웹 서버 로그를 이용한 웹 시스템 행위 모델 자동 생성

윤영동, 배정호, 채흥석
부산대학교 컴퓨터공학과
부산광역시 금정구 장전 2 동 부산대학교 자연대연구실험동 305 호
{crayyd, jhbae83, hschae} @pusan.ac.kr

Automatic generation of web system behavioral model using web server log

Young-Dong Yoon, Jung-Ho Bae, Heung-Suk Chae
Dept. of Computer Science and Engineering, Pusan National University

요 약

웹 시스템의 사용자들이 웹을 이용하면 그에 대한 기록이 웹 서버에 남는다. 기록된 웹 서버 로그를 분석하면 웹 시스템에 어떤 부류의 사용자가 있는지 각 사용자들이 어떤 방식으로 웹 시스템을 사용하는지 알 수 있다. 이러한 사용자들의 행위를 행위 모델로 표현할 수 있다면 웹 시스템에 대한 명세가 없더라도 행위 모델을 분석하여 시스템의 명세를 추정하고 테스트하기 용이해진다. 뿐만 아니라 웹 시스템 사용자들의 사용자 별 이용 패턴 및 통계정보 역시 알 수 있다. 그러나 웹 서버 로그를 일일이 담당자가 분석하여 행위 모델을 생성하기에 웹 서버 로그는 그 양이 너무 방대하다. 따라서 본 논문은 웹 서버 로그를 이용하여 웹 시스템의 행위 모델을 자동 생성하는 방법에 대해서 제시한다.

1. 서론

UML 에서는 객체들의 행위들에 대해 묘사한 모델을 행위 모델(Behavioral model)이라고 한다. 클래스나 데이터의 집합과 같은 구조 모델(Structural model) 만으로는 어떤 시스템의 소프트웨어를 표현할 수 없다. 따라서 소프트웨어의 각 부분이 각 사용자들에 의해 어떠한 제어 흐름을 가지고 어떠한 방식으로 동작하는지 소프트웨어의 행위에 대한 부분을 명시한 것이 행위 모델이다. 행위 모델은 정형 검증, 테스트, 코드 자동 생성 등 다양한 분야에서 활용된다.

이러한 행위 모델을 생성하는 방법은 두 가지이다. 하나는 명세를 통해 명세가 요구하는 이상적인 시스템의 행위를 모델링 하는 방법이고 다른 하나는 실제로 시스템이 사용된 내역을 통해 이미 행해진 행위를 모델링 하는 방법이다.

명세를 알 수 없는 어떤 시스템이 있다면 해당 시스템이 사용된 내역 즉, 시스템 로그를 통해 행위 모델을 생성할 수 있다. 시스템이 만약 웹 시스템이라면 해당 시스템의 로그로는 웹 로그가 있다. 웹 로그를 통해 웹 시스템의 행위 모델을 생성할 수 있다면 행위 모델의 분석을 통해 웹 시스템의 명세에 대해 추정할 수 있을 뿐만 아니라 각각의 사용자들에 의해 어떠한 패턴으로 사용되고 있는지에 대한 통계나 경향 정보 등과 같은 명세에는 없는 정보 역시 얻을 수 있다. 이러한 정보들을 얻을 수 있다면 시스템의 어떤 부분에 보완이나 수정이 필요한지도 알아낼 수 있기 때문에 시스템의 유지 보수 및 개선사항 도출 역시 용이해진다.

하지만 명세가 없는 시스템에서 이러한 행위 모델을 생성하기 위해 시스템의 로그를 직접 분석하는 것은 크게 두 가지 문제점을 가진다. 첫 번째는 사람의 손으로 수행하는 작업의 필연적인 문제인 오류가 발생하기 쉽다는 점(error-prone)이고 두 번째는 복잡한 시스템일수록 행위 모델 생성에 걸리는 기간이 기하급수적으로 길어진다는 점(time-consuming)이다. 행위 모델은 시스템의 모든 실행 가능한 행위들의 흐름을 표현하므로 복잡도가 높다. 시스템에 따라 수천 수만 가지 이상일 수 있는 행위의 흐름을 수동 작업하는 것은 현실적으로 어렵다. 따라서 자동으로 완전하고 정확한 행위 모델을 생성하기 위한 기법의 연구가 필요하다.

본 연구에서는 웹 서버의 로그를 이용하여 행위 모델을 자동 생성하는 방법에 대하여 제시한다.

2. 관련연구

웹 로그를 이용하여 웹 사용자의 의미 있는 패턴, 프로파일, 추세 등에 대한 정보를 발견하기 위한 데이터 마이닝 기술을 웹 마이닝이라고 한다. [1]은 로그 데이터를 이용해서 path traversal tree 를 생성하고 traversal 의 빈도를 분석해서 데이터 마이닝을 하는 법을 소개한다. Path 의 빈도를 분석하여 트리를 생성하는 내용은 유용할 수 있으나 별도로 생성하는 모델이 없고 사용자의 타입이 다양한 경우에 대한 대응이 없다. [2]는 웹 서버 로그뿐만 아니라 다양한 루트를 통해 5 단계의 수집가능한 로그 정보를 제시한다. 표 1 는 [2]에서 제시한 5 단계의 웹 로그를 설명한다.

<표 1> Summary of logging features and mining opportunities[2]

Level	Logged information	Logging facilities	Mining opportunities
1	-consumer IP -invokedWS -timestamp -HTTP status code	-Web server	-service utilization -consumer tracking -failure rates
2	-level 1 + -SOAP request & response -timestamps	-HTTP listener & logger	-level 1 + -WS execution time -analysis of SOAP messages
3	-invoked WS & operation -SOAP request & response	-WS container -SOAP handlers	-level 2
4	-level 3 + -consumer-side activity	-WS container -SOAP handlers	-level 3 + -client-side activity
5	-level 4 + -workflow information	-level 4 + -Web services	-level 4 + -Web services workflows

Logged information 은 해당 단계에서 수집되어야 할 로그 정보이고 Logging facilities 는 해당 단계의 로그가 수집되기 위해 시스템이 갖추어야 하는 요소들이다. Mining opportunities 는 해당 단계의 로그를 통해 유추해낼 수 있는 정보를 의미한다. [2]에서 제시하는 다양한 로그 수집 방법을 이용하면 상대적으로 많은 정보를 확보하여 높은 수준의 데이터 마이닝이 가능하다. 그러나 1 단계의 로그를 제외한 나머지는 웹 서버 이외에 시스템이 해당 로그를 수집할 수 있는 별도의 환경을 갖추어야 하기 때문에 일부의 웹 시스템을 제외하면 적용하기 어렵다.

3. 연구 배경

3.1 웹 서버 로그

웹 서버의 종류에 따라 웹 로그의 포맷은 조금씩 다를 수 있지만 중요한 부분은 표준으로 지정되어 있다. 아파치 웹 서버는 NCSA 에서 지정한 표준인 NCSA Common Log Format(CLF)와 NCSA Extended Log Format(ELF) 포맷을 지원한다. CLF 포맷은 아파치 웹 서버를 설치하면 기본적으로 적용되는 포맷으로 액세스 로그 정보만 있는 것이고 ELF 포맷은 CLF 포맷에서 리퍼럴 로그와 에이전트 로그가 추가된 포맷이다.

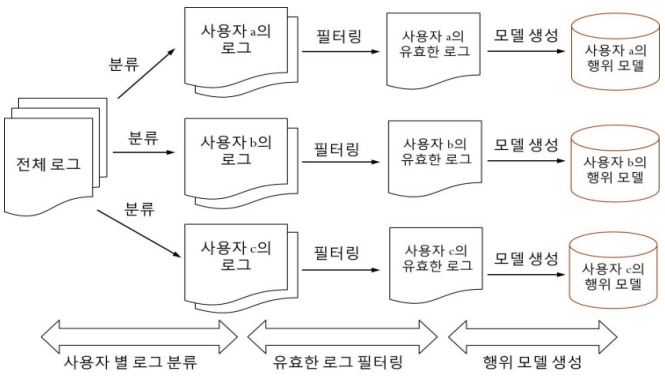
본 연구에서는 CLF 와 ELF 포맷의 로그 모두를 고려한다.

3.2 행위 모델

본문의 행위 모델은 상태 기계 표기법(State machine notation)을 따른다. 상태 기계는 시스템이나 컴포넌트, 객체의 행위를 표현하기 위해 사용되는 UML 의 표준 동적 행위 모델이다. 상태 기계는 상태, 전이 그리고 실행 가능한 이벤트로 구성되어 있다. 상태는 초기상태와 상태로 나뉘어 진다. 초기 상태는 최초의 상태를 의미하고 그 이후의 상태는 초기 상태를 제외한 상태이다.

4. 행위 모델의 생성 절차

행위 모델 생성 절차는 사용자 별 로그 분류, 유효한 로그 필터링, 행위 모델 생성으로 구성된다. 사용자 별 로그 분류는 전체 웹 서버 로그를 사용자 별로 나누는 절차이다. 유효한 로그 필터링은 분류된 사용자 별 로그에서 행위 모델 생성에 유효한 로그만을 필터링하는 절차이다. 행위 모델 생성은 앞의 절차에서 생성된 각 사용자 별 유효한 로그를 통해 행위 모델을 생성하는 절차이다.



(그림 1) 행위 모델 생성 절차

그림 1 은 행위 모델 생성 절차를 도식화한 그림이다.

4.1 사용자 별 로그 분류

각 사용자 별로 행위 모델을 생성하기 위해서는 먼저 전체 로그를 사용자 별로 분류할 필요성이 있다.. 사용자 별 로그 분류 절차는 원격 호스트 IP, 원격 사용자 원격 로그인 명, 참조한 페이지, 사용자 에이전트 정보를 조합하여 그림 5 와 같이 각각의 사용자를 나누고 전체 로그를 각 사용자 별로 분류한다. CLF 포맷의 경우에는 원격 호스트 IP, 원격 사용자, 원격 로그인 명만을 이용하여 사용자를 구분하고 ELF 포맷의 경우 참조한 페이지(referrer)와 사용자 에이전트 정보(user agent) 또한 고려하여 사용자를 구분한다. 웹 서버의 운영에 있어서 사용자들을 임의로 등록하지 않는 한 원격 사용자와 원격 로그인 명으로는 구분되지 않기 때문에 CLF 포맷의 경우에는 원격 호스트의 IP 가 주된 사용자 구분 기준이고 ELF 포맷의 경우에는 사용자 에이전트 정보를 추가로 고려하기 때문에 좀 더 정확한 사용자 구분이 가능해진다.

<표 2> 실제 사용중인 웹 시스템의 예제 로그

```
164.125.34.127 - - [22/Nov/2012:15:07:20 +0900] "GET /index.php HTTP/1.1" 200 21404 31200
164.125.34.127 - - [22/Nov/2012:15:07:20 +0900] "GET /style.css HTTP/1.1" 304 - 0
164.125.34.127 - - [22/Nov/2012:15:07:20 +0900] "GET /img/title_body_quick.jpg HTTP/1.1" 304 - 0
95.67.106.110 - - [22/Nov/2012:15:07:27 +0900] "GET /board.php?id=see_0801_proposala_new HTTP/1.1" 200 22553 107762
95.67.106.110 - - [22/Nov/2012:15:07:27 +0900] "GET /view.php?id=see_0801_proposala_new&lan=&page=10&sn1=&divpage=1&sn=off&ss=on&sc=on&select_arrange=hit&desc=desc&no=18136 HTTP/1.1" 200 14351 242601
95.67.106.110 - - [22/Nov/2012:15:07:32 +0900] "POST /comment_ok.php HTTP/1.1" 200 11812 1638003
69.163.96.25 - - [22/Nov/2012:15:07:33 +0900] "GET /view.php?id=see_0801_proposala_new&lan=&page=10&sn1=&divpage=1&sn=off&ss=on&sc=on&select_arrange=hit&desc=desc&no=18136 HTTP/1.1" 200 170827 2090404
```

```
164.125.34.127 - - [22/Nov/2012:15:07:45 +0900] "GET /main.php HTTP/1.1" 200 9116 31200
164.125.34.127 - - [22/Nov/2012:15:07:57 +0900] "GET /skin/nzeo_ver3_modified_garu/style.css HTTP/1.1" 304 - 0
164.125.34.127 - - [22/Nov/2012:15:07:57 +0900] "GET /skin/nzeo_ver3_modified_garu/btn_memobox.gif HTTP/1.1" 304 - 0
164.125.34.127 - - [22/Nov/2012:15:08:04 +0900] "GET /board.php?id=CCD_1202_notice HTTP/1.1" 200 42432 93600
164.125.34.127 - - [22/Nov/2012:15:08:45 +0900] "GET /view.php?id=CCD_1202_notice&lan=&page=1&sn1=&divpage=1&sn=off&ss=on&sc=on&select_arrange=headnum&desc=asc&no=22 HTTP/1.1" 200 21523 140400
164.125.34.127 - - [22/Nov/2012:15:09:42 +0900] "POST /bbs/comment_ok.php HTTP/1.1" 200 11812 187200
164.125.34.127 - - [22/Nov/2012:15:09:42 +0900] "GET /view.php?id=CCD_1202_notice&lan=&page=1&sn1=&divpage=1&sn=off&ss=on&sc=on&select_arrange=headnum&desc=asc&no=22 HTTP/1.1" 200 31727 241400
164.125.34.127 - - [22/Nov/2012:15:09:50 +0900] "GET /board.php?id=CCD_1202_week1 HTTP/1.0" 200 32451 405601
```

표 2 은 실제 사용중인 시스템의 웹 서버 로그 예제이다. 아파치 서버 2.2 를 사용하며 CLF 포맷으로 웹 서버 로그를 기록한다. 해당 로그는 CLF 포맷이기 때문에 ip 를 통해 사용자를 구분한다. 예제 로그에서 164.125.34.127 의 ip 에 해당하는 사용자 로그만을 분류하고 간략히 나타내기 위해 요청 메소드의 세부 패러미터 부분을 생략하면 표 3 과 같은 결과를 얻을 수 있다. 즉 표 3 는 ip 로 164.125.34.127 을 사용하는 사용자의 로그 요약이다.

<표 3> 164.125.34.127 사용자의 로그 요약

요청 시간	요청 메소드 및 요청 파일
22/Nov/2012:15:07:20	GET /index.php
22/Nov/2012:15:07:20	GET /style.css
22/Nov/2012:15:07:20	GET /img/title_body_quick.jpg
22/Nov/2012:15:07:45	GET /main.php
22/Nov/2012:15:07:57	GET /skin/nzeo_ver3_modified_garu/style.css
22/Nov/2012:15:07:57	GET /skin/nzeo_ver3_modified_garu/btn_memobox.gif
22/Nov/2012:15:08:04	GET /board.php
22/Nov/2012:15:08:45	GET /view.php
22/Nov/2012:15:09:42	POST /bbs/comment_ok.php
22/Nov/2012:15:09:42	GET /view.php
22/Nov/2012:15:09:50	GET /board.php

표 3 을 보면 164.125.34.127 의 ip 를 사용하는 사용자의 브라우저에서 웹 서버에 요청한 정보와 요청한 시간을 알 수 있다. 예제에서 확인 가능한 요청 내역으로는 특정 페이지에 대한 요청, 웹 페이지를 꾸미는 그림 파일에 대한 요청, 스타일시트 파일에 대한 요청이 있다. 요청 메소드가 GET 인 것은 웹 서버로부터 읽어 오기 위한 요청이고 POST 인 것은 웹 서버에 쓰기 위한 요청이다. 각 사용자 별로 행위 모델을 생성하고 분석하기 위해서는 이와 같은 사용자별 로그 분류가 선행되어야 한다.

4.2 유효한 로그 필터링

웹 서버에 대한 모든 요청이 행위 모델 생성에 있어서 유효한 로그는 아니다. 로그에서 상태(state)나 전이 이벤트(Transition event) 또는 액션(action)을 규정지을 수 있는 최소한의 요청은 바로 해당 행위를 구현한 웹 페이지에 대한 요청이다. 웹을 꾸미기 위한 그림파일이나 스타일시트 파일 등은 이에 해당하지 않으므로 행위 모델 생성에 있어서 유효한 요청이 아니라고 볼 수 있다. 따라서 사용자 별로 분류된 로그 중에서도 유효한 로그는 웹 페이지에 대한 요청을 포함하는 로그이므로 웹 페이지를 기술한 파일을 요청하는 로그만을 유효한 로그로 판단한다. 본문에서는 "jsp, php, html" 로 기술한 웹 페이지만을 대상으로 하였다. 표 4 은 표 3 의 로그에서 웹 페이지에 요청이 포함된 로그 즉, 유효한 로그만 필터링한 것이다.

<표 4> 164.125.34.127 사용자의 유효한 로그 요약

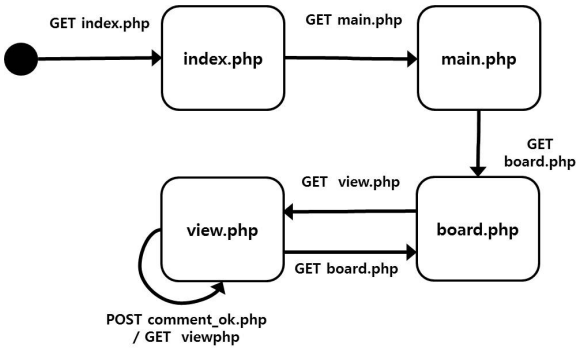
요청 시간	요청 메소드 및 요청 파일
22/Nov/2012:15:07:20	GET /index.php
22/Nov/2012:15:07:45	GET /main.php
22/Nov/2012:15:08:04	GET /board.php
22/Nov/2012:15:08:45	GET /view.php
22/Nov/2012:15:09:42	POST /bbs/comment_ok.php
22/Nov/2012:15:09:42	GET /view.php
22/Nov/2012:15:09:50	GET /board.php

4.3 행위 모델 생성

유효한 로그에 대한 필터링이 끝났으면 남은 로그를 통해 초기상태, 상태, 전이이벤트, 액션을 생성한다. 행위 모델의 생성은 다음의 규칙을 따른다.

최근 한 시간 이내에 웹 서버에 로그 기록이 없는 ip 로 접속한 사용자는 웹 서버에 로그 기록을 남기는 순간 초기상태가 된다. 즉 마지막 행위를 수행한 후 한 시간을 초과하면 이후의 행위는 새로운 사용자의 행위로 간주한다. 사용된 로그의 포맷이 ELF 일 경우에는 ip, 사용자 에이전트 정보 두 가지를 고려하여 사용자를 구분한다.

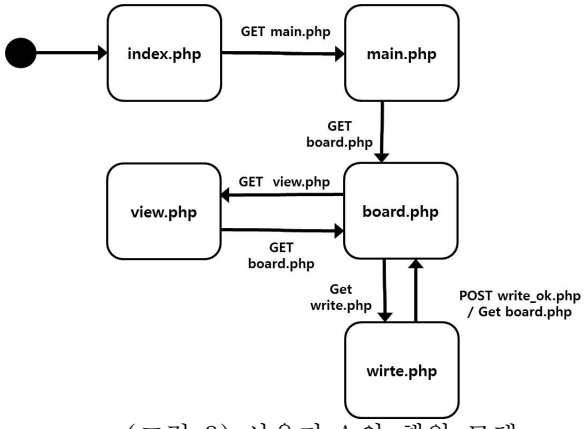
하나의 유효한 요청 이후 1 초 이내에 또 다른 유효한 요청이 있을 경우 전자의 요청은 전이 이벤트로 간주하지만 후자의 이벤트는 액션으로 간주한다. 이는 사용자가 1 초 이내에 두 가지 행위를 하지 않을 것이라는 가정과 실제 웹에서는 하나의 행위를 수행하기 위해 여러 개의 페이지가 사용될 수 있다는 사실을 고려한 것이다.



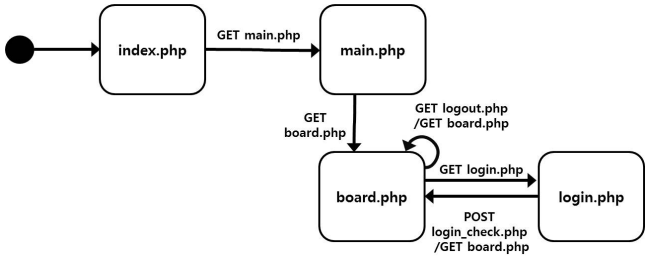
(그림 2) 예제 로그로 생성한 행위 모델

그림 2 은 표 3 에 의해 생성된 행위 모델이다. 정확하게 표현하면 164.125.34.127 ip 를 사용하는 사용자가 2012 년 11 월 22 일 15 시 7 분 20 초부터 2012 년 11 월 22 일 15 시 9 분 50 초 사이에 이 웹 시스템에 행한 행위를 모델링한 행위 모델이다.

그림 3 과 그림 4 은 이러한 행위 모델 생성 방법을 통해 이 웹사이트의 다른 사용자(사용자 A, 사용자 B)의 로그로 행위 모델을 생성한 것이다.

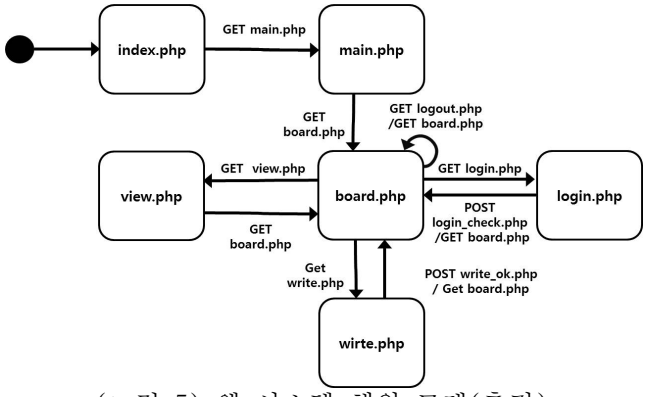


(그림 3) 사용자 A 의 행위 모델



(그림 4) 사용자 B 의 행위 모델

사용자 A 와 B 의 행위 모델을 164.125.34.127 ip 를 사용하는 사용자의 행위 모델과 합치면 그림 5 와 같은 행위 모델이 생성된다.



(그림 5) 웹 시스템 행위 모델(추정)

만약 웹 서버의 로그가 충분히 많고 그림 5 의 행위 모델이 웹 서버의 모든 로그에 대한 행위를 포함한다면 잠정적으로 이 모델이 시스템의 행위 모델이라 추정할 수 있다. 하지만 웹 서버의 모든 로그에 대한 행위를 포함하지 않는다면 해당 행위를 모델에 추가함으로써 시스템의 행위 모델을 좀더 완전하게 만들 수 있다.

5. 결 론

본 논문은 웹 시스템의 웹 서버 로그로부터 사용자를 분류하고 행위와 관련된 요소들을 필터링하여 행위 모델을 생성하는 기본적인 방법을 제시하였다. 웹 서버 로그만으로 웹 마이닝을 하는 것은 얻을 수 있는 정보의 한계가 있다. 하지만 웹 서버 로그가 아닌 표 1 의 level 2-5 에 해당하는 로그들은 반드시 로그를 수집할 수 있는 별도의 환경이 구축되어야 하기 때문에 해당 로그를 필요로 하는 마이닝 기법은 모든 웹 시스템에 적용하기에는 부적합하다. 웹 서버 로그만으로 필요한 데이터를 마이닝 할 수 있다면 모든 웹 서버 시스템에 해당 기법을 적용할 수 있을 것이다. 본 논문에서는 웹 서버 로그를 이용하여 웹 시스템 행위 모델을 자동 생성 하기 위한 방법을 제시하였지만 아직까지 실제 명세를 통해 생성한 행위와의 비교 검증을 위한 실험이 선결과제로 남아 있다. 또한 프로세스 마이닝 알고리즘을 적용하여 생성된 행위 모델과의 비교 연구도 필요하다. 따라서 웹 서버 로그에서 얻은 정보를 프로세스 마이닝 알고리즘에 적용할 수 있도록 웹 서버 로그를 워크 플로우 로그로 변환하는 기법에 대한 연구를 진행하고 프로세스 마이닝 알고리즘 적용 시 생성된 행위 모델의 장단점을 본 연구와 비교하는 연구 역시 진행할 것이다.

참고문헌

[1] M.-S. Chen, J. S. Park, and P. S. Yu, “Efficient data mining for traversal patterns,” IEEE Trans. on Knowledge and Data Eng., 10(2):209–221, March/April, 1998.

[2] S. Dustdar and R. Gombotz, “Discovering Web Service Workflows Using Web Services Interaction Mining,” International Journal of Business Process Integration and Management, 1(4):256-266, 2006

[3] R. Cooley, B. Mobasher, J. Srivastava, ”Web mining : Information and pattern discovery on the World Wide Web,” In: International Conference on Tools with Artificial Intelligence, pp.558-567, Nov, 1997.

[4] H. Mannila, H. Toivonen, “Discovering Generalized episodes using minimal ocurrences,” In: Proc. Second International Conference on Knowledge Discovery and Data Mining, Portland, Oregon, pp.146-151, 1996.