한국시뮬레이션학회 논문지
Vol. 22, No. 4, pp. 109-118 (2013. 12)
http://dx.doi.org/10.9709/JKSS.2013.22.4.109
ISSN 1225-5904

DEVS 형식론 기반의 Dynamic Reliability Block Diagram과 GPU 가속 기술을 이용한 신뢰도 분석 방법

하 솔[1] · 구남국[2†] · 노명일[3]

GPU-accelerated Reliability Analysis Method using Dynamic Reliability Block Diagram based on DEVS Formalism

Sol Ha · Namkug Ku · Myung-Il Roh

ABSTRACT

This paper adopts the system configuration to assess the reliability instead of making a fault tree (FT), which is a traditional method to analyze reliability of a certain system; this is the reliability block diagram (RBD) method. The RBD method is a graphical presentation of a system diagram connecting the subsystems of components according to their functions or reliability relationships. The equipment model for the reliability simulation is modeled based on the discrete event system specification (DEVS) formalism. In order to make various alternatives of target system, this paper also adopts the system entity structure (SES), an ontological framework that hierarchically represents the elements of a system and their relationships. To enhance the calculation time of reliability analysis, GPU-based accelerations are adopted to the reliability simulation.

Key words : Reliability analysis, dynamic reliability block, diagram, DEVS (Discrete Event System Specification), GPGPU (General-Purposed computing on Graphics Processing Units)

요 약

전통적으로 신뢰도 분석에 사용되는 Fault Tree Analysis의 경우 관련 분야의 전문가가 필요하고 작성자의 판단에 따라 신뢰도 분석 결과가 달라진다. 반면, Reliability Block Diagram의 경우 시스템 구성도나 Process Flow Diagram (PFD), Piping and Instrument Diagram (P&ID)을 기반으로 하기에 작성에 필요한 비용과 시간이 절감되는 장점이 있다. 본 논문에서는 Dynamic Reliability Block Diagram과 이산 사건 시뮬레이션에 널리 사용되는 DEVS 형식론을 이용하는 신뢰도 분석 방법을 제안한다. 또한 시스템 모델링 방법론 중 하나인 System Entity Structure/Model Base의 개념을 도입함으로써 다양한 설계 대안에 대한 신뢰도 분석 모델을 자동으로 생성할 수 있도록 하였다. 그리고 Reliability Block Diagram을 이용한 신뢰도 분석 시 오래 소요되는 계산 시간을 단축시키기 위해 GPU 가속 기술을 신뢰도 분석 시뮬레이션에 접목하였다.

주요어 : 신뢰도 분석, 동적 신뢰도 블록 다이어그램, 이산 사건 시스템 형식론, GPGPU

*본 연구는 (a) 지식경제부 산업원천기술개발사업(10035331, 시뮬레이션 기반의 선박 및 해양플랜트 생산 기술 개발), (b) 서울대학교 공학연구소, (c) 서울대학교 해양시스템공학연구소의 지원으로 이루어진 연구 결과의 일부임을 밝히며, 이에 감사드립니다.
접수일(2013년 7월 5일), 심사일(2013년 9월 9일), 게재 확정일(2013년 9월 24일)
[1] 서울대학교 공학연구소
[2] 서울대학교 해양플랜트창의인재양성사업단
[3] 서울대학교 조선해양공학과 및 해양시스템공학연구소
주 저 자 : 하솔
교신저자 : 구남국
E-mail; knk80@snu.ac.kr

1. 서 론

시스템의 신뢰도 분석은 항공, 우주, 원자력, 철도, 선박 등의 다양한 분야에서 필요로 하고 있다. 시스템의 신뢰도 분석에는 전통적인 방법인 fault tree analysis (FTA)를 비롯하여 Petri-Nets, reliability block diagram (RBD) 등의 여러 가지 방법이 있다(Shin & Seong, 2008).

Fault tree analysis (FTA) 방법은 시스템의 고장 확률을 분석하기 위해 사용되는 가장 전통적인 방법 중 하나

이다(Vesely et al., 1981). FTA는 시스템의 고장 원인을 분석하고 상위 시스템의 고장 원인을 하위 항목의 조합으로써 표현하는 트리 구조 기반의 신뢰도 분석 방법이다. Kim(2012)은 선박 기관 의장실에 대해 FTA 방법을 이용하여 fault tree를 생성하였으며, 이에 대한 고장 확률 계산을 위해 Markov chain 방법과 Bayesian network 방법을 사용하였다. 그러나 이 2가지 방법은 모두 fault tree analysis가 수행되었다는 전제 하에서 진행되며, fault tree analysis는 분석 자체가 오래 걸릴 수 있으며, 훈련된 전문가가 필요하고 또한 순차적 처리나 타이밍, 수리 등의 요소를 고려하기 어려운 단점이 있다.

신뢰도를 분석하는 다른 방법으로는 reliability block diagram (RBD)을 이용한 방법이 있다. RBD를 이용한 방법은 FTA와는 달리 시스템의 구성도를 그대로 채용한다는 장점이 있다. 그러나 신뢰도 분석 시 Markov chain이나 몬테칼로 방법을 사용하기 때문에 계산 시간이 다소 오래 소요된다는 단점이 있다.

본 논문에서는 시스템의 구성을 그대로 사용할 수 있는 RBD 방법을 사용하였으며, 신뢰도 분석 시뮬레이션을 효율적으로 구성하기 위해 이산 사건 시뮬레이션에 널리 사용되는 형식론인 이산 사건 시스템 형식론(Discrete Event System Specification; DEVS)과 시스템 모델링 방법론인 System Entity Structure/Model Base (SES/MB) 개념을 도입하였다. 또한 RBD에서 단점이 되는 계산 시간을 단축하기 위해 GPU 기반의 가속 기술을 신뢰도 분석 시뮬레이션에 적용하였다.

본 논문의 2장에서는 신뢰도 분석 방법인 FTA와 RBD에서 설명한다. 3장에서는 RBD를 구성하기 위해 도입한 SES/MB의 개념과 DEVS 형식론 기반의 모델에 대해서 설명한다. 4장에서는 RBD를 이용한 신뢰도 분석 시뮬레이션의 계산속도 향상을 위해 도입한 GPU 가속 기술에 대해 설명한다. 5장에서는 제안한 방법을 예시에 적용하여 신뢰도를 분석한 사례에 대해 기술하였으며, 마지막으로 6장에서는 결론에 대해 서술한다.

2. 신뢰도 분석 방법

시스템의 신뢰도를 분석하기 위한 방법으로 fault tree analysis (FTA)와 reliability block diagram (RBD)이 있다. 본 장에서는 2가지 방법을 이용하여 대상 시스템의 신뢰도를 분석하는 방법에 대해 간략히 설명하고 장단점에 대해 분석하였다.

2.1 Fault Tree Analysis

2.1.1 Static FTA

FTA는 가장 전통적으로 사용되는 신뢰도 분석 방법 중 하나이다(Vesely et al., 1981). FTA는 논리합(OR)이나 논리곱(AND) 등의 간단한 논리 게이트(logical gate)를 이용하여 시스템이 고장나는 원인에 대해 top-down 방식의 트리(tree) 구조로 분석하는 방법이다. FTA의 최상위 노드에는 시스템 고장이라는 항목이 위치해 있으며, 상위 시스템의 고장 원인이 트리의 하위 노드에 위치하고 이들의 논리 연산으로 조합된다. Fig. 1은 간단한 직렬 시스템을 FT로 표현한 예를 보여준다. 직렬 시스템에서 밸브 A와 밸브 B 중 하나만 고장이 나더라도 시스템 전체가 동작하지 않기 때문에 논리합을 의미하는 OR 게이트로 FT가 구성되어 있는 것을 확인할 수 있다.

FT에서는 시스템의 고장의 야기하는 최하위 사건들을 논리 연산자를 이용하여 조합함으로써 상위 시스템의 고장을 표현한다. 따라서 Fig. 1과 같이 고장을 야기하는 사건과 논리 연산자 역할을 하는 게이트(gate) 들을 조합하여 트리(tree) 구조로 표현한다. Fig. 2는 FT에서 사용되는 게이트의 예를 보여준다.

FT는 시스템의 고장 원인을 단위 사건의 조합으로 나타냄으로써 시스템의 여러 가지 고장 요소를 분석하고 시스템의 취약 부분을 분석하는 것에 도움을 주는 장점이 있다. 또한 고장이 발생했을 때 고장 원인을 단시간 내에 찾아내고 이에 대해 적절히 대처할 수 있는 일종의 기준

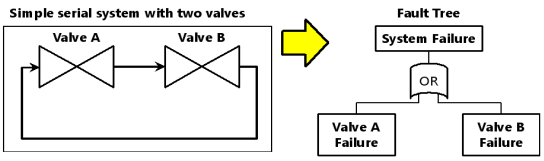

Fig. 1. Simple fault tree for a simple serial system with two valves

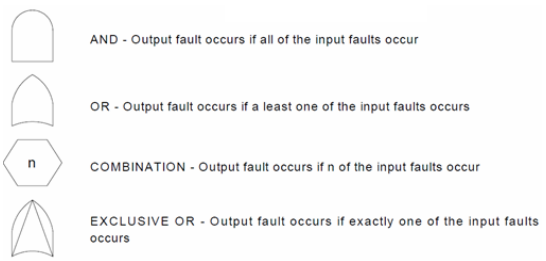

Fig. 2. Gate of the fault tree (Stamatelatos, 2002)

을 제시한다. 그러나 FT를 구성하는 주요 항목 중 하나인 게이트는 논리 연산자로서의 기능만을 수행하기에 동적인 상태의 시스템 고장 확률을 분석하기에는 다소 무리가 있다. 특히 사고 발생 원인의 시간에 따른 발생순서나 상호 의존 관계에 대해서는 반영하지 못한다.

2.1.2 Dynamic FTA

앞 절에서 설명한 것과 같이 static FTA에서는 사용되는 논리 게이트가 논리 연산자에 국한되어 있기에 사고 발생 원인의 시간에 따른 발생순서나 상호 의존 관계를 명확히 반영하기 힘들다. Fig. 3은 2개의 밸브로 하나의 스위치로 구성된 시스템을 예로서 보여준다. 밸브의 동작 여부가 밸브 이전에 설치된 스위치에 의존적이기 때문에 스위치의 동작 여부에 따라 밸브의 동작이 영향을 받는다. Fig. 3의 예와 같이 스위치보다 밸브 A가 먼저 고장 날 경우에는 밸브가 고장 남과 동시에 스위치가 작동하여 시스템의 흐름이 여분의 장비인 밸브 B를 통하여 흐를 수 있다. 따라서 그 후 스위치가 고장 나더라도 시스템은 고장 나지 않는다.

그러나 Fig. 4와 같이 스위치가 먼저 고장 날 경우에는 결과가 달라진다. 이후 밸브 A가 고장 나더라도 스위치가 제 기능을 발휘하지 못하기에 밸브 B로의 전환이 되지 않아 전체 시스템이 작동하지 않게 되는 결과를 가져온다.

2개의 예제에서 동일하게 스위치와 밸브 A가 고장났으나, 고장 순서에 따라 시스템이 작동하느냐의 여부가 결정된다. 따라서 이러한 상황을 고려하기 위해 static FTA에 사고 발생순서나 상호 의존 관계를 고려하기 위한 게이트를 추가로 고려한 dynamic FTA 방법이 도입되었다. Dynamic FTA에서는 앞의 예제와 같이 사고 발생순서나 여분의 장비, 그리고 상호 의존 관계를 고려할 수 있도록 Functional Dependency Gate (FDEP), Spare Gate (SG), Priority-AND Gate (PAND), Sequence-Enforcing Gate (SEQ) 등의 게이트를 도입하였다(Fig. 5 참조).

Fig. 6은 Fig. 3의 예제를 dynamic FT를 이용하여 표

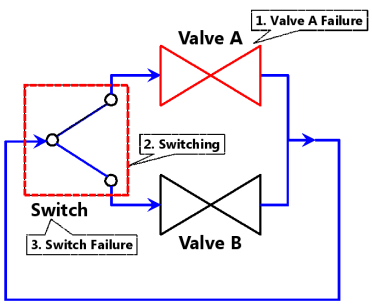

Fig. 3. Simple redundant system with two valves and a switch: switch failure after valve 'A' failure

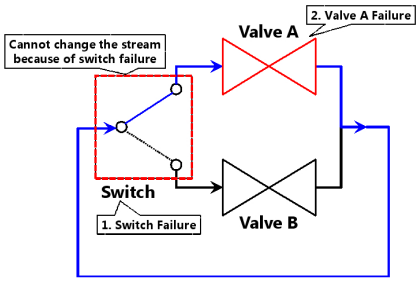

Fig. 4. Simple redundant system with two valves and a switch: valve 'A' failure after switch failure

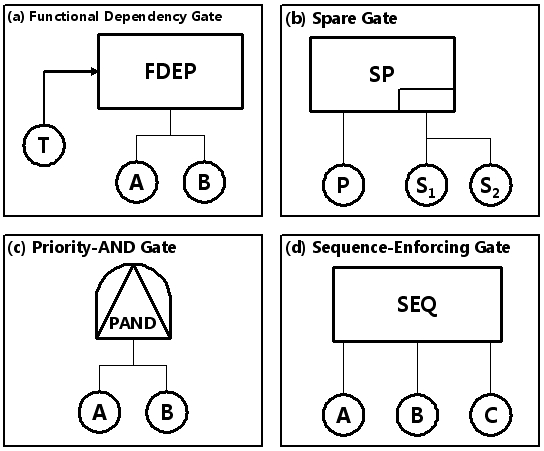

Fig. 5. Gates of the DFT: (a) functional dependency (FDEP) gate, (b) spare (SP) gate, (c) priority-and (PAND) gate, and (d) sequence-enforcing (SEQ) gate

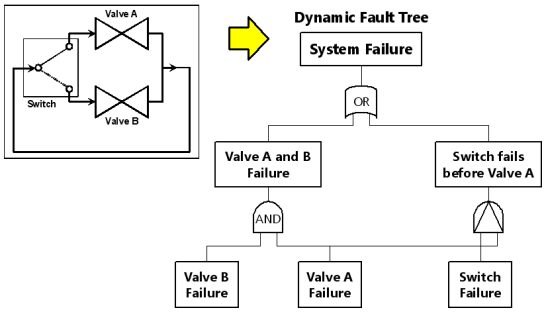

Fig. 6. Gates of the DFT: (a) functional dependency (FDEP) gate, (b) spare (SP) gate, (c) priority-and (PAND) gate, and (d) sequence-enforcing (SEQ) gate

현한 예시를 보여준다. Fig. 6에서 볼 수 있듯이 스위치와 밸브 A의 고장 순서를 반영하기 위해 PAND 게이트를 도입한 것을 확인할 수 있다.

2.1.3 FT를 이용한 신뢰도 계산

FT를 이용하여 신뢰도를 계산하는 방법은 Fig. 7의 과정으로 진행된다. 먼저 주어진 시스템을 분석하여 FT를 작성한다. 그리고 작성한 FT를 기반으로 Binary Decision Diagram (BDD), Markov chain, Bayesian network 등의 방법을 이용하여 시스템의 신뢰도(고장 확률; failure probability)을 계산한다.

BDD를 이용한 방법은 static FT를 계산하는 가장 빠른 방법이다. static FT는 AND, OR와 같은 논리 게이트로만 구성되어 있기에 논리 연산자의 계산 공식을 이용하여 손쉽고 빠르게 시스템 신뢰도를 계산할 수 있다. 그러나 dynamic FT에 대해서는 논리 연산자의 계산 공식만으로는 계산이 불가능하기에 적용할 수 없다는 단점이 있다.

Markov chain을 이용한 방법은 BDD를 이용한 방법과는 달리 dynamic FT에도 적용 가능하다. FT를 생성한 후 최하위 노드에 위치한 고장 원인의 발생 여부(operating or failure)를 조합하여 다수의 상태를 만든다. 그리고 각각의 상태에 대해 FT를 이용하여 작동 여부를 판단하고, Markov chain의 계산 결과와 조합하여 시스템의 고장 확률을 계산한다. 그러나 이 방법은 FT와는 별개로 다시 Markov chain을 생성해야 하기에 FT와는 다소 연계성이 떨어진다. 또한 Makov chain을 구성하는 상태의 개수도 고장 원인의 개수(n)가 증가함에 따라 기하급수적으로 증가($2^n \times n!$)하기에 장비의 개수가 늘어날수록 계산 시간 역시 기하급수적으로 늘어난다.

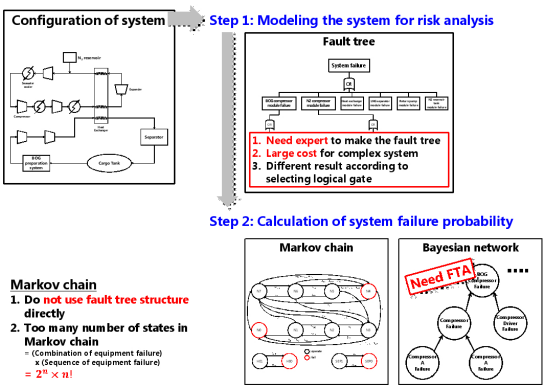

Fig. 7. Disadvantage of FTA-based reliability analysis method

Dynamic FT 기반의 시스템 신뢰도를 계산하는 또 하나의 방법으로 Bayesian network를 이용하는 방법이 있다(Bobbio et al., 2001). Bayesian network는 Markov chain과는 달리 FT의 구조를 그대로 채용하면서 이를 확률 계산에 사용되는 Bayesian network로 변환하는 알고리즘을 제시한다. 계산 속도나 범용성 측면에서 Markov chain보다 빠른 결과를 도출한다는 장점을 가지고 있다.

2.2 Reliability Block Diagram

시스템의 신뢰도 분석을 위해 전통적으로 사용되어 온 FTA는 관련 분야의 전문가와 FT 관련 전문가가 주어진 시스템을 분석하여 FT를 작성해야하는 단점이 있다. 시스템이 복잡할수록 FT의 작성 비용과 시간이 더 소모되며, 결과물은 FT도 전문가의 관련 분야 지식과 적절한 논리 게이트의 사용 여부에 따라 그 결과가 달라진다. Bayesian network 방법을 이용하면 dynamic FT에 대한 신뢰도 계산 결과를 단시간 내에 얻는다는 장점이 존재하지면 근본적으로 FT를 만들어야 한다는 점은 변하지 않는다.

신뢰도를 분석하기 위한 방법 중 또 다른 방법으로 Reliability Block Diagram (RBD)을 이용한 방법이 존재한다. RBD를 이용한 방법은 시스템을 분석하고 고장 여부의 인과 관계를 면밀히 분석하여 이를 FT로 작성하는 것과는 달리 시스템을 구성하는 하위 장비들의 연관 관계를 그대로 채용하는 방식을 말한다. RBD는 시스템의 구성도를 그대로 채용하여 변환할 수 있으며, 각 하위 장비의 고장 여부에 따라 전체 시스템의 고장 여부를 확인할 수 있는 알고리즘을 제시한다.

RBD를 이용하면 FT를 작성하는 과정을 거치지 않고 기존에 작성된 시스템 구성도나 PFD (Process Flow Diagram), P&ID (Piping & Instrument Diagram)을 이용하여 손쉽게 신뢰도 분석을 위한 모델을 작성할 수 있

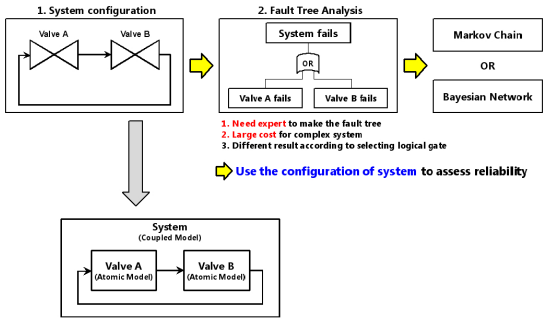

Fig. 8. Reason for the use of a RBD

다. 또한 시스템 구성도와 하위 시스템 간의 연결 관계를 기반으로 하기에 하위 장비들 간의 인과 관계도 FT보다 명확하게 정의할 수 있다. 본 논문에서는 RBD를 정의하는 방법으로써 구조적 모델을 표현하는 System Entity Structure (SES)와 행위 모델을 표현하는 Model Base (MB)를 도입하였다.

3. DEVS 형식론 기반 Reliability Block Diagram을 이용한 신뢰도 분석 방법

RBD를 이용한 신뢰도 분석 방법을 이용하기 위해서는 RBD의 근간이 되는 시스템 구성도, PFD, P&ID로부터 효율적으로 RBD를 모델링할 수 있는 도구를 필요로 한다. 또한 몬테칼로 방법을 이용하여 신뢰도를 분석하고 고장 확률을 계산하기 위해서는 RBD와 연계하여 시뮬레이션을 효율적으로 수행할 수 있어야 한다.

본 논문에서는 Zeigler et al.(2000)이 제안한 시스템의 모델링 방법론을 이용하여 RBD를 모델링하고 이를 이용하여 시뮬레이션을 수행하였다. Zeigler et al.(2000)은 이산 사건 시뮬레이션을 효율적으로 구성할 수 있는 형식론이 이산 사건 시스템 명세(Discrete Event System Specification; DEVS)를 제시하였다. 또한, 모의 대상인 시스템을 하위 시스템의 구조적인 연결 관계를 나타내는 system entity structure (SES)와 동적인 행동을 담당하는 model base (MB)로 분할하여 표현하는 방법을 제시하였다. 본 논문에서는 DEVS 형식론과 SES/MB 모델링 방법론을 RBD와 연계하여 신뢰도 분석 시뮬레이션에 적용하였다.

3.1 System Entity Structure를 이용한 시스템 모델링 및 설계 대안 자동 생성 방법

SES란 시스템을 구성하는 하위 시스템의 구조적인 연결 관계를 트리(tree) 구조로 표현한 것을 말한다. SES에서 사용하는 객체와 객체 간의 구조적인 연결 관계는 하나의 상위 객체가 여러 개의 하위 객체로 구성되는 것뿐만 아니라 종류를 구분하는 것도 가능하다. 이로 인해 SES는 pruning이라는 과정을 통해 종류를 선택함으로써 여러 가지 설계 대안을 생성해 낼 수 있으며, 이는 신뢰도 분석 시 주어진 시스템에 대한 여러 가지 설계 대안을 검토할 수 있도록 도움을 준다.

Fig. 9는 육상 및 해상의 LNG 생산 플랜트에서 사용되는 DMR (dual mixed refrigerant) 공정의 구성도 예시이다(Ha, 2013). Fig. 9의 예시뿐만 아니라 LNG 생산 플랜트에 사용되는 여러 가지 공정을 분석하여 Fig. 10과

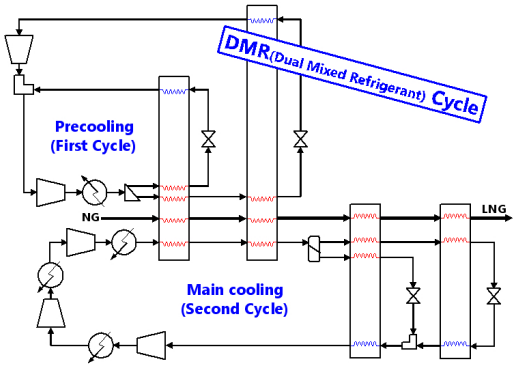

Fig. 9. Configuration of the DMR (dual mixed refrigerant) cycle for LNG production plants

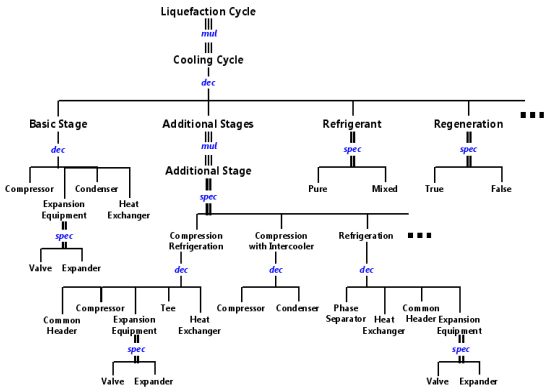

Fig. 10. System entity structure for the LNG liquefaction cycle

같이 대상 분야의 SES를 구성할 수 있다.

SES를 pruning 하는 과정을 거치면 다음과 같이 여러 가지 설계 대안에 해당하는 Pruned Entity Structure (PES)를 생성할 수 있다. 이는 다음 장에 설명할 MB 내의 DEVS 모델과 합성하여 신뢰도 분석을 위한 시뮬레이션 모델로 구성 가능하다.

3.2 DEVS 형식론을 이용한 신뢰도 분석 방법

주어진 시스템의 신뢰도를 분석하기 위해서는 시뮬레이션 수행 동안 매 단위 시간마다 각 장비에 대한 고장 여부를 확인하는 기능과 이에 대한 유지 보수 기능이 기본적으로 필요하다. 그리고 하위 시스템의 고장 유무를 종합적으로 검토하여 전체 시스템이 정상적으로 동작하는지를 판단하는 기능도 필요하다.

본 논문에서는 신뢰도 분석 시뮬레이션을 위해 기본적으로 필요한 4가지 기능을 다음과 같이 정의하였다.

(1) 단위 시간 동안의 고장 여부 확인
(2) 고장에 따른 유지보수 작업
(3) 다수 장비에서 유입되는 신호에 대한 정합성 판단
(4) 하위 시스템의 고장 여부를 종합하여 전체 시스템의 정상 동작 여부 확인

4가지 기능은 DEVS 형식론 기반의 단위 모델(atomic model)로 구성하였으며, 본 논문에서는 이 중에서 다음의 3가지 기능을 소개한다.

3.2.1 단위 시간 동안의 고장 여부 확인

단위 시간 동안 단위 장비나 시스템의 고장 여부는 Fig. 11과 같이 구성된 'Equipment Failure' 단위 모델로 구성하였다. 'Equipment Failure' 단위 모델은 시뮬레이션 시작 시 매 단위 시간마다 장비의 작동 여부를 확인하는 'THROW' 상태를 유지한다. 매 단위 시간마다 주어진 확률에 따라 장비의 작동 여부를 결정하며, 그 결과에 따라

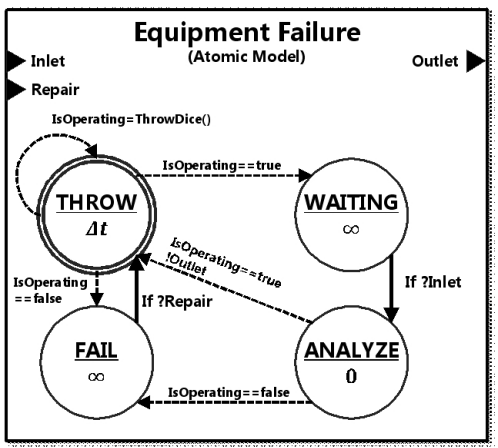

Fig. 11. DEVS atomic model to check the failure of a system

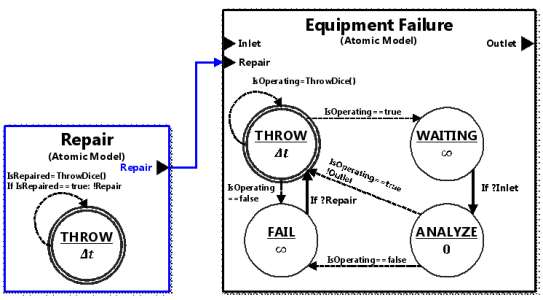

Fig. 12. DEVS atomic model to repair failed system

장비가 고장나면 'FAIL' 상태로, 그렇지 않으면 'WAITING' 상태로 전환한다.

3.2.2 고장에 따른 유지 보수 작업

대부분의 시스템은 시스템의 하위 구성 요소에 고장이 발생할 것을 대비하여 일정 주기 간격으로 유지 보수를 수행하는 과정을 거친다. 유지 보수 작업의 과정을 반영하기 위해 Fig. 12와 같이 유지 보수를 수행할 수 있는 기능을 'Repair' 단위 모델로 구성하였다. 'Repair' 단위 모델은 Fig. 12와 같이 'Equipment Failure' 모델과 연결하여 사용된다.

3.2.3 전체 시스템의 정상 동작 여부 확인

시뮬레이션 단위 시간이 진행되는 동안 'Equipment Failure' 모델과 'Repair' 모델을 이용하여 각 장비에 대한 고장 여부를 결정한다. 그리고 최종적으로 각 장비의 고장 여부를 종합하여 전체 시스템이 정상 동작 하는지에 대한 판단을 수행해야 한다. 'Analyzer' 단위 모델은 시뮬레이션 단위 시간 동안 결정된 각 장비들의 고장 여부를 종합하기 위해 전체 시스템을 순환하는 신호를 발생시킨다. 그리고 이 신호가 전체 시스템을 순환하고 다시 'Analyzer' 모델로 돌아오느냐 여부에 따라서 현 시간의 전체 시스템 고장 여부를 판단한다.

4. GPU 가속 기술을 이용한 계산 시간 단축

FT를 작성하는 것과 비교할 때 RBD는 소요 비용과 시간이 단축되고 사용자가 쉽게 내용을 파악할 수 있다는 장점이 있다. 그러나 Markov chain이나 몬테칼로 방법을 이용하여 신뢰도(고장 확률)를 계산하기에 Bayesian network를 이용하는 FT에 비해서 계산 시간이 다소 오래 걸린다는 단점이 있다. 따라서 본 논문에서는 GPU 가속 기술을

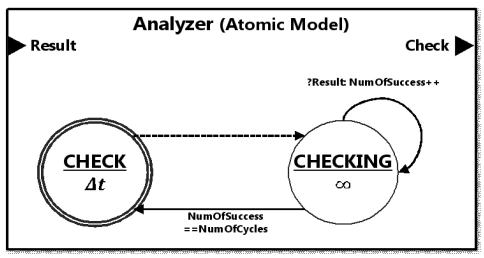

Fig. 13. DEVS atomic model to analysis whole system failure

3장에서 설명한 DEVS 기반의 신뢰도 분석 시뮬레이션 방법과 접목하여 몬테칼로 방법을 이용한 계산 방법의 계산 소요 시간을 단축시키고자 하였다.

4.1 병렬 처리 방법의 종류

병렬 처리 방법으로는 클러스터 컴퓨팅(Cluster Computing), 그리드 컴퓨팅(Grid Computing), 그리고 GPGPU (General-Purpose computing on Graphics Processing Units) 방법 등이 있다. 이 중 GPGPU (General-Purpose computing on Graphics Processing Units: 그래픽 처리 장치를 통한 일반 목적의 컴퓨팅) 방법은 컴퓨터 그래픽스를 위한 계산만 다루는 GPU (Graphics Processing Units)를 사용하여 CPU (Central Processing Unit)가 전통적으로 취급했던 응용 프로그램들의 계산을 수행하는 기술이다.

GPGPU 방법은 프로세서가 부담을 갖는 작업을 그래픽 프로세서가 대신 처리함으로써 시스템의 효율을 높이는 것을 기본으로 한다(Nguyen, 2007). GPGPU 방법은 비용적인 면을 크게 절감할 수 있고 사용자가 손쉽게 하드웨어를 확보할 수 있다는 장점이 있다. GPU를 이용한 병렬 처리 기술로는 대표적으로 nVIDIA에서 개발 중인 CUDA (Compute Unified Device Architecture)와 AMD에서 개발 중인 CTM (Close To the Metal)이다. 본 논문에서는 신뢰도 분석 시뮬레이션의 속도 개선을 위해 GPU를 이용한 병렬 처리 기술인 CUDA를 적용하였다.

4.2 GPU를 이용한 신뢰도 분석 시뮬레이션의 병렬 처리 과정

GPU를 이용한 병렬 처리 기술인 CUDA를 이용하여 신뢰도 분석 시뮬레이션을 수행하는 과정은 다음의 4가지 과정으로 진행된다.

(1) DEVS 모델 초기화
(2) CPU에서 GPU로의 데이터 복사
(3) 단위 시간 시뮬레이션 수행
(4) GPU에서 CPU로의 데이터 복사 및 후처리

각각에 대한 상세한 설명은 다음과 같다.

4.2.1 DEVS 모델의 초기화

계산 수행을 위해 DEVS 모델을 초기화 한다. 이 때 초기화된 데이터는 CPU에서 관리하는 메모리인 host memory에 저장된다. DEVS 모델을 초기화 하는 과정에 대한 pseudo code는 다음과 같다.

```
// allocate host memory
host_models = AllocateModels(x_size, y_size)

// initialize models
For each model in host_models
  model.Initialize()  // initialize model
End For each
```

4.2.2 CPU에서 GPU로의 데이터 복사

초기화가 완료되면 GPU에서 연산을 수행하기 위해 host memory에 저장한 데이터를 GPU의 device memory로 복사한다. Device memory로 데이터를 복사하게 위해 CUDA에서 사전에 정의된 함수를 이용한다. 이에 대한 pseudo code는 다음과 같다.

```
// allocate device memory
device_models = cudaMalloc(x_size * y_size)

// copy from host to device
cudaMemcpy(host_models, device_models, x_size * y_size)
```

4.2.3 단위 시간 시뮬레이션 수행

GPU는 복사한 데이터를 이용하여 다수의 산술연산장치의 집합인 Grid를 이용하여 병렬 연산을 수행한다. 단위 시간 동안의 시뮬레이션에는 DEVS 형식론에 따라 각 모델의 output function을 먼저 실행한다. 이에 의해 발생한 출력을 처리하기 위해 다음으로 external transition function과 병렬 처리 시 발생하는 입출력, 상태의 충돌을 처리하기 위한 confluent transition function을 실행한다. 그리고 마지막으로 internal transition function을 실행함으로써 단위 시간의 시뮬레이션을 마치게 된다. 이에 대한 pseudo code는 다음과 같다.

```
// setup grid property
SetGridProperty(threads, blocks)

// output function
OutputFunc<<<blocks, threads>>>(device_models)

// external transition function
ExTransFunc<<<blocks, threads>>>(device_models)

// confluent transition function
ConfTransFunc<<<blocks, threads>>>(device_models)

// time synchronize
__syncthreads()

// internal transition function
IntTransFunc<<<blocks, threads>>>(device_models)
```

4.2.4 GPU에서 CPU로의 데이터 복사 및 후처리

단위 시간만큼 진행 후 GPU에서의 연산이 종료되면 계산 결과를 CPU의 host memory로 다시 복사한다. CPU에서는 host memory에 저장된 데이터를 이용하여 시스템 고장 여부 확인 및 후처리 작업을 진행한다. 이에 대한 pseudo code는 다음과 같다.

```
// copy from device to host
cudaMemcpy(device_models, host_models, x_size * y_size)

// post process
ConfirmSystemFailure(host_models)
```

5. DEVS 형식론 기반의 RBD를 이용한 신뢰도 분석 예시

본 논문에서 제안한 방법의 효용성을 검증하기 위해 다음의 2가지 예시에 대해 제안한 방법을 적용하였다. 그리고 GPU 가속 기술을 적용하였을 때의 개선 사항을 분석하기 위해 적용 전과 적용 후의 계산 시간을 비교 분석하였다.

5.1 Firewater pumping system

Fig. 14는 간략화한 firewater pumping system의 구성도를 보여준다. Fig. 14에서 볼 수 있듯이 firewater pumping system은 여분을 포함하여 2개의 fire pump를 가지고 있으며, 이를 작동시킬 수 있는 1개의 engine을 가지고 있다.

이를 이용하여 FTA 방법과 논문에서 제안한 방법을 이용하여 신뢰도를 분석한 결과는 다음 표와 같다.

Table 1에서 볼 수 있듯이 RBD는 몬테칼로 방법을 이용하여 신뢰도를 계산하기에 FTA에 비하여 계산 시간이 오래 소요되는 것을 확인할 수 있다. 그러나 GPU 가속 기술을 도입하였을 때는 계산 시간이 1/5로 줄어들며, FTA 방법을 이용한 계산 시간과 비교할 때 그 차이가 크지 않음을 확인할 수 있다.

5.2 Hypothetical computer system

Fig. 15는 간략화한 hypothetical computer system의 구성도를 보여준다. Fig. 15에서 볼 수 있듯이 hypothetical computer system은 프로세스 시스템(process system), 메모리 시스템(memory system), 응용 프로그램 (application), 그리고 2개의 버스(bus)로 구성된다.

각각의 하위 시스템은 다수의 하위 장비와 여분의 장비로 구성되며, 일부 장비들은 의존적인 관계를 가지고

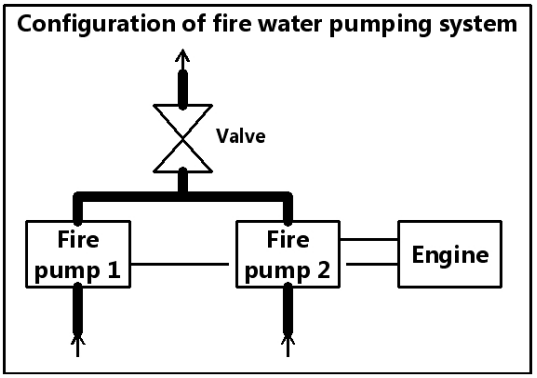

Fig. 14. Configuration of firewater pumping system

Table 1. Comparison of reliability analysis results for the firewater pumping system

Reliability analysis method	Probability of system failure	Execution time
FTA	3.040%	0.49 [sec]
RBD based on DEVS	3.016%	4 [sec]
RBD based on DEVS and GPGPU	3.016%	0.78 [sec]

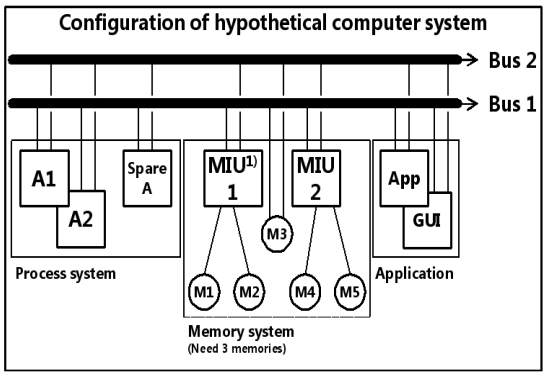

Fig. 15. Configuration of hypothetical computer system

Table. 2. Comparison of reliability analysis results for the hypothetical computer system

Reliability analysis method	Probability of system failure	Execution time
FTA	4.899%	6.3 [sec]
RBD based on DEVS	4.854%	403.4 [sec]
RBD based on DEVS and GPGPU	4.854%	34.1 [sec]

있기에 dynamic FTA나 DRBD를 이용하여 신뢰도를 분석하였다. 시스템의 신뢰도를 분석한 결과는 Table 2와 같다. Table 2에서 볼 수 있듯이 RBD를 이용한 방법으로는 신뢰도 분석을 위한 계산 시간이 많이 소요되는 것을 확인할 수 있다. 그러나 GPU 가속 기술을 도입하였을 때는 10배 정도의 성능 향상이 있으며, FTA를 이용한 방법과 비교할 때도 계산 시간이 크게 차이나지 않는 것을 확인할 수 있다.

6. 결 론

본 논문에서는 주어진 시스템의 신뢰도를 분석하기 위해 reliability block diagram (RBD)을 도입하였다. Fault tree (FT)를 이용한 방법의 경우 FT 작성을 위해 관련 분야의 전문가가 필요하며, 작성 시간과 결과도 전문가의 판단에 의존되는 경향이 많다. 그러나 RBD의 경우 FT에 비해 시스템 구성도나 PFD, P&ID를 기반으로 보다 쉽게 작성할 수 있다는 장점이 있다.

RBD의 작성과 여러 가지 설계 대안에 대한 신뢰도 분석을 지원하기 위해 이산 사건 시뮬레이션에서 사용되는 DEVS 형식론과 시스템을 모델링하는 방법론인 SES/MB의 개념을 채용하였다. 그리고 신뢰도 분석을 위한 4개의 DEVS 단위 모델을 제안하였다. 또한 RBD를 이용한 신뢰도 분석 시 단점이 되는 계산 시간의 향상을 위해 GPU 기반의 가속 기술을 도입하였다. 그리고 일부 예시에 대해서 적용하여 FT 방법과의 계산 시간 비교를 통해 GPU 가속 기술 도입의 효용성을 확인하였다. 향후에는 다양한 예제에 대해서 제안한 방법을 적용하여 보다 상세한 검증을 시도할 예정이며, RBD를 구성하는 장비들을 DEVS 형식론이 아닌 다른 모델링 기법을 도입하여 계산 시간 향상을 도모할 예정이다.

References

1. Bobbio, A., Portinale, L., Minichino, M. and Ciancamerla, “Improving the analysis of dependable systems by mapping fault trees into Bayesian networks”, Reliability Engineering & System Safety, Vol. 71, No. 3, pp. 249-260, 2001.3.
2. Ha, S., Ontological Modeling for Process and Reliability Simulation of LNG FPSO Liquefaction Cycle based on the DEVS formalism considering Ship Motion Effect, Ph.D Thesis, Seoul National University, 2013.
3. Kim, S.T., Quantitative Reliability Analysis of Ship Machinery Systems using Dynamic Fault Trees and Bayesian Networks, Master Thesis, Seoul National University, 2012.
4. Nguyen, H., GPU Gems 3: Programming Techniques for High Performance Graphics and General-Purpose Computation, Addison-Wesley Education Publishers, Inc., 2007.
5. Shin, S.K. and Seong, P.H., “Review of various dynamic modeling methods and development of an intuitive modeling method for dynamic systems”, Journal of Korean Nuclear Society, Vol. 40, No. 5, pp. 375-386, 2008.
6. Stamatelatos, M., Fault Tree Handbook with Aerospace Applications Version 1.1, 2002.
7. Vesely, W.E., Goldberg, F.F., Roberts, N.H. and Haasl, D.F., Fault tree handbook, NUREG-0492, US Nuclear Regulatory Commission, 1981.
8. Zeigler, B.P., Praehofer, H. and Kim, T.G., Theory of modeling and simulation, Academic press New York, NY, 2000.

하 솔 (hasol81@snu.ac.kr)

2003 서울대학교 조선해양공학과 학사
2013 서울대학교 조선해양공학과 박사
2013.3~현재 서울대학교 공학연구소 선임연구원

관심분야 : 해양 플랜트 설계 최적화, 모델링&시뮬레이션, 수중운동체(잠수함, 어뢰) 교전 시뮬레이션, 격자 기반 시뮬레이션(Cellular Automata, Lattice Gas Automata, Lattice Boltzmann Method)

구 남 국 (knk80@snu.ac.kr)

2004 서울대학교 조선해양공학과 학사
2012 서울대학교 조선해양공학과 박사
2012.3~현재 서울대학교 해양기술인력양성사업단/공학연구소 선임연구원

관심분야 : 해양플랜트 설계 및 생산, 다물체계 동역학 및 제어, 용접/전처리 로봇

노 명 일 (miroh@snu.ac.kr)

1998 서울대학교 공과대학 조선해양공학과 학사
2000 서울대학교 공과대학 조선해양공학과 석사
2005 서울대학교 공과대학 조선해양공학과 박사
2005~2007 서울대학교 공학연구소/해양시스템공학연구소 선임연구원
2007~2012 울산대학교 조선해양공학과 조교수
2013~현재 서울대학교 조선해양공학과 부교수

관심분야 : 전산선박설계 및 생산, 시뮬레이션 기반 설계 및 생산, 최적 설계, 해양구조물 설계