17-3-R06

# 마크 업 문서의 대수적 분석에 의한 분할과 변환

(A Decomposition and Transformation of Markup Documents with Algebraic Analysis)

곽 동 규, 최 종 명, 유 재 우
숭실대학교 컴퓨터학과
{coolman, jmchoi, cwyoo}@computing.ssu.ac.kr

## 요 약

현재 많은 응용프로그램들이 XML로 표현된 정보를 이용하여 서비스하고 있다. 특정 응용프로그램을 위해 작성된 XML 문서를 다른 응용프로그램에서 사용하기 위해서는 적절한 변환이 필요하다. 본 논문은 대수학적 방법을 이용해서 XML 문서를 분할하고, 변환하는 방법을 소개한다. 이 방법을 사용하기 위해서 논문에서는 XML 문서의 태그간의 관계를 연산으로 정의하고, 연산의 특성을 이용해서 문서의 경계를 구분하는 방법과 경계가 구분된 문서를 분할하는 방법을 제시한다. 또한, 분할된 문서의 태그들을 그룹핑해서 m:n으로 변환이 가능한 단위를 정의하고, 이 단위를 이용해서 XML 문서를 변환하는 방법을 소개한다. 대수학적 방법을 이용한 문서의 변환은 일반적인 XML 문서에 적용할 수 있고, 경계를 구분하여 정보의 이해가 용이한 문서를 생성하는 장점을 가지고 있다. 논문에서는 문서 변환의 예로 HTML 문서를 VoiceXML로 변환하는 HTMLtoVoiceXML을 소개한다.

## 1. 서론

XML은 데이터를 표현하기 위한 W3C의 표준 마크 업 언어로서 점차 다양한 분야에서 다양한 목적으로 사용되고 있기 때문에 특정 응용프로그램의 데이터를 표현하기 위해 사용된 XML 문서를 필요에 따라 다른 응용프로그램을 위한 XML 문서로 변환해야 하는 요구가 많아지고 있다. 예를 들어, 현재 데스크탑 컴퓨터의 웹 브라우저를 위해 작성된 HTML 혹은 XHTML 문서를 음성 단말기를 위한 VoiceXML[3] 혹은 핸드폰을 위한 WML 등의 문서로 변환해야 할 필요성이 높아지고 있다.

XHTML을 VoiceXML 혹은 WML 등의 문서로 변환의 특징은 많은 데이터를 갖는 문서를 소량의 데이터를 갖는 여러 개의 문서로 변환해야한다는 것이다. 따라서 이러한 형태의 문서 변환을 수행하기 위해서는 기존 문서를 의미와 문법에 맞는 보다 작은 문서로 분할하고, 분할된 문서를 원하는 형태의 문서로 변환하는 방법이 필요하다. 반대로 VoiceXML 혹은 WML 문서를 XHTML 문서로 변환하기 위해서는 여러 개의 VoiceXML 혹은 WML 문서를 결합해서 하나의 XHTML 문서를 생성해야한다.

본 논문에서는 XML 문서를 분할 혹은 결합을 통해서 다른 XML 문서로 변환해야 하는 필요성을 만족시키기 위해서 XML 문서를

대수학적인 집합으로 표현하는 방법을 소개한다. 대수학적으로 표현된 XML 문서는 괄호와 + 및 * 연산자를 이용해서 표현할 수 있다. + 연산자는 XML 문서의 원소(element)가 느슨하게 연결된 것을 의미하고, * 연산자는 원소들이 강하게 연결된 것을 의미한다. XML 문서에서 * 연산자는 + 연산자에 대해서 배분 법칙을 적용할 수 있기 때문에 배분 법칙을 통해서 XML 문서를 문법에 맞는 작은 여러 개의 XML 문서로 분할할 수 있다. 배분 법칙을 역으로 적용하면 작은 여러 개의 XML 문서를 하나의 XML 문서로 결합하는 것도 가능하다.

분할된 XML 문서는 XSLT[1]와 유사한 변환 규칙에 따라 다른 XML 문서로 변환될 수 있다. 이때 각 XML 응용은 개발자의 의도가 다르기 때문에 다른 XML 응용 문서로 1:1 매핑이 이루어질 수 없고, m:n의 매핑이 이루어진다. 따라서 논문에서는 m:n의 매핑을 처리하기 위해서 그룹핑 방법을 사용한다.

본 논문은 2장에서 관련 연구를 소개하고, 3장에서는 XML을 표현하기 위한 집합과 집합의 원소를 정의한다. 4장에는 XML의 대수학적 정의를 이용한 문서 분할/병합과 문서 변환 방법을 소개하고, 5장에는 논문에서 제시한 방법에 따라 XML 문서를 분할하고 변환하는 예제를 소개한다. 마지막으로 6장에서는 결론과 향후 연구 과제를 밝힌다.

## 2. 관련 연구

### 2.1 Record-Boundary Discovery[2]

문서에서 논리적인 분할 단위를 찾기 위한 Record-Boundary Discovery 방법은 HTML 문서에서 태그들의 사용 패턴을 분석하여 문서의 경계에 사용되는 태그를 정의하고, 문서의 정보 반복을 분석하여 경계를 찾는 방법이다. 이 방법은 HTML에서 나타나는 태그 중에서 HT(Highest-Count Tags)와 IT(Identifiable Separator Tags)를 정의하고, 문서에 나타나는 정보를 SD(Standard Deviation), RP(Repeation-Tag Pattern), OM(Ontology-Matching) 분할 방법으로 나누어 문서에서 경계를 추출한다. 이 방법은 HTML 문서를 효과적으로 분할 할 수 있는 방법을 제공하지만, 사용 패턴을 분석할 수 없는 일반적인 XML 문서에는 적용할 수 없는 문제점을 가지고 있다.

### 2.2 A Formal Data Model and Algebra for XML[4]

마크업 언어를 대수학적으로 정의하기 위한 연구는 XML 문서를 데이터베이스에서 관리하기 위한 XML Query의 연구를 통해 이루어졌다. David[4]는 XML 문서에서 정보를 찾기 위한 Selection과 Joins의 연산을 정의하였다. 또한 찾아낸 정보를 사용자가 원하는 형태로 바꾸기 위해 Distinct, Unorder, Sort 등의 연산을 정의하였다. 이런 대수학적 정의는 문서에서 정보가 가지는 위치를 표현하는 방법이므로 문서가 가지고 있는 문법적인 요소를 포함하고 있지는 않아서, 문서의 문법적 구조를 통한 분석에는 적합하지 않다.

## 3. XML을 표현하기 위한 집합 구조

XML 문서는 태그간의 결합으로 구성되어 있다. 태그간의 결합은 태그의 연속 혹은 하나 이상의 연속된 태그를 포함하는 구조를 갖는다. 따라서 본 논문에서는 태그들의 집합을 이용해서 XML 문서를 표현하는 방법을 사용한다. 집합 TAG는 XML 문서의 태그를 원소로 갖는 집합이다. XML 문서는 태그들의 결합으로 이루어져 있으므로 집합TAG의 원소는 XML 문서를 분석하는 가장 작은 단위이다.

정의 1. 집합 TAG

TAG = {a | a는 태그}

본 논문에서 제안하는 모든 집합은 집합 TAG 원소간의 연산으로 이루진다. 왜냐하면 XML 문서의 기본적인 단위는 태그, 속성, PCDATA인데, 속성과 PCDATA는 하나의 태그에 소속되어 있기 때문에 생략하여도 문서의 구조를 설명하는데 문제가 없다. 그러므로 집합 TAG의 원소는 태그 구조를 설명하는 집합의 가장 작은 단위로 볼 수 있다.

집합 TAG는 정보를 갖지 않는 ε이라는 특별한 태그를 갖고 있다.

정의 2. 집합 TAG의 ε 원소

ε ∈ TAG인 태그가 있다. ε 태그는 정보를 갖지 않는다.

XML 문서는 집합 TAG가 포함하고 있는 원소간의 두 가지 형태의 결합을 통해 이루어진다. 하나는 연속된 태그의 결합이고, 다른 하나는 포함된 태그의 결합이다. 정의 3은 XML 문서에서 태그들의 결합을 연산으로 표현한 것이다. 모든 XML 문서는 정의 3의 두 연산을 통해 표현할 수 있다. 이 연산에서 XML 문서의 구조와는 무관한 PCDATA 내용은 생략된다.

정의 3. 집합 TAG 원소간의 연산

연속된 태그 "<a>P1</a>P2"는 "a · b"이고, 포함된 태그 "<a>P1</a>"는 "(b)"이며, ((a)) = (a)이다. 단, a, b ∈ TAG이고, P1과 P2는 PCDATA이다.

집합 TAG 원소간의 연산을 통해 생성된 식은 집합 TAG의 원소가 아니다. 그러므로 연산으로 생성된 식을 원소로 가지는 집합을 통해 XML 문서들의 특성을 일반적으로 분석할 수 있다. 정의 4의 집합 XA는 집합 TAG 원소에 정의 3의 연산을 적용해서 생성되는 식들의 집합이다.

정의 4. 집합 XA

XA = {x | x = a · b 혹은 x = (a), a, b ∈ TAG 혹은 XA}

모든 XML 문서는 집합 XA의 원소이다. 하지만 이 집합의 모든 원소가 XML 문서는 아니다. 왜냐하면 모든 XML 문서는 가장 상위에 존재하는 하나의 태그를 가져야 하는데 집합 XA에는 그렇지 않은 원소도 존재하기 때문이다. 이러한 원소는 XML 문서를 분석하는데 있어 XML 문서를 작은 단위로 나누고 분석하는 과정에 나타난다. 또한, 집합 XA의 두 연산에 대하여 자유롭게 연산할 수 있고, 다항식은 하나의 루트 태그를 갖는 XML 특성을 제외하면 다른 모든 특성을 만족한다. 그러므로 XML 문서는 집합 XA 중 루트 태그를 하나만 갖는 문서로 정의할 수 있다.

집합 XA의 원소 중에는 XML 문서가 아닌 원소가 존재한다. 그러므로 모든 DTD의 집합 DS와 집합 XA의 부분집합 XAD(d)를 정의하여 DTD에 유효한 집합을 설명한다. 집합 XAD(d)는 집합 XA의 원소 중 어떤 DTD에 유효한 XML 문서를 원소로 갖는 집합이다.

정의 5. 집합 DS

DS = {d | d는 DTD}

정의 6. 집합 XAD(d)

XAD(d) = {x | x ∈ XA이고, x는 d에 유효한 문서, d ∈ DS}

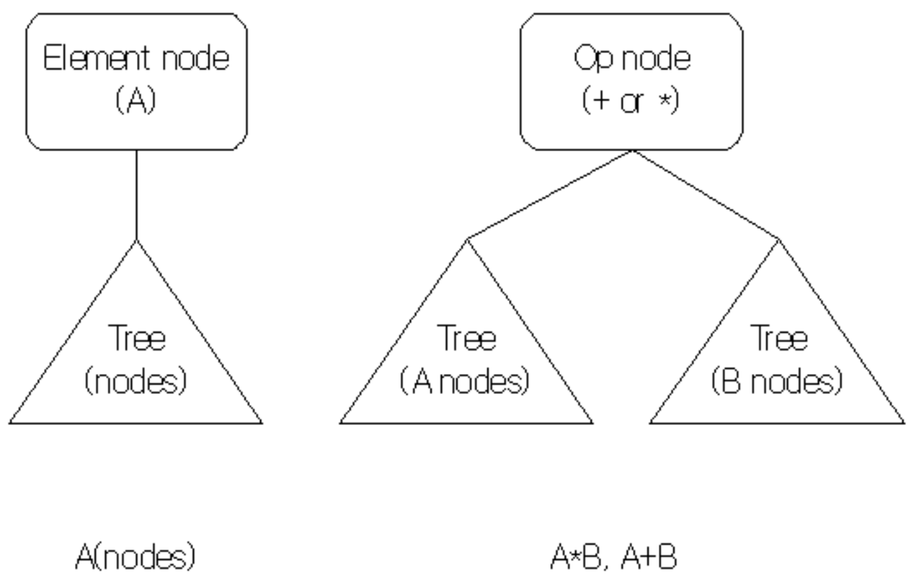
지금까지 정의한 두 연산 중 " · "은 연속된 태그의 결합만을 나타낸다. 그러나 연속된 태그의 결합은 문법적으로 SEQ와 OR 두 가지 종류가 있으므로, DTD에 유효한 문서를 가진 집합 XAD(d)에서 " · " 연산은 두 가지 종류가 있다. 정의 7은 이 두 가지 종류의 연산을 정의하여 문법적으로 다른 두 연산을 구분한다.

집합 XAD는 집합 XA의 원소 중에서 어떤 DTD에 유효한 XML 문서를 원소로 갖는 집합이다. 집합 XAD에서 " · " 연산의 정의는 두 가지로 구분된다. 즉, DTD에서 OR로 되어있는 태그의 연속은 "+"이고 SEQ로 되어있는 태그의 연속은 " * "이다. 이러한 정의는 집합 XAD의 원소가 문법적 특성을 포함하게 한다.

정의 7. 집합 XAD(d)의 연산

d의 "<!ELEMENT a (b,c)>"와 매칭되는 "(b · c)"는 "(b * c)"이고, d의 "<!ELEMENT a (b|c)>"와 매칭되는 "(b · c)"는 "(b+c)"이다.

집합 XAD(d)와 DTD d에 유효한 모든 XML 문서의 집합은 동치이다. 그러므로 집합 XAD(d)만으로 DTD d에 유효한 모든 XML 문서의 특성을 확인할 수 있다. 정의 7의 연산은 (그림 1)과 같은 자료구조로 표현할 수 있다.




(그림 1) 세 연산의 자료구조

하나의 XML 문서는 그림 1과 같은 자료구조를 통해 하나의 루트를 갖는 트리의 형태로 표현한다.

## 4. 문서 연산

### 4.1 문서의 분할과 병합

문서의 분할과 병합은 서로 다른 DTD를 갖는 두 XML 문서를 변환할 때 응용프로그램의 특성에 따라 상대적으로 큰 XML 문서는 분할하고, 작은 XML 문서는 병합하여 정보의 이해가 쉬운 XML 문서로 만드는데 목적이 있다. 문서의 분할이나 병합에 있어 필요한 정보가 누락되거나 경계에 따르지 않는 분할과 병합은 정보의 이해를 더욱 어렵게 하는 요인이 된다. 그러므로 정보의 누락이 없고 단락에 맞는 문서의 분할과 병합이 중요하다.

집합 XAD(d)의 원소는 문법적 구조를 포함하고 있다. 문법적 구조의 특성은 "+"과 " * ", "( )" 연산을 통해 나타난다. 그러므로 이 연산의 특성을 활용하여 문법적 구조에 따른 XML 문서를 분석할 수 있다. 두 연산 "+"와 " * "는 모두 태그가 순차적으로 결합되어 있을 때 표현되는 연산이다. 하지만 "+"는 문법적으로 OR의 의미를 가지고 있고, " * "는 SEQ의 의미를 가지고 있다. OR는 선택적으로 표현될 정보를 기술하는데 사용되고, SEQ는 꼭 표현될 정보를 기술하는 경우 사용된다. 문서 경계의 구분은 문서에서 독립된 정보를 찾아내는 것이므로, 문법적 SEQ로 결합한 태그가 OR로 결합한 태그보다 응집력이 더 높다. 정리 1의 집합 XAD(d)의 배분 법칙은 이런 성질을 이용하여 문서 경계를 구분하는 문서 분할을 가능하게 한다. 지면상 증명은 생략한다.

정리 1. 집합 XAD의 배분 법칙

X ∈ XAD, X=(A*(B+C)) 인 XML 문서 X는 (A*B), (A*C)로 분리될 수 있고, (A*B) ∈ XAD, (A*C) ∈ XAD이다.

집합 XAD(d)의 배분 법칙은 크기가 큰 하나의 XML 문서를 두 개 또는 여러 개의 문서로 분할하며, 분할된 문서는 XAD(d)의 원소이다. 또한 분할된 문서에는 누락된 정보가 전혀 없다. 이렇게 분할된 문서를 형제 문서라고 한다. 배분 법칙을 역으로 적용하면 두 개 또는 여러 개의 문서의 병합 과정도 정보

의 누락이 없이 병합할 수 있다.

(표 1)의 알고리즘은 (그림 1)의 자료구조를 갖는 XML 문서를 분할하는 알고리즘이다.

(표 1) 문서 분할 알고리즘

```
분할_위치_찾기(candidateTag, node)
i = 0
if(node == OpNode)
  if(node.op == op.+)
    if(node. lift_node == candidateTag)
      if(node.right_node == canididateTag)
        candidateList[i] = node
      end_if
      else if(node.right_node == OpNode)
        if(node.rigth_node == op.+)
          candidateList[i] = node
        end_if
      end_else if
    end_if
  end_if
end_if
if(getChildCount(node) == 0)
  exit
end_if
else
  childNode[] = getChild(node)
  for i = 1 to length[childNode]
    listAdd(candidateList,
      분할_위치_찾기(candidateTag, childNode[i]))
end_else

문서_분할()
▷ cPcdata는 분할하고자 하는 문서의 개수
candidateList
분할_위치_찾기(candidateTag, rootNode)
for i = 1 to length[candidateList]
  if(pcdataCount[candidateList[i]] >= cPcdata)
    ▷ candidateList[i]위치에서 문서를 나눈다.
  end_if
end_for
▷ 분할할 수 없는 문서
```

## 4.2. 문서 변환

서로 다른 DTD를 가진 두 XML 문서의 변환은 태그간의 변환이고, 두 XML의 태그는 사용목적과 응용프로그램의 특성에 맞게 정의되어 있어 1:1 변환이 아닌 m:n으로의 변환이 수행되어야 한다. 변환 단위는 XML 문서의 일부분이고, 앞에서 정의한 집합 XA의 특성과 일치한다.

변환 함수를 정의함에 있어 이 함수의 정의역과 공역을 정의하여야 하며, 정의역과 공역은 XML 문서의 일부분이 된다. 본 논문은 XML 문서의 일부분을 부분 문서라 한다. 부분 문서는 DTD d에 나타나는 태그들이 d에 기술된 연산으로 생성할 수 있다. 그러므로 d에 나타나는 태그들을 원소로 갖는 집합 XADT(d)의 연산으로 부분 문서를 설명한다. 집합 XADT(d)는 집합 TAG의 부분 집합이고, 집합 DS의 한 원소 d에 나타나는 태그들의 집합이다.

정의 8. 집합 XADT(d)

XADT(d) = {t | t ∈ TAG, t는 DTD d에 나타나는 태그}

집합 XADT(d)의 원소가 d에 나타나는 연산으로 결합하면, 함수에서 사용할 수 있는 정의역과 공역을 정의할 수 있다. 집합 XADsub(d)는 집합 XADT(d)의 연산을 통해 정의역과 공역으로 정의되는 집합이다.

정의 9. 집합 XADsub(d)

XADsub(d) = {s | s = t1 + t2 or t1 * t2 or (t1) t1 and t2 ∈ XADT(d), s는 XAD의 한 원소의 단항 이거나 다항 and '+', '*', '( )'는 DTD에 있는 연산}

집합 XADsub(d)의 한 원소는 XAD(d)의 한 원소가 나타내는 문서의 부분문서가 된다. DTD x의 집합 XADsub(x)와 DTD y의 집합 XADsub(y)의 순서쌍을 정의하여 관계를 만들고, 집합 XADsub(x)의 모든 원소가 정의역이 되면 이 관계는 함수가 된다. 변환 함수에서 삭제되어야만 하는 태그는 ε태그와 순서쌍을 만들어 삭제되게 된다. 이 변환 함수는 각 XML 태그의 특성에 따라 정의되어야 하며 언

어의 의미적 요소를 분석하여야만 가능하다.

변환 함수는 XML 문서와 DTD만으로 생성하기 어렵다. 변환은 각 태그의 의미하는 바에 따라 변환 함수를 정의해야 하는데 XML 문서와 DTD만으로는 각 태그가 의미에 관한 정보가 미약하기 때문이다. 또한 DTD에 나타나는 태그는 표준이 없어 DTD의 순서쌍에 따라 각기 다른 변환 함수를 정의하여야 한다. 그러므로 변환 함수는 각 경우에 따라서 다르게 다루어져야 한다.

(표 2) 문서변환 알고리즘

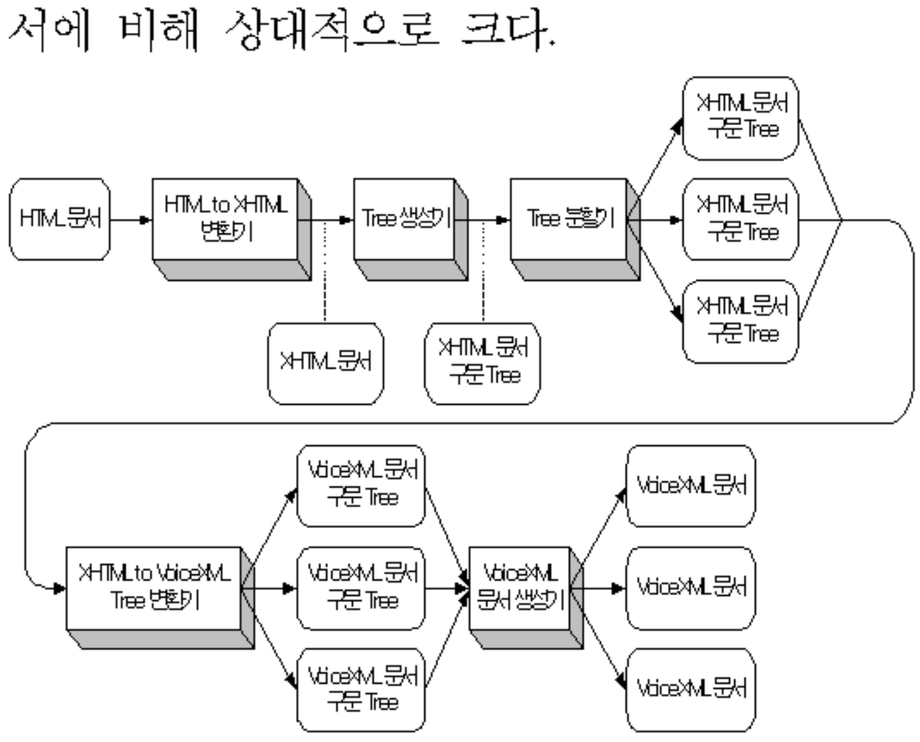
```
태그변환(AXmlNode, BXmlNode)
if(AXmlNode = op)
    태그변환(AXmlNode.lift_node, BXmlNode)
    태그변환(AXmlNode.right_node, BXmlNode)
end_if
else if(AXmlNode = element || BXmlNode = pcdata)
    for i ← 1 to length[X]
        if(AXmlNode = X[i])
            VXmlNode ← Y[i]
        end_if
        else
            /* 다른 변환 관계 적용 */
            break
        end_else
        AXmlNode ← AXmlNode.child
        BXmlNode ← BXmlNode.child
    end_for
    if(BXmlNode ≠ pcdata)
        태그변환(AXmlNode, BXmlNode)
    end_if
end_else if

문서변환(AXmlRootNode)
if(AXmlRootNode = html)
    ▷ BXmlRootNode 생성
    태그변환(AXmlRootNode.child,
BXmlRootNode.child)
end_if
else
/* DTD에 유효한 문서가 아님.*/
end_else
```

XML 문서는 경우에 맞는 변환 규칙과 (표 2)의 알고리즘을 이용하여 변환한다.

## 5. XML 문서 변환 예

마크업 언어의 대수학적 변환 방법을 통해 XML 문서를 변환하는 것을 실제로 예를 들어 보이기 위해 HTML 문서를 VoiceXML로 변환하는 경우를 생각해보자. HTML은 웹 브라우저를 이용해서 시각적인 정보를 제공하는 마크업 언어이고, VoiceXML은 전화와 같은 음성 정보를 전달하기 위한 XML 응용이다. 두 언어의 정보 전달 매체는 비교하면 HTML은 시각적인 매체이고, VoiceXML은 청각적인 매체를 사용한다. 즉, 정보의 내용은 같으나 응용프로그램에서 사용하는 DTD가 다르게 기술되어 문서 변환이 필요하다. 하지만 VoiceXML은 청각적 매체가 시간적인 제약을 가지는 반면 HTML은 시간적인 제약을 받지 않는다. 그래서 HTML 문서가 VoiceXML 문서에 비해 상대적으로 크다.




(그림 2) HTMLtoVoiceXML 시스템 구조도

문서의 변환에는 XHTML DTD의 분석이 필요하고 그 분석을 바탕으로 분할 가능한 태그를 찾아야 한다. (표 3)은 XHTML DTD의 일부로 이 DTD의 분석을 통해 "<table>" 태그가 분할 가능 태그임을 알 수 있다.

문서를 변환하기 위해서 HTML은 XML 응용이 아니기 때문에 초기에 HTML 문서를 XHTML 문서로 변환하고, 이것을 VoiceXML로 변환하는 방법을 사용한다. 그 다음에는 XHTML 문서와 DTD를 이용해서 분할 가능한 트리를 생성한다. 생성한 트리를 분할하고

변환 관계를 이용하여 문서를 변환한다. 즉 HTML 문서를 VoiceXML 문서로 변환하는 과정은 (그림 2)와 같다.

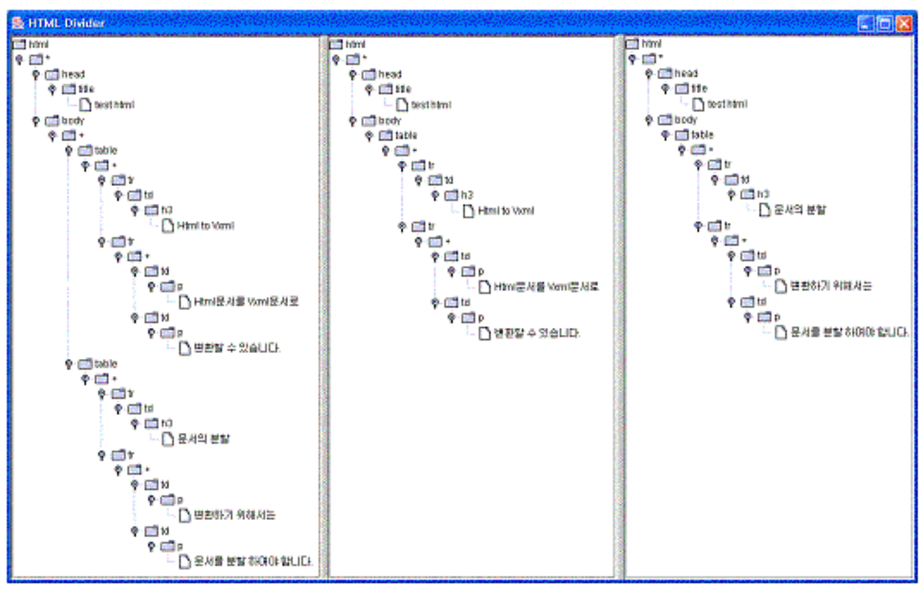
(표 3) XHTML DTD의 일부

```
http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd
<!ENTITY % block "p | %heading ? >
<!ENTITY % Block "(%block; | form | %misc;)*">
<!ELEMENT html (head, body)>
<!ELEMENT head (%head.misc;, ((title, %head.misc;,
(base, %head.misc;)?| (base, %head.misc;)))>
<!ELEMENT title (#PCDATA)>
<!ELEMENT body %Block;>
<!ENTITY % Block "(%block; | form | %misc;)*">
<!ENTITY % block "p | %heading; | div | %lists; |
%blocktext; | fieldset | table"
```

(표 4) 변환할 XHTML 문서

```
<html>
<head><title>test html</title></head>
<body>
<table><tr><td><h3>Html to Vxml</h3></td></tr>
 <tr><td><p>Html문서를 Vxml문서로</p></td>
 <td><p>변환할 수 있습니다.</p></td><td>
  <a href = "soongsil.ac.kr">숭실대학교</a>
 </td></tr></table>
<table><tr><td><h3>문서의 분할</h3></td></tr>
 <tr><td><p>변환하기 위해서는</p></td>
  <td><p>문서를 분할 하여야 합니다.</p></td>
 </tr></table></body>
</html>
```

(표 4)의 문서 내용을 분할하는 예는 (그림 3)과 같이 트리로 나타난다.



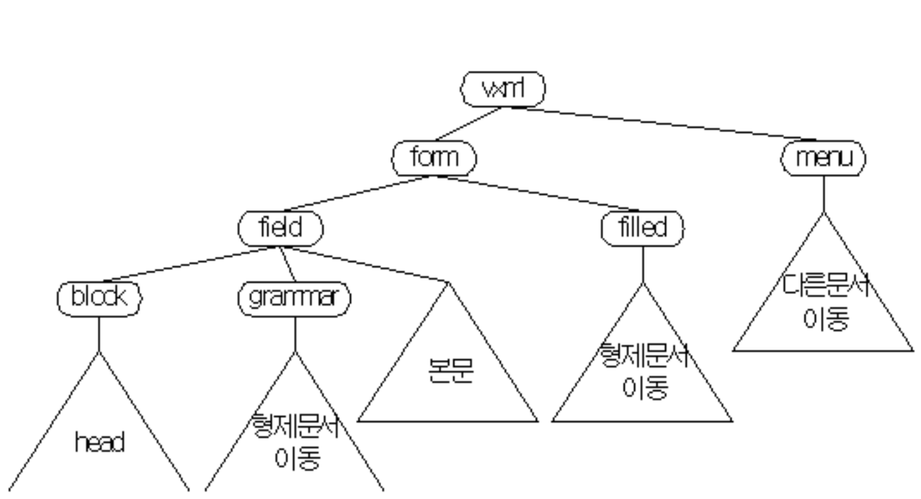
(그림 3) 분할된 HTML 문서

(표 4)는 DTD에 합당한 XHTML 문서이다. 이 문서는 "<table>" 태그를 이용한 문서이며 한 테이블은 문서의 경계이므로 분할하여야 한다. 그러므로 "<table>" 태그를 분할 후보 태그로 하여 문서를 분할할 수 있다.

(그림 3)에서 보인 트리는 XHTML 문서를 분할 가능한 트리의 형태로 나타낸 것이고, 이 트리를 이용하여 두 개의 트리로 분할할 수 있다. 분할된 각각의 트리는 각각 하나의 문서로 변화될 수 있으며, 각 트리를 변환하여 정보의 이해가 쉬운 문서를 생성할 수 있다.

문서의 변환은 사용자가 HTML 문서에 표현되어 있는 정보의 내용을 음성으로 듣는 것을 목적으로 하고 있고, HTML 문서의 특성상 문서에는 다른 문서로의 링크가 존재한다. 또한 한 HTML 문서는 다수의 VoiceXML 문서로 변환되기 때문에 형제 문서로의 이동도 고려되어야 한다. 그러므로 VoiceXML 문서의 출력은 HTML 문서의 내용을 읽어 HTML 문서를 분할하고 각 형제문서를 VoiceXML 문서로 변환하여 출력하고 사용자의 요구에 따라 형제문서로의 이동과 다른 문서로의 이동이 이루어져야 한다.

사용자가 이해하기 쉬운 VoiceXML 문서를 생성하기 위해 앞의 시나리오에 알맞는 문서를 생성하여야 하는데 그 문서의 구조는 본문 변환 내용과 형제 문서의 링크, 다른 문서로의 링크로 구성되어야 한다. VoiceXML에서 제공하고 있는 태그의 정의를 이용하여 이런 문서의 구조와 시나리오를 지니는 문서를 생성하면 (그림 4)와 같은 트리의 형태이다.




(그림 4) 생성될 VoiceXML 문서의 구성

(그림 4)는 생성될 형제 문서의 구조를 보여 주고 있다. 문서 변환을 위해 변환 함수를 정의할 때, 변환 함수는 (그림 4)와 같은 구조의 문서를 생성하는 관계로 정의하여야 한다. (표 5)은 앞의 구조와 같은 문서를 생성하는 변환 함수의 일부분이다.

(표 5) 적용될 변환 규칙

| XHTML 태그 | VoiceXML 태그 |
|---|---|
| <html>node</html> | <vxml><form id="id name"> node</form></vxml> |
| <head><title>PCDATA </title></head> | <block><prompt>PCDATA </prompt></block> |
| <body>node</body> | <field id ="id name">node</field> |
| PCDATA | <prompt>PCDATA</prompt> |
| <img src="..."></img> | ε |
| ... | ... |

변환 함수는 각 태그에 대하여 어떻게 변환하여야 하는지를 표현하고 있다. 한 예로 " " 태그는 XHTML에서 사용자에게 문자열을 출력하는 태그이다. 그러므로 VoiceXML에서는 "<prompt>"로 변환하여 사용자에게 음성으로 문자의 내용을 전달하여야 한다. 또한 "<img>" 태그는 사용자에게 그림을 보여주는 태그이다. 하지만 음성을 통해서는 그림 이미지를 전달할 방법이 없기 때문에, 이 태그는 ε으로 대응시켜 삭제한다.

(표 6)은 생성된 VoiceXML문서 중 하나이다. 이 문서는 사용자에게 XHTML 문서의 내용을 전달하고 형제 문서로의 링크가 이루어졌으며 XHTML 문서의 링크 또한 이루어져있다.

생성된 VoiceXML 문서에서 형제 문서의 이동은 내용이 출력되는 동안의 DTMF 입력으로 이루어져 있고 XHTML 문서에서 링크는 문서가 끝난 후 DTMF 입력을 통해 표현되어 있다. 생성된 문서는 음성 매체를 이용하는 사용자에게 정보의 이해가 쉬운 크기이다. 다른 XML 문서의 변환 또한 각 매체에 맞는 길이로 생성되어야 하며 태그의 의미에 맞는 변환이 이루어져야 한다.

(표 6) 생성된 VoiceXML 문서

```
<vxml version="2.0">
<form id="brother0">
    <block><prompt>test html</prompt></block>
    <field name="body">
        <grammar>1번</grammar>
        <prompt>Html to Vxml</prompt>
        <prompt>Html문서를 Vxml문서로</prompt>
        <prompt>변환할 수 있습니다.</prompt>
        <prompt>숭실대학교</prompt>
    </field>
    <filled>
        <if cond="brother0=='1번'"><goto next="test1.vxml"/></if>
        <goto next="test1.vxml"/>
    </filled>
</form>
<menu>
    <prompt>0번숭실대학교입니다.</prompt>
    <choice dtmf="0" next="soongsil.ac.kr"/>
</menu></vxml>
```

## 6. 결론

XML은 점차 넓은 분야에서 다양한 목적으로 사용되기 때문에 XML로 표현된 데이터를 다른 형태의 XML 문서로 변환해야할 필요성이 높아지고 있다. 기존에 마크업 언어로 작성된 문서를 변환하기 위한 연구들이 수행되었지만, 일반적인 XML 문서에는 적용되기 어려운 문제점들을 가지고 있었다. 이러한 문제를 해결하기 위해서 본 논문에서는 XML 문서를 대수학적 방법으로 표현하고, 연산을 이용해서 문서를 분할하는 방법을 소개하였다.

서로 다른 DTD를 갖는 XML 문서의 변환을 위해서 XML 문서의 태그간의 관계를 대수학적인 집합을 이용해서 표현하고, 그 집합에 적용될 수 있는 연산을 정의하고, 연산의 특성을 찾아내는 방법을 사용하였다. 이 연산은 XML 문서가 생성된 DTD의 구조를 요소로 포함하고 있기 때문에 분할 시 정보의 결합도가 높은 부분을 DTD에 유효한 여러 개의 문서로 분할할 수 있다. 즉, 집합 XAD의

정의는 XML 문서를 분석에서 XML 문서만으로 분석하는 기존의 방법을 벗어나 DTD 문서에서의 정의를 문서 분석에 포함하였다.

집합 XA의 정의는 변환을 수학적인 관계로 정의할 수 있고, 그 관계의 정의역의 정의에 따라 관계가 함수로 정의할 수 있음을 보였다. 이는 응용프로그램의 특성에 맞는 XML 문서를 생성하고 이 문서는 정보의 이해가 쉬운 장점이 있다. 또한 기존의 통계적인 방법이나 사용패턴의 연구에서 벗어나 새로운 표준의 등장과 함께 문서를 분할할 수 있는 방법을 제시하였다. 하지만 XML 개발자의 의도와 부합하는 태그의 의미를 분석할 수 있는 방법이 없어 문서 변환기를 자동화로 생성하지 못하였고, 각 XML 변환을 위한 변환 함수를 검증할 수 있는 방법을 제시하지 못하였다. 또한, 집합 XA는 대수학적인 항등원의 정의가 미비하고 역원의 정의가 없어 체나 군에 속하지 못하는 집합이다. 이런 두 정의가 이루어져 체나 군의 집합에 속하게 된다면 그 집합들이 가지고 있는 정리를 사용하여 XML 문서의 정보를 연구하는데 더 많은 도움을 줄 것이다.

## 참고문헌


[1] XSL Transformations (XSLT) Version 1.0 http://www.w3.org/TR/1999/REC-xslt-19991116.

[2] David W. Embley, Y. S. Jiang, Yiu-Kai Ng, "Record-Boundary Discovery in Web Documents", SIGMOD Conference, pp. 467-478, 1999.

[3] Voice Extensible Markup Language (Voice XML) Version 2.0, http://www.w3c.org/TR/2001/CR-voicexml20-20030220.

[4] David Beech, Ashok Mallhotra, Michael Rys, A Formal Data Model and Algebra for XML, http://www-db.stanford.edu/dbseminar/Archive/FallY99/malhotra-slides/malhotra.pdf, 1999.



곽 동 규

2002년 서경대학교 응용수학과 학사

2002년-현재 숭실대학교 컴퓨터학과 석사과정

관심분야 : 프로그래밍언 어, XML.

최 종 명

1992년 숭실대학교 전자계산학과 학사

1996년 숭실대학교 전자계산학과 석사

2003년 숭실대학교 컴퓨터학과 박사

관심분야 : 시각 프로그래밍, 멀티패러다임시스템, XML

유 재 우

1976년 숭실대학교 전자계산학과 학사

1985년 한국과학기술원 전산학과 박사

1983년~현재 숭실대학교 컴퓨터학부 교수

1986~87년, 1996~97년 코넬대학교, 피츠버그대학교 객원교수

1999~2000년 한국정보과학회 프로그래밍언어 연구회 위원장

관심분야 : 프로그래밍언어, 컴파일러, 인간과 컴퓨터 상호작용