2001년 한국정보처리학회 춘계 학술발표논문집 제8권 제1호

이동 에이전트를 이용한 웹 Link-Fail 시스템

정유정*, 김창원*, 정채영**
*조선대학교 전산통계학과
e-mail:gamjiyu@hanmail.net

Link-Fail System Using Mobile Agent on Web

Yu-Jeong Jeong* , Chang-Won Kim*, Chae-Young Chung**
*Dept of Computer Science & Statistics, Chosun University

요 약

현재 인터넷의 웹 환경에서 일반인들이 보다 쉽게 웹상의 대량의 정보를 효율적으로 분류,검색 하기 위해 많은 기술들이 개발되어 사용되고 있다. 본 논문에서는 이러한 정보의 방대함과 모호성으로 정확한 정보를 발견하기 어려웠을 경우 시스템 관리자가 링크 되어있는 다량의 문서 오류를 빠르고 쉽게 체크할 수 있는 시스템 구현방법을 제안하여 검증한결과 Link-Fail된 문서만 찾아 관리자로 하여금 빠른 시간내에 수정 보완할 수 있도록 알려줌으로서 효율적인 측면과 시간절약 측면에서 커다란 향상을 가져올 수 있었다.

1. 서 론

정보제공 시스템은 다양한 기술의 발전과 사용자의 새로운 요구에 의해서 급속히 성장하고 있다. 이와 같은 성장은 하드웨어 기술, 통신 기술, 그리고 사용자 인터페이스 발전으로 인하여 지속될 것이며 이미 여러 분야 즉, 전자 도서관, 원격 교육, 전자상거래 등에서 정보 시스템을 활용하고있다.[1]

그러나 현재 인터넷의 웹 환경에서는 질의와 무관한 불필요한 문서들이 검색되는 경우가 종종 발생한다. 그래서 일반인들이 보다 쉽게 웹상의 대량의 정보를 효율적으로 분류, 검색하기 위해 검색엔진, 에이전트 등의 많은 기술들이 개발되어 사용되고 있다.

인터넷에서 정보를 검색하는 작업은 이미 보편화되었지만, 네트워크에 산재되어있는 정보의 방대함과 대부분 정보검색에 사용하는 단어의 모호성으로 정확한 정보를 발견하기가 어렵다.

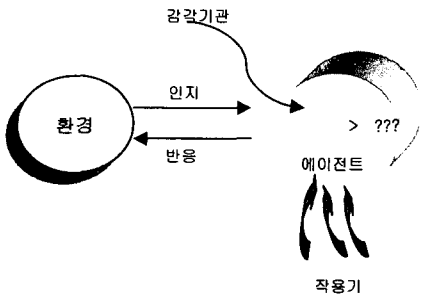
본 논문에서는 이러한 정보의 방대함과 모호성으로 정확한 정보를 발견하기 어려웠을 경우 시스템 관리자가 링크되어있는 다량의 문서 오류를 빠르고 쉽게 체크할 수 있는 시스템을 구현하고자한다.

본 논문에서 다룰 내용은 다음과 같다. 서론에 이어 2장에서는 에이전트의 정의와 형태분류에 대해 알아보고, 3장에서는 제안한 웹 Link-Fail 시스템을 설계하며, 4장에서는 실험 및 평가, 5장에서는 결론 및 향후 연구방향을 제시한다.

2. 에이전트 정의

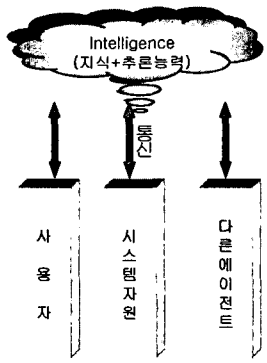
에이전트는 인공지능에서 오래 전부터 연구되어온 분야로서 1990년대 초부터 독립된 분야로 인식되기 시작한 개념이다. 에이전트는 여러 학자들에 의해 개념이 정립되었는데 대표적으로 S. Russell과 P.Norvig은 그림1과 같은 에이전트 시스템을 정의한다.

이 그림 1에서 "???" 부분은 가장 합리적인 에이전트를 의미하는 것으로 이 부분의 설계가 효과적인 에이전트 시스템을 구축하는 주요 요소가 된다.[2]



(그림 1) 에이전트 시스템

또한 M.Wooldridge와 N. Jennings는 에이전트가 자동성이나 다른 에이전트간 통신 능력,시간에 따른 환경변화에 대응하고 목표 지향적인 특성을 갖는 하드웨어 또는 소프트웨어 기반 컴퓨터시스템이라고 정의하였다. K.Takeda는 에이전트를 소프트웨어 에이전트와 하드웨어 에이전트로 나누고 소프트웨어 에이전트는 자동적으로 인간과 통신하여 작동하는 목표 지향적인 시스템이라고 하였고, 이에 학습 기능이 첨가되면 하드웨어 에이전트로 분류하였다.



(그림 2) 에이전트의 기본개념

에이전트는 그림2에서와 같이 인지, 반응, 목표, 환경이 중요한 요소가 되며 이들간을 묶는 시스템 구성이 중요하다. 따라서 그림3에서처럼 내부에 자신만의 지식 베이스와 추론 능력으로부터 사용자, 시스템 자원, 다른 에이전트와 통신하는 능력이 에이전트의 기본개념이다.[3]

3. 에이전트의 형태 분류

S. Russell과 P.Norving은 에이전트의 여러 형태들을 제시해 놓았는데 바람직한 에이전트의 구성 및 기능은 다음과 같다.

유형1) 단순한 반응형태의 에이전트(simple reflex agent)이다. 이 형태의 에이전트는 자신의 지식베이스에서 인지된 상태와 정확히 일치하는 반응만을 수행한다. 즉, "조건부-결론부"의 관계가 명확해야 한다는 것이다. 그러나 일반적으로 인지할 수 있는 환경은 비결정적이고 동적이기 때문에 항상 이 관계가 성립된다고 볼 수 없으므로 응용범위가 매우 좁은 단점을 갖는다.

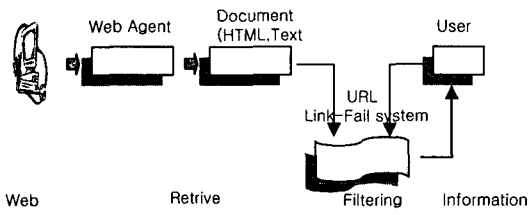
유형 2) 실세계의 여러 경우를 고려할 수 있는 에이전트(agent that keep track of the world)이다. 이 형태의 에이전트는 인지된 상태의 범위에 관한 내부 지식을 갖고있다. 즉 인지된 상태가 지식 베이스의 조건부와 정확히 일치하는 것이 없더라도 인지된 상태의 범위로부터 유사한 결론부를 찾아낼 수 있게 된다. 그러나 내부지식에 대한 유지 또한 메모리나 크기면에서 많은 오버헤드를 필요로 한다.

유형 3) 목표 지향적 에이전트(goal-based agent)이다. 이 형태의 에이전트는 인지에 대한 반응은 목표가 주어졌을 경우 보다 정확히 수행할 수 있다는 것을 기본적인 전제로 하고 있다. 그러나 모든 인지 가능한 환경 자체가 비 결정적 ,동적 특성 외에는 접근하기가 쉽지않다는 특성을 갖고있으므로 항상 목표가 주어질 수만은 없다는 것 또한 단점이 될 수 있다.

유형 4) 융통성을 갖는 에이전트(utility-based agent)이다. 이 형태의 에이전트는 인지에 대한 반응을 목표에 대해 얼마만큼 만족하는지를 사용자 중심으로 그 목표에 대한 만족도를 수치화하고 있다. 따라서 다른 형태들보다 융통성을 갖고 있다. 그러나 역시 (유형 4)에서와 같은 점과 더불어 수치화 척도등에 대한 객관성등이 해결해야 할 과제이다.[4][5]

4. 제안한 웹 link-fail 시스템 설계

본 논문에서 구현한 시스템의 구성도는 그림3과 같다. 해당 정보를 웹 에이전트를 이용하여 수집하고 각 해당 정보들을 사용자의 관심도와 함께 필터링 과정을 거쳐 사용자가 짧은 시간에 다량의 문서중 link-fail 정보를 제공하는 것이 본 시스템의 목적이다.



(그림 3) 시스템 구성도

본 논문에서 구현한 Link-Fail 웹 에이전트의 주된 기능은 웹 서버상의 정보를 가져와 HTML 테그 제거 등의 전처리 과정을 거쳐 필터링 부분의 입력으로 보내는 것이다. 웹 에이전트는 URL Manager, Information Collector의 모듈과 Robotparser 모듈 로 구성되어있다. Information Collector는 웹 서버에 접속을 하여 웹 문서를 수집하고,URL Manager는 웹 문서에 링크 되어진 웹 서버의 주소를 추출하고 Robotparser는 매개시스템으로부터 전송된 질의를 해석하고 정보 원천에 적합한 질의 언어로 바꾸는 역할을 한다.

또한 상대 시스템의 과부하를 막기위한 방법으로 페이지 요구문 사이에 지연문을 넣거나 robotx.txt 파일을 참고하여 로봇 배제 규약을 준수하도록 하였다.

본 논문에서 적용한 알고리즘적 기술은 표1과 같다.

5. 실험 및 평가

본 논문에서 제안한 위의 알고리즘을 적용하여 Link-Fail 시스템으로 먼저 사용자들이 많이 접속하는 http://www.daum,net 사이트에 접속하여 Link-Fail 오류 문서가 몇 개나 되는지 실험을 해보았다. 실험에 적용한 결과 total 294개의문서 중 10개의 Link-Fail 문서가 검출되었으며 해당 사이트로 직접 이동하여 검색해 본 결과 본 논문에서 구현한 시스템에서 적용하여 찾아낸 문서 모두가 Link-Fail 문서임을 검증할 수 있었다. 검증된 문서 결과는 그림4와 같다.

6. 결론 및 향후 연구 방향

기존의 웹 정보시스템에서는 정보가 많아짐으로서 오류 링크 문서를 찾는데 한계가 있었다.

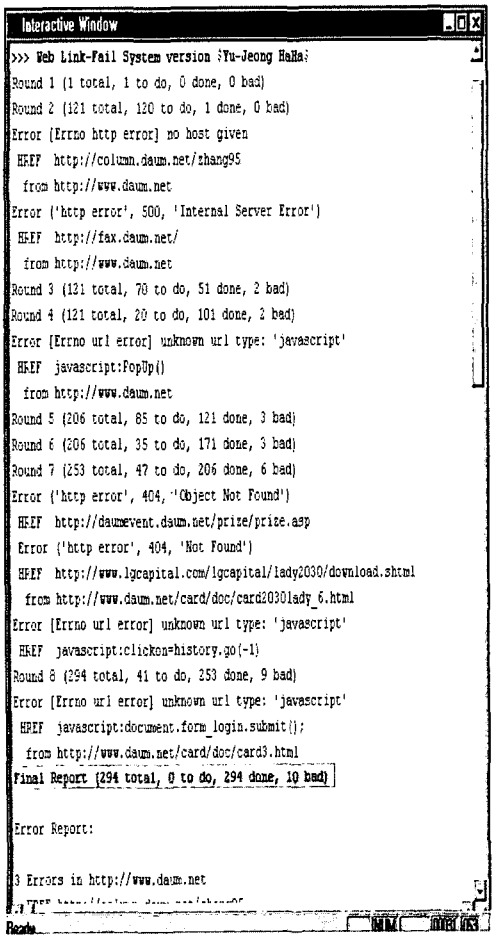
본 논문에서 제안한 Link-Fail 시스템으로 체크하여 다량의 문서를 동시에 작업 처리하여 Link-Fail 된 문서만 찾아 관리자로 하여금 빠른 시간내에 수정

```
//
def main():
    checkext = CHECKEXT
    verbose = VERBOSE
    maxpage = MAXPAGE
    roundsize = ROUNDSIZE
    dumpfile = DUMPFILE
    restart = 0
    norun = 0
//
class Checker:
    checkext = CHECKEXT
    verbose = VERBOSE
    maxpage = MAXPAGE
    roundsize = ROUNDSIZE
    nonames = NONAMES
    validflags = tuple(dir())
//
class MyHTMLParser(sgmllib.SGMLParser):
   def __init__(self,url, verbose=VERBOSE,
checker=None):
        self.myverbose = verbose
        self.checker = checker
        self.base = None
        self.links = {}
        self.names = []
        self.url = url
        sgmllib.SGMLParser.__init__(self)
```

(표 1) 제안된 알고리즘

보완할 수 있도록 알려줌으로서 효율적인 측면과 시간절약 측면에서 커다란 향상을 가져올 수 있었다.

앞으로 다 방향 에이전트 연구와, 검색 시스템과 원격관리 모듈간의 작업에서의 보안관련 사항에 대한 연구가 진행되어져야 한다고 사료된다.



(그림 4)제안한 Link-Fail 시스템 구현 결과

참고문헌

[1]Gio Wiederhold, "Extrapolating Trends for Information Technology" 첨단정보기술연구소 소식지 2호,2000.

[2]이달상,이춘근,"사이버 스페이스 상에서멀티에이전트 구축을 위한 연구",멀티미디어 학회,'98 춘계학술발표논문집,pp.63

[3]S.Russell, P.Norving,op,cit,pp.35-49

[4]조영임,"지능적 추론 및 통신 방법에 의한 지식 기반 다중 에이전트 시스템",평택대학교 논문집,제9집2호,pp.477-489,1997

[5]Daniela Florescu, Alon Levy and Alberto Mendelzon,"Database Techniques for the World Wide Web: A Survey"ACM SIGMOD Record 27:3, September 1998,59-74.