문서 영상의 논리적인 구조 분석을 위한 구문론적인 접근 방식

(A Syntactic Approach for Logical Structure Analysis of Document Images)

이 경 호 † 최 윤 철 †† 조 성 배 ††
(Kyong Ho Lee) (Yoon Chul Choy) (Sung Bae Cho)

요 약 본 논문에서는 다수의 페이지로 구성된 복잡한 구조의 문서로부터 SGML/XML에 기반한 전자 문서를 생성하기 위한 구문론적인 구조분석 방법을 제안한다. 특히 제안된 파싱 기법은 텍스트 라인을 기본 단위로 하는 기존 연구보다 논리적인 계층 구조를 보다 정확하고 빠르게 생성하기 위하여 텍스트 영역의 계층적인 트리 구조를 입력으로 받아들인다. 또한 문서 유형의 논리적인 구조 정보와 기하적인 특성을 효과적으로 기술할 수 있는 문서 모델을 정의하고, 이의 자동 생성과 점증적인 학습 방법을 제안한다. 제안된 방법의 성능을 평가하기 위하여 과학 기술 논문으로부터 스캐닝한 372개의 논문 영상으로 실험한 결과, 제안된 방법은 기존 연구와 달리 다수의 문서 영상으로 구성된 문서에 대하여 논리적인 구조분석과 문서 모델의 자동 생성을 효율적으로 지원하였다. 특히 제안된 방법은 논리적인 구조분석의 최종 결과로서 SGML/XML 문서를 생성하기 때문에 문서의 재 사용성과 호환성을 높인다.

Abstract This paper presents a syntactic method for sophisticated logical structure analysis which transforms document images with multiple pages and hierarchical structure into an electronic document based on SGML/XML. To produce a logical structure more accurately and fastly than previous works of which the basic units are text lines, the proposed parsing method takes as input text regions with hierarchical structure. Furthermore, we define a document model that is able to describe geometric characteristics and logical structure information of document class efficiently and present its automated and incremental learning method. Experimental results with 372 images scanned from technical journal documents show that the method has performed logical structure analysis successfully and generated a document model automatically. Particularly, the method generates SGML/XML documents as the result of structural analysis, so that it enhances the reusability of documents and independence of platform.

1. 서 론

인터넷과 전자도서관의 급속한 확산에 따라 기존의 종이를 이용한 정보 매체에서 전자 문서로의 전환이 가속화되고 있다. 그러나 전자 문서의 활발한 보급에도 불구하고 종이 문서의 양도 급속도로 증가하고 있다. 이는 인간이 기본적으로 종이 형태의 문서를 선호함을 반영하는 것이다. 그러나 종이 문서는 전자 문서와 비교하여 저장, 검색, 갱신, 그리고 전송 등의 문서 처리의 다양한 면에 있어서 비효율적이다.

한편 SGML(standard generalized markup language) [1]과 XML (extensible markup language) [2]은 논리적인 계층 구조를 표현할 수 있다는 장점 때문에 전자도서관과 인터넷 등에서 전자 문서의 표준 포맷으로 자리 잡았다. 따라서 문서 영상으로부터 논리적인 구성 요소를 추출하여 SGML/XML 문서를 생성하는 논리적인 구조분석 방법의 개발은 매우 중요하다.

일반적으로 인간은 문서를 구성하는 텍스트 영역의

† 정 회 원 : NIST 연구원
kyongho@nist.gov
†† 종신회원 : 연세대학교 컴퓨터과학과 교수
ycchoy@rainbow.yonsei.ac.kr
sbcho@csai.yonsei.ac.kr
논문접수 : 2000년 10월 14일
심사완료 : 2001년 5월 17일

기하적인 특성으로부터 제목 또는 단락 등의 논리적인 구성 요소를 식별하고, 이를 병합하여 절 구조와 같은 복합적인 구성 요소를 식별함으로써 문서의 논리적인 계층 구조를 인식한다. 이와 같이 텍스트 영역의 기하적인 특성으로부터 직접적인 식별이 가능한 논리적인 요소를 주 구조(primary structure)라고 하며 이미 식별된 다수의 구성 요소들을 병합함으로써 추출 가능한 구성 요소를 부 구조(secondary structure)라고 한다[3]. 따라서 문서 영상으로부터 SGML/XML 문서를 자동 생성하기 위해서는 주 구조는 물론이고 부 구조에 대한 논리적인 구조분석이 이루어져야 한다.

그러나 기존 연구의 대부분[4],[5],[6],[7],[8]은 주 구조에 해당하는 구성 요소만을 추출하기 때문에 계층적인 구조 정보를 지원하지 않는다. 또한 주로 단일의 문서 영상을 처리 대상으로 하기 때문에 다수의 문서 영상으로 구성된 복잡한 구조의 문서를 지원하지 않는다[9],[10],[11].

한편 문서 영상의 논리적인 구조분석을 위해서는 문서 유형에 대한 지식을 표현한 문서 모델이 요구된다. 문서 영상으로부터 논리적인 계층 구조의 효과적인 추출을 위해서는 문서 유형의 기하적인 특성과 논리적인 계층 구조에 대한 다양한 정보를 표현할 수 있는 문서 모델과 실험 영상으로부터 문서 모델을 자동 생성할 수 있는 방법이 요구된다. 그러나 기존 연구의 대부분은 단순한 수준의 문서 모델을 제공하거나 문서 모델의 자동 생성을 지원하지 않는다[5],[6],[7].

따라서 본 논문에서는 다수의 문서 영상으로 구성된 복잡한 구조의 문서를 대상으로 논리적인 구조분석을 위한 구문론적인 방법을 제안한다. 일반적으로 문서 영상을 구성하는 텍스트 영역은 각각 제목 또는 단락 등에 해당하는 헤더(header) 또는 바디(body)로써의 기능을 한다. 특히 헤더와 바디는 해당 텍스트 영역의 기하적인 특성에 따라 다양한 종류로 구별된다. 본 논문에서는 헤더와 바디를 문서의 기능 구성 요소로 정의하고, 다양한 종류의 헤더와 바디로 구성된 계층적인 트리 구조를 기능 구조 트리(functional structure tree)라고 정의한다.

제안된 방법은 구조분석의 정확성과 처리 속도의 향상을 위하여 기존 연구와 달리 텍스트 라인이 아닌 기능 구조 트리에 제안된 문서 모델에 기반한 파싱 기법을 적용하여 각각의 노드에 레이블이 부여된 논리 구조 트리(logical structure tree)를 생성한다. 이를 위하여 기하적인 특성이 유사한 인접한 텍스트 라인을 병합하여 헤더와 바디의 순차적인 집합을 생성하고, 헤더의 반복적인 특성에 기반하여 순차적인 집합을 반복 분할하면서 기능 구조 트리를 하향식으로 생성한다.

한편 본 논문에서는 문서 모델을 효율적으로 표현할 수 있는 언어인 DSDL (document structure description language)을 제안한다. DSDL은 논리적인 계층 구조를 기술하기 위하여 문서 유형이 포함할 수 있는 주 구조의 기하적인 특성은 물론이고 부 구조가 포함할 수 있는 구성 요소의 종류와 순서 그리고 빈도수 등에 대한 다양한 정보를 기술한다. 특히 본 논문은 문서 모델의 자동 생성과 더불어 기존 모델에 새로운 종류의 문서 구조를 반영할 수 있는 점증적인 학습(incremental learning) 방법을 제안한다. 또한 문서 모델에 포함된 논리적인 구조 정보와 기하적인 특성으로부터 각각 SGML DTD (document type definition)와 DSSSL(document style semantics and specification language) [12] 스타일 시트를 생성한다.

제안된 방법의 성능을 평가하기 위하여 IEEE Transactions on Pattern Analysis and Machine Intelligence(TPAMI)로부터 스캐닝한 372개의 논문 영상으로 실험한 결과, 제안된 방법은 기존 연구와 달리 다수의 문서 영상으로 구성된 문서에 대하여 논리적인 구조분석과 문서 모델의 자동 생성을 효율적으로 지원하였다. 특히 제안된 방법은 논리적인 구조분석의 최종 결과로서 SGML/XML 문서를 생성하기 때문에 문서의 재 사용성과 호환성을 지원한다.

본 논문의 구성은 다음과 같다. 2절에서는 관련 연구를 통하여 문서 영상의 논리적인 구조분석 방법에 대한 기존의 연구 결과를 간략히 기술한다. 3절에서는 문서 모델을 기술하기 위하여 정의된 DSDL을 자세히 설명한다. 4절에서는 제안된 구조분석 방법을 기능 구조 트리의 생성, 논리 구조 트리의 생성, 그리고 SGML 문서 생성의 세 단계로 구분하고, 각 단계에 대한 자세한 설명을 기술한다. 5절에서는 문서 모델의 자동 생성 및 점증적인 학습 기법을 소개하고, 6절에서는 실험 결과를 통하여 제안된 방법의 성능을 기존 연구와 비교 및 분석한다. 마지막으로 7절에서는 결론 및 향후 연구 방향을 기술한다.

2. 관련 연구

일반적으로 문서 영상을 전자 문서로 변환하기 위한 문서 영상의 분석 및 이해(document image analysis and understanding)에 관한 연구는 문서 영상으로부터 텍스트 라인, 그림, 테이블 등을 분할 및 식별하는 기하적인 구조분석과 이로부터 논리적인 계층 구조를 추출하는 논리적인 구조분석의 두 단계로 구성된다[13], [14]. 한편 기하적인 구조분석과 달리 논리적인 구조분석에 관한

연구 결과는 미비한 실정이며 최근 들어 전자 문서의 활발한 보급에 힘입어 이에 대한 관심이 증가하고 있다.

일반적으로 논리적인 구조분석에 관한 기존 연구는 크게 인공지능 기법에 기반한 모델 매칭(model matching) 방법과 파싱 기법에 기반한 구문론적인 방법의 두 가지로 구분되며, 표현 범위와 방법은 다르지만 문서 유형에 대한 지식을 표현하기 위하여 문서 모델을 사용한다. 본 절에서는 <표 1>과 <표 2>와 같이 논리적인 구조분석에 관한 기존의 연구 결과를 모델 매칭 방법과 구문론적인 방법으로 구분하고, 추출 가능한 구조의 종류, 문서 모델의 표현 범위, 그리고 문서 모델의 자동 생성 여부의 세 가지 측면에서 각각의 특징과 문제점을 간략히 기술한다.

기존 연구의 대부분은 주로 단일의 문서 영상을 대상으로 단순한 수준의 구조 정보를 추출한다. 따라서 다수의 문서 영상으로 구성된 복잡한 구조의 문서에 대한 정교한 수준의 논리적인 구조분석 방법의 개발이 요구된다. 이를 위해서는 주 구조에 해당하는 텍스트 영역의 기하적인 특성은 물론이고 부 구조가 포함할 수 있는 구성 요소의 종류와 순서 그리고 반복 횟수 등에 대한 다양한 정보를 기술할 수 있는 문서 모델이 요구된다.

표 1 성능 평가 기준

기준 / 기호	추출 가능한 구조의 종류	문서 모델의 표현 범위
△	주 구조	구체적인 표현 기법 없이 단순한 규칙 또는 일반적인 특성
□	제한된 수준의 계층 구조	논리적인 구성 요소의 기하적인 특성을 기술하며 논리적인 계층 구조를 포함하지 않음
○	주 구조+부 구조	제한적인 수준의 기하적인 특성과 논리적인 계층 구조 기술
◉	다수의 문서영상으로 구성된 문서에 대한 계층 구조	다양한 종류의 기하적인 특성과 구성 요소 사이의 관계 및 빈도수 등을 포함한 논리적인 계층 구조 기술

3. 문서 모델

본 절에서는 문서 유형의 기하적인 특성과 논리적인 계층 구조에 대한 문서 모델을 기술할 수 있는 언어인

표 2 논리적인 구조분석 방법

구분	연구	연도	특 징	구조 종류	문서모델 표현범위	문서모델 자동생성	처리 대상
구문론적인 방법	[5]	1992	화소의 길이와 빈도수 등의 기하적인 특성을 문서 문법(document grammar)으로 기술	□	○	×	논문
	[6]	1993					
	[15]	1992	논리적인 구성 요소의 상대적인 위치 등을 기술	□	○	×	목차
	[16]	1993	퍼지 논리(fuzzy logic)에 기반하여 오류 허용하며 비교 함수를 이용하여 유사한 것을 선택	◉	◉	×	책
	[17]	1993	페이지 라인의 크기와 유형 및 정렬 방식 등의 기하적인 특성과 공간 관계 등을 페이지 문법(page grammar)으로 기술	○	◉	×	책
	[18]	1994	텍스트 라인 사이의 연결관계에 해당하는 그래프로부터 최소 비용의 패스를 탐색하며 문법은 인접한 레이블의 기하적인 특성과 순서에 대한 확률 값을 포함	△	□	×	책 등
	[19]	1997	전자문서를 SGML 문서로 변환하기 위하여 SGML DTD에 기반한 문법인 DSD(document structure description)제안	◉	◉	○	전자문서
	[20]	1999					
모델매칭방법	[7]	1996	기하적인 특성에 대한 규칙과 레이아웃 모델에 기반	□	○	×	논문
	[8]	1996	레이아웃에 대한 지식과 제어 과정을 규칙 형태로 기술	□	○	×	신문
	[9]	1988	논리적인 계층 구조와 상대적인 크기와 위치에 대한 특성을 결정 트리(decision tree)로 기술	○	○	○	서신
	[10]	1992					
	[11]	1992	기하 구조 트리의 변환을 위하여 단 유형에 대한 일반적인 특성을 4개의 변환 규칙으로 정의	□	△	×	신문
	[21]	1997	흰 공백의 크기와 위치 정보를 이용하여 계층 구조를 생성하며 레이블의 식별을 위하여 프로토타입 기술	○	○	×	논문
	[22]	1998					
	[23]	1999	계층 구조에 대한 통계적인 확률 분포를 기술한 문서 모델인 N-Gram 제안	○	○	○	신문
	[24]	1997	목차의 구조에 대한 일반적인 지식을 이용	□	△	×	목차
	[25]	1998	추출 대상에 대한 기하적인 특성 기술	△	□	×	서신 등
	[26]	1995	기하적인 특성, 빈도수, 그리고 어휘 정보 기술	△	□	×	서신
	[27]	1999	술어논리(predicate logic)로 기술한 기하적인 지식을 이용하여 객체를 식별하며 문자 인식 결과로부터 계층 구조 생성	○	□	×	매뉴얼

DSDL을 자세히 기술한다. DSDL은 제안된 논리 구조 트리에서 중간 노드에 해당하는 부 구조 각각에 대하여 이에 포함될 수 있는 구성 요소의 종류, 순서, 그리고 반복 횟수 등에 대한 정보를 정규 수식으로 표현한다. 본 논문에서는 이를 해당 부 구조의 내용 모델(content model)이라고 정의한다.

또한 주 구성 요소 각각에 대하여 이를 만족하는 텍스트 영역의 기하적인 조건으로서 단 유형, 텍스트 라인의 숫자와 높이, 텍스트 영역의 전후 여백, 검은 화소의 밀도 분포, 그리고 정렬 방식 등에 관한 특성을 기술한다.

```
<!ELEMENT Document          (Title, Author, Affil, Abstract, Keyword, Sec-Body)>
<!ELEMENT Title             #( FUNTION_TYPE: HEADER
                               COLUMN_TYPE: SINGLE
                               MIN_LINE_HEIGHT: 31
                               MAX_LINE_HEIGHT: 42
                               MIN_LINE_NUMBER: 1
                               MAX_LINE_NUMBER: 3
                               MIN_SPACE_BEFORE: 105
                               MAX_SPACE_BEFORE: 120
                               MIN_SPACE_AFTER: 60
                               MAX_SPACE_AFTER: 70
                               MIN_BLACK_PIXEL_DENSITY: 0.307
                               MAX_BLACK_PIXEL_DENSITY: 0.525
                               JUSTIFY: CENTER)>
<!ELEMENT Author            #( ... )>
<!ELEMENT Affil             #( ... )>
<!ELEMENT Abstract          #( ... )>
<!ELEMENT Keyword           #( ... )>
<!ELEMENT Sec-Body          (Section*, Reference )>
<!ELEMENT Section           (Sec-Header, Paragraph*, Sub-Section*)>
<!ELEMENT Sub-Section       (Sub-Sec-Header, Paragraph*, Sub-Sub-Section*)>
<!ELEMENT Sub-Sub-Section   (Sub-Sub-Sec-Header, Paragraph*)>
<!ELEMENT Reference         (Sec-Header, Ref-Item*)>
<!ELEMENT Sec-Header        #( ... )>
<!ELEMENT Sub-Sec-Header    #( ... )>
<!ELEMENT Sub-Sub-Sec-Header #( ... )>
<!ELEMENT Paragraph         #( ... )>
<!ELEMENT Ref-Item          #( ... )>
```

그림 1 문서 문법의 예

DSDL은 <그림 1>과 같이 기호 "#"을 사용하여 부 구조의 내용 모델과 주 구조의 기하적인 특성 정보를 구별한다. 특히 본 논문에서는 DSDL에 의하여 기술된 문서 모델을 문서 문법(document grammar)이라고 정의한다. 한편 SGML/XML에서 문서 유형에 대한 논리적인 구조 정보는 DTD에 의하여 기술된다. 따라서 제안된 방법은 논리적인 구조 정보와 더불어 기하적인 특성을 기술하기 위하여 <그림 2>와 같이 SGML/XML DTD의 엘리먼트(element) 선언 방법을 확장하여 DSDL을 정의한다. 특히 DSDL은 SGML/XML과 마찬가지로 결정적(deterministic) [28] 내용 모델만을 허용한다. 예를 들어, 두 개의 그룹으로 구성된 비 결정적 내용 모델 ((b, c) | (b, d))에 대하여 파서는 입력 값 b로부터 적절한 그룹을 선택할 수 없다. 따라서 기존의 연구 결과 [29] 등을 적용하여 결정적인 내용 모델 (b, (c | d))로 변경되어야 한다.

```
element_declaration ::= "<ELEMENT", element_type (content_model | geometric_property), ">"
element_type ::= generic_identifier
content_model ::= model_group
model_group ::= "(", content_token (connector, content_token)*, ")", occurrence_indicator?
content_token ::= element_token | model_group
element_token ::= generic_identifier, occurrence_indicator?
generic_identifier ::= name
name ::= name_start_character, name_character*
name_character ::= name_start_character | digit
name_start_character ::= "A" | . . | "Z" | "a" | ... | "z"
digit ::= "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
connector ::= "," | "|" | "&"
occurrence_indicator ::= "?" | "+" | "*"
geometric_property ::= "#", "(", functional_structure_type, column_type, line_height, line_number,
space_before, space_after, black_pixel_density, justification, ")"
functional_structure_type ::= "FUNCTION_TYPE:", funtional_type_indicator
funtional_type_indicator ::= "HEADER" | "BODY"
column_type ::= "COLUMN_TYPE:", column_type_indicator
column_type_indicator ::= "SINGLE" | "DOUBLE"
line_height ::= min_line_height, max_line_height
min_line_height ::= "MIN_LINE_HEIGHT:", real
max_line_height ::= "MAX_LINE_HEIGHT:", real
line_number ::= min_line_number, max_line_number
min_line_number ::= "MIN_LINE_NUMBER:", integer
max_line_number ::= "MAX_LINE_NUMBER:", integer
space_before ::= min_space_before, max_ space_before
min_space_before ::= "MIN_SPACE_BEFORE:", real
max_ space_before ::= "MAX_SPACE_BEFORE:", real
space_after ::= min_space_after, max_ space_after
min_space_after ::= "MIN_SPACE_AFTER:", real
max_ space_after ::= "MAX_SPACE_AFTER:", real
black_pixel_density ::= min_black_pixel_density, max_black_pixel_density
min_black_pixel_density ::= "MIN_BLACK_PIXEL_DENSITY:", real
max_black_pixel_density ::= "MAX_BLACK_PIXEL_DENSITY:", real
justification ::= "JUSTIFY:", justification_indicator
justification_indicator ::= "LEFT" | "RIGHT" | "CENTER" | "INDENT"
integer ::= digit+
real ::= ".", digit+ | digit+, ".", digit*
```

그림 2 DSDL의 E-BNF 형식

4. 구문론적인 구조 분석 방법

본 절에서는 논리적인 구조분석 방법을 기술한다. 제안된 방법은 <그림 3>과 같이 기능 구조 트리의 생성, 논리 구조 트리의 생성, 그리고 SGML 문서 생성의 세

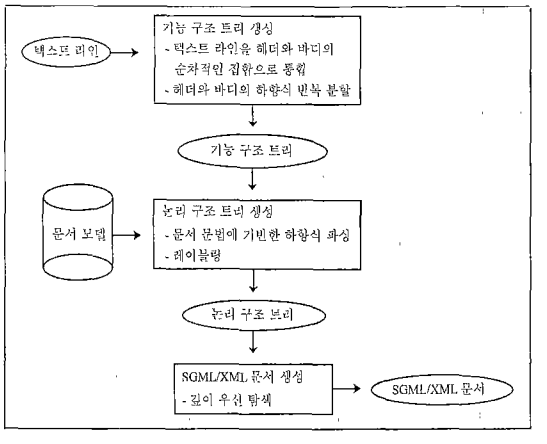


그림 3 논리적인 구조분석 과정

단계로 구성된다. 제안된 방법은 먼저 기하적인 구조분석의 결과로부터 추출된 텍스트 라인을 병합하여 헤더와 바디의 순차적인 집합을 생성하고, 이를 반복 분할하면서 계층적인 기능 구조 트리를 생성한다.

두 번째는 문서 문법을 기능 구조 트리에 적용하여 논리 구조 트리를 생성한다. 특히 제안된 파싱 기법은 구조분석의 정확성과 처리 속도의 향상을 위하여 화소 또는 텍스트 라인의 집합이 아닌 기능 구조 트리를 대상으로 한다. 마지막으로 제안된 방법은 논리 구조 트리를 깊이 우선 탐색(depth first traversal) 하면서 논리적인 구조분석의 최종 결과로서 SGML/XML 문서를 생성한다. 각 단계에 대한 자세한 설명은 다음과 같다.

4.1 기능 구조 트리의 생성

제안된 방법은 형태 심리학(gestalt psychology)[30]에서 사용하는 세 가지의 일반적인 원칙을 적용하여 기하적인 특성이 유사한 인접한 텍스트 라인을 병합하여 헤더와 바디의 순차적인 집합을 생성한다. 즉, 근접성(proximity)의 원칙에 따라 서로 다른 객체 사이의 줄 간격은 동일한 객체에 속하는 텍스트 라인 사이의 줄 간격 보다 크다. 또한 유사성(similarity)의 원칙에 따라 동일한 객체에 속하는 텍스트 라인의 기하적인 특성은 서로 유사하다. 마지막으로 연속성(contiguity)의 원칙을 적용하여 서로 다른 종류의 단 영역에 속하는 텍스트 라인은 구별된다.

추출된 헤더와 바디 각각은 이를 구성하는 텍스트 라인의 기하적인 특성에 따라 서로 다른 종류로 구별된다. 예를 들어, 문서의 제목과 절 제목은 서로 다른 종류의 헤더로 분류되며 요약과 단락 역시 서로 다른 종류의 바디로 구별된다.

일반적으로 문서의 본문은 다수의 절을 포함하며 각각의 절은 계층적으로 중첩된 하위 레벨의 절로 구성된다. 또한 각각의 절 구조는 절 제목에 의하여 식별되며 하위 레벨의 절 제목은 이를 포함하는 상위 레벨의 절 제목보다 뒤에 위치한다. 따라서 제안된 방법은 헤더와 바디의 순차적인 집합으로부터 계층 구조의 기능 구조 트리를 생성하기 위하여 문서를 구성하는 구조의 반복적인 특성을 이용한다.

제안된 방법은 <그림 4>와 같이 반복되는 헤더를 기준으로 순차적인 집합을 반복 분할하면서 기능 구조 트리를 하향식으로 생성한다.

예를 들어, 실험에 사용된 TPAMI에 속하는 논문의 본문 영역으로부터 추출된 헤더와 바디의 순차적인 집합과 이로부터 생성된 기능 구조 트리는 각각 <그림 5>와 <그림 6>과 같다.

1. 헤더와 바디의 순차적인 집합을 생성한다.
   (a) Gestalt psychology의 원칙에 기반하여 텍스트라인을 다양한 종류의 헤더와 바디로 통합한다.
2. 헤더와 바디의 순차적인 집합을 반복 분할한다.
   (a) 반복적인 패턴을 형성하거나 하위 레벨 헤더에 해당하는 반복 헤더를 찾는다.
   (b) 순차적인 집합을 반복 헤더를 기준으로 분할한다.
3. 단계 2를 분할 가능한 집합이 더 이상 존재하지 않을 때까지 반복 적용한다.

그림 4 기능 구조 트리 생성 알고리즘

S = { H1, B1, B1, B1, B1, B1, H2, B1, B1, B1, B1, H2, B1, B1, B1, H3, B1, B1, B1, H3, B1, B1, B1, H2, B1, B1, H3, B1, B1, H3, B1, B1, B1, H3, B1, B1, H2, B1, B1, B1, H1, B1, H2, B1, H3, B1, B1, B1, H3, B1, B1, B1, B1, H3, B1, B1, B1, H2, B1, B1, H3, B1, B1, B1, H3, B1, B1, B1, B1, H3, B1, H2, B1, B1, H1, B1, B1, H2, B1, B1, H3, B1, B1, B1, H2, B1, B1, B1, B1, B1, H3, B1, B1, B1, B1, H3, B1, B1, H3, B1, B1, B1, H3, B1, B1, B1, H2, B1, H1, B1, B1, B1, B1, H1, B1, B1, B1, H2, B1, B1, B1, H2, B1, B1, B1, H2, B1, H2, B1, B1, B1, H3, B1, H1, B1, B1, B1, H1, B1, B1, B1, H1, B1, H1, B2, B2, B2, B2 }

그림 5 헤더와 바디의 순차적인 집합

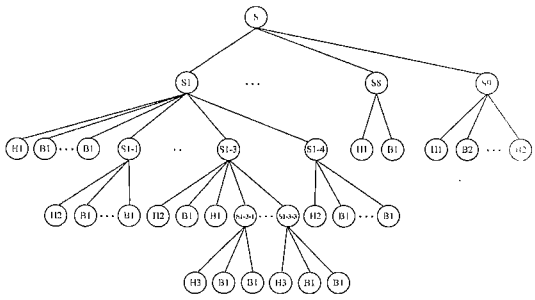


그림 6 기능 구조 트리

S = {S1, S2, S3, S4, S5, S6, S7, S8, S9}
S1 = {H1, B1, B1, B1, B1, B1, H2, B1, B1, B1, B1, H2, B1, B1, B1, H3, B1, B1, B1, H3, B1, B1, B1, H2, B1, B1, H3, B1, B1, H3, B1, B1, B1, H3, B1, B1, H2, B1, B1, B1},
S2 = {H1, B1, H2, B1, H3, B1, B1, B1, H3, B1, B1, B1, B1, B1, H3, B1, B1, B1, H2, B1, B1, H3, B1, B1, B1, B1, H3, B1, B1, B1, B1, H3, B1, B1, H2, B1, B1, B1},
S3 = {H1, B1, B1, H2, B1, B1, H3, B1, B1, H3, B1, B1, B1, H2, B1, B1, B1, B1, B1, H3, B1, B1, B1, B1, B1, H3, B1, B1, B1, H3, B1, B1, B1, H3, B1, B1, B1, H2, B1, B1, B1},
S4 = {H1, B1, B1, B1, B1},
S5 = {H1, B1, B1, B1, H2, B1, B1, B1, H2, B1, B1, B1, H2, B1, B1, H2, B1, B1, B1, B1, H3, B1},
S6 = {H1, B1, B1, B1},
S7 = {H1, B1, B1, B1},
S8 = {H1, B1},
S9 = {H1, B2, B2, B2, B2}

그림 7 H1을 기준으로 S를 분할한 결과

이와 같이 집합을 반복적으로 분할하는 것은 트리 구조를 하향식으로 확장해 가는 과정에 해당한다. 예를 들어, 집합 S를 반복하는 헤더인 H1을 기준으로 분할한 결과와 이에 해당하는 트리 구조는 각각 <그림 7>과

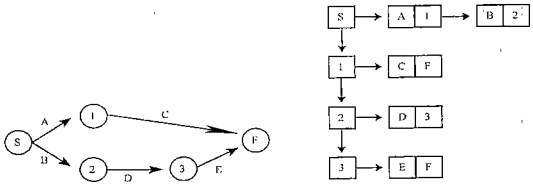
같다. 한편 집합이 하위 레벨의 절 제목에 해당하는 단일 헤더만을 포함한다면 이를 기준으로 집합을 분할한다. 제안된 분할 과정은 분할 가능한 집합이 더 이상 존재하지 않을 때까지 반복 적용되며 다단의 중첩 구조로 구성된 문서로부터 계층 구조를 추출할 수 있다. 특히 생성된 기능 구조 트리를 구성하는 각각의 단말 노드는 해당 텍스트 영역의 기하적인 특성을 포함한다.

4.2 논리 구조 트리의 생성

제안된 파싱 기법은 기능 구조 트리를 하향식 깊이 우선 탐색하면서 기능 구조 트리를 구성하는 중간 노드의 계층 구조와 단말 노드의 기하적인 특성이 문서 문법의 해당 조건을 만족하는지의 여부를 검사한다. 뿌리 노드(root node)를 포함한 중간 노드에 대하여 자식 노드가 단말 노드 또는 중간 노드인지의 여부에 따른 처리 과정은 다음과 같다. 먼저 자식 노드가 단말 노드인 경우, 자식 노드의 기하적인 특성을 만족하는 엘리먼트의 이름을 레이블로 부여한다. 자식 노드가 중간 노드인 경우, 허용 가능한 엘리먼트의 이름을 부여한 후 해당 엘리먼트의 내용 모델이 적합한지의 여부를 검사한다.

만일 중간 노드의 파싱 과정에서 자식 노드를 만족하는 엘리먼트 선언이 존재하지 않을 경우, 제안된 방법은 중간 노드에 선택 가능한 또 다른 문법을 적용하기 위하여 백트래킹(backtracking)한다. 예를 들어, <그림 6>에서 중간 노드S의 자식 노드 S9에 <그림 1>의 문서 모델에서 엘리먼트 *Sec-Body*의 문법을 적용한다고 하자. 이때 레이블 *Section*과 *Reference*의 적용이 가능하다. 따라서 제안된 방법은 먼저 레이블 *Section*을 부여한 후, S9의 자식 노드가 *Section*의 내용 모델을 만족하는지의 여부를 검사한다. 그러나 해당 내용 모델은 두 번째 단말 자식 노드인 B2의 기하적인 특성을 만족하는 엘리먼트를 포함하지 않는다. 따라서 제안된 방법은 부모 노드 S9로 백트래킹하여 허용 가능한 또 다른 엘리먼트인 *Reference*의 내용 모델을 적용한다.

특히 제안된 방법은 중간 노드에 대한 파싱 과정의 속도 향상을 위하여 내용 모델을 테이블로 재구성한다. 먼저 내용 모델에서 순서에 상관없음을 나타내는 "&" 기호를 순서를 가진 관계로 변환한다. 예를 들어, 내용 모델 (A & B)을 (A B) | (B A)로 변환한다. 제안된 방법은 내용 모델 정보를 상태 전이도(transition diagram)로 표현한 후 이를 다시 상태 전이 테이블(transition table)로 재구성한다. 각각의 구성 요소는 해당 테이블에 대한 시작 포인터를 포함한다. 예를 들어, 내용 모델 AC | BDE를 상태 전이도와 테이블로 구성한 결과는 <그림 8>과 같다.



(a) 상태 전이도 (b) 상태 전이 테이블

그림 8 상태 전이도 및 상태 전이 테이블

결과적으로 제안된 파싱 기법은 각각의 노드에 문법에 명시된 레이블을 부여하여 논리 구조 트리를 완성한다. 예를 들어, <그림 1>의 문법을 기반으로 <그림 6>의 기능 구조 트리에 하향식 파싱 기법을 적용하여 생성된 논리 구조 트리는 <그림 9>와 같다.

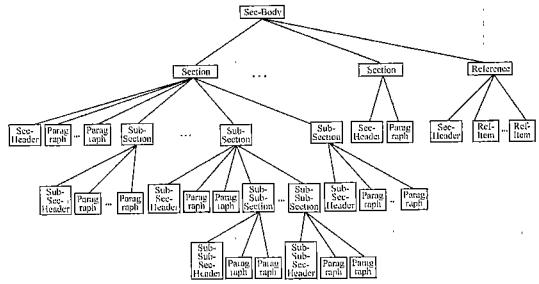


그림 9 논리 구조 트리

4.3 SGML 문서의 생성

일반적으로 사용자가 문서를 처음부터 끝까지 읽어가는 순서는 논리 구조 트리를 깊이 우선 탐색하는 과정으로 볼 수 있다. 제안된 방법은 논리 구조 트리를 구성하는 단말 노드와 중간 노드 각각에 대하여 서로 다른

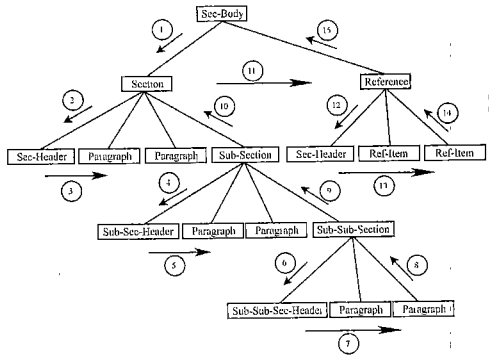


그림 10 논리 구조 트리의 깊이 우선 탐색 과정

방법을 적용하여 SGML/XML 문서를 생성한다.

먼저 논리 구조 트리를 깊이 우선 탐색하면서 중간 노드를 만나면 해당 레이블을 이름으로 갖는 엘리먼트의 시작 태그를 출력하며 해당 노드를 벗어날 때 끝 태그를 출력한다.

```
<Sec-Body>
  <Section>
    <Sec-Header>OCR results</ Sec-Header>
    <Paragraph> OCR results </Paragraph>
    <Paragraph> OCR results </Paragraph>
    <Sub-Section>
      <Sub-Sec-Header> OCR results </Sub-Sec-Header>
      <Paragraph> OCR results </Paragraph>
      <Paragraph> OCR results </Paragraph>
      <Sub-Sub-Section>
        <Sub-Sub-Sec-Header> OCR results </Sub-Sub-Sec-Header>
        <Paragraph> OCR results </Paragraph>
        <Paragraph> OCR results </Paragraph>
      </Sub-Sub-Section>
    </Sub-Section>
  </Section>
  <Reference>
    <Sec-Header> OCR results </ Sec-Header>
    <Ref-Item> OCR results </Ref-Item>
    <Ref-Item> OCR results </Ref-Item>
  </Reference>
</Sec-Body>
```

그림 11 SGML 문서의 생성 결과

예를 들어, <그림 10>에서 탐색 과정 ①과 ⑩에 의하여 각각 엘리먼트 *Section*의 시작 태그와 끝 태그를 출력한다. 한편 단말 노드는 문서의 텍스트 영역과 직접적인 대응 관계를 갖는 주 구조에 해당한다. 따라서 단말 노드를 만나면 먼저 해당 레이블의 이름을 갖는 엘리먼트의 시작 태그를 출력하고, 해당 텍스트 영역의 문자 인식 결과와 끝 태그를 출력함으로써 엘리먼트를 생성한다. 따라서 제안된 방법은 <그림 10>과 같이 논리 구조 트리를 깊이 우선 탐색하면서 <그림 11>과 같은 SGML 문서를 생성한다. 한편 생성된 SGML 문서는 문서 모델로부터 자동 생성된 DTD에 대하여 검증이 가능하다.

5. 문서 모델의 자동 생성

본 절에서는 샘플 영상으로부터 문서 모델의 자동 생성과 점증적인 학습 방법을 기술한다. 제안된 방법은 기능 구조 트리의 생성, 기능 구조 트리의 일반화, 레이블링(labeling), 그리고 문서 모델의 생성의 네 단계로 구성된다. 여기서 기능 구조 트리를 생성하는 방법은 "4.1 기능 구조 트리의 생성"과 동일하다. 특히 제안된 방법은 전자 문서의 표준 포맷인 SGML과 DSSSL을 지원하기 위하여 문서 모델에 포함된 논리적인 구조 정보와 기하적인 특성으로부터 각각 SGML DTD와 DSSSL 스타일 시트를 자동 생성한다.

5.1 기능 구조 트리의 일반화

본 절에서는 <그림 6>과 같은 기능 구조 트리로부터 <그림 12>와 같은 일반화된 기능 구조 트리를 생성하는 방법을 기술한다. 이를 위하여 제안된 방법은 문서를 구성하는 구조의 반복적인 특성에 기반한다. 예를 들어, 과학기술 논문의 경우, 본문은 계층적으로 중첩된 다수의 절로 구성된다. 제안된 방법은 특정한 중간 노드가 포함하는 하부 구조를 일반화하기 위하여 반복하는 공통 구조를 식별하고 이를 Kleene *로 표현한다.

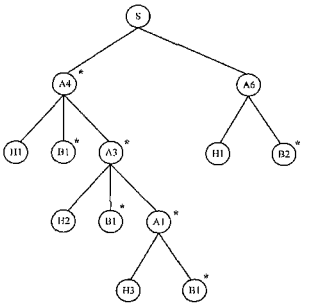


그림 12 일반화된 기능 구조 트리

일반적으로 논리적으로 동일한 수준의 절 구조는 기능 구조 트리에서 동일한 레벨(level)에 위치한다. 예를 들어, <그림 6>의 기능 구조 트리에서 절 제목은 2번째 레벨에 위치한다. 따라서 제안된 방법은 기능 구조 트리를 상향식으로 각각의 레벨을 일반화하면서 상위 노드가 포함할 수 있는 하부 구조의 정규 수식을 추출한다. 제안된 일반화 알고리즘은 <그림 13>과 같다.

1. 기능 구조 트리의 각 레벨에 다음의 일반화 과정 (a) 또는 (b)를 적용하여 일반화 패턴을 추출한다. 특히 동일한 일반화 패턴을 자식으로 갖는 부모 노드는 보조 기호로 대체한다.
   (a) 반복적인 패턴, $(a, \ldots, a)$,을 식별하여 이를 일반화 패턴, (a^*).으로 표현한다. 여기서 기호 "a"은 바디 또는 일반화 패턴을 나타내는 보조 기호에 해당한다.
   (b) 자식 노드의 일반화 패턴이 각각 (S) 와 (S, C)인 두 노드 A와 B를 (B^*)로 일반화한다. 여기서 S는 단일 헤더이거나 단일의 헤더와 같은 유형이며 한 개 이상의 바디로 구성된 순차적인 집합이다. 또한 C는 보조 기호로 표현된 하위 레벨의 절에 해당한다.
2. 일반화 과정이 더 이상 적용되지 않을 때까지 단계 1을 반복한다.

그림 13 기능 구조 트리의 일반화 알고리즘

예를 들어, <그림 6>의 기능 구조 트리에서 최하위 레벨인 레벨 5를 일반화한 결과는 <그림 14>와 같다.

S1-2-1 = (H3, B1*), S1-2-2 = (H3, B1*),
S1-3-1 = (H3, B1*), S1-3-2 = (H3, B1*), S1-3-3 = (H3, B1*),
S2-1-1 = (H3, B1*), S2-1-2 = (H3, B1*), S2-1-3 = (H3, B1*),
S2-2-1 = (H3, B1*), S2-2-2 = (H3, B1*), S2-2-3 = (H3, B1*), S2-2-4 = (H3, B1)
S3-1-1 = (H3, B1*), S3-1-2 = (H3, B1*),
S3-2-1 = (H3, B1*), S3-2-2 = (H3, B1*), S3-2-3 = (H3, B1*),
S3-2-4 = (H3, B1)
S5-4-1 = (H3, B1)

그림 14 그림 6에서 레벨 5의 일반화 패턴

본 논문은 반복적인 노드의 집합을 Kleene *로 표현한 결과를 일반화 패턴(generalized pattern)이라고 정의한다. 예를 들어, **S1−2−1=(H3**, B1, B1, B1)로부터 반복적인 패턴 (B1, B1, B1)을 (B1*)로 대체하여 일반화 패턴 (**H3**, B1*)을 생성한다. 실제로 일반화 패턴은 상위 노드가 포함할 수 있는 자식 노드의 종류와 순서 그리고 빈도수에 대한 정규 수식에 해당한다.

제안된 방법은 일반화 패턴을 효과적으로 추출하기 위하여 자식 노드에 대하여 동일한 일반화 패턴을 갖는 부모 노드는 동일한 이름의 보조 기호(auxiliary symbol)로 표현한다. 특히 본 논문은 동일한 헤더와 바디로 시작하는 두 패턴 (a, b)와 (a, b*)를 동일한 형태로 간주한다.

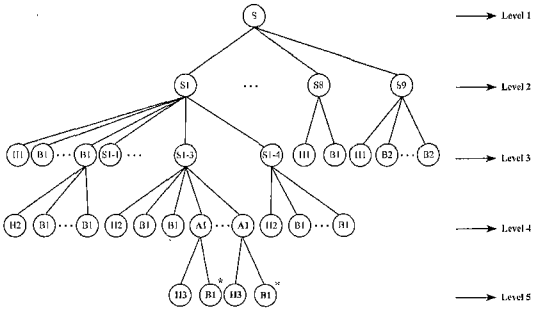


그림 15 그림 6에서 최하위 레벨을 일반화한 결과

따라서 <그림 15>와 같이 일반화 패턴 (H3, B1*)와 (H3, B1)은 보조 기호 A1으로 대체된다. 또한 레벨 4에 속하는 노드의 집합으로부터 반복적인 패턴 (B1, , B1)과 (A1, , A1)을 식별하고, 이를 각각 (B1*)와 (A1*)로 대체하여 <그림 16>과 같은 일반화 패턴을 추출한다.

S1-1 = (H2, B1*), S1-2 = (H2, B1*, A1*), S1-3 = (H2, B1*, A1*), S1-4 = (H2, B1*)
S2-1 = (H2, B1, A1*), S2-2 = (H2, B1*, A1*), S2-3 = (H2, B1*)
S3-1 = (H2, B1*, A1*), S3-2 = (H2, B1*, A1*), S3-3 = (H2, B1)
S5-1 = (H2, B1*), S5-2 = (H2, B1*), S5-3 = (H2, B1), S5-4 = (H2, B1*, A1)

그림 16 레벨 4의 일반화 결과

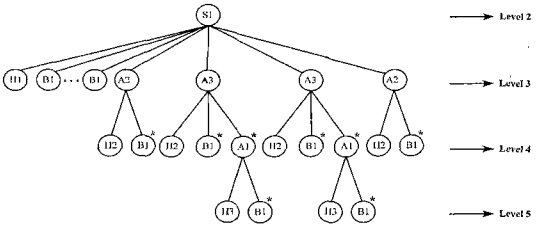


그림 17 레벨 4의 일반화 결과를 보조 기호로 표현한 결과

마찬가지로 레벨 4의 결과를 바탕으로 레벨 3을 일반화한다. 예를 들어, <그림 17>에서 S1=(H1, B1, B1, B1, B1, B1, A2, A3, A3, A2)의 일반화 과정은 다음과 같다. 전술한 바와 같이 제안된 방법은 먼저 동일한 모양의 반복적인 패턴 (B1,..., B1)과 (A3,..., A3)를 식별하여 <그림 18>과 같은 정규 수식 S1=(H1, B1*, A2, A3*, A2)을 생성한다.

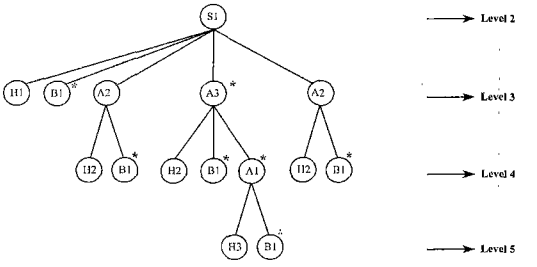


그림 18 반복적인 패턴의 일반화 결과

일반적으로 문서를 구성하는 절은 하위 레벨의 절을 선택적으로 포함할 수 있다. 따라서 자식 노드의 패턴이 각각 (S)와 (S, C)인 두 노드 A와 B는 (B*)로 일반화한다. 여기서 S는 단일의 헤더 또는 헤더와 한 개 이상의 동일한 종류의 바디로 구성된 순차적인 집합이며 C

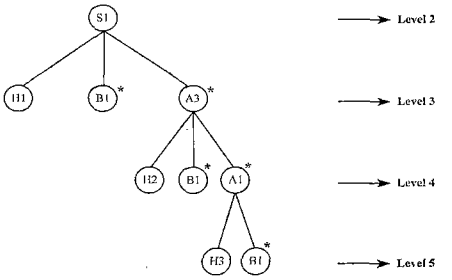


그림 19 S1의 일반화 결과

는 보조 기호로 표현된 하위 레벨의 절 구조에 해당한다. 예를 들어, A2와 A3는 하부 구조에 해당하는 일반화 패턴으로써 모두 공통적인 접두사 A2를 포함한다. 따라서 제안된 방법은 <그림 19>와 같이 정규 수식 (A2, A3*, A2)를 (A3*)로 통합하여 S1의 일반화 패턴으로써 (H1 B1* A3*)를 추출한다. <그림 6>의 기능 구조 트리를 레벨 3까지 일반화한 결과는 <그림 20>과 같다.

```
S1 = S2 = S3 = A4, S4 = A5, S5 = A4, S6 = S7 = S8 = A5, S9 = A6,
where A4 = (H1 B1* A3*), A5 = (H1 B1*). and A6 = (H1 B2*)
```

그림 20 레벨 3의 일반화 결과

마지막으로 노드 S의 자식 노드인 (S1, S2, S3, S4, S5, S6, S7, S8, S9)의 일반화 과정은 다음과 같다. 먼저 반복적인 패턴 (S1, S2, S3)과 (S6, S7, S8)을 식별하여 일반화 패턴 (A4* A1 A4 A1* A5)를 생성한다. 또한 제안된 일반화 알고리즘의 1(b)를 적용하여 최종적으로 <그림 12>와 같이 일반화 패턴 (A4* A5)을 생성한다.

5.2 레이블링

제안된 방법은 일반화된 기능 구조 트리에 논리적인 의미를 부여하기 위하여 사용자로부터 각각의 노드에 대한 적절한 레이블을 입력 받는다. 예를 들어, <그림 12>에 레이블을 부여한 결과는 <그림 21>과 같다.

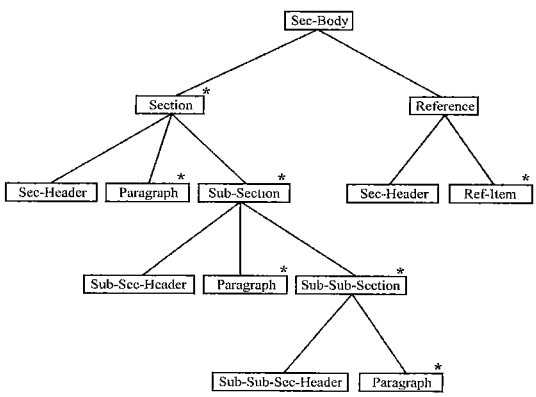


그림 21 일반화된 기능 구조 트리에 레이블을 부여한 결과

5.3 문서 모델의 생성

전술한 바와 같이 기능 구조 트리에서 중간 노드는 부 구조에 해당하며 단말 노드는 주 구조에 해당한다. 따라서 제안된 방법은 각각의 중간 노드에 대하여 자식 노드에 대한 일반화 패턴 정보를 이용하여 내용 모델을 생성한다. 또한 각각의 단말 노드가 포함하는 기하적인 속성 값을 이용하여 주 구조에 대한 기하적인 조건을 기술한다.

예를 들어, <그림 21>에 너비 우선 탐색(breadth-first traversal)을 적용하여 추출한 문서 모델은 <그림 22>와 같다.

```
<!ELEMENT Sec-Body (Section*, Reference)>
<!ELEMENT Section (Sec-Header, Paragraph*, Sub-Section*)>
<!ELEMENT Sub-Section (Sub-Sec-Header, Paragraph*, Sub-Sub-Section*)>
<!ELEMENT Sub-Sub-Section (Sub-Sub-Sec-Header, Paragraph*)>
<!ELEMENT Reference (Sec-Header, Ref-Item*)>
```

그림 22 그림 21로부터 추출된 문서 모델의 예

일반적으로 문서 모델은 다수의 학습 데이타로부터 생성된다. 한편 문서 유형에 속하는 새로운 문서의 구조를 반영하기 위하여 문서 모델에 대한 점증적인 학습 방법이 요구된다. 제안된 방법은 이러한 문제에 대한 효과적인 해결 방법을 제공한다.

먼저 다수의 실험 데이타로부터 문서 모델을 생성하기 위하여 제안된 방법은 먼저 부 구조 각각에 대하여 가능한 문서 모델을 통합한다. 특히 보다 간결한 형태의 정규 수식을 추출하기 위하여 인수 분해(factorization) 기법을 적용한다. 예를 들어, 내용 모델 (a c), (a d), (b c), 그리고 (b d)에 인수 분해 과정을 적용한 결과는 ((a | b) | (c | d))에 해당한다.

본 논문은 이를 위하여 기존에 논리 함수 (boolean function)의 최적화 분야에서 사용하는 알고리즘 [31]을 적용한다. 마찬가지로 제안된 방법은 새로운 문서의 일반화 결과를 위와 같은 방법으로 기존 문법에 간단히 통합할 수 있기 때문에 문서 모델의 점증적인 학습을 지원한다. 마지막으로 <그림 23>과 <그림 24>와 같이 제안된 방법은 문서 모델로부터 SGML DTD와 DSSSL 스타일 시트를 생성한다.

```
<!ELEMENT Sec-Body          --  (Section*, Reference)>
<!ELEMENT Section           --  (Sec-Header, Paragraph*, Sub-Section*)>
<!ELEMENT Sub-Section       --  (Sub-Sec-Header, Paragraph*, Sub-Sub-Section*)>
<!ELEMENT Sub-Sub-Section   --  (Sub-Sub-Sec-Header, Paragraph*)>
<!ELEMENT Reference         --  (Sec-Header, Ref-Item*)>
<!ELEMENT Sec-Header        --  (#PCDATA)>
<!ELEMENT Sub-Sec-Header    --  (#PCDATA)>
<!ELEMENT Sub-Sub-Sec-Header --  (#PCDATA)>
<!ELEMENT Paragraph         --  (#PCDATA)>
<!ELEMENT Ref-Item          --  (#PCDATA)>
```

그림 23 문서의 모델의 논리적인 구조를 SGML DTD로 표현한 결과

```
(element Sec-Header (make paragraph    font-size: 24pt
                                       font-weight: 'bold
                                       quadding: 'left
                                       line-spacing 24pt
                                       space-before. 12pt
                                       space-after: 12pt ))
(element Paragraph (make paragraph     font-size: 24pt
                                       font-weight. 'medium
                                       quadding: 'left
                                       first-line-start-indent: (if(first-sibling?)
                                                                    0pt
                                                                    Paragraph-Indent)
                                       space-before 0pt
                                       space-after: 0pt
                                       color: (RGB-COLOR 0. 0. 0.)))
```

그림 24 DSSSL 스타일 시트의 일부

6. 실험 결과 및 성능 분석

제안된 방법의 성능을 평가하기 위하여 1999년 1월부터 6월 사이에 발행된 TPAMI에 속하며 372개의 논문 영상으로 구성된 정규 논문 26편을 대상으로 실험하였다. 제안된 구조분석 시스템은 다수의 페이지로 구성된 각각의 논문에 속하는 텍스트 라인의 집합을 입력으로 받아들인다. 특히 본 논문에서는 헤더와 바디의 식별률을 계산하기 위하여 각각의 논문에 최근에 본 연구팀이 개발한 기하적인 구조 분석 방법[32]을 적용하여 텍스트 라인의 집합을 추출하고, 이로부터 헤더와 바디의 집합을 생성하여 검증용 데이타(*ground truth dataset*)를 마련하였다. 제안된 방법의 성능을 정량적으로 평가한 결과는 <표 3>과 같다.

표 3 성능 평가

논문	텍스트 라인 수	헤더와 바디의 수 (헤더의 수)	잘못 식별된 헤더와 바디의 수 (잘못 식별된 헤더의 수)
1	576	84(10)	0(0)
2	1259	149(26)	1(0)
3	731	99(16)	0(0)
4	1144	104(14)	3(0)
5	971	144(29)	1(3)
6	1353	170(48)	0(0)
7	1051	150(22)	2(0)
8	1239	120(19)	2(0)
9	722	92(12)	0(0)
10	826	101(14)	0(0)
11	608	81(17)	5(0)
12	703	78(13)	4(0)
13	1383	208(43)	1(2)
14	1238	140(15)	0(0)
15	618	91(16)	0(0)
16	751	89(15)	0(0)
17	601	74(18)	1(0)
18	730	74(14)	2(0)
19	1089	156(39)	1(0)
20	914	115(30)	0(1)
21	756	110(27)	0(0)
22	953	116(19)	1(0)
23	1030	121(24)	1(0)
24	860	97(19)	1(0)
25	778	118(29)	0(0)
26	1239	162(19)	5(0)
합계	24123	3043(560)	31(6)

6.1 성능 분석

본 논문에서는 구조분석 방법의 정확성과 처리 속도의 두 가지 측면에서 제안된 방법의 성능을 분석한다. 제안된 구조분석 방법은 기능 구조 트리의 생성과 파싱의 두 단계로 구성된다. 또한 기능 구조 트리의 생성 과정은 헤더와 바디의 식별과 계층 구조의 생성의 두 단계로 구성된다. 따라서 본 논문에서는 구조분석의 정확성을 평가하기 위하여 헤더와 바디의 식별률, 계층 구조 생성의 정확성, 그리고 논리적인 객체의 식별률의 세 가지 평가 기준을 제안한다.

헤더와 바디의 식별률 면에 있어서, 제안된 방법은 <표 3>과 같이 3,043개의 기능 구성 요소 중에서 31개의 식별에 실패하여 98.9%의 식별률을 보였다. 한편 Summers[22]는 360개의 영상으로 구성된 기술 보고서(technical report)를 대상으로 실험 결과를 제시하였다. Summers의 연구 결과는 본 논문과 동일한 실험 데이타를 사용하지는 않지만 텍스트 라인을 입력으로 받아들이는 가장 최근의 연구 결과이다. 정량적인 실험 결과에 의하면 제안된 방법은 90.1%의 식별률을 보인 Summers의 방법보다 나은 결과를 보였다. 이는 제안된 방법이 보다 다양한 종류의 기하적인 특성과 인접한 페이지를 모두 고려하기 때문이다. 예를 들어, 단락은 인접한 페이지에 분할되어 위치하는 경우가 다수 존재하며 절 제목은 줄 간격은 물론이고 밀도 등의 다양한 종류의 기하적인 특성에 의하여 구별된다.

실험 결과, 식별된 헤더와 바디의 오류 분석은 다음과 같다. 오류는 크게 바디의 통합과 분할 그리고 헤더의 삽입과 삭제의 네 가지로 분류할 수 있다. 먼저 바디의 통합 오류의 대부분은 불규칙한 정렬 방식의 텍스트 라인으로 시작하며 리스트 또는 수식을 포함하는 단락이 이전 단락에 통합된 경우이었다. 한편 분할 오류의 대부분은 수식이 단락의 시작 텍스트 라인으로 잘못 식별된 경우에 해당하였다.

그밖에 편집 오류에 의하여 바디가 통합 또는 분할되었으며 단락의 일부 텍스트 라인이 비 텍스트 객체의 캡션으로 잘못 식별되었다. 한편 헤더의 삭제와 삽입 오류는 모두 세 번째 단계의 절 제목을 식별하지 못하거나 바디에 속하는 텍스트 라인을 세 번째 단계의 절 제목으로 잘못 식별한 경우에 해당하였다.

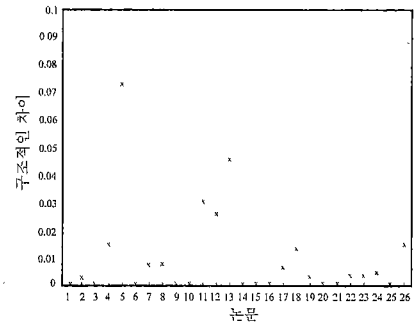


그림 25 구조적인 차이에 대한 실험 결과

계층 구조의 정확성을 실험하기 위하여 본 논문은 구조 분석의 결과로 생성된 계층 구조와 정확한 구조 사이의 구조적인 차이를 계산하였다. 이를 위하여 트리 간의 차이(difference)를 계산하는 기존 연구인 Zhang과 Shasha의 방법[33]을 적용하였다. 본 논문은 동일한 레벨의 중간 노드는 동일한 레이블을 갖으며 각 연산의 비용을 1로 가정하였다. 특히 보다 의미 있는 차이를 계산하기 위하여 추출된 차이를 생성된 트리 구조와 정확한 트리 구조의 합으로 정규화 하였다. 실험 결과 <그림 25>와 같이 평균적인 차이는 0.01로 나타났다.

논리적인 객체의 식별률의 경우, 제안된 시스템은 전체 실험 데이타에 대하여 파싱 오류 없이 레이블링을 수행하였다. 실험 결과, 논리적인 객체의 식별 오류는 모두 헤더와 바디의 식별 오류에 기인하였으며 정확히 식별된 헤더와 바디에 잘못된 레이블이 부여된 경우는 존재하지 않았다.

제안된 파싱 기법은 처리 속도의 향상을 위하여 텍스트 라인이 아닌 기능 구성 요소인 헤더와 바디를 처리 대상으로 한다. 일반적으로 구문론적인 방법은 빈번한 백트래킹을 허용하기 때문에 처리 시간의 대부분이 파싱 과정에 소요된다. 본 논문에서는 전체 실험 영상을 대상으로 텍스트 라인과 기능 구성 요소의 빈도수를 조사하였다. 조사 결과, 텍스트 라인과 기능 구성 요소의 빈도수는 각각 24,123과 3,043으로서 1 : 0.126의 비율을 나타났다. 따라서 제안된 구조분석 방법은 파싱의 기본 단위가 화소 또는 텍스트 라인인 방법보다 처리 속도가 보다 빠르다.

6.2 관련 연구와의 비교

본 논문은 다수의 페이지를 포함하는 문서 영상으로부터 SGML/XML 문서의 자동 생성을 위하여 논리적인 구조분석과 문서모델의 자동생성 방법을 제안한다. 일반적으로 기존 연구의 대부분 [5],[6],[7],[8],[11],[15],[17],[18],[24],[25],[26],[27]은 단일의 문서 영상을 처리 대상으로 하기 때문에 다수의 페이지로 구성된 문서를 지원하지 않는다. 또한 문서모델의 자동생성 및 점증적인 학습방법을 제공하지 않기 때문에 오류 영상을 지원하기 위하여 문서 모델의 설계 및 테스트 과정을 여러 번 반복하여야 한다.

기존 연구의 대부분은 논리적인 객체의 식별을 위하여 화소의 길이와 빈도수 [5],[6], 텍스트 영역의 상대적인 위치와 크기 [9],[10],[15], 그리고 흰 공백의 위치와 크기 [21],[22] 등의 단순한 정보만을 고려하기 때문에 제한된 수준의 구조 분석을 지원한다. 반면에 제안된 방법은 주 구조의 추출을 위하여 텍스트 라인의 단 유형, 높이, 정렬 방식, 밀도, 그리고 줄 간격 등의 다양한 종류의 기하적인 특성을 고려하며 부 구조의 식별을 위하여 자식의 종류, 순서, 그리고 빈도수 등을 이용한다.

제안된 방법의 처리 속도를 기존의 구문론적인 방법과 비교하면 다음과 같다. 일반적으로 구문론적인 구조분석에서 전체 처리 시간의 70% 이상이 재귀적인 파싱에 소요된다[5],[6]. 따라서 보다 빠른 파싱을 위하여 적은 수의 문법과 입력이 요구된다. 기존 연구 [5],[6]은 파싱의 기본적인 단위로써 화소를 대상으로 하기 때문에 많은 수의 문법을 필요로 한다. 한편 텍스트 라인에 기반한 [17]은 처리 속도의 향상을 위하여 인접한 텍스트 라인을 병합한 텍스트 영역을 파싱의 기본 단위로 사용하는 것이 바람직하다고 기술한다. 따라서 제안된 방법은 계층 구조를 갖는 기능 구성 요소에 기반하기 때문에 입력 단위가 화소 또는 텍스트 라인인 기존 연구보다 보다 빠른 처리 속도를 제공한다.

한편 관련 연구 [19],[20]는 본 논문과 같이 다수의 페이지로 구성된 문서에 대하여 구문론적인 구조분석과 문서모델의 자동생성 방법을 제안한다. 그러나 파싱의 기본 단위로서 문자를 사용하기 때문에 텍스트로 이루어진 전자 문서를 처리 대상으로 하며 기하적인 특성에 기반한 문서 영상의 구조분석을 지원하지 않는다.

7. 결론 및 향후 연구 방향

SGML/XML은 논리적인 구조 정보를 표현할 수 있으며 이 기종간의 호환이 가능하다는 장점 때문에 전자 문서의 표준 포맷으로 널리 사용되고 있다. 따라서 본 논문에서는 다수의 문서 영상으로 구성된 복잡한 구조의 문서로부터 SGML/XML에 기반한 전자 문서를 생성하기 위한 구문론적인 구조분석 방법을 제안한다. 특히 제안된 구조분석 방법은 처리 속도의 향상을 위하여 계층적인 구조의 기능 구조 트리에 기반한다. 이를 위하

여 형태 심리학의 세 가지 원칙을 적용하여 기능 구성 요소를 추출하고, 이로부터 헤더의 반복적인 특성을 적용하여 기능 구조 트리를 하향식으로 생성한다.

또한 구조분석의 정확성을 위하여 문서 모델을 효율적으로 표현할 수 있는 언어인 DSDL을 제안한다. DSDL은 논리적인 계층 구조를 기술하기 위하여 문서 유형이 포함할 수 있는 주 구조의 기하적인 특성은 물론이고 부 구조가 포함할 수 있는 구성 요소의 종류와 순서 그리고 빈도수 등에 대한 다양한 정보를 기술한다. 특히 본 논문은 오류 문서를 문서 모델에 쉽게 반영하기 위하여 문서 모델의 자동 생성과 점증적인 학습 방법을 제안하였으며 문서 모델에 포함된 논리적인 구조 정보와 기하적인 특성으로부터 각각 SGML DTD와 DSSSL 스타일 시트를 생성한다.

제안된 구조분석 방법은 기능 구조 트리를 하향식 깊이 우선 탐색하면서 DSDL로 기술된 문법을 적용하여 논리 구조 트리를 생성하고, 구조분석의 최종 결과로써 SGML/XML 문서를 생성한다. 제안된 방법의 성능을 분석하기 위하여 다양한 종류의 문서 영상에 실험한 결과, 기존 연구와 달리 다수의 문서 영상으로 구성된 문서에 대하여 논리적인 구조분석과 문서 모델의 자동 생성을 효율적으로 지원하였다. 특히 제안된 방법은 논리적인 구조분석의 최종 결과로서 SGML/XML 문서를 생성하기 때문에 문서의 재 사용성과 호환성을 높인다.

향후 본 연구에서는 논리적인 구조분석 방법을 체계적으로 분석할 수 성능 평가 기준에 관한 연구를 진행할 계획이다. 예를 들어, 문서 모델의 자동 생성 기법을 평가하기 위하여 자동 생성된 문서 모델을 전문가의 수작업에 위하여 생성된 문서 모델과 비교하는데 있어서의 체계적인 평가 기준이 요구된다.

참 고 문 헌

[1] International Organization for Standardization, Information Processing-Text and Office Systems-Standard Generalized Markup Language (SGML), *ISO/IEC 8879*, 1986.

[2] World Wide Web Consortium, *Extensible Markup Language (XML) 1.0*, http://www.w3c.org/TR/REC-xml, 2000.

[3] K. M. Summers, "Toward a Taxonomy of Logical Document Structures," *Proc. Dartmouth Institute for Advanced Graduate Studies (DAGS'95)*, pp. 124~133, Boston, May 1995.

[4] G. Nagy, J. Kanai, M. Krishnamoorthy, M. Thomas, and M. Viswanathan, "Two Complementary Techniques for Digitized Document Analysis," *Proc. ACM Conf. Document Processing Systems*, pp. 169~176, 1988.

[5] G. Nagy, S. Seth, and M. Viswanathan, A Prototype Document Image Analysis System for Technical Journals, *IEEE Computer*, Vol. 25, No. 7, pp. 10~22, Jul. 1992.

[6] M. Krishnamoorthy, G. Nagy, S. Seth, and M. Viswanathan, Syntactic Segmentation and Labeling of Digitized Pages from Technical Journals, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 15, No. 7, pp. 737~747, Jul. 1993.

[7] G.S.D. Farrow, C.S. Xydears, J.P. Oakley, A. Khorabi and N.G. Prelcic, "A Comparison of System Architectures for Intelligent Document Understanding," *Signal Processing: Image Communication*, Vol. 9, pp. 1~19, 1996

[8] D. Niyogi and S. N. Srihari, An Integrated Approach to Document Decomposition and Structural Analysis, *Int'l Journal of Imaging Systems and Technology*, Vol. 7, pp.330~342, 1996.

[9] A. Dengel and G. Barth, High Level Document Analysis Guided By Geometric Aspects, *Int'l Journal of Pattern Recognition and Artificial Intelligence*, Vol. 2, No. 4, pp.641~655, 1988.

[10] A. Dengel, R. Bleisinger, R. Hoch, F. Fein, and F. Hnes, From Paper to Office Document Standard Representation, *IEEE Computer*, Vol. 25, No. 7, pp. 63~67, Jul. 1992.

[11] S. Tsujimoto and H. Asada, Major Components of a Complete Text Reading System, *Proc. IEEE*, Vol. 80, No. 7, pp. 1133~1149, Jul. 1992.

[12] International Organization for Standardization, Information Technology-Text and Office Systems-Document Style Semantics and Specification (DSSSL), *ISO/IEC 10179*, 1996.

[13] L. O'Gorman and R. Kasturi, *Document Image Analysis*, IEEE Computer Society, 1995.

[14] G. Nagy, Twenty Years of Document Image Analysis in PAMI, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 22, No. 1, pp. 38~62, Jan. 2000.

[15] G. A. Story, L. O'Gorman, D. Fox, L. L. Schaper, and H. V. Jagadish, The RightPages Image-Based Electronic Library for Alerting and Browsing, *IEEE Computer*, Vol. 25, No. 9, pp. 17~26, Sept. 1992.

[16] T. Hu and R. Ingold, "A Mixed Approach toward an Efficient Logical Structure Recognition from Document Images," *Electronic Publishing: Origination, Dissemination and Design*, Vol. 6, No. 4, pp. 457~468, 1993.

[17] A. Conway, "Page Grammars and Page Parsing: A Syntactic Approach to Document Layout Recognition," *Proc. Second Int'l Conf. Document Analysis and Recognition*, pp. 761~764, 1993.
[18] Y. Tateisi and N. Itoh, "Using Stochastic Syntactic Analysis for Extraction a Logical Structure from a Document Image," *Proc. Int'l Conf. Pattern Recognition*, Vol. 2, pp. 391~394, Oct. 1994.
[19] B. Klein and P. Fankhauser, "Error Tolerant Document Structure Analysis," *Proc. IEEE Int'l Forum on Research and Technology on Advances in Digital Libraries*, pp. 116~127, 1997.
[20] B. Klein and A. Abecker, "Distributed knowledge-based parsing for Document Analysis and Understanding," *Proc. IEEE Int'l Forum on Research and Technology on Advances in Digital Libraries*, pp. 6~15, May 1999.
[21] D. Rus and K. Summers, Geometric Algorithms and Experiments for Automated Document Structuring, *Mathematical and Computer Modelling*, Vol. 26, No. 1, pp. 55~83, 1997.
[22] K. M. Summers, *Automatic Discovery of Logical Document Structure*. Ph.D. Thesis, Cornell University, Aug. 1998.
[23] O. Hitz, L. Robadey, and R. Ingold, Analysis of synthetic document images, *Proc. Fifth Int'l Conf. Document Analysis and Recognition*, pp.374~377, Bangalore, India, Sep. 1999.
[24] C. Lin, Y. Niwa, S. Narita, Logical Structure Analysis of Book Document Image Using Contents Information, *Proc. Fourth Int'l Conf. Document Analysis and Recognition*, Vol. II, pp. 1048~1051, 1997.
[25] T. Kochi, and T. Saitoh, A Layout-Free Method for Element Extraction from Document Images, *Proc. Workshop on Document Analysis System*, pp. 336~345, 1998.
[26] T. A. Bayer and H. Walischewski, "Experiments on Extracting Structural Information from Paper Documents using Syntactic Pattern Analysis," *Proc. of the Third Int'l Conf. Document Analysis and Recognition*, pp.476~479, 1995.
[27] M. Worring, and A. W.M. Smeulders, "Content-based Internet Access to Paper Documents," *Int'l Journal on Document Analysis and Recognition*, Vol. 1, No. 4, pp. 209~220, 1999.
[28] A. V. Aho, R. Sethi, and J. D. Ullman. *Compilers: Principles, Techniques, and Tools*. Addison-Wesley, 1986.
[29] A. Bruggemann-Klein, "Regular Expressions into Finite Automata," *Theoretical Computer Science*, Vol. 120, No. 2, pp. 197~213, Nov. 1993.
[30] K. Koffka, *Principles of Gestalt Psychology*. Harcourt, Brace and World, New York, 1935.
[31] A. R. R. Wang, *Algorithms for Multi-level Logic Optimization*. Ph.D. Thesis, The University of California, Berkeley, 1989.
[32] K. H. Lee, Y. C. Choy and S. B. Cho, "Geometric Structure Analysis of Document Images: A Knowledge-based Approach," *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 22, No. 11, pp. 1224~1240, Nov. 2000.
[33] K. Zhang and D. Shasha, Simple Fast Algorithms for the Editing Distance between Trees and Related Problems, *SIAM Journal on Computing*, Vol. 18, No. 6, pp. 1245~1262, 1989.

이 경 호
1995년 2월 연세대학교 전산과학과 졸업(이학사). 1997년 2월 연세대학교 컴퓨터과학과(공학석사). 2001년 2월 연세대학교 컴퓨터과학과(공학박사). 2001년 4월 ~ 현재 National Institute of Standards and Technology(NIST) 객원연구원. 관심분야는 멀티미디어 문서처리, XML/SGML, 문서영상의 이해

최 윤 철
1773년 서울대학교 전자공학과 졸업(공학사). 1975년 6월 Univ. of pittsburgh(공학석사). 1979년 6월 Univ. of california, Berkeley, Dept. of IE70R(공학박사), 1979년 8월 ~ 1982년 7월 Lockheed 사 및 Rockwell International사 책임연구원 1982년 9월 ~ 1784년 1월 Univ. of washington 전산학과 박사과정. 1990년 9월 ~ 1992년 1월 Univ. of Massachusetts 연구교수. 1984년 3월 ~ 현재 연세대학교 컴퓨터과학과 교수. 관심분야는 멀티미디어, 컴퓨터 그래픽스, 가상 현실, 지리정보시스템

조 성 배
1988년 연세대학교 전산과학과 졸업(이학사). 1990년 한국과학기술원 전산학과(석사). 1991년 ~ 현재 한국과학기술원 인공지능연구센터 참여연구원. 1993년 한국과학기술원 전산학과(박사). 1993년 ~ 1995년 ATR 인간정보통신연구소 객원연구원. 1995년 ~ 현재 연세대학교 컴퓨터과학과 부교수. 관심분야는 신경망, 패턴인식, 지능정보처리, 진화 연산