者のコメントなどの要素を持つ．DTD で定義される要素数は 18，文書数は 8 である．この XML 文書を複製して以下のデータ量を持つ XML 文書を作成した．

D_{xml} (単位は MB)					
0.5	1.0	1.5	2.0	2.5	3.0

実行する XSL プログラムとして，上記の XML 文書から指定した論文のデータを抽出して表示させるものを用いた．そして以下の二つの場合について元の XML 文書のデータ量及び変換後の HTML 文書のデータ量を変えながら実行時間を測定した．

- サーバ側で HTML に変換してからクライアント側に転送する場合
- クライアントに XML データを転送して HTML に変換する場合

変換後のデータ量を変化させる場合は，変換前のデータ量を 1MB に固定して変換後のデータ量を以下に示す値とした．

D_{html} (単位は MB)				
0.1	0.4	0.8	1.2	1.5

変換に用いた XSL プログラムの主な部分を以下に示す．

```
<-- 取り出す情報の選択 -->
<xsl:template match="mgroups">
<xsl:apply-templates select="mgroup[@name<2]"/>
<-- この数字を変化させることで変換後のデータ量を
変化させる -->
</xsl:template>
<-- コメントの表示 -->
<xsl:template match="mgroup">
<xsl:value-of select="@name"/>
<xsl:copy-of select="children/mgroup/coment" />
<xsl:apply-templates />
</xsl:template>
```

変換プログラムには Java を使い，XML パーサは XML4J-3.1.0，XSL プロセッサは lotusxsl-1.0.1 を使用した．サーバ側での実行には Java Servlet（jswdk-1.0.1）を用いた．測定に使用したハードウェアは以下の通りである．

lowp CPU:UltraSPARCII 296MHz, メモリ:1GB, OS:SunOS 5.6

highp CPU:PentiumIII 1GHz, メモリ:512MB, OS：Windows2000

ここで highp の方が lowp よりも処理能力が高い．ネットワークについてはローカルエリアのイーサネットを使用した．各ハードウェアとスイッチングハブは 100BaseTx で接続されている．

実行時間の測定方法としては，プログラム中で date メソッドを用いてそれぞれの処理の開始時間と終了時間を測り，その差をとることで測定している．測定した時間は，データを受信するのにかかる時間（T_{net}）,XML 文書の構文解析時間（T_{par}）,XML 文書を HTML に変換するのにかかった時間（T_{tr}）, 総実行時間（T_{total}）である．ここで，T_{total} を

$$T_{total} = T_{net} + T_{par} + T_{tr}$$

としている．また，$T^s_{tr}, T^s_{par}, T^s_{net}, T^s_{total}$ をそれぞれサーバで XSL 変換を行った場合の $T_{net}, T_{par}, T_{tr}, T_{total}$ とし，$T^c_{tr}, T^c_{par}, T^c_{net}, T^c_{total}$ をそれぞれクライアントで XSL 変換を行った場合の $T_{net}, T_{par}, T_{tr}, T_{total}$ とおく．

3.2 ハードウェアの性能差が小さい場合

ハードウェアにサーバとクライアントの両方とも highp を使用した場合の T_{net}, T_{par}，T_{tr} を以下の場合について調べた．

1. 変換後のデータ量を 0.8MB に固定し変換前のデータ量を変化させる
2. 変換前のデータ量を 1MB に固定し変換後のデータ量を変化させる

1 の結果について横軸を変換前データ量，縦軸を時間としたサーバ側とクライアント側それぞれの T_{net}，T_{par}，T_{tr}，T_{total} のグラフを図 2 に示す．D_{xml} を変化させた場合，実行時間の増加は T_{par} に支配されている．サーバ側では T_{net} は一定である．

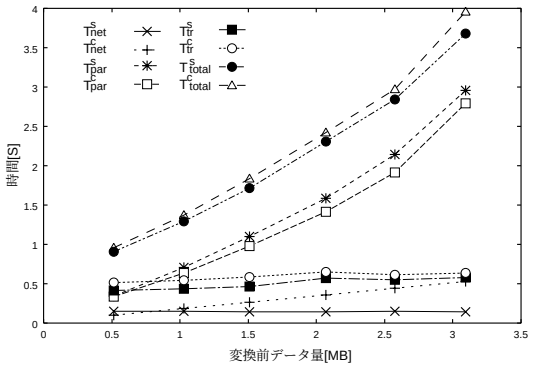

図 2: 変換後 0.8MB 固定 性能差小 サーバとクライアントの実行時間比較

2 の結果について横軸を変換後データ量，縦軸を時間としたサーバ側とクライアント側それぞれの T_{net}，T_{par}，T_{tr}，T_{total} のグラフを図 3 に示す．D_{html} を変

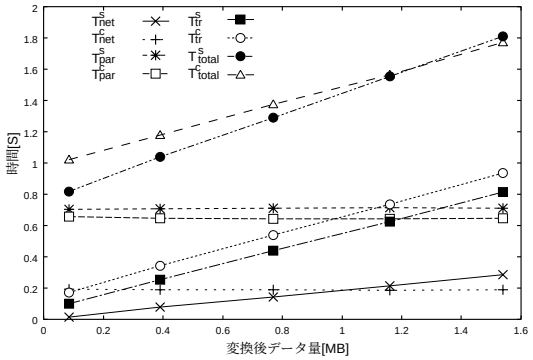

図 3: 変換前 1MB 固定 性能差小 サーバとクライアントの実行時間比較

化させた場合は，実行時間の増加は T_{tr} がもっとも影響を与えており，T_{par} は一定である．クライアント側の T_{net} も一定である．

3.3 ハードウェアの性能差が大きい場合

highp をサーバ，lowp をクライアントとして使用した場合と lowp をサーバ，highp をクライアントとして使用した場合のそれぞれで時間を測った．条件は 3.2 節と同じく，以下の場合について調べた．

1. 実行回数を 1 回，変換後のデータ量を 0.8MB に固定し変換前のデータ量を変化させる
2. 実行回数を 1 回，変換前のデータ量を 1MB に固定し変換後のデータ量を変化させる

3.3.1 サーバ側が高性能な場合

1 の結果から，D_{xml} を変化させた場合サーバ側，クライアント側の両方とも，実行時間の増加は T_{par} に支配されている．T_{tr} の増加率は T_{par} の増加率と比べるとかなり小さいので，全体としてみた場合はほぼ一定と見なせる．それぞれの時間を見た場合，性能差が小さい場合に比べて，T_{par} および T_{tr} に大きな差が現れていることがわかった．

2 の結果からは，D_{html} を変化させた場合，サーバ側で XSL 変換を行った場合の実行時間の増加は T_{net} の増加によるものが大部分を占めるが，クライアント側で XSL 変換を行った場合の実行時間の増加は T_{tr} に支配されていることがわかった．

3.3.2 クライアント側が高性能な場合

1 の結果から，D_{xml} を変化させた場合の実行時間の増加はやはり T_{par} に支配されており，T_{tr} はサーバ側，クライアント側ともに全体から見ればほぼ一定であることがわかった．

2 の結果からは，D_{html} を変化させた場合は実行時間の増加はサーバ側，クライアント側の両方で T_{tr} に支配されていることがわかった．

4 実行時間モデルの検証

4.1 実験結果の考察

3 節の実験結果より以下の性質が求められた．

1. 通信時間はサーバ側では変換後データ量，クライアント側では変換前のデータ量が大きく寄与する．
2. 構文解析時間は変換前のデータ量とハードウェア性能によって決定される．
3. 変換時間は変換前後のデータ量とハードウェア性能によって決定される．

また，図 2 のグラフより T_{par} は $D_{xml} \log_2 D_{xml}$ に比例する関数で近似できることがわかる．このことが他の XML 文書集合についても成り立つかを調べるために，http://www.acm.org/sigs/sigmod/record/xml で配布されている文献情報に関する XML 文書について T_{par} を測定した．この文書は DTD に存在する要素数が 19，文書数 919，文書の大きさの平均は 2.2KB である．文書を順次追加して D_{xml} を増大させながら T_{par}

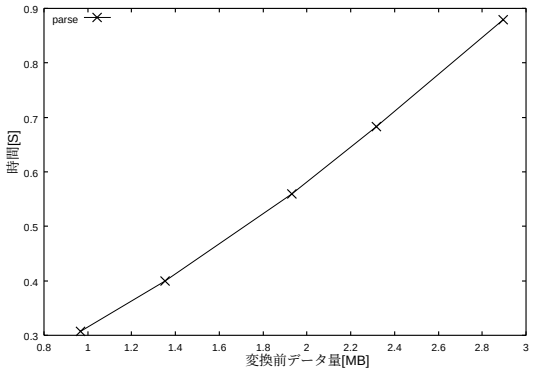

図 4: 与えられる D_{xml} についての T_{par}

を測定した．測定した T_{par} のグラフを図 4 に示す．この結果，同程度の大きさを持つ XML 文書の集合について，その合体の構文解析時間 T_{par} は $D_{xml}\log_2 D_{xml}$ に比例する関数として近似できるといえる．

4.2 実行時間の予測式

$T_{net}, T_{par}, T_{tr}, D_{xml}, D_{html}$ の項目は，いずれも XSL プログラムを実行するたびに測定し記録することが，無視できる手間で可能である．これらの測定値を利用して，最適な実行サイトを求めることを考察する．2.3 節のモデルの詳細化を行ない，ハードウェアの負荷状況，ネットワークの混雑状況が一定であり，同一の XSL プログラムを用いた場合の実行時間を，処理すべきデータの大きさ D_{xml}, D_{html}, および n から予測する理論式を以下に示す．

データ量を変化させた場合，T_{net} および T_{tr} はほぼ直線であるため，それぞれ D_{xml} と D_{html} の一次式で表現する．T_{par} は $D_{xml}\log_2 D_{xml}$ に比例する関数として表現する．それぞれの式は以下のようになる．

$T^s_{net} = n(aD_{html} + b_0)$
$T^s_{par} = c_0 D_{xml}\log_2 D_{xml} + d_0$
$T^s_{tr} = n(e_0 D_{xml} + f_0 D_{html} + g_0)$
$T^c_{net} = aD_{xml} + b_1$
$T^c_{par} = c_1 D_{xml}\log_2 D_{xml} + d_1$
$T^c_{tr} = n(e_1 D_{xml} + f_1 D_{html} + g_1)$

上記ではネットワーク転送速度は T^s_{net}, T^c_{net} の係数に含まれることになる．またクライアントとサーバの負荷状況を加味した処理性能比は $T^s_{par}, T^s_{tr}, T^c_{par}, T^c_{tr}$ の係数に現れることになる．

3 節の実験データから求めた係数の一例を以下に示す．

$T^s_{net} = n(0.1766D_{html} + 0.001916)$,
$T^s_{par} = 0.07822D_{xml}(\log_2 D_{xml} + 9)$
$T^s_{tr} = n(0.07104D_{xml} + 0.4684D_{html} - 0.009105)$
$T^c_{net} = 0.1661D_{xml} + 0.01669$,
$T^c_{par} = T^s_{par}$
$T^c_{tr} = T^s_{tr}$

4.3 実行時間予測式の変動要因

サーバとクライアントのそれぞれの実行時間は，ハードウェアの処理性能の比と，負荷状況によって決まると考えられる．実際の実行時間はハードウェアの処理性能の比に各サイトの負荷状況を加味したものとなる．同一の XML データを各サイトで同じ処理として例えば構文解析を行なうことにより，負荷状況を加味したハードウェアの処理性能の比が求まる．負荷状況が変化した場合は，同一データの実行時間が変化することにより検出できる．4.3 節の予測式において，ハードウェアの性能比および負荷状況の変化の影響を受ける項目は, サーバ側の T^s_{par}，T^s_{tr} およびクライアント側の T^c_{par}，T^c_{tr} である．

ネットワークの混雑状況により通信時間は変化すると考えられる．ネットワークの混雑状況の影響を受ける予測式の項目は，T^s_{net} および T^c_{net} である．逆にこれ以外の項目は混雑状況の影響を受けないといえる．

XSL プログラムに依存して，変化すると考えられる項目は，T^s_{tr}, T^c_{tr}, および D_{html} である．D_{html} は XSL プログラムの出力結果の大きさであるからこれもプログラムに依存する．これに対し, XML データの転送時間および構文解析時間は XSL プログラムとは独立に決まる．

4.4 実行時間の予測問題 – 負荷一定の場合

ここでの予測問題とは，XSL プログラム，ハードウェアの負荷状況およびネットワークの混雑状況が一定であるとの仮定のもとで，与えられた XSL プログラム, D_{html}, D_{xml}, n から，サーバおよびクライアントでの総実行時間を予測する問題である．両者の予測時間が精度の良いものであれば，効率のよい実行サイトを正確に決定できる．

以下では，D_{html} および D_{xml} の 10 個の組み合わせについて，$T^s_{tr}, T^s_{par}, T^s_{net} T^c_{tr}, T^c_{par}, T^c_{net}$ の測定結果が存在していると仮定する．これらの測定結果から予測式の係数を求め，$n = 1$ として，与えた D_{html} および D_{xml} に関する予測時間を計算する．予測時間の精度を検証するために，同一のパラメータに対する実測

変換前データ量 *[MB]*		1	2	1	1	3.5
変換後データ量 *[MB]*		0.8	0.8	0.4	1.2	4.0
サーバ側	実験値 [s]	1.279	2.223	1.038	1.554	6.943
実行時間	予測値 [s]	1.278	2.249	1.017	1.537	6.718
	誤差 [%]	0.1	1.2	2.0	1.1	3.2
クライアント側	実験値 [s]	1.258	2.457	1.096	1.469	6.512
実行時間	予測値 [s]	1.319	2.468	1.130	1.508	6.608
	誤差 [%]	4.9	0.5	3.1	2.7	1.5

表 1: 実験値と予測値の比較

時間を実験により測定し，比較を行なった．結果を表 1 に示す．全体的に誤差は 5%程度であり，過去に複数のデータ量に関する測定値が存在していれば，データ量が変化した場合についての実行時間をほぼ正しく予測する事ができるといえる．

また D_{xml} が 1MB，D_{html} が 0.8MB の組み合わせについて，与えた n に関する予測時間を計算し，実験値と予測値の比較を行った．その結果，誤差はサーバ側で 1%，クライアント側で 5%程度であった．この場合でも実行時間をほぼ正しく予測できたといえる．

4.5 実行時間の予測 – 負荷変動のある場合

XSL プログラムに異なるものが使用される場合や，ハードウェアの負荷状況，ネットワークの混雑状況に変化があった場合は，予測式の係数を再計算する必要がある．過去に測定した値のうち，4.3 節で述べた 変動の影響を受ける項目は，再計算に用いることはできず，破棄する必要がある．これに対し，変動の影響を受けない項目は，新たな係数の導出にも使用することができる．

予測式の係数を決定する計測値が不足する場合は，変動前の測定値を，予測の精度を犠牲にして使用することが考えられる．例としては，XSL プログラムとして新たな未知のものが与えられた場合に，既知の XSL プログラムに対する測定値で代用するなどがある．

以下の実験では，サーバの負荷が変動した場合の実行時間予測について検討する．サーバの負荷が変動し，処理効率が低下したと仮定する．負荷状況の変化は，同一データに対する T^s_{par}, T^s_{tr} の変化として検出できる．この 1 回の測定値から T^s_{par}, T^s_{tr} の新たな係数を算出し，他の項目については変動前の測定値を用いて予測時間を求めた．この予測時間と実測した実行時間を表 2 に示す．

誤差は 5%以内に収まっており，過去の実行データに対しては，負荷が変動した場合についても実行サイトの判定に使用できる精度の予測ができたといえる．

5 最適な実行場所を選ぶための判断基準

以下では，ネットワークの混雑状況およびハードウェアの負荷状況について，変動の少ない状態が継続して，実行時間の記録が得られているとの前提のもとで，効率の良い実行場所を決める判断基準を考察する．

5.1 実行時間が逆転する条件

通信時間は

$nT^s_{net} = T^c_{net}$

となる時点で逆転するため，

D_{xml}/D_{html}

の値が小さいほど，少ない回数でサーバ側とクライアント側の転送時間が逆転する．

構文解析時間と変換時間は逆転することはなく，ハードウェアの性能により決定される．

実行時間全体として考えたときは D_{html} の転送時間と変換時間の差が重要である．

$T_t = (T^c_{tr} - T^s_{tr})$

とおくと，

$T^s_{net} < T_t$

の場合，回数を重ねた場合に受ける影響は変換時間によるものとなるため，実行時間が逆転する事はない．

$T^s_{net} > T_t$

の場合，一回の処理で生じる差は

$T^s_{net} - T_t$

となり，

$T^c_{net} + (T^c_{par} - T^s_{par}) = n(T^s_{net} - T_t)$

となる時点で実行時間の逆転が起こる．

変換前データ量 *[MB]*		1	2	1	1
変換後データ量 *[MB]*		0.8	0.8	0.4	1.2
サーバ側	実験値 [s]	7.403	13.717	7.824	13.47
実行時間	予測値 [s]	7.433	13.756	7.989	12.935
	誤差 [%]	0.41	0.28	2.1	4.0

表 2: サーバの負荷が変動した場合の実験値と予測値の比較

5.2 最適な実行場所を選ぶ判断基準

ハードウェアの性能差が小さい場合には D_{xml}/D_{html} を判断基準とすることができる．5.1 節よりサーバとクライアントのハードウェア性能及び負荷状況が同じだとすると，

D_{xml}/D_{html}

の値が 1 以上の場合はサーバ側，1 を下回る場合はクライアント側で実行した方がよい．複数回実行する場合にはこの値を実行回数が越える場合には，クライアント側で処理を行った方がよいことが予想できる．逆に，1〜2 回しか処理が行われないデータについては，ほとんどの場合サーバ側で処理を行うべきである．

ハードウェアの性能差が大きい場合には T_{net} の差と T_{tr} の差のどちらがより影響を与えるかということが問題となる．T_{net} の差が与える影響はデータ量と転送速度によって決まり，T_{tr} の差が与える影響はハードウェアの性能比によって決まる．そこで，与えられた D_{xml}, D_{html} についての

$(T^c_{net} + T^c_{par} - T^s_{par})/(T^s_{net} - T_t)$

を計算し，その値が 1 以上の場合はサーバ側，1 を下回る場合はクライアント側で実行した方がよい．複数回実行する場合には，この値を実行回数が越えればクライアント側で処理を行った方がよいと思われる．

6 おわりに

本稿ではクライアントとサーバの分散環境で XML 文書を処理する応用において，XSL 変換を最適に実行する方式を決定する手法として，サーバでの実行時間とクライアントでの実行時間をそれぞれ予測し，最適な実行サイトを選択する手法を考察した．実行するたびに処理時間の構成を記録することにより，新たに XSL 変換を行う際の実行時間を予測する式を示した．ハードウェアの負荷等の環境が一定の場合と，一部変動する場合とについて，予測値と実測値がよく一致することも確認した．今後の課題としては，さらに多様な環境での本手法の評価を行うことと，本手法を組み込んだ XSL アプリケーションを試作することである．

参考文献

[1] Extensible Markup Language (XML) 1.0 W3C Recommendation, http://www.w3.org/TR/REC-xml.

[2] Extensible Stylesheet Language (XSL)Version 1.0 W3C Working Draft March 2000, http://www.w3.org/TR/XSL/.

[3] XSL Transformations (XSLT)Version 1.0 W3C Recommendation November 1999, http://www.w3.org/TR/xslt.html.

[4] P.Krishna, 喜連川 優: インターネット上におけるサービス時間とサーバ負荷を減少させるためのスケーラブルプロトコル, 電子情報通信学会技術研究報告 DE99-81〜89[データ工学]1999 年 10 月.