규칙 기반 라우팅 구성 장애 진단 알고리즘에 관한 연구

황태인*, 조강홍*, 정진욱*
*성균관대학교 전기전자 및 컴퓨터공학부
e-mail : tihwang@songgang.skku.ac.kr

A Study on the Algorithm for Rule-based Routing Configuration Fault Diagnosis

Tae In hwang*, Kang Hong Cho*, Jin Wook Chung*
*Department of Electric, Electronic and Computer Engineering
Sungkyunkwan University

요 약

이 논문에서는 시스템의 라우팅 구성 장애를 진단하기 위한 규칙과 알고리즘을 제시하였다. 라우팅 구성 장애 관리를 위하여 네트워크 구성 관리 규칙, 라우팅 구성 장애 진단 규칙을 제안하였으며 후향 추론 알고리즘을 기반으로 이런 규칙간의 상호 연동을 위하여 메타 규칙을 적용하였다. 제안한 규칙과 알고리즘을 시나리오에 기반하여 규칙, Blackboard, 목표의 변화 과정을 보여줌으로써 실험 결과를 제시하였다. 시스템의 TCP/IP 네트워크 구성 관리와 관련하여 시스템에서 발생할 수 있는 네트워크 장애들 중에서 라우팅 구성 장애를 진단하기 위한 규칙 및 추론 알고리즘을 제안함으로써 이질적이고 급변하는 네트워크 환경에 쉽게 대처할 수 있는 시스템 개발을 위한 방법론을 제시하고자 한다

1. 서론

컴퓨터 통신 기술이 발달함에 따라 네트워크는 점차 방대하고 복잡해지게 되었고 이에 따라 네트워크 장비들 뿐만 아니라 이런 장비들에 연결되어 있는 각기 다른 운영체제가 적재된 시스템들 역시 계속해서 증가하였다. 하지만 기하 급수적으로 증가하고 있는 이런 피관리 시스템들의 수에 비해 이들을 관리하기 위한 전문 지식을 갖춘 관리자는 터무니 없이 부족한 상태이다. 이로 인해 시스템의 네트워크 구성 장애를 자동으로 관리해줄 수 있는 시스템에 관한 연구가 필요하게 되었다[1][2].

현재까지 개발되어져 있는 시스템들은 대부분 일반적인 프로그래밍 기법에 의해서 네트워크 구성 장애를 자동으로 진단하고 복구하는 시스템이다. 하지만, 현재와 같이 이질적이고 계속해서 변화해 가는 네트워크 환경에서 정적으로 프로그램된 시스템들은 적응력이 떨어지며 변화해 가는 네트워크 환경에 대처하기 힘들다. 그렇기 때문에 이런 시스템은 바람직하지 않으며 규칙이나 사례 기반 시스템에 관한 연구가 필요하다.

현재까지 네트워크 장애 관리를 위한 시스템으로 LODES 와 CRITTER 가 구현되었다[3][4]. LODES 는 LAN 상의 장애 검출 및 위치 확인을 위한 규칙 기반 전문가 시스템이며, CRITTER 는 네트워크를 사례에 기반하여 관리하기 위한 시스템이다. 하지만 이런 구현된 시스템들과 관련 연구들[5][6][7]에 따르면 장애가 발생한 네트워크의 위치를 추적하고 진단하기 위한 연구는 현재 수행된 상태이지만 규칙 및 사례에 기반하여 시스템의 네트워크 구성 장애를 자동으로 진단하고 복구하기 위한 연구는 이루어 지지 않았다.

이 논문에서는 시스템의 TCP/IP 네트워크 구성 관리와 관련하여 시스템에서 발생할 수 있는 네트워크 장애들 중에서 라우팅 구성 장애를 진단하기 위한

규칙 및 추론 알고리즘을 제안하고 이를 통해 이질적이고 급변하는 네트워크 환경에 쉽게 대처할 수 있는 시스템 개발을 위한 방법론을 제시하고자 한다

2. 라우팅 구성 장애 진단 규칙

생성 규칙은 전문지식을 나타내는 가장 빠른 방법이며 이 방법은 전문가 시스템에서 가장 많이 사용되고 있다. 사실과 규칙을 이용하여 결론에 도달하며 이 결론은 또한 새로운 사실이 된다. 이 논문에서는 후향 추론을 기반으로 한다. 추론 방법의 선택은 전문가 시스템을 응용하는 분야의 특성에 따라서 선택이 이루어지게 된다. 전향 추론은 설계 및 계획형 문제에 주로 이용되며 후향 추론 방식은 진단 및 분류 등의 문제에 주로 이용된다. 라우팅 장애를 진단하고 복구하기 위한 추론 방법으로는 후향 추론이 적합하다.

제안한 규칙은 추후 확장을 고려하여 문제를 네트워크 구성 장애로 정의하고 장애 유형은 다음과 같이 네 가지로 정의하였다.

표 1. 네트워크 구성 장애 유형

장애유형	설명
라우팅 구성장애	디폴트 라우터 설정 오류 및 동적 라우팅 실패로 인하여 발생하는 외부 네트워크와의 통신 장애를 말한다.
도메인 네임 서비스 구성장애	도메인 네임 서버 설정 오류로 인하여 도메인 네임 을 IP 주소로 변환하지 못하기 때문에 발생하는 장애를 말한다.
인터넷 데몬 프로세스 구성장애	데몬 프로세스 구동 실패로 인하여 응용 프로그램이 올바르게 동작하지 않기 때문에 발생하는 장애를 말한다.
IP 중복 장애	서로 다른 시스템이 같은 IP 를 사용하고 있을 경우 발생할 수 있는 장애를 말한다.

규칙을 표현하기 위해 후향 추론 방식을 지원하는 LEVEL 5 쉘에서 제공하는 DISPLAY, CHAIN 명령을 이용하였다. DISPLAY 명령을 사용하여 중간에 중요한 정보들을 출력하였다. 또한 각각 분리되어 있는 전문가 모듈 사이에 제어를 넘겨주고 받기 위해 CHAIN 명령을 사용한 메타 규칙을 적용하였다. 각 모듈 사이에 통신은 Blackboard 를 이용하였으며, Blackboard 는 각 모듈에 의해 결정된 문제에 대한 정보를 담고 있으며 이 정보는 문제 해결을 위해 다른 모듈에서 사용되어진다[8].

이 논문에서는 규칙을 두 개의 모듈 즉, 네트워크 구성 관리 규칙 모듈, 라우팅 구성 장애 진단 규칙 모듈로 나누었으며 각각의 규칙을 표기하기 위해 다음과 같이 정의하였다.

[정의 1] 각 모듈이 추구하는 목표의 집합을 G 라고 정의하며 원소의 집합은 다음과 같다. G = { g11, g12, g211, g212, g221, g222 }

[정의 2] 현존 사실과 추론에 의해 새로 생성된 사실들의 집합을 F 라고 정의하며 원소의 집합은 다음과 같다. F = { f1, f2, f3, f4, f5, f6, f7, f8, f9, f10 }

[정의 3] 추론 중에 어떤 사실을 얻기 위해 필요한 질의함수의 집합을 Q 라고 정의하며 원소의 집합은 다음과 같다. Q = {q1, q2, q3, q4, q5, q6, q7, q8, q9, q10 }

[정의 4] 질의함수 수행 후에 발생할 수 있는 결과값 즉 사건의 집합을 E 라고 정의하며 원소들의 집합은 다음과 같다. E = { unix, win, linux, up, down, exist, notexist, noresponse, response , success, fail, match, notmatch }

[그림 1]과 [그림 2]는 위에서 설명한 명령어와 정의들을 이용하여 실제로 라우팅 구성 장애 진단 규칙을 표현한 그림이다.

M1 모듈

```
GOAL  g11, g12

RULE 1 : IF  q1 = UNIX  THEN  g11  AND  f1
RULE 2 : IF  q2 = fail  AND  f1
         THEN  g12  AND  f2  AND  CHAIN(M21)

END
```

g11 : 시스템 유형 판별 g12 : 장애 유형 판별 f1 : 시스템 유형 = 유닉스 f2 : 장애 유형 = 라우팅 구성 장애	q1 : 시스템 운영체제 검사 q2 : 1 홉까지의 경로 추적 결과 검사

그림 1. 네트워크 구성 관리 규칙 모듈

[그림 1]의 네트워크 구성 관리 규칙 모듈의 목표는 시스템의 운영체제를 판별하고 발생한 장애가 네트워크 구성 장애 유형 중 어떤 장애 유형에 속하는지를 판별하는 것이다.

```
GOAL  g211, g212

RULE 3 : IF  q3 = down
         THEN  g211  AND  f3  AND  DISPLAY (f3)
RULE 4 : IF  q4 = exist
         THEN  g211  AND  f4  AND  DISPLAY(f4)
RULE 5 : IF  q5 = notmatch
         THEN  g211  AND  f5  AND  DISPLAY(f5)
RULE 6 : IF  q6 = notmatch
         THEN  g211  AND  f6  AND  DISPLAY(f6)
RULE 7 : IF  q5 = match
         THEN  g211  AND  ¬ f5
RULE 8 : IF  q6 = match
         THEN  g211  AND  ¬f6
RULE 9 : IF  q7 = noresponse
         AND  q8 = exist  AND  ¬f5  AND  ¬ f6
         THEN  g21  AND  f7  AND  DISPLAY(f7)
RULE 10 : IF  q8 = notexist
         THEN  g211  AND  f8  AND  DISPLAY(f8)
RULE 11 : IF  q9 = exist  OR  q10 = exist
         THEN  f8  AND  f9  DISPLAY(f9)
RULE 12 : IF  q9 = notexist  AND  q10 = notexist
         THEN  f8  AND  f10  DISPLAY(f10)
RULE 13 : IF  CHAIN(M22)  THEN  g212

END
```

g211 : 라우팅 장애 유형 판별 g212 : 장애 복구 모듈로 전이 f3 : 장애 원인 = 인터페이스 다운 f4 : 장애 원인 = 케이블 연결 상태 불량 f5 : 장애 원인 = 네트워크 주소 변경 f6 : 장애 원앤 = 서브넷 마스크 변경 f7 : 장애 원인 = 디폴트 게이트웨이 다운 또는 디폴트 게이트웨이 주소 변경 f8 : 장애 원인 = 디폴트 엔트리 부재 f9 : 장애 원인 = 디폴트 엔트리 삭제 f10 : 장애 원인 = 라우팅 데몬 프로세스 다운	q3 : 인터페이스 상태 검사 q4 : 로그 정보에 "disconnected cable" 엔트리 존재 유무 검사 q5 : 백업된 네트워크 주소와 현재 네트워크 주소 일치 여부 검사 q6 : 백업된 서브넷 마스크 주소와 현재 서브넷 마스크 주소 일치 여부 검사 q7 : 디폴트 게이트웨이 주소로의 ping 수행 결과 검사 q8 : 라우팅 테이블에 "default" 또는 "0.0.0.0" 엔트리 존재 유무 검사 q9 : Defaultrouter 파일 존재 유무 검사 q10 : 라우팅 데몬 프로세스 존재 유무 검사

그림 2. 라우팅 구성 장애 진단 규칙 모듈

[그림 2]의 라우팅 구성 장애 진단 규칙 모듈의 목표는 시스템내의 라우팅 구성 장애를 유발한 원인이 무엇인지를 식별하는 것이며 식별된 원인을 라우팅 장애 복구 규칙 모듈에 전달하는 것이다.

위에서 제안한 규칙들을 이용하여 새로운 사실을 유추해내기 위해서는 추론 알고리즘이 필요하다. 이 논문에서는 후향 추론 알고리즘을 기반으로 다른 규칙들이 추후에 서로 연결 가능할 수 있도록 하기 위하여 메타 규칙을 적용하여 라우팅 구성 장애를 진단하기 위한 알고리즘을 다음과 같이 제시한다.

```
Initialization :
    Table Blackboard
    Table RuleTable
    Stack GoalTable
    GoalTable.Push(M.G)

while(!GoalTable.Empty())
{
   GoalEntry = GoalTable.Pop();
   while((TD = Scan(RuleTable.CC,GoalEntry)) != NotFound)
   {
      if(TD.PC == TU)
      {
         Blackboard.Add(CC.F);
         if(TD.CC == CHAIN(M))
         {
           M = RuleTable.Transition();
           GoalTable.Push(M.G);
         }
         TD.Fire();
      }
      else if(TD.PC == FA) TD.Discard();
      else if(TD.PC == CHAIN(M))
      {
         RuleTable.Transition(M);
         GoalTable.Push(M.G);
      }
   }
}
```

G = Goals F = Facts M = Modules A = active rule D = discarded rule TD = triggered rule	CC = conclusion clause PC = premise clause FD = fired rule FR = free clause FA = false clause TU = true clause	Scan(CC, GoalEntry) : CC 와 Goal 이 일치하는 active rule 스캔 Add() :테이블에 엔트리 추가 Discard() : rule 을 discard 시킴 Transition() : 다른 모듈로 전이 Pop() : GoalTable 스택에서 있는 Goal 을 Pop 함 Push() : GoalTable 스택에 Goal 을 Push 함 Empty() : GoalTable 스택이 비어 있는지 검사 Fire() : Rule 을 fire 시킴

그림 3. 라우팅 구성 장애 진단을 위한 후향 추론 알고리즘

[그림 3]은 라우팅 구성 장애 관리를 위한 후향 추론 알고리즘이다. 초기화 과정에서 GoalTable, Blackboard, RuleTable 을 생성한다. GoalTable 에 첫 번째로 실행되는 생성규칙 모듈의 G(Goals)을 추가한다. 목표 항목(GoalEntry)과 RuleTable 의 결론부(RuleTable.CC)가 일치하는 규칙이 있는지 찾는다. 만약 여러 개가 있다면 첫번째 규칙을 적용(trigger)한다. 적용된 규칙은 TD 상태로 바뀐다. 만약 적용된 규칙의 조건부(TD.PC)가 참이라면 Blackboard 에 결론부의 사실들(CC.F)을 추가하고 적용된 규칙은 FD(fired rule) 상태로 바뀐다. 만약 적용된 규칙의 결론부에 CHAIN 명령이 있다면 다른 RuleTable 로 전이하며 GoalTable 에 바뀐 생성규칙 모듈의 G 을 추가한다. 만약 적용된 규칙의 조건부가 거짓이라면 적용된 규칙은 폐기(Discard)된다. 만약 적용된 규칙의

조건부에 CHAIN 명령이 있다면 결론부와 마찬가지로 다른 RuleTable 로 전이하고 새로운 목표가 GoalTable 에 추가된다. 위와 같은 과정이 적용된 A(active rule)가 없을 때 까지 반복된다. 적용된 A 가 없다면 새로운 목표가 팝(pop)되고 또 다시 위와 같은 과정이 반복된다. 만약 GoalTable 에 G 가 모두 팝되고 없다면 프로세스는 종료한다.

3 실험 및 고찰

디폴트 엔트리가 삭제된 경우의 장애 진단 과정에서 규칙, Blackboard, 모듈, Goal 의 변화 과정을 보임으로써 제안한 규칙 및 알고리즘을 검증하였다.

표 2. 장애 진단 과정

단계	규칙 번호	규칙 상태 천이	조건절 번호	조건절 상태	Black board	Goal Table
0	-	-	-	-	-	g11,g12
1	1	A->TD ->FD	(1)-1	TU	f1	g12
2	2	A->TD ->FD	(2)-1 (2)-2	TU TU	f1,f2	g211 g212
3	10	A->TD ->FD	(10)-1	TU	f1 f2,f8	g212
4	11	A->TD ->FD	(11)-1 (11)-2	TU FA	f1,f2 f8,f9	g212
5	13	-	(13)-1	CHAIN	f1,f2 f8,f9	g221 g222

[표 2]는 디폴트 엔트리가 삭제되었을 경우의 장애 진단 과정을 나타내고 있다. 디폴트 엔트리가 삭제된 경우의 장애 진단 과정 중에 Blackboard 에 디폴트 엔트리 부재(f8)와 디폴트 엔트리 삭제(f9)라는 장애 원인이 추가 되었으며 CHAIN 명령에 의해 라우팅 구성 장애 복구 생성 규칙 모듈로 전이하여 장애 복구를 위한 목표인 g221 과 g222 가 GoalTable 에 추가됨을 알 수 있다.

4. 결론

이 논문에서는 시스템에서 발생할 수 있는 라우팅 구성 장애 관리의 중요성을 인지하고, 이질적인 네트워크 환경에 대처할 수 있는 규칙 기반의 라우팅 구성 장애 진단 시스템 구현을 위한 규칙과 알고리즘을 제안하였다. 후향 추론 알고리즘을 기반으로 다른 규칙과의 상호 연동을 위하여 메타 규칙을 적용하였으며 라우팅 구성 장애 관리를 위한 규칙을 두 개의 모듈로 나누었다. 네트워크 구성 관리 규칙 모듈은 시스템의 운영체제 유형과 여러 네트워크 구성 장애 유형을 분류하기 위한 규칙들을 포함하였다. 라우팅 구성 장애 진단 규칙 모듈은 장애가 발생하게 된 원인을 발견하기 위한 규칙들을 포함하였다. 제안한 규칙과 알고리즘을 검증하기 위해 디폴트 엔트리가 삭제된 경우의 규칙, Blackboard, 모듈, Goal 의 변화 과정을 제시하였다.

이 논문에서 제안한 규칙 및 알고리즘을 이용하여 실제적으로 라우팅 구성 장애 진단 시스템의 구현에 적용한다면 시스템에서 발생한 라우팅 구성 장애를 규칙에 기반하여 자동으로 진단할 수 있을 것이며 대규모의 네트워크에 존재하는 피관리 시스템에서 발생한 라우팅 구성 장애를 관리자가 직접 진단하는데 드는 시간과 비용을 줄일 수 있을 것으로 기대된다.

참고문헌

[1] R. J. Clack, R. Hutchins, S. W. Register, “The Role of Human Operators in Configuring, Managing, and Troubleshooting Interconnected Computer Networks”, IEEE, pp.4201-4206, 1995

[2] Show-Way Yeh, Chunan-lin Wu, Hong-Dah Sheng, “Expert System Based Automatic Network Fault Management System”, IEEE, pp.767-774, 1989

[3] Toshiharu Sugawara, “A Cooperative LAN Diagnostic and Observation Expert System”, IEEE, pp.667-674, 1990

[4] L. Lewis, “A Case-Based Reasoning Approach to the Management of Faults in Communications Networks”, IEEE, pp.114-120, 1993

[5] 김순철, 최영수, 정진욱, “LAN 세그먼트 관리를 위한 PC 기반의 RMON 에이전트 및 관리자 시스템에 관한 연구”, 정보처리학회 추계학술발표논문집, 제 6 권, 제 2 호, pp.186-191, 1999

[6] 유승근, 최영수, 정진욱, “네트워크 및 시스템 관리를 위한 웹 기반 통합 관리 시스템의 설계 및 구현”, 정보처리학회 추계학술발표논문집, 제 6 권, 제 2 호, pp.173-178, 1999

[7] 조강홍, 안성진, 정진욱, 박형우, “RMON MIB 을 이용한 LAN 성능 및 장애 파라미터 추출에 관한 연구”, 정보처리학회 추계학술발표논문집, 제 4 권, 제 2 호, pp.943-948, 1997

[8] B.Hayes-Roth, “A Blackboard architecture for control”, Artificial Intelligence, Vol.26, pp.255-321, 1985