한국해양공학회 2001년도 춘계학술대회 논문집 PP.32~37

T-invariant의 후방추론 기법을 이용한 시스템의 고장진단

정영미* · 정석권** · 유삼상***

Fault Diagnosis Using Backward Chaining of T-invariance

Young-Mi Jung* · Seok-Kwon Jeong** · Sam-Sang You***

Key Words : Petri Net(페트리네트), T-invariance(트랜지션 불변성), Fault Diagnosis(고장진단), Backward Chaining(후방추론)

This paper describes a noble fault diagnosis method using inter node search technique in PN model. First, a complicated fault system is modeled as PN graphic expressions. Next, to find out sources for faults on which we focus, the PN model is analyzed using the backward chaining of T-invariance. In this step, the technique of inter node search is applied for reducing some range of sources in a fault. Also, composing method of incidence matrix in PN is proposed. Then, it makes the diagnosis system to very flexible system because new knowledges about the sources in a fault can be added easily to conventional systems. Finally, the proposed method is applied to the automobile trouble diagnosis system to confirm the validity of the method.

1. 서 언

최근, 각 산업분야의 운영체제는 자동화의 실현으로 시스템을 24시간 가동하는 것이 일반적이다. 시스템의 원활하고 안정적인 가동을 위해서는 고장 발생시 이를 신속히 진단하여 원인을 규명할 수 있는 고장진단 기술이 요구된다. 고장 원인 규명은 시스템의 거대화 고성능화로 인해 고장을 일으키는 각 요소간의 관계가 매우 복잡하므로 전문적인 지식과 오랜 경험을 필요로 한다. 그러나 전문인력의 잦은 직장이동과 전문가의 부족현상 등의 제 문제점을 감안하면, 이상적으로는 전문가가 아닌 초보자라도 쉽게 고장을 진단할 수 있는 고장진단 시스템의 구축이 바람직하다.

종래의 통계적 처리수법이나 패턴분류 중심의 진단법은 고장현상에 대한 가능한 원인만을 제시하였다. 그러나 최근에는 원인의 탐색단계에서 원인후보의 효과적인 압축과 그 추론과정에 대한 시각화 및 이해의 용이성이 동시에 요구되고 있다. 또, 고장진단시스템은 새로운 지식이 창출될 경우, 이의 추가나 변경이 용이하도록 유연성을 가져야 한다.

고장진단 관련연구로는 PN(Petri Nets)을 이용한 고장진단 연구가 있다. Cai[2]등이 제안한 후방추론법은 해석상 접속행렬(incidence matrix)이 정방행렬로 제한되어 있다. 따라서 이 방법은 시스템의 접속행렬이 정방행렬이 아닐 경우, dummy 플레이스를 부가하여 정방행렬을 구성하는 수법을 제안하였다. 그러나 dummy 플레이스의 부가규칙은 명확히 기술되어 있지 않으므로 해석의 모호성을 초래한다. 또한, 임[1]등은 선형대수 이론에 근거한 T불변성(T-invariance)을 이용하여 후방추론 알고리즘을 제시하였다. 그러나, 이 방법은 고장현상에 대한 가능한 모든 원인을 제공하므로 직접적인 원인 규명에는 비효율적이다. 뿐만 아니라 새로운 지식의 추가도 용이하지 않다.

본 논문에서는 이러한 문제점을 보완하기 위하여 임등이 제안한 기존의 방법에 중간노드 검색법을 부

* 부경대 대학원 메카트로닉스공학과
** 부경대 기계공학부
*** 한국해양대 기계 · 정보공학부

가하여 고장원인 범위를 압축함으로써 효과적인 원인 규명이 가능한 방법을 제안한다. 또한, 새로운 지식의 추가가 용이하도록 접속행렬의 합성법을 제안한다. 제안된 방법은 자동차의 시동상태불량에 대한 원인 규명에 적용되어 그 타당성이 입증된다.

2. PN의 T-invariant성을 이용한 후방추론

2.1 PN의 표현

PN의 그래프 표현은 모델링된 시스템을 시각적으로 이해하기 쉽도록 해준다. PN은 사건의 상태나 조건에 대응되는 플레이스(place) P ($P=\{p_1, p_2, \cdots, p_m\}$ $m \geq 0$)와 사건의 진행이나 명령수행을 의미하는 트랜지션(transition) T ($T=\{t_1, t_2, \cdots, t_n\}$ $n \geq 0$) 그리고 방향성을 가지며 사건의 흐름을 나타내는 아크(arc)로 시스템을 모델링한다. 또한, PN은 시스템의 거동을 나타내기 위해 토큰을 P 내부에 표시하여 특정의 상태나 조건이 충족됨을 정의한다. P 내의 토큰의 분포는 마킹(marking) M으로 나타내며 시스템 전체의 동적 상황을 모의 실험하는데 사용한다. Fig. 1은 PN의 요소를 도식적으로 나타낸다.

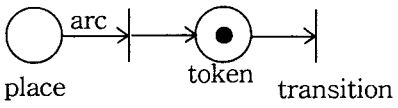

Fig. 1 Expression of PN

토큰은 T의 발화규칙[6]에 따라 P간을 이동하게 되며, 이는 정보의 흐름을 나타내므로 시스템의 거동을 모의실험하는 데 유용하다. T의 발화규칙은 일반 PN의 발화규칙[6]을 따른다. M은 T의 발화를 통해 토큰의 재분포가 이루어져 새로운 마킹 M_k로 되어 다음 상태로의 천이를 나타낸다.

2.2 T-iavariant성을 이용한 후방추론

후방추론은 고장현상에 대해서 그 원인을 규명하는 한 방법이다. 본 논문에서는 고장과 원인의 관계가 "A→B", 또는 "A OR B→ C"로서 "A→C", "B→ C"로 되는 Horn-clause(HC)[5]로 간주한다. HC에서 A와 B는 각각 명제이다. PN 모델링시, 모든 명제는 각각 하나의 P로, 각 규칙(각 명제간의 인과관계)은 하나의 T로 표현된다.

T불변성을 이용한 후방추론은 모델링된 PN에 새로운 T를 부가하여 시스템을 해석한다. 즉, 고장이 발생하면 그 고장을 나타내는 P(결과P)에 새로운 T(결과T)를 부가하여 현상의 발생을 표현한다. 또한, 고장의 원인을 나타내는 P(원인P)에도 새로운 T(원인T)를 부가하여 모든 원인P가 고장현상의 원인을 표현한다. 이렇게 PN을 변형하면 선형대수 이론에 근거하여 다음에 정의된 T불변벡터를 구할 수 있다.

[정의1] $Ax=0$을 만족하는 0이 아닌 벡터 x를 T불변벡터라 한다. 여기서, A는 PN의 접속행렬로서 $A=O(t_j, p_i)-I(p_i, t_j)$이다. O는 트랜지션 t_j에서 플레이스 p_i로의 출력행렬, I는 플레이스 p_i에서 트랜지션 t_j로의 입력행렬을 나타낸다.

구해진 T불변벡터는 결과T와 원인T를 벡터 내에 반드시 포함한다. 이때 포함된 원인T가 구하고자 하는 고장현상에 대한 원인이 된다. T불변성을 이용한 후방추론의 원리를 이하에서 설명한다.

PN로 모델링된 고장진단 시스템은 식(1)의 상태방정식을 이용하여 수치적으로 해석된다.

$$M_{k+1}=M_k+A \cdot t_k \tag{1}$$

여기서, $k=1,2 \ldots$이고, M_k는 k상태에서의 마킹을, t_k는 k상태에서의 T의 발화벡터로서 발화일 경우 1을 비발화일 경우 0을 가진다. A는 [정의1]에서의 접속행렬로서 시스템의 고유한 값을 갖게 된다.

Fig. 2는 T불변성을 설명하기 위한 PN모델이다.

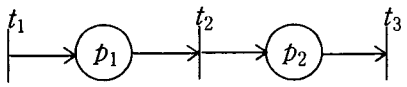

Fig. 2 Principle of backward chaining

우선, 고장이 없는 초기상태를 가정하면, 초기마킹 M_0는 0이다. 이 상태에서 임의의 고장원인이 존재하면, 이는 원인T의 t_1에 대응하므로 원인P의 p_1에 토큰이 생성된다. 원인 p_1은 인과관계에 의해 결과P의 t_2에 대응되며 발화조건을 만족시켜 결과P의 p_2

에 토큰을 생성시킨다. 결국, 최종적으로 결과T의 t_3가 발화하게 됨으로써 결과P의 p_2의 토큰이 소멸되어 최종마킹 M_n 또한 0으로 된다. 그러므로 식(1)에서 보듯이, M_0와 M_n이 0인 상태에서의 T불변성을 이용하면, $Ax=0$ $(x\neq 0)$을 만족하는 벡터 x는 목적 t_3와 원인 t_1을 동시에 포함한다. 이로써 고장원인 t_1의 추론이 가능하게 된다.

$Ax=0$ $(x\neq 0)$을 만족하는 T불변벡터 x는 선형변환을 이용하여 구할 수 있다. 여기서 A는 $m\times n$의 행렬, x와 0은 $n\times 1$ 및 $m\times 1$의 열벡터이다. $Ax=0$를 전치시키면, $x^T A^T = 0^T$로 된다. 구하고자 하는 x^T는 식(2)를 이용하여 구할 수 있다.

$$D A^T = 0 \qquad (2)$$

D는 $n\times n$의 대각행렬로서, 적당한 선형변환에 의해 A^T의 각 행을 0으로 만드는 행벡터로 구성되며 이 벡터들이 구하고자 하는 x^T의 해가 된다. x^T는 컴퓨터를 이용한 연산 프로그램을 통해 용이하게 구해진다.

3. 중간노드 검색을 통한 고장진단

3.1 중간노드 검색법

고장과 그 원인의 관계가 단순히 한 개의 명제만으로 구성되지 않는, 보다 복잡한 인과관계를 갖는 일반적인 형태를 가정하면, 이상에서 논의한 진단법은 비효율적이다. 왜냐하면, 구해진 T불변벡터들은 현재의 고장과는 무관할 가능성이 있는 모든 원인들을 동시에 제공하기 때문이다. 어떤 고장에 대한 가능성이 있는 원인이 다수 개 존재할 경우, 이들을 모두 확인, 점검하는 것은 대단히 비경제적이다. 따라서, 고장에 대한 직접적인 원인을 나타내는 해의 범위를 최대한 압축할 수 있는 방안이 필요하다. 본 논문에서는 이에 대한 해결방안의 하나로써, 중간노드 검색법을 제안한다.

이 방법은 구하고자 하는 최종 결과T에 대한 최초원인T를 직접적으로 구하지 않고, 결과T 직전의 중간노드에 대응하는 T를 순차적으로 확인해 가는 방법이다.

Fig. 3은 PN로 모델링된 시스템에 중간노드 검색법을 사용하는 과정을 나타내기 위한 한 예이다. 그림에서는 최초원인 P3, P4에 의해 P2의 고장이, P6에 의해 P5의 고장이 발생함을 보이고 있다. 또한, P2와 P5로 인해 최종적으로 고장 P1이 발생하는 구성으로 되어 있다. 우리가 주목하는 고장이 P1이면, 임[1]의 방법에 의한 진단결과는 P3, P4, P6가 고장원인으로 주어지게 된다. 그러나, 이들 3 요소는 P1의 고장에 대한 가능성을 갖는 해이기는 하나, 이 모두가 동시에 직접적인 고장요인으로 작용하고 있는 것은 아니다. 만일, P6만이 직접적인 고장요인으로 작용하고 있다면, 해로 나타난 이 3 요소 모두를 점검하는 것은 명확히 비효율적이다. 이 경우에는 P2와 P5를 먼저 확인한 후에 그 결과에 따라 다음 단계로의 고장을 확인하는 것이 고장원인의 범위를 압축할 수 있는 한 방법임을 알 수 있다.

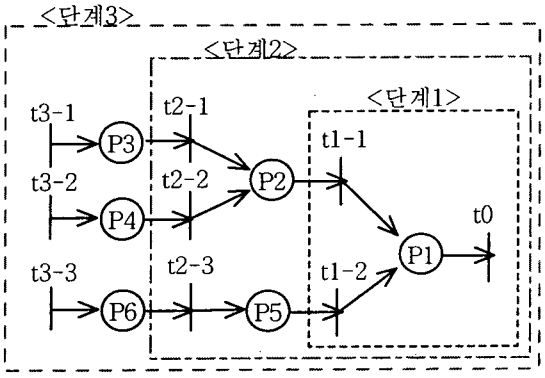

Fig. 3 Inter node search method

고장 P1에 대한 원인을 나타내는 중간노드 P2와 P5의 대응 T는 t1-1과 t1-2이다. 따라서, 이 방법은 T불변벡터의 해에 포함된 이들 T를 먼저 추출하여 이들에 대응하는 P의 상태를 확인하는 절차를 반드시 거치게 된다. 이때, P2에 해당하는 현상이 없을 경우, 해에 포함되었던 P2의 대응T인 t1-1은 제거된다. 그리고, t1-2가 새로운 결과T로 간주되어 이 T와 t3-3에 이르는 시스템에 대한 진단을 계속하게 된다. 이로써 t3-3가 최종적인 해로서 얻어지게 되며, 이의 점검만을 통해 고장요인의 해소가 가능하게 되므로 효율적인 고장진단이 수행된다.

이와 같이 중간노드 검색법은 단계별 접속행렬로부터 T불변벡터를 구하여 고장을 확인함으로써 원인 범위를 압축할 수 있다. 단계별 접속행렬은 다음에 정의된 규칙에 의해 전체 접속행렬로부터 구할

수 있다.

규칙1] ① 단계별 접속행렬은 해당 단계에서의 결과 T로 간주하는 t_c와 $\triangle P$ 관계인 한개의 p_c를 행으로, 그 p_c와 $\blacktriangle T$ 관계인 n개의 t_j를 열로 하는 $1\times n$의 크기로 구성된다. 단, $\triangle P$는 결과$T(\ t_c)$의 열에 대해서 행렬의 요소가 $(\ p_i,\ t_c)$=-1의 값인 $P(\ p_c)$를 의미하고, $\blacktriangle T$는 p_c의 행에 대해서 행렬의 요소가 $(\ p_c,\ t_j)\neq 0$의 값인 모든 T를 의미한다.

생성된 접속행렬로부터 T불변벡터를 구하여 원인 T를 찾는다. 임의의 단계에서 구해진 T는 다음 단계의 새로운 결과T로 된다. 이러한 과정은 최종의 원인P까지 반복된다. 고장의 발생에서 최종 원인까지의 전체적인 추론 과정은 각 단계에서 해가 아닌 T항을 제거한 접속행렬들을 합성함으로써 얻을 수 있다. 접속행렬의 합성칙은 규칙2, 3으로 정의된다.

규칙2] 각 단계별 접속행렬 B_i는 p_i를 행으로 t_j를 열로 하고 정의1에 나타낸 $O(\ t_j,\ p_i)-I(\ p_i,\ t_j)$에 따른 고유값을 가진다. 여기서, p_i와 t_j는 B_i에서 사용된 P와 T를 각각 의미한다.

규칙3] 전체PN의 접속행렬 B는 단계별 접속행렬 B_i의 합성으로 정의된다. 전체PN은 $P\times T$의 차원을 가지며 P와 T는 다음 식으로 정의된다.

$$P=\sum_{k=1}^{n} p_k - D_p \tag{3}$$

$$T=\sum_{k=1}^{n} t_k - D_t \tag{4}$$

D_p는 동일 P의 중복도, D_t는 동일 T의 중복도이다. 이때, B_i의 각 요소 b_{ij}는 $b_{ij}\in(p_i, t_j)$일때 b_{ij}의 고유값을 그대로 가지며 $b_{ij}\notin(p_i, t_j)$일때는 항상 0을 갖는다.

3.2 시스템의 증설

모델링된 시스템에 새로운 지식을 추가할 필요성이 있을 경우, 기존의 PN에 추가될 새로운 PN을 포함한 전체의 접속행렬은 앞 절에서 제시한 규칙2, 3을 이용함으로써 계산된다.

PN으로 모델링된 기존의 시스템 C에 새로운 지식을 나타내는 시스템 D가 Fig. 4와 같이 추가될 경우, 전체 접속행렬 B_t는 접속행렬의 합성칙에 따라 다음과 같이 구해지게 된다.

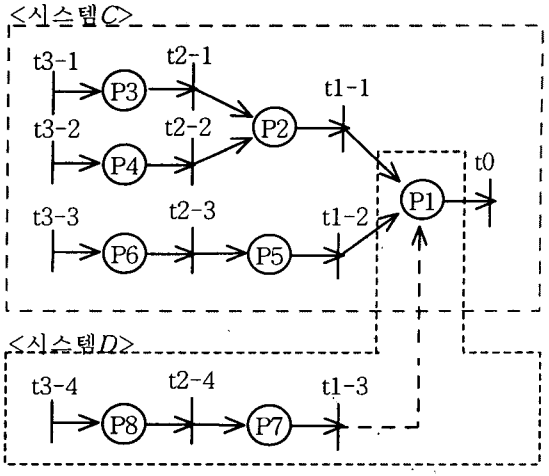

Fig. 4 Addition of a new system

우선, 기존의 시스템 C의 접속행렬 B_c는 식(5)와 같이 구해진다.

$$B_c = \begin{array}{c} \\ \text{P1} \\ \text{P2} \\ \text{P3} \\ \text{P4} \\ \text{P5} \\ \text{P6} \end{array} \begin{array}{c} \begin{array}{ccccccccc} \text{t0} & \text{t1-1} & \text{t1-2} & \text{t2-1} & \text{t2-2} & \text{t2-3} & \text{t3-1} & \text{t3-2} & \text{t3-3} \end{array} \\ \begin{bmatrix} -1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 \end{bmatrix} \end{array} \tag{5}$$

또한, 추가되는 새로운 시스템 D의 접속행렬 B_d는 식(6)과 같다.

$$B_d = \begin{array}{c} \\ \text{P1} \\ \text{P7} \\ \text{P8} \end{array} \begin{array}{c} \begin{array}{cccc} \text{t0} & \text{t1-3} & \text{t2-4} & \text{t3-4} \end{array} \\ \begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix} \end{array} \tag{6}$$

이 두 행렬을 합성한 전체PN의 접속행렬 B_t는 최종적으로 식(7)로 구해진다.

$$B_t = \begin{array}{c} \\ \text{P1} \\ \text{P2} \\ \text{P3} \\ \text{P4} \\ \text{P5} \\ \text{P6} \\ \text{P7} \\ \text{P8} \end{array} \begin{array}{c} \begin{array}{ccccccccccccc} \text{t0} & \text{t1-1} & \text{t1-2} & \text{t2-1} & \text{t2-2} & \text{t2-3} & \text{t3-1} & \text{t3-2} & \text{t3-3} & \text{t1-3} & \text{t2-4} & \text{t3-4} \end{array} \\ \begin{bmatrix} -1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix} \end{array} \tag{7}$$

3.3 중간노드 검색에 의한 고장진단의 적용 예

Fig. 5는 A/T 자동차에 있어서 Table 1과 같은 시동시의 고장현상을 PN으로 모델링한 것이다.

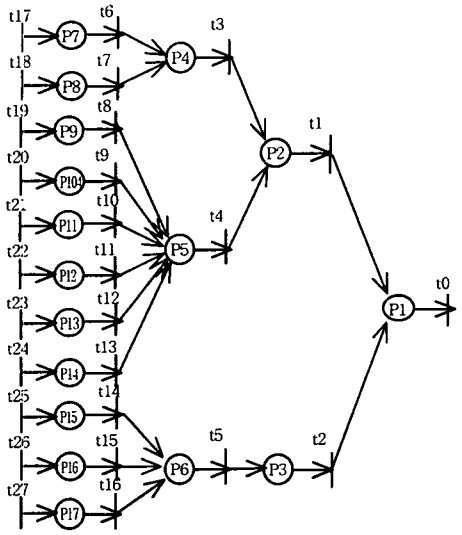

Fig. 5 PN model of the troubles in A/T motors

Table 1 The meaning of each place

place	state	place	state
P1	시동상태 불량	P10	파워TR불량
P2	스타팅모터는 작동됨	P11	CPS 및 배전기, 컨넥터의 접촉불량
P3	탁탁 부딪치는 소리만 남	P12	컨트롤 릴레이의 불량
P4	불꽃은 튀고 있음	P13	메인배선 접지불량
P5	불꽃이 튀지 않음	P14	ECU 불량
P6	BATTERY 방전	P15	스타팅모터가 작동이 부실
P7	컨넥터의 접촉불량	P16	누전
P8	진공호수의 빠짐,누설	P17	충전시간 부족
P9	점화코일 불량		

우선 Fig. 5의 PN의 접속행렬 B는 식(8)과 같이 구해진다. 이를 중간노드 검색법에 따라 고장현상의 발생에 대해 추론해 가면, Fig. 5는 4단계로 구분되고 단계별 접속행렬은 규칙1 에 따라 구해진다. 본 절에서는 스타팅 모터는 작동하면서 시동이 걸리지 않는 고장현상에 대한 최초원인을 진공호수의 빠짐에 의한 것으로 가정하고 다음과 같은 추론을 행한다. 먼저, 1단계로 시동이 걸리지 않음을 표시하는

B =

	t0	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11	t12	t13	t14	t15	t16	t17	t18	t19	t20	t21	t22	t23	t24	t25	t26	t27
P1	-1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
P2	0	-1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
P3	0	0	-1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
P4	0	0	0	-1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
P5	0	0	0	0	-1	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
P6	0	0	0	0	0	-1	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
P7	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
P8	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
P9	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
P10	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
P11	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
P12	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
P13	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
P14	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	1	0	0	0	0
P15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	1	0	0	0
P16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	1	0	0
P17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	1	0

(8)

결과T가 $t0$이므로 결과P인 $p1$을 행으로, t_j=[$t0$ $t1$ $t2$]를 열로 하는 접속행렬A_1은 다음과 같이 된다.

$$A_1 = \begin{matrix} \\ p1 \end{matrix} \begin{matrix} t0 & t1 & t2 \\ [\,-1 & 1 & 1\,] \end{matrix}$$

이때, T불변벡터는 후방추론에 의해 [1 1 0], [1 0 1]로 구해진다. 이는 결과T인 $t0$에 대해 그 원인T가 $t1$, $t2$임을 나타내므로 $t1$, $t2$의 발화를 가능하게 하는 각 플레이스 $p2$, $p3$의 상태를 확인함으로써 직접적으로 작용한 원인T를 구할 수 있다. Table 1에서 $p2$는 스타팅 모터는 작동되지만 시동이 걸리지 않는 경우를, $p3$는 부딪치는 소리가 나면서 시동이 걸리지 않는 경우를 각각 나타낸다. 스타팅 모터는 작동된다고 가정하면, 확인된 원인T는 $t1$이다.

2단계에서는 $t1$이 결과T가 되므로 접속행렬 A_2는 $t1$의 열에 0이 아닌 값을 가지는 P인 $p2$를 행으로 $p2$의 행에 대해 0이외의 값을 가지는 T인 t_j=[$t1$ $t3$ $t4$]를 열로 하여 다음과 같이 된다.

$$A_2 = \begin{matrix} \\ p2 \end{matrix} \begin{matrix} t1 & t3 & t4 \\ [\,-1 & 1 & 1\,] \end{matrix}$$

T불변벡터는 [1 1 0], [1 0 1]로 구해지므로 직접적으로 작용하는 원인T는 $t3$와 $t4$에 해당하는 $p4$, $p5$의 상태인 불꽃이 튀는지의 여부를 확인함으로써 알 수 있다. 여기서, 불꽃이 튄다고 가정하고 2단계에서 구해진 원인T $t3$를 결과T로 하여 3단계를 수행하면, 접속행렬A_3는 다음과 같다.

$$A_3 = \begin{matrix} \\ p4 \end{matrix} \begin{matrix} t3 & t6 & t7 \\ [\,-1 & 1 & 1\,] \end{matrix}$$

이로부터 T불변벡터 [1 1 0], [1 0 1]을 얻을 수 있다. 이 단계의 원인T는 $t6$, $t7$임을 알 수 있고 이에

해당하는 $p7$, $p8$의 상태인 컨넥터의 접촉불량, 또는 진공호수의 빠짐이나 누설을 확인하여 직접 작용한 원인을 알아낼 수 있다. 진공호수의 빠짐에 의해 일어난 것으로 가정하였으므로, 4단계의 접속행렬은 다음과 같다.

$$A_4 = \begin{matrix} \\ p7 \end{matrix} \begin{matrix} t6 & t17 \\ \end{matrix} \begin{bmatrix} -1 & 1 \end{bmatrix}$$

T불변벡터가 [1 1]로 얻어지므로, 결국 고장의 원인은 진공호수의 빠짐으로 확정할 수 있다.

전체적인 추론과정은 지금까지의 중간노드 검색과정을 규칙2, 3에 따라 합성함으로써 알 수 있다. $t2$, $t4$와 $t7$항은 제거되었으므로, 전체 접속행렬 A는 식(9)와 같이 구해진다.

$$A = \begin{matrix} \\ P1 \\ P2 \\ P4 \\ P7 \end{matrix} \begin{matrix} t0 & t1 & t3 & t6 & t7 & t17 \\ \end{matrix} \begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix} \quad (9)$$

식(9)에서, T불변벡터를 구하면, [1 1 1 1 1 1]을 얻게 된다. 이로부터, 지금까지의 모든 추론 과정을 볼 수 있으며 시동상태 불량에 대한 최종원인은 진공호수의 빠짐인 사실을 알 수 있다. 만약, 중간노드 검색 없이 전체 접속행렬의 T불변벡터를 구한다면, 이 고장에 대해 총 11가지의 원인이 제공되므로 고장처리를 위해서는 이들 모두를 확인해야만 한다. 그러나, 중간노드 검색법은 이 고장에 직접적으로 작용하고 있는 원인을 2가지로 제한시켜 주므로 효율적인 고장처치가 가능해 진다. 그러므로, 전체 원인을 모두 검색함으로써 야기되는 시간적 낭비와 인력의 소모를 최소화할 수 있다. 또한, 이 방법은 원인의 추론 과정 전체를 시각화 해주기 때문에 구축된 알고리즘을 쉽게 이해할 수 있도록 해준다.

4. 결 언

본 논문에서는 복잡한 고장 시스템의 효율적인 진단 방안의 하나로서 중간노드 검색법을 제안하였다. 제안된 방법은 T불변성의 후방추론 기법을 이용하여 PN으로 모델링된 고장현상의 직접적인 원인만을 진단할 수 있는 방법이다. 또한, 기존의 제안된 방법과는 달리 새로운 지식의 추가가 용이하도록 유연성을 갖도록 하는 방안도 제안하였다. 이를 위해 고장에 관해 발견된 새로운 지식을 기존 구축된 시스템에 추가할 필요성이 있을 경우, 기존의 PN 접속행렬에 추가PN의 접속행렬을 용이하게 합성할 수 있도록 하는 제 규칙을 정의하였다. 마지막으로 제안된 방법은 자동차의 시동상태불량에 대한 고장진단 예에 적용하여 그 유효성이 확인되었다.

본 논문에서 제안한 방법은 향후, 선박 또는 해양 플랜트와 같이 복잡한 시스템의 신속한 고장처리에 유효하게 사용될 수 있을 것으로 기대된다.

참고문헌

1) 임재걸, "논리 프로그램의 페트리넷 모형에서 후방추론 방법", 한국정보과학회, '97가을 학술발표 논문집 Vol. 24, No. 2, pp. 23~26, 1997
2) Yiang Cai et al, "ペトリネットの 構造解析と計數プレースの付加法", 計測自動制御學會論文集 Vol. 29, No. 12, pp. 1458~1464, 1993
3) 정석권, 양주호 "페트리네트의 계층화를 통한 시퀀스제어계의 설계(Ⅰ), -SFC에 근거한 페트리네트의 분할", 한국해양공학회지, Vol. 13, No. 3, pp. 106~115, 1999
4) 정영미, 정석권, 양주호, "PLC베이스의 시퀀스 제어 시스템 해석을 위한 페트리네트의 분할과 합성" 한국동력기계공학회 춘계학술대회논문집, pp. 57~63, 2000
5) N. J. Nilsson, "Principles of Artificial Intelligence" Palo Alto, CA, Tioga, 1980
6) 일본전기학회, シーケンス制御工學, 1988