Journal of the Korean Institute
of Industrial Engineers
Vol.22, No.1, March, 1996

고장나무의 비관련 사상들에 대한 축소방법*

A Method of Reducing the Irrelevant Events in a Fault Tree*

이해상**, 홍정식***, 이창훈****

Hae-Sang Lee**, Jung-Sik Hong***, Chang-Hoon Lie****

Abstract

Fault tree analysis is NP-hard problem. In this paper, we present a method which reduces size of the fault tree by eliminating the irrelevant events. Irrelevant event is the event which has no contribution to the system failure. In a fault tree, the irrelevant events occur due to the existence of the replicated events. By investigating the structure of the replicated events we establish the conditions which characterize the irrelevant events. Based on these conditions we present the computational algorithm which eliminate the irrelevant events. Complexity of the algorithm is shown to be polynomial and so, this algorithm can be utilized efficiently in FTA.

1. 서 론

고장나무분석(Fault Tree Analysis, 이하 FTA로 표기 됨) 작업은 크게, 정성적 분석과 정량적 분석 두 가지로 대별된다. 정성적 분석에서는 최소절단집합(min cut set)을 구하는 것이 대표적 사례이고 정량적 분석에서는 최상위 사상(Top Event, 이하 TE로 표기 됨)의 발생확률을 구하는 것이 대표적 사례이다[6]. 최소절단 집합을 구하는 작업과 TE의 발생확률을 계산하는 작업 둘다 계산시간은 중복사상(Replicated Event 이하 RE로 표기됨)에 좌우된다[8].

따라서, FTA에서 RE는 매우 중요한 역할을 하며, RE에 조건을 주어 고장나무(Fault Tree, 이하 FT로 표기 됨)를 간단히 한 후에

* 본 연구는 부분적으로 한국원자력연구소의 재정적 지원에 의하여 수행되었음.

** SDS(Sam-Sung Data System)

*** 서울산업대학교 산업공학과(Dept. of Industrial Eng. Seoul Nat'l Polytechnic Univ.)

**** 서울대학교 산업공학과(Dept. of Industrial Eng. Seoul Nat'l Univ.)

TE의 발생확률을 구하는 알고리즘이 개발되고 있으며[1], 또한 절단집합들 중에서 최소 절단집합을 찾아 내는데 RE의 특성을 이용하는 작업이 수행되고 있다[7]. 한편으로, RE의 발생위치를 파악하여, FT를 소규모의 모듈로 쪼개어, FTA를 용이하게 하는 연구가 최근까지 수행되고 있다[5,9]. 그러나 주어진 FT에서 중복사상을 구조적으로 분석하여, FTA에 전혀 무관한 사상(irrelevant event)을 제거하는 연구는 수행되지 못한 상태이다. 본 논문은 부울린 대수(Boolean algebra)의 흡수 법칙(absorption law)을 FTA에 적용하는 작업과 또한 중복사상의 구조적 특성을 규명하는 작업을 토대로 FTA에 전혀 무관한 사상(irrelevant event)을 특징짓는 조건을 새로이 확립한다. 그리고 이러한 조건을 바탕으로 FT의 크기를 축소시키는 계산 알고리즘을 제시한다.

이러한 알고리즘은 기존의 대규모 FT를 분석하는 각종 패키지의[2,3,4] 입력부분에 장착되어, FT의 크기를 축소시킴으로써, FTA의 계산시간을 대폭 줄일 것으로 예상된다.

2절에서는 FT의 중복사상의 구조적 특성을 분석하여 비관련 사상을 특징짓는 조건을 확립한 정리를 제시한다. 3절에서는 정리를 바탕으로 하여 FT크기를 축소하는 알고리즘을 제시하고, 알고리즘의 복잡도가 다항시간임을 입증함으로써, FTA 계산시간의 축소를 보인다. 4절에서는 실례를 분석하여, 알고리즘의 작동과정을 보이고, 5절에서는 결론 및 추후 연구방향을 제시한다.

2. 중복사상의 구조 분석

본 절에서는 중복사상의 구조분석을 통해 비관련 사상을 특징 짓는 조건을 도출한다.

2.1 AND-OR 게이트 축소

먼저, 주어진 FT를 정리하여 FT를 표준화하고자 한다. 여러 FT에서 AND 게이트의 구성요소로 AND 게이트가 있으며, OR 게이트의 구성요소로도 OR게이트가 있다(편의상 AND 게이트와 OR 게이트만 고려하고자 한다). 이는 시스템의 표현상 이점이 있고, FT 구축 과정상에서 자연스럽게 발생되는 현상이다. 그러나 논리적인 관점에서, AND 게이트의 구성요소인 AND 게이트는 상위 AND 게이트로 통합될 수 있다. OR 게이트 역시 마찬가지다. 이러한 통합작업을 거치고 나면, 특정 AND 게이트의 구성요소는 기본 사상(Basic Event 이하 BE로 표기 됨)들과 OR 게이트들로만 이루어지며 마찬가지로 특정 OR 게이트의 구성 요소는 BE들과 AND 게이트들로 이루어진다.

여기서, 다음 용어들을 간략히 정의하자([그림 1] 참조).

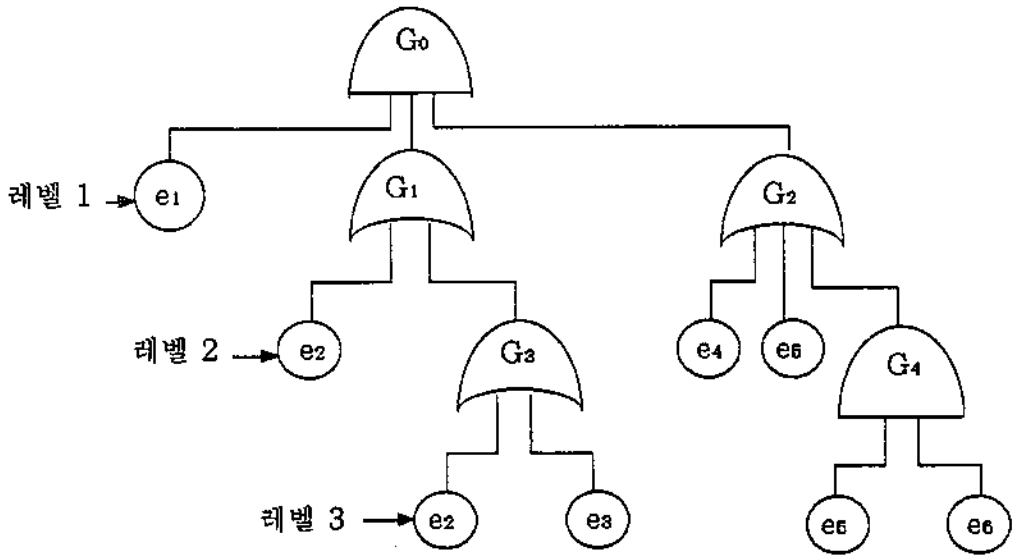
e_i : i번째 BE를 나타냄

X_i : e_i의 상태변수, 즉 $X_i = \begin{pmatrix} 1, & e_i \text{ 발생} \\ 0, & e_i \text{ 미발생} \end{pmatrix}$

G_j : j번째 게이트를 나타냄

e_i의 레벨 : TE인 G_0로 부터 e_i까지를 연결짓는 경로(path)상에 놓인 게이트의 개수

G_j의 레벨 : TE인 G_0로 부터 G_j까지를 연결짓는 경로(path)상에 놓인 게이트의 개수



G_0의 successor : $\{e_1, G_1, G_2\}$
e_6의 predecessor : $\{G_4\}$
G_2의 descendant : $\{e_4, e_5, G_4, e_6\}$
레벨 3의 e_5의 ancestor : $\{G_0, G_2, G_4\}$

그림 1. FT의 게이트와 BE를 특징짓는 개념들

e_i의 predecessor : e_i와 TE간 경로상에서 e_i에 인접한(adjacent) 게이트

G_j의 predecessor : G_j와 TE간 경로상에서 G_j에 인접한(adjacent) 게이트

G_j의 successor : G_j를 predecessor로 갖는 게이트나 BE의 집합

e_i의 ancestor : e_i와 TE간 경로상에 있는 모든 게이트들의 집합

G_j의 ancestor : G_j와 TE간 경로상에 있는 모든 게이트들의 집합

G_j의 descendant : G_j를 ancestor로 하는 게이트나 BE들의 집합

중복해서 발생되는 BE로 인해, 이상의 개념으로 하나의 FT가 충분히 표현되지 못한다. FT의 데이타 구조는 후에 알고리즘 설명에서 기술될 것이다.

2.2 비관련 사상의 제거

위의 AND-OR 게이트 축소후에, 간단한 경우의 비관련 사상이 나오게 된다. 즉, AND 게이트나 OR 게이트의 successor에 속하는 기본 사상들 중에서 중복이 존재하는 경우이다. 이 경우는 당연히 중복사상들 중 하나의 사상을 제외한 나머지 사상을 제거하면 된다.

다음, 부울린 대수의 흡수 법칙을 고려하자.

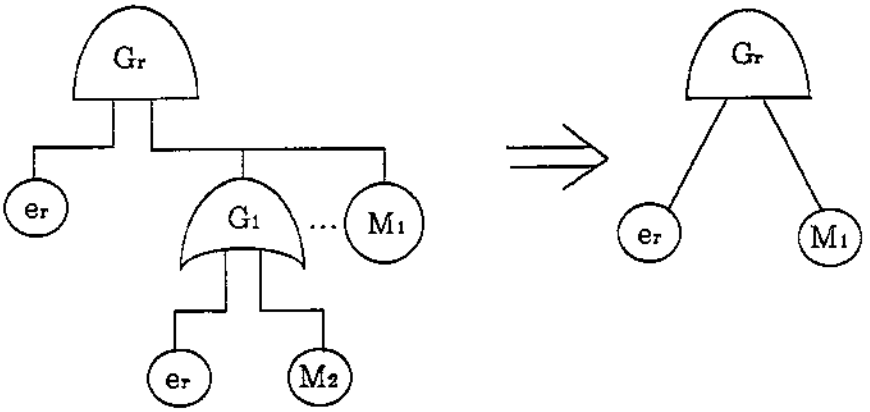
○ $A * (A+B) = A$ (1)

○ $A+(A * B) = A$ (2)

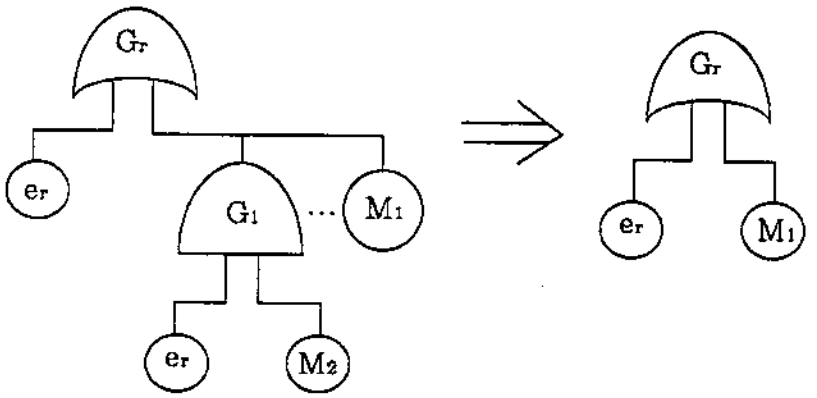
(1)과 (2)를 FT에 적용한 것이 보조정리 1에 나와 있다.

보조정리 1

(a) RE e_r의 predecessor가 AND 게이트 G_r이다. 이때, G_r의 successor에 속하는 게이트들 중 e_r을 successor로 갖는 게이트는 제거된다.([그림 2(a)] 참조)
(여기서 특정 게이트의 제거라 함은, 그 게이트와 그 자신의 descendant가 제거됨을 뜻한다.)



(a) 보조정리 1(a)의 비관련 사상 제거



(b) 보조정리 1(b)의 비관련 사상 제거

그림 2. 보조정리 1 유형의 중복사상

(b) RE e_r의 predecessor가 OR 게이트 G_r이다. 이때, G_r의 successor에 속하는 게이트들 중 e_r을 successor로 갖는 게이트는 제거된다.([그림 2(b)] 참조)

증명)

(a) [그림 2(a)]의 좌측 FT의 구조 함수, $\phi(G_r)$는 다음과 같다.
여기서, M_1과 M_2는 임의의 하부 FT (sub fault tree)이다.

$$\phi(G_r) = X_r \cdot \phi(G_1) \cdot X_{M_1} \quad (3)$$

$\phi(G_1) = 1-\overline{X_r} \cdot \overline{X}_{M_2}$이므로, 이를 (3)에 대입하면,

$$\phi(G_r) = X_r \cdot (1-\overline{X_r} \cdot \overline{X}_{M_2})X_{M_1}$$
$$= X_r X_{M_1} - X_r \cdot \overline{X_r} \cdot X_{M_1} \cdot \overline{X}_{M_2} \quad (4)$$

$X_r \cdot \overline{X_r} = 0$이므로, (4)식을 정리하면

$$\phi(G_r) = X_r \cdot X_{M_1}$$

이고, 이는 [그림 3(a)]의 우측 FT이다.

(b) (a)와 마찬가지 방법으로 증명 가능하다.

Q.E.D.

보조정리 1은 AND 게이트나 OR 게이트의 successor에 RE가 존재할 때, successor에 속하는 게이트들(레벨 1)에 대한 제거 기준을 제공한다. 이를 확장하면, 특정 게이트의 successor에 중복사상이 존재할 때, 이 게이트를 TE로 할때, 레벨이 2인 게이트들에 대한 제거기준을 생각해 볼 수 있다. 레벨 2인 게이트들에 대한 제거기준은 보조정리 2로 수식화된다.

보조정리 2.

(a) 특정 AND 게이트 G_r의 successor에 RE인 e_r이 존재한다.
이 때 G_r로 부터 레벨 2인 AND 게이트들 중 e_r을 successor로 가지고 있는 게이트가 존재할 경우, 그 게이트의 e_r은 제거된다.([그림 3(a)] 참조)

(b) 특정 OR 게이트 G_r의 successor에 RE인 e_r이 존재한다.
이 때 G_r로 부터 레벨 2인 OR 게이트들 중 e_r을 successor로 가지고 있는 게이트가 존재할 경우, 그 게이트의 e_r은 제거된다.
([그림 3(b)] 참조)

증명)

(a) [그림 3(a)]의 좌측 FT의 구조함수는 다음과 같다.

$$\phi(G_r) = X_r \cdot \phi(G_1) \cdot X_{M_1} \quad (5)$$
$$\phi(G_1) = \phi(G_2) + X_{M_2} - \phi(G_2) \cdot X_{M_2} \quad (6)$$
$$\phi(G_2) = X_r \cdot X_{M_3} \quad (7)$$

(6)과 (7)을 (5)에 대입하면

$$\phi(G_r) = X_r[X_r \cdot X_{M_3} + X_{M_2} - X_r \cdot X_{M_3} \cdot X_{M_2}]X_{M_1} \quad (8)$$

$X_r^2 = X_r$이므로, (8)을 정리하면,

$$\phi(X_r) = [X_r \cdot X_{M_3} + X_r \cdot X_{M_2} - X_r \cdot X_{M_3} \cdot X_{M_2}]X_{M_1} \quad (9)$$

(9)는 [그림 3(a)]의 우측 FT의 구조함수이다.

(b) (a)와 마찬가지로 증명 가능하다.

Q.E.D.

이제 보조정리 1과 보조정리 2를 합하여 일반화 하면, 특정 게이트가 중복사상 e_r을 successor로 갖고 있을때, 이 특정 게이트로부터 임의의 레벨 n에 위치해 있는 게이트의 제거기준을 확립하는 것이다. 보조정리 1과 보조정리 2를 토대로 한 비관련 사상의 일반적인 제거 기준이 정리 1에 나와 있다.

〈정 리 1〉

(a) 특정 AND 게이트 G_r이 RE e_r을 successor로 가지고 있다. 이때 G_r의 descendant 중 임의의 레벨 2k에 있는 AND 게이트의 경우, e_r을 successor로 갖고 있으면, e_r이 제거되고, 레벨 2k+1에 있는 OR 게이트의 경우 e_r을 successor로 갖고 있으면 OR 게이트 자체가 제거된다.

(b) 특정 OR 게이트 G_r이 RE e_r을 successor로 가지고 있다. 이때 G_r의 descendant 중 임의의 레벨 2k에 있는 OR 게이트의 경우, e_r을 successor로 갖고 있으면, e_r이 제거되고, 레벨 2k+1에 있는 AND 게이트의 경우 e_r을 successor로 갖고 있으면 AND 게이트 자체가 제거된다.

증명)

(a) 임의의 레벨에 속한 AND 게이트와 OR 게이트에 대해서도 보조정리 1과 보조정리 2의 내용이 일반화되서 적용된다는 것을 모든 가능한 경우의 구조함수 나열로는 증명하기가 불가능하다.

그러나, 보조정리 2(a) 항목을 역으로 적용해 보자.

즉, 특정 AND 게이트 G_r가 e_r을 successor로 갖고 있을 때, G_r로 부터 레벨 2인 AND 게이트들의 successor에서 e_r이 제거되는 것을 역으로 생각하면, e_r이 없는 AND 게이트에 e_r을 논리적 오류없이 successor로 추가시킬 수 있다.

그림 4와 같이 레벨 2마다 AND 게이트 e_r을 successor로 추가하자. 그러면, 정리 1(a)는 보조정리 1(a)와 보조정리 2(a)에 따라서 성립함을 알 수 있다.

(b) 역시 마찬가지 방법으로 증명가능하다.

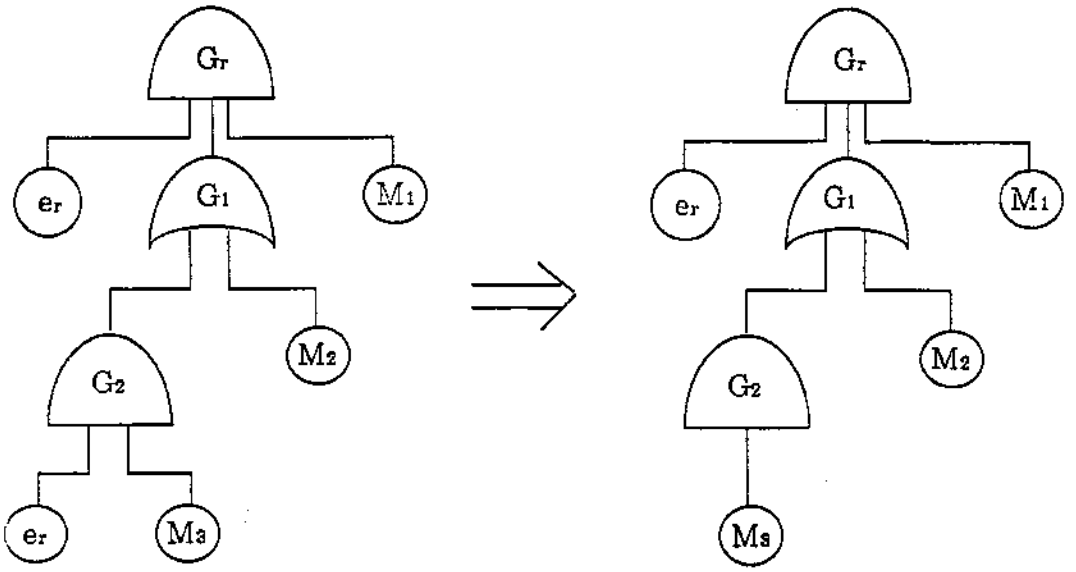
Q.E.D.

3. 중복사상 축소 알고리즘 및 실례

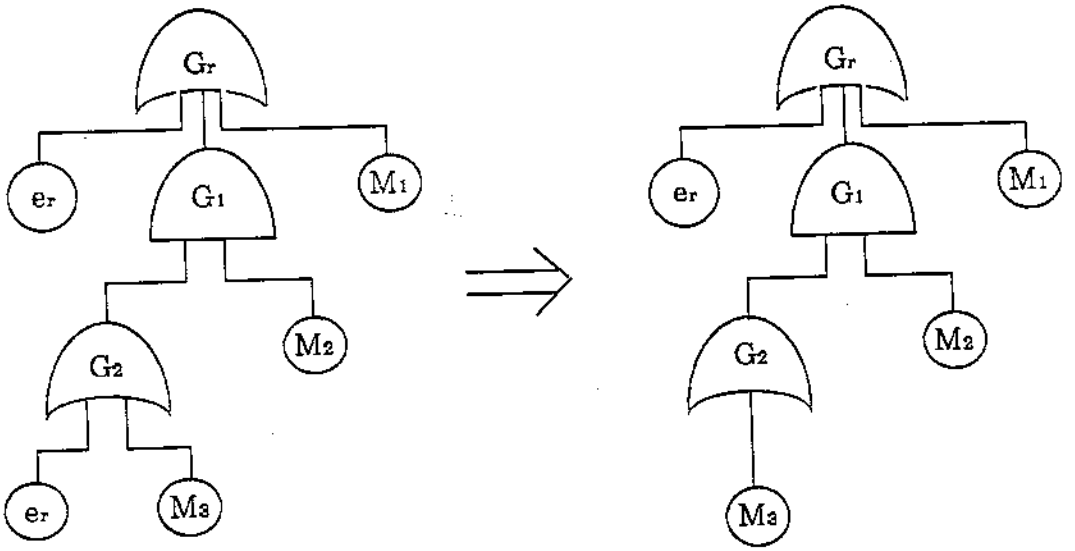
정리 1의 내용을 토대로 해서 FT의 크기를 축소하는 알고리즘을 구축하기 위해서 먼저 AND-OR 게이트 축소를 시행하여야 한다.

AND-OR 게이트 축소는 AND 게이트의 successor에 AND 게이트가 있거나 OR 게이트의 successor에 OR 게이트가 있는 경우 이를 통합하는 것이다. 알고리즘의 구축을 위해서, 주어진 FT를 입력시킬 때, 다음과 같은 데이타 구조를 사용하고자 한다.

즉, TE로 부터 레벨 k에 있는 j번째 게이트의 경우 (게이트 번호는 편의상 상위 레벨부터, 그리고 특정레벨의 좌측부터 붙여나간다) 이의 predecessor 게이트와 successor에 속하는 BE와 게이트들로 표현된다. 즉,



(a) **보조정리 2**(a)의 비관련 사상 제거



(b) **보조정리 2**(b)의 비관련 사상 제거

그림 3. 보조정리 2 유형의 중복사상

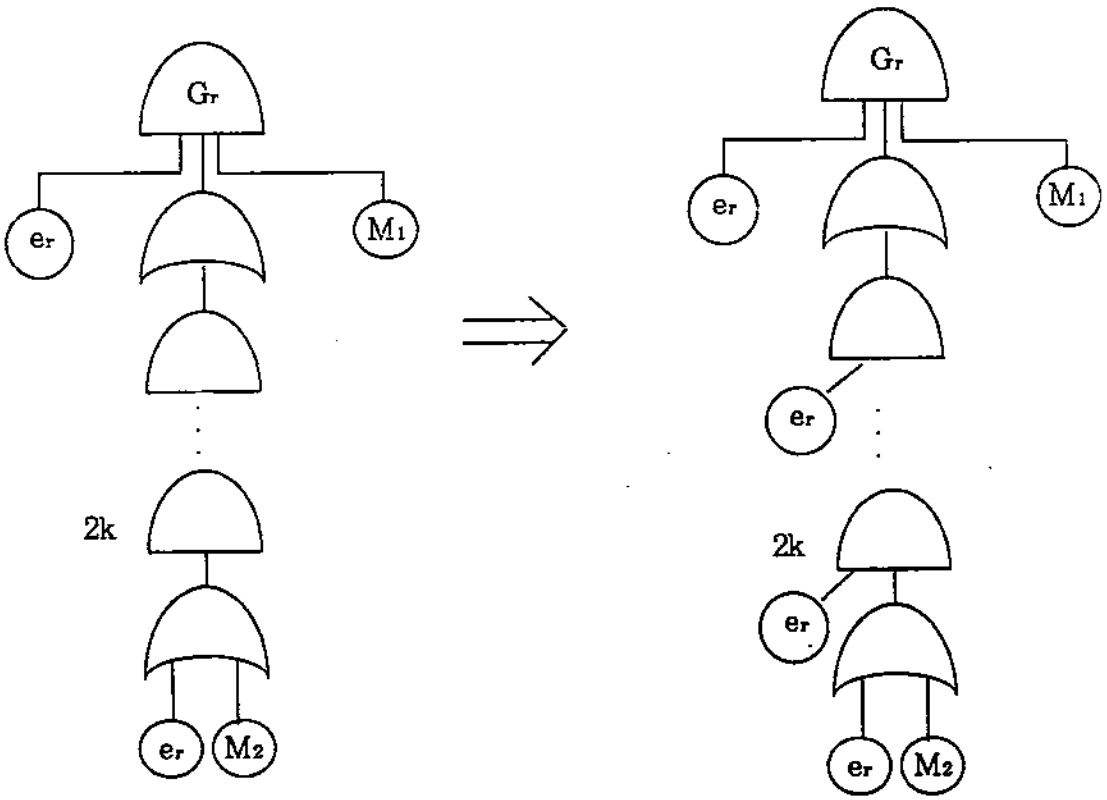


그림 4. 정리 1의 증명 예시 그림

$G^i_{jk} = I,\ B^i_j, \Phi^i_j, k$

여기서, G^i_{jk}는 레벨이 k이고 predecessor가 G_I인 j번째 게이트를 지칭한다. 그리고 I는 게이트의 종류를 나타내며 (I=0은 OR 게이트, I=1은 AND 게이트를 나타낸다.)

B^i_j는 게이트 G^i_j의 successor에 속하는 BE들의 집합이고, Φ^i_j는 게이트 G^i_j의 successor에 속하는 게이트 들의 집합이다. k는 물론 레벨을 지칭한다.

그러면, AND-OR 축소는 G^i_{jk}의 경우, G^i_{jk}의 successor 게이트인 G^j_{pk+1}과 각자의 I 변수의 값을 체크함으로써 이루어 진다. 알고리즘은 아래와 같다. N은 FT의 총 게이트 수이고 M은 총 BE의 수이다.

■ AND-OR REDUCTION ALGORITHM

```
DO k=1 to N
  DO j=min{G_J^J}_k to max {G_J^J}_k
    IF I[{G_J^J}_k] = I[{G_P^J}_{k+1}]
      UPDATE B{G_J^J}_k = B{G_J^J}_k ∪ B
                          {G_P^J}_{k+1}
             Φ{G_J^J}_k = Φ{G_J^J}_k ∪
                          Φ{G_P^J}_{k+1}
      REMOVE {G_P^J}_{k+1}
    Otherwise j = j+1
  Otherwise k = k+1
```

AND-OR 축소 알고리즘은 각 게이트들이 각자의 successor만을 체크함으로써 수행된다. 따라서 알고리즘 수행시간은 O(N)이 됨을 쉽게 알 수 있다. 이제 축소된 FT를 가지고 비관련 사상을 제거하는 알고리즘을 고려하자.

정리 1에 따라, 이는 특정 게이트 $\{G_h^l\}_{k+s}$의 successor에 속하는 BE들이($B\{G_h^l\}_{k+s}$), G_h^l의 ancestor에 속하는 특정 게이트 $\{G_j^k\}_k$의 successor에 속하는 BE와($B\{G_j^j\}_k$) 중복되는 사상(RE)을 갖고 있느냐에 따라 결정된다. 즉, RE가 존재할 경우 ($B\{G_j^j\}_k \cap B\{G_h^l\}_{k+s} \neq \phi$), G_j^j와 G_h^l의 게이트가 서로 동일하면, $B\{G_h^l\}$ 중 RE만 제거되고, 서로 다르면 $\{G_h^l\}_{k+s}$ 자체가 제거된다. 알고리즘은 다음과 같다.

■ FT REDUCTION ALGORITHM

```
Do k=1 to N
  Do j = min{G_j^j}_k to max{G_j^j}_k
   IF B{G_j^j}_k ∩ B{G_h^l}_{k+s} = φ
   IF I{G_j^j}_k = I{G_h^l}_{k+s}
    UPDATE B{G_h^l}_{k+s} = B{G_h^l}_{k+s}
    − {B{G_j^j}_k ∩ B{G_h^l}_{k+s}}
   IF I{G_j^j}_k ≠ I{G_h^l}_{k+s}
    REMOVE {G_h^l}_{k+s}
  Otherwise j = j+1
 Otherwise k = k+1
```

이상의 FT 축소 알고리즘은 TE로 부터 레벨 k인 게이트$\{B_j^j\}_k$의 경우, B_j^j의 ancestor에 속하는 k개의 게이트들과 RE 공유 여부및 게이트의 일치도를 체크하게 된다.

따라서, 전체 게이트 갯수 N이 레벨에 따라 $N_1, N_2, \cdots, N_k$로 나누어 진다고 할때($N_1+N_2+\cdots+N_k=N$), 알고리즘의 계산시간은 $(N_1+2N_2+\cdots+KN_k)M$에 비례함을 쉽게 알 수 있다. 또한 레벨이 클수록 그에 비례해서 RE 공유여부의 체크시간이 커지므로 최대 시간은 모든 i에 대해서 $N_i=1$일 때이다. 이때 걸리는 시간은 $\frac{N(N+1)}{2}M$에 비례하므로 $O(N^2 \cdot M)$이 된다.

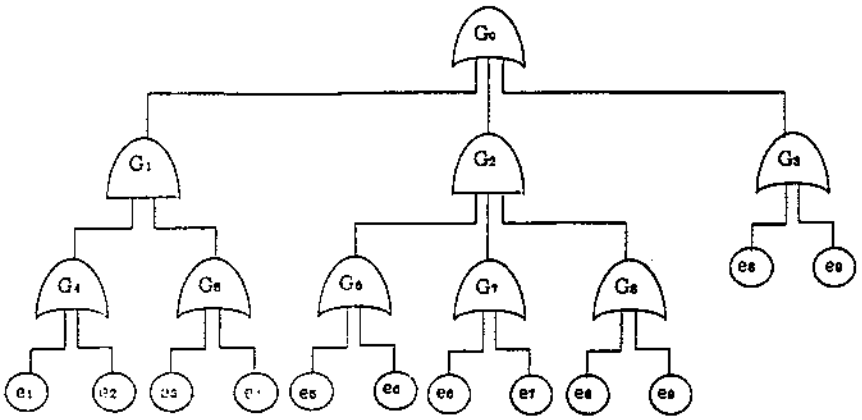
3.2 알고리즘 적용 예제

[5]에 나와 있는 FT를 고려하자([그림 5(a)]). 이 FT는 9개의 게이트와 12개의 BE로 이루어져 있다. 먼저, AND-OR 축소 알고리즘을 적용하면, G_3가 G_0와 통합되어 [그림 5(b)]와 같이 된다. 다음 RE 축소의 경우 게이트 G_8이 자신의 ancestor인 G_0와 e_8, e_9를 공유하고 있다. 그리고, G_8과 G_0 둘다 OR 게이트 이므로 e_8과 e_9만 제거된다. 이 경우 G_8이 e_8, e_9로만 이루어 졌으므로 G_8이 제거된 결과가 되었다([그림 5(c)]참조).

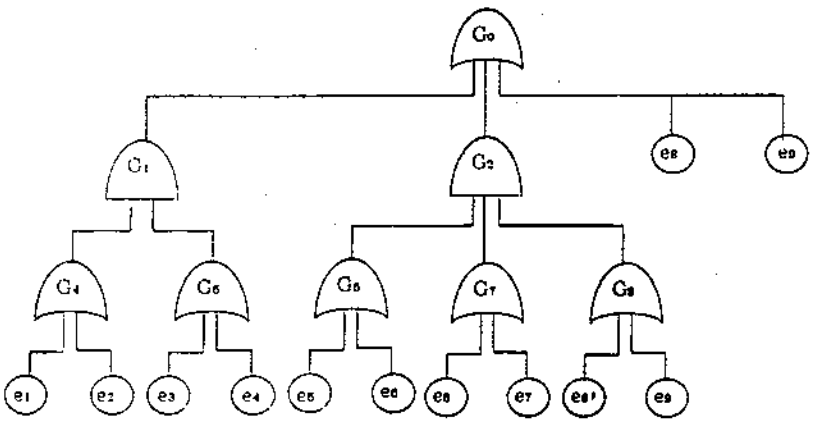
4. 결론 및 추후 연구 방향

FTA에 있어 TE의 발생 확률을 계산하는 문제는 NP-hard 문제이다. 물론, RE가 없는 경우는 선형시간만에 TE의 발생확률이 계산된다. 따라서, FTA에 있어 중요한 것은 RE의 갯수 및 발생위치이다. 본 논문에서 우리는 TE의 발생에 관련이 없는 비관련 사상을 특징짓는 조건을 정립하였고, 이를 통해 FT의 크기를 축소하는 알고리즘을 제시하였다. 알고리즘의 복잡도는 다항시간(polynomial time)임을 보였다.

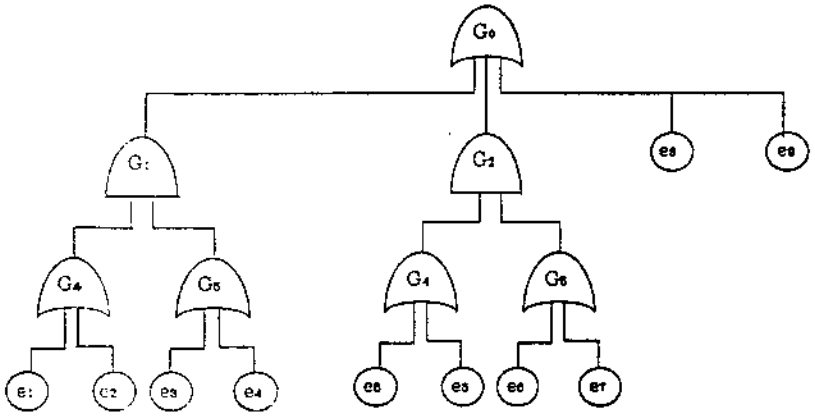
본 논문에서 제시된 알고리즘은 현재 원자



(a) 예제 FT



(b) AND-OR 축소 후의 FT



(c) 비관련 사상 축소후의 FT

그림 5. 알고리즘 적용 예제 FT

력 연구소에서 기개발한 FTA 계산 패키지의 개선을 위해 활용되고 있으며, 계산시간상에 있어서 30% 정도의 절감을 보여주고 있으며, 계속적인 사례연구가 수행중에 있다. 추후에, 본 논문에서 제시된 FT 축소 후에 FT에 남아 있는 RE들의 구조를 규명하는 연구가 수행되야 할 것이다.

참 고 문 헌

[1] Arnborg, S., "A Reduced State Enumeration-Another Algorithm for Reliability Evaluation", IEEE Trans. Reliability, R-27, pp 101-105, 1981.

[2] Corynen, F.C., A fast bottom-up algorithm for computing the cut sets of noncoherent fault trees, NUREG/CR-5242, US Nuclear Regulatory Commission, Washington, DC, October 1988.

[3] Erdmann, R. C., Leverenz, and F.L. Kirch, WHAMCUT, a computer code for fault tree evaluation, EPRI NP-803, Science Applications Inc. Los Altos, CA, June 1978.

[4] Fussell, J.B., Henry, E.B. and Marshall, N. H., MOCUS-a computer program to obtain minimal cut sets from fault trees, ANCR-1156, Aeroject, Nuclear Co., Idaho Falls, Idaho, March 1974.

[5] Kohda, T., E.J.Henley and K. Inoue, "Finding Modules in Fault Trees", IEEE Trans. Reliability, Vol. 38, pp 165-176, 1989.

[6] Lee, W.S., D.L. Grosh, F.A. Tillman and C.H. Lie., "Fault Tree Analysis, Methods and Applications-A Review", IEEE Trans. Relibility, Vol. 34, pp 194-203, 1985.

[7] Russell, K.D. and D.M. Rasmuson, "Fault Tree Reduction and Quantification-an Overview of IRRAS Algorithms", RESS, Vol. 40, pp 149-164, 1993.

[8] Willie, R. R., Computer-aided fault tree analysis, ORC 78-14, Operations Research Center, University of California, Berkeley, California, August 1978.

[9] Wilson, J.M., "Modularizing and Minimizing Fault Trees", IEEE Trans. Reliability, Vol. 34, pp 320-322, 1985.

95년 3월 최초 접수, 96년 2월 최종 수정