순차상태전이금지(FSSTP)를 이용한 교착상태 관리제어를 위한 재구성 방법

Reconfiguration method for Supervisor Control in Deadlock status Using FSSTP(Forbidden Sequence of State Transition Problem)

송 유 진*, 이 은 주, 이 종 근
(Yu-Jin Song, Eun Joo Lee, and Jong-Kun Lee)

Abstract : The object of this paper is to propose a method to deal with the problem of modeling user specifications in approaches based on supervisory control and Petri nets. However, most of Petri Net approaches are based on forbidden states specifications, and these specifications are suitable the use of tool such as the reachability graph. But these methods were not able to show the user specification easily and these formalisms are generally limited by the combinatorial explosion that occurs when attempting to model complex systems. Herein, we propose a new efficient method using FSSTP (Forbidden Sequences of State-Transitions Problem) and theory of region. Also, to detect and avoid the deadlock problem in control process, we use DAPN method (Deadlock Avoidance Petri nets) for solving this problem in control model.

Keywords : reconfiguration, supervisor control, petri nets, discrete event system, forbidden sequences of state-transitions problem, deadlock avoidance petri net, constrained synchronous reachability graph

I. 서론

최근에 사용자 요구사항을 반영하는 이산사건 제어를 위한 제어기설계 문제에 많은 연구 관심을 가지고 있으며[1], 특히 감독제어(supervisory control)을 기반으로 사용자 요구사항에 충실한 모델링 문제를 다루는 데 많은 연구 초점이 집중되고 있다. 제어기 설계에 있어서 플랜트 모델은 모든 가능한 제조 공정을 명시하는데 반하여, 사용자 요구사항은 제한된 공간으로 플랜트의 동작을 한정하는 특성을 가지고 있다[1,2]. 따라서 유연생산 시스템이 분산처리 되는 환경으로의 전환시점에서 제어 모델링 문제의 가장 어려운 점은 최종 제어기가 사용자의 요구들을 모두 만족시킨다는 것을 보증하여야 하는 것이다.

기존의 패트리넷을 이용한 제어 모델링 연구에서는 상태금지문제(FSP: Forbidden State Problem)나 상태전이금지문제(FSTP: Forbidden State-Transition Problem)등을 정의하고 해당 문제를 해결하는 방법을 제시하였다[3,4]. 상태금지문제는 교착상태 문제에 도달하는 상태나 사용자 요구사항에 의해 금지해야 할 상태를 찾아내고 해당 상태에 도달하지 못하도록 제어하는 것이다. 또한, 상태전이금지문제는 어떤 상태에 도달하는 여러 경로 중에서 사용자 요구사항에 의하여 금지할 경로를 찾고 해당 경로에 해당하는 트랜지션들을 점화하지 못하게 제어하는 것이다.

그러나 [5,6]에서 소개한 바와 같이 패트리넷을 이용한 제어 모델링 문제에서 병렬로 작업하는 기계들에 어떤 순서를 주어서 병합하는 경우에는 상태금지문제나 상태전이금지문제로 설명할 수 없음을 알 수 있다. 이의 해결을 위해, 순차상태 전이금지 문제인 FSSTP(Forbidden Sequences of State-Transitions Problem)를 정의한다.

그리고 프로세스 모델과 사용자 요구사항을 합성하는 강제적 동기식 도달 가능 그래프(CSRG: Constrained Synchronous Reachability Graph)와 Ghaffari의 영역이론(theory of region) [3, 4]을 이용하여 사용자 요구사항에 충실한 제어모델을 생성하였다[5,6]. 한편, 실시간으로 움직이는 플랜트 모델에서의 자원 할당은 자원을 다루는 시스템의 많은 형태에서 중요한 제어 책무를 가지고 있다. 자원할당을 통해 시스템이 수행되는 많은 플랜트 모델의 경우에서 자원 프로세스는 시스템에 입력, 요구, 응답, 사용 그리고 요구된 자원들을 해제한 후 시스템에서 나가게 된다.

이러한 상황에서 시스템은 교착상태에 빠질 수 있는데, 교착상태를 해결하는 것은 실시간으로 자원을 할당하는 시스템에서 매우 중요한 문제이다. 즉, 이러한 시스템이 원활한 수행이 가능한지를 판가름 하는 문제라고 볼 수 있다.

따라서, 플랜트 모델에 교착상태와 같은 문제가 생기는 경우 사용자 요구사항에 충실한 제어 모델을 생성할 수 없으며, 수행이 불가능하다. 이에, 본 연구에서는 플랜트 모델에 교착상태와 같은 문제가 있는지 미리 확인하고, 문제가 있는 경우엔 이 문제를 해결한 후 사용자 지향의 제어 모델을 구축하고자 한다.

본 연구에서는 이미 검증된바 있는 교착상태 회피 방법인 DAPN(Deadlock Avoidance Petri Net)을 이용하여 초기 플랜트 모델이 보유한 문제를 해결하여 재구성하고, 재구성된 플랜트 모델과 사용자 요구사항 모델을 이용하여 제어된 모델을 구축하는 방법을 제안하고자 한다.

II. 제어모델

1. 순차상태전이금지 문제

순차상태전이금지 문제(FSSTP: Forbidden Sequences of State-

* 책임저자(Corresponding Author)
논문접수 : 2007. 10. 9., 채택확정 : 2008. 1. 23.
송유진, 이종근 : 창원대학교
(syj@changwan.ac.kr/jklee@changwon.ac.kr)
이은주 : 한양대학교(eunjoolee@hanyang.ac.kr)

Transitions Problem)에 대한 정의를 위하여 그림 1의 예를 이용하고자 한다.

그림 1(a)는 서로 독립적으로 어떠한 제약 없이 동작하는 두 개의 머신에 관한 플랜트 패트리넷 모델이다. 그림 1(b)는 이 두 머신을 합성한 도달가능 그래프(reachability graph) 이다. 그림 1(b)에서, 마킹 M_0 에서부터 마킹 M_2 까지 $t_4, t_1t_2t_3t_4$, $t_1t_4t_2t_3\cdots$ 등 몇 개의 경로가 존재하고 있음을 알 수 있으며, 그림 1(c)에서와 같이 사용자 요구조건이 경로 $t_1t_4t_2t_3$ 를 금지하고자 한다고 가정했을 때 해당 경로를 제거하여도 기존의 모든 마킹은 허용될 수 있으며 모든 트랜지션은 점화 가능하다는 것을 알 수 있다.

이러한 문제를 기존의 상태금지문제로 본다면 M_2 를 금지상태로 하여 M_2 를 제거한다. 그러나, M_2 를 제거하면 $t_1t_4t_2t_3$ 뿐만 아니라 허용해야 하는 $t_4, t_1t_2t_3t_4, \cdots$ 등의 경로까지도 제거가 된다. 따라서, 상태금지 문제가 아님을 알 수 있다. 또한 이 문제를 상태전이금지문제로 본다면, 문제 해결을 위하여 $t_1t_4t_2t_3$ 를 제거한다. 그러나, $t_1t_4t_2t_3$ 를 제거하면 허용해야 하는 $t_1, t_1t_4, t_1t_2\cdots$ 등의 경로까지도 제거되므로 사용자 요구사항을 만족하지 못하게 된다.

따라서, 그림 1에서 설명하는 문제는 기존의 상태금지문제나 상태전이금지문제와는 다른 개념의 문제임을 알 수 있다[3,5,6]. 이 새로운 문제를 순차상태전이금지문제라 명하고 다음의 정의 1로 정의한다.

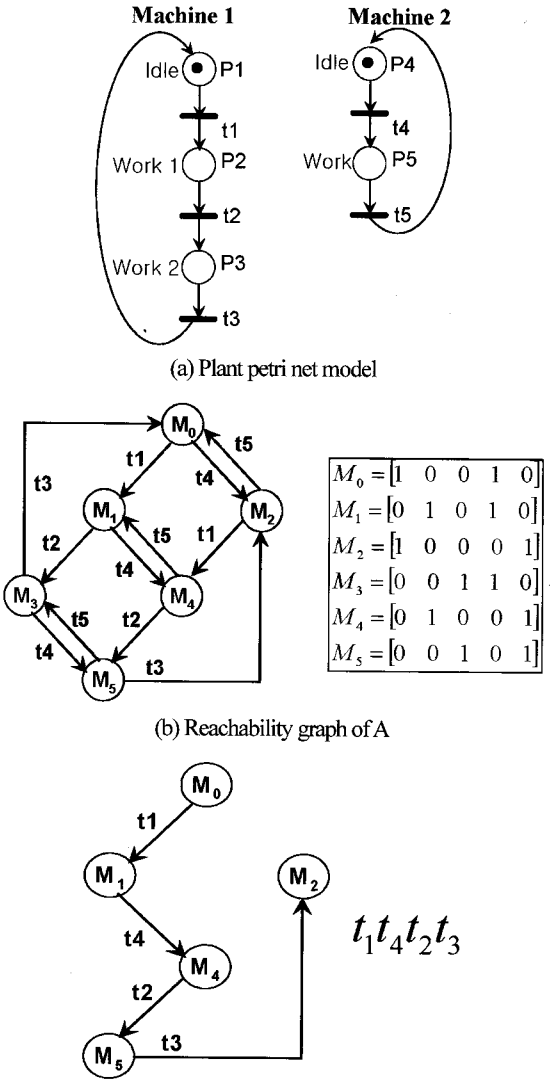


그림 1. 순차상태전이금지문제의 예제모델.
Fig. 1. Example model of FSSTP.

정의 1 [5,6]: 순차상태전이금지문제

플랜트 모델 (N_p, M_0) 에서 원하는 동작은 다음의 조건들을 반드시 만족해야 한다.

(a) $M_0 \in DB$ (DB: 원하는 동작들)

(b) $\forall M \in DB,\ \exists \sigma \mid (M_0 \xrightarrow{\sigma} M) \in DB$

(σ : 점화가능한 트랜지션들의 경로)

(c) $\exists (M \xrightarrow{\gamma} M') \in R(N_p, M_0)$ and $(M \xrightarrow{\gamma} M') \notin DB$

(γ : 금지되어야 할 트랜지션들의 경로, $R(N_p, M_0)$: 패트리넷 모델의 도달 가능 그래프)

2. 강제된 동기식 도달 가능 그래프

사용자 요구사항 패트리넷 모델과 플랜트 패트리넷 모델을 병합하는 방법으로 강제된 동기식 도달 가능 그래프(CSRG: Constrained Synchronous Reachability Graph)를 제안하였다[5,6]. 이전 연구에서 제안된 동기식 접근 가능 그래프는 Ramadge와 Wonham의[3] 연구로부터 기본 개념을 설계하였으며, 문서로 된 사용자 요구사항에 대한 패트리넷 모델과 플랜트 패트리넷 모델을 통합하는 방법을 기반으로 하였다. 사용자 요구사항 패트리넷 모델의 트랜지션은 플랜트 모델의 트랜지션을 이용하여 설계한다.

강제적 동기식 도달 가능 그래프는 두 모델(사용자 요구사항 패트리넷 모델과 플랜트 패트리넷 모델)에 대한 도달 가능 그래프를 병합할 때 기존의 카테시안 프로덕트로 얻을 수 있는 도달 가능 그래프보다 작은 크기의 결과를 산출한다. 또한 동기식 도달 가능 그래프를 적용한 결과로 순차상태전이금지문제를 쉽게 찾아낼 수 있다.

2.1 Ghaffari의 영역이론의 소개

일반적으로 두 개 이상의 넷을 통합하려 하는 경우, 주어진 문제에 대한 오토마타와 동일한 형태인 영역이론을 이용한다. Ghaffari가 재정의한 영역이론은 오토마타의 영역이론을 수정한 것으로 사용자 요구사항에 맞는 모델을 구축하기 위하여 초기 플랜트 패트리넷 모델에 추가할 제어 플레이스를 산출한다[3,4]. Ghaffari의 영역이론을 이용한 사용자 요구사항에 맞는 제어기 생성은 이미 검증된 방법이며, 본 연구 또한 사용자 요구사항에 맞는 시스템의 재구성이 목적이므로 원하는 제어기 생성에 Ghaffari의 영역이론을 이용한다. 본 연구에서는 먼저, 강제적 동기식 도달 가능 그래프를 생성한 후Ghaffari의 영역이론을 이용하기 위하여 정리 1에 해당하는 수식을 산출하여 제어기를 생성한다.

참고문헌 [3]은 금지 동작을 제어하기 위해서 초기 패트리넷에 제어 플레이스를 추가하는 영역이론의 새로운 시도를 제안했다. 이러한 Ghaffari의 연구는 다음의 정리로 요약할 수 있다.

정리 1 [3]: 다음을 만족하는 제어 플레이스 $\{(M_0(p_c), C(p_c,.))\}$ 의 집합 P_c 가 존재하면, 도달 가능 그래프가 $RG(N,M_0)$ 인 패트리넷 (N,M_0) 이 반드시 존재한다:

1. $RG(N,M_0)$ 에서 (N,M_0) 의 플레이스 p 의 연결과 관련하여 (1)을 만족하는 사이클;

$$\sum_{t\in T} C(p,t)\vec{\delta}[t] = 0, \forall \delta \in \Delta \quad (1)$$

$\vec{\delta}$ 는 비방향 사이클 δ 에서 트랜지션 t의 모든 발생 횟수의 합을 나타내고 Δ 는 도달 가능성 그래프 $RG(N,M_0)$ 의 비방향 사이클들의 집합이다. $\vec{\delta}$ 를 δ 의 계수벡터라 부르기도 한다.

2. $RG(N,M_0)$ 에서 마킹과 관련하여 다음의 (2)를 만족하는 도달가능 식;

$$M_0(p) + C(p,.)\vec{\Gamma}_M \geq 0, \forall M \in R \quad (2)$$

$\vec{\Gamma}_M$ 는 Γ_M 의 계수벡터이다. Γ_M 은 초기상태 M_0에서 M까지의 비방향 궤도이고 δ 와 유사하다. R은 초기모델의 도달 가능 그래프를 나타낸다.

3. M 으로부터 점화하지 않는 t 로 구성한 (M,t) 에 대하여, (3)을 만족하는 적어도 하나의 제어 플레이스 p_c 가 존재한다.

$$M(p_c) = M_0(p_c) + C(p_c,.)\vec{\Gamma}_M + C(p_c,t) < 0 \quad (3)$$

2.2 영역이론에 적용할 수식의 자동추출

영역이론에 적용할 수식은 플랜트 모델과 사용자 요구사항에 대한 제어능력을 반영한 모델을 결합한 결과인 동기식 도달 가능 그래프의 반복적용으로 추출할 수 있다. 이 때 다음 두 개의 중요한 규칙을 준수해야 한다.

규칙 1: 사용자 요구사항을 적용한 모델의 트랜지션은 두 개의 패트리넷 모델(플랜트 모델과 사용자 요구사항 모델)에서 동시에 점화 가능 할 때만 점화 가능하다.

규칙 2: 플랜트 모델의 점화 가능한 트랜지션은 만일 이 트랜지션이 사용자 요구사항 모델에도 존재하나 두 개의 모델에서 동시에 동작이 불가능하다면 이 트랜지션의 점화는 금지되어야 한다.

위와 같은 규칙을 바탕으로 영역이론에 적용할 방정식을 추출하기 위해 동기식 도달 가능 그래프를 이용한 알고리즘을 나타내면 다음과 같다.

알고리즘 1: 영역이론에 적용할 수식 추출 알고리즘

The marking M_i is composed of the places of different Petri Nets models.

Inputs: M_0, LT

Outputs: AS, FS, CS

Step 1 $AM \leftarrow \{M_0\}$;

Step 2 while $(AM \neq \varnothing)$ *do,*

Step 2.1 $M_i = next\_marking\_of(AM)$; $AM \leftarrow AM - \{M_i\}$

Step 2.1.1 $T_{Mi} \leftarrow transitions\_from\ (M_i, LT)$; *Step 2.1.2 while* $(T_{Mi} = \varnothing)$ *do,*

Step 2.1.2.1 $MT = next\_successor\_in\ (T_{Mi})$;

$M_i = (t_{ik}, M_k)$; $T_{Mi} \leftarrow T_{Mi} - \{MT\}$;

Step 2.1.2.2 $t_{ik} \leftarrow transition\ (MT)$;

Step 2.1.2.3 $M_k \leftarrow marking\ (MT)$;

*Step 2.1.2.4 (*Applying rule 2*)*

if enabled_in_plant (t_{ik}) *and not (enabled_in_specification* (t_{ik})*) then*

Step 2.1.2.4.1 $\forall \sigma \in AS / M_0[\sigma > M_i$; $FS \leftarrow FS \cup \{\sigma\, t_{ik}\}$;

Step 2.1.2.4.2 $FM \leftarrow FM \cup \{M_k\}$; *end_if;*

else if enabled_in_plant (t_{ik}) *then*

Step 2.1.2.4.3 If not (M_k *in PM) then*

Step2.1.2.4.3.1 $\forall \sigma \in AS / M_0[\sigma > M_i$;

$AS \leftarrow AS \cup \{\sigma\, t_{ik}\}$;

Step 2.1.2.4.3.2 $AM \leftarrow AM \cup \{M_k\}$; *end_if*

else

Step2.1.2.4.3.3 $\forall \sigma' \in AS / M_k[\sigma' > M_i$;

$CS \leftarrow CS \cup \{\sigma'\, t_{ik}\}$;

end_if;

end_if;

end_while;

Step 2.1.3 $PM \leftarrow PM \cup \{M_i\}$;

end_while;

end.

여기서,

M_0는 초기 마킹이며,

AM는 승인된 마킹이고,

PM는 수행된 마킹들의 집합을 의미하며,

AS는 승인된 순서들의 집합을 나타내며,

FS는 금지된 순서들의 집합이며,

CS는 사이클의 집합을 의미한다.

정리 2: 규칙 1을 증거하는 트랜지션 t_{ik} / M_i [$t_{ik} > M_k$ 과 t_{ik} 이 존재할 때, $\forall \sigma \in AS / M_0$ [$\sigma > M_i$ 이고 $\sigma \cdot t_{ik} \in AS$이다.

증명: t_{ik} 가 규칙 1을 증거한다면 이 트랜지션은 플랜트 패트리넷 모델과 사용자 요구사항 패트리넷 모델에서 동시에 점화 가능하다는 것을 의미한다. 순서 $\sigma \in AS/M_0$ [$\sigma > M_i$ 가 존재한다면, M_i 는 AM에 해당된다. 그 결과, M_i [$t_{ik} > M_k$ 에서의 M_k 또한 AM에 해당되며, 순서 $\sigma \cdot t_{ik}$ 는 M_0 로부터 인증된 마킹 M_k 에 도달하게 한다. 이것은 $\sigma \cdot t_{ik}$ 가 AS에 해당됨을 의미한다.

정리 3: 규칙 2를 규명하는 트랜지션 t_{ik} / M_i [$t_{ik} > M_k$ 와 t_{ik} 가 있다면, $\forall \sigma \in AS/M_0$ [$\sigma > M_i$ 이고 $\sigma \cdot t_{ik} \in FS$이다.

증명: t_{ik} 가 규칙 2에 해당된다면 이 트랜지션은 사용자

요구사항 패트리넷 모델에서 점화할 수 없다는 것을 의미한다. 즉 M_i에서 M_k로 점화되는 트랜지션은 금지되며 MK는 FM에 해당된다. $M_k \in$ FM 인 순서 $\sigma \in AS/M_0$ $[\sigma > M_i$가 존재한다면, 순서 $\sigma \cdot t_{ik}$를 허용해서는 안되며 이 순서는 FS에 해당된다.

3. 예제플랜트 모델에의 적용

그림 2(a)는 예제 플랜트 모델이고, 그림 2(b)는 사용자 요구사항에 대한 제어를 반영한 모델이다.

사용자 요구사항을 반영한 모델은 다음과 같은 제약사항을 갖는다.

- 머신 1은 버퍼가 가득 차 있는 경우에는 동작할 수 없다.
- 머신 2는 버퍼가 비어 있는 경우에 동작할 수 없다.

그림 2의 플랜트 모델과 사용자 요구사항 모델을 [5] 에서 제안한 순차상태전이금지문제를 해결하기 위한 합성법 동기식 도달 가능 그래프를 이용하고 Ghaffari의 영역이론을 이용하여 사용자 요구사항에 따른 제어된 모델을 구축하고자 한다. 그림 3은 그림 2(a)와 2(b)를 이용하여 동기식도달가능그래프를 구한 그림이다.

그림 3을 이용하여 도달 가능식과 도달 금지식을 구할 수 있지만, 사이클식은 존재하지 않음을 알 수 있다. 또한, 마킹 M_7에서 트랜지션이 더 이상 점화 할 수 없는 교착상태임을 알 수 있다.

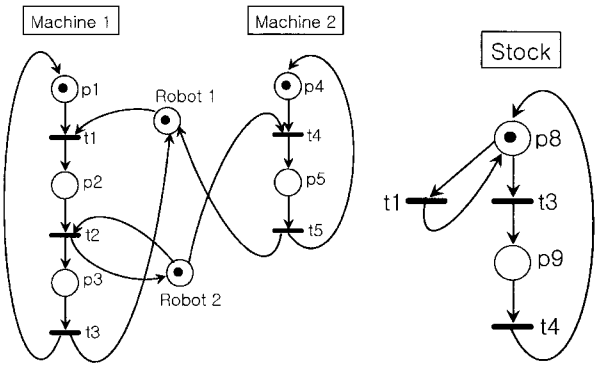
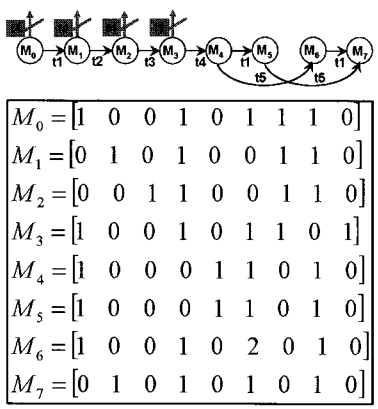


그림 2. 플랜트 모델과 사용자 요구사항 모델.

Fig. 2. Plant model and user specification model.



$$
\begin{aligned}
M_0 &= [1\ 0\ 0\ 1\ 0\ 1\ 1\ 1\ 0] \\
M_1 &= [0\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 0] \\
M_2 &= [0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 0] \\
M_3 &= [1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1] \\
M_4 &= [1\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 0] \\
M_5 &= [1\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 0] \\
M_6 &= [1\ 0\ 0\ 1\ 0\ 2\ 0\ 1\ 0] \\
M_7 &= [0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0]
\end{aligned}
$$

그림 3. 그림 2의 동기식 도달 가능 그래프.

Fig. 3. CSRG of Fig. 2.

이와 같이 플랜트 모델에 교착상태가 있는 경우는 앞에서 제시한 알고리즘을 적용하여 제어모델을 재구성하기가 불가능하다.

그러므로, 본 예에서 사용자가 원하는 대로 제어된 모델을 구축하기 위해서는 먼저 플랜트 모델을 검증할 필요가 있으며, 따라서 본 연구에서는 먼저 교착상태가 발생한 플랜트 모델을 수정하여 교착 상태 문제를 해결한 후 제어모델을 구축하는 방법을 제안하고자 한다.

III. DAPN 을 사용한 플랜트 모델의 재구성

2장에서 살펴본 바와 같이 교착상태와 같은 문제를 내포하고 있는 플랜트 모델에서는 사용자 요구사항에 의한 제어된 모델을 구축할 수 없음을 보았다. 따라서 이 장에서는 플랜트 모델이 보유하고 있는 교착상태와 같은 문제를 DAPN 알고리즘을 이용하여, 플랜트 모델을 재구성함으로써, 사용자 지향의 제어 모델을 에러 없이 구축하고자 한다.

기존의 패트리넷은 플랜트 모델의 특성상 작업을 하는 기계 부분과 부품을 의미하는 자원의 부분이 모두 하나의 플레이스 종류로 기술되어 있으므로 해서 교착상태의 최대의 문제점인 분배와 취득의 부분을 명확히 표현해낼 수 없었다. 따라서 작업 플레이스P의 집합과 자원 공유 플레이스R의 집합으로 분리하여 표현하는 DAPN을 이용하여 플랜트 모델 재구성에 이용하고자 한다[7].

정의 2: DAPN

$DAPN = (P, R, T, I, O, M_0)$

여기서, $P = \{p1, p2, ..., pn\}$:

작업 플레이스들의 집합 $(n \geq 0)$

$R = \{r1, r2, ..., rm\}$:

자원공유 플레이스들의 집합 $(m \geq 0)$

$T = \{t1, t2, ..., tk\}$:

트랜지션들의 집합 $(k \geq 0)$

$P \cap R = \phi \quad P \cap T = \phi \quad R \cap T = \phi$

$\forall p \in P \quad |\bullet p| = 1 \ \ and \ \ |p \bullet| = 1$

$I : P \times T \rightarrow N \quad R \times T \rightarrow N,$

I 는 입력 함수

$O : T \times P \rightarrow N \quad T \times R \rightarrow N,$

O 는 출력 함수

$P \neq 0 \ \ and \ \ R \neq 0 \ \ and \ \ T \neq 0$

$M_0 \in M = \{M | M : P \rightarrow N\},$

M_0 는 초기 마킹

N : 양의 정수들의 집합

정의 3: 교착상태

$DAPN = (P, R, T, I, O, M_0)$ 에서 $Ps \subset \{P \cup R\}$ 인 Ps 가 만약 $\forall tj \in T$ 인 경우, 상태가 다음과 같으면 이를 교착상태라고 한다.

$$\#(Ps, I(tj)) \geq \#(Ps, O(tj))$$

$DAPN$ 모델은 플랜트 모델의 작업에 기반을 둔 작업 플레이스와 자원 공유 플레이스 사이의 모든 관계를 보여주기

위해 행렬로 표현된다. 이것은 작업에 기반을 둔 서브넷들로 분해될 수 있다는 것을 의미한다.

정의 4: 서브넷

$DAPN' = (P', R', T', I', O')$,

$DAPN = (P, R, T, I, O)$ 일 때,

만약, $P' \subseteq P, R' \subseteq R, T' \subseteq T$,

$I' = I \cap (\{P' \cup R'\} \times T')$,

$O' = O \cap (T' \times \{P' \cup R'\})$ 이면,

$DAPN' \subseteq DAPN$ 이고, 이 때, $DAPN'$ 은 $DAPN$ 의 서브넷이라 한다.

정의 5: Incidence Matrix

M_{PR} 은 $DAPN$ 의 incidence matrix이다.

M_{PRi} 는 $DAPN_i$ 의 incidence matrix이다.

$DAPN_i \subseteq DAPN$

$\forall M_{PRi} \subseteq M_{PR}, (1 \le i \le n)$

그러므로, $\bigcup_i M_{PRi} = M_{PR}$

정의 6은 작업에 기반을 둔 작업 플레이스와 자원공유 플레이스 사이의 모든 관계를 보여주는 행렬에 관한 정의이다.

정의 6: Extended Incidence Matrix

M_{PRi}' 은 M_{PRi} 의 extended incidence matrix이다.

$\#(p', M_{PRi}') = Lim$

$Lim = \max(\#(p, M_{PRi}), 1 \le i \le n)$

$p' \in M_{PRi}', p \in M_{PRi}$

또한 $F_{M_{PR1}}^{M_{PRn}'}$ 은 extended incidence matrix $M_{PR1}', M_{PR2}', \ldots\ldots, M_{PRn}'$ 들의 연합이다.

즉, $F_{M_{PR1}}^{M_{PRn}'} = \sum_{i=1}^{n} M_{PRi}'$

여기서, 작업 플레이스들의 수는,

$$\#(p, F_{M_{PR1}}^{M_{PRn}'}) = \max(\#(p, M_{PRi}), (1 \le i \le n))$$

자원공유 플레이스들의 수는,

$$\#(r, F_{M_{PR1}}^{M_{PRn}'}) = \max(\#(r, M_{PRi}), (1 \le i \le n))$$

그림 2(a)의 플랜트 모델을 위의 정의 4, 정의 5와 정의 6에 의해 $F_{M_{PR1}}^{M_{PR2}'}$ 을 구하면 다음 그림 4와 같은 행렬이 된다.

그림 4의 $F_{M_{PR1}}^{M_{PR2}'}$ 행렬을 이용한 단계 표현은 그림 5와 같고, 그림 5의 단계별 로봇의 상태는 표 1과 같다.

여기서 B_step은 각 단계에 해당하는 작업 플레이스가 수행되기 위해 필요한 트랜지션의 점화전(before) 자원의 상태이고 A_step은 각 단계에 해당되는 작업 플레이스가 수행되기 위해 필요한 트랜지션의 점화 후(after) 이루어지는 자원의 상태를 말한다.

각 단계의 자원의 값은 정의 7을 사용하여 계산될 수 있다.

정의 7: 상태 값 (SV : Status Value)

SV_0 : 로봇의 초기 상태

$SV_i = SV_{i-1} - \#(r, B\_step((i+1)/2))$,

$$F_{M_{PR1}}^{M_{PR2}'} = M_{PR1}' + M_{PR2}' =$$

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 2 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \begin{matrix} p1,4 \\ p2,5 \\ p3 \\ robot\ 1 \\ robot\ 2 \end{matrix}$$

그림 4. Extended incidence matrix들의 합 $F_{M_{PR1}}^{M_{PR2}'}$.

Fig. 4. Sum $F_{M_{PR1}}^{M_{PR2}'}$ of extended incidence matrix.

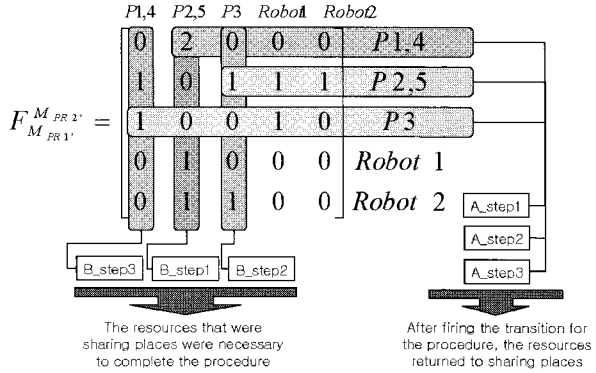


그림 5. $F_{M_{PR1}}^{M_{PR2}'}$ 행렬을 이용한 단계 표현.

Fig. 5. Step expression using $F_{M_{PR1}}^{M_{PR2}'}$ matrix.

표 1. 그림 5의 단계별 로봇의 상태.

Table 1. Robot status for each steps in Fig. 5.

	B_step1	B_step2	B_step3	A_step1	A_step2	A_step3
robot1	1	0	0	0	1	1
robot2	1	1	0	0	1	0

표 2. 그림 5의 상태값 테이블.

Table 2. Status value table of Fig. 5.

	SV0	SV1	SV2	SV3	SV4	SV5	SV6
robot1	1	0	0	0	1	1	2
robot2	1	0	0	-1	0	0	0

$1 \le i \le n$, i is an odd number, r is robot.

$SV_j = SV_{j-1} + \#(r, A\_step((j/2))$

$2 \le j \le n$, j is an even number, r is robot.

표 1을 정의 7의 상태값(SV)으로 나타내면 표 2와 같다.

표 2를 통해 그림 2(a)의 플랜트 모델은 SV3의 음수 값으로 인해 교착상태가 생기게 됨을 알 수 있다. 이러한 교착상태를 회피하기 위해서는 우선, 교착상태를 일으키는 음수 값을 없애야 한다. SV3의 음수 값을 제거하기 위해 SVi의 형태

표 3. 그림 5의 재구성된 상태값 테이블.
Table 3. Reconfigured status table of Fig. 5.

	SV0	SV1	SV2	SV3	SV4	SV5	SV6
robot1	1	0	0	0	1	1	2
robot2	1	0	1	0	1	1	1

표 4. 그림 5의 재구성된 상태값 테이블.
Table 4. Reconfigured status table of Fig. 5.

	SV0	SV1	SV2	SV3	SV4	SV5	SV6
robot1	1	0	0	0	1	1	1
robot2	1	0	1	0	1	1	1

$$F^{M_{PR2'}}_{M_{PR1'}} = \begin{array}{c} \begin{matrix} P1,4 & P2,5 & P3 & Robot1 & Robot2 \end{matrix} \\ \begin{bmatrix} 0 & 2 & 0 & 0 & (0) \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & (1) & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \end{array} \begin{array}{l} P1,4 \\ P2,5 \\ P3 \\ Robot\ 1 \\ Robot\ 2 \end{array}$$

$$F^{M_{PR2'}}_{M_{PR1'}} = \begin{bmatrix} 0 & 2 & 0 & 0 & (1) \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & (0) & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \begin{array}{l} P1,4 \\ P2,5 \\ P3 \\ Robot\ 1 \\ Robot\ 2 \end{array}$$

그림 6. 행렬 $F^{M_{PR2'}}_{M_{PR1'}}$.

Fig. 6. Matrix $F^{M_{PR2'}}_{M_{PR1'}}$.

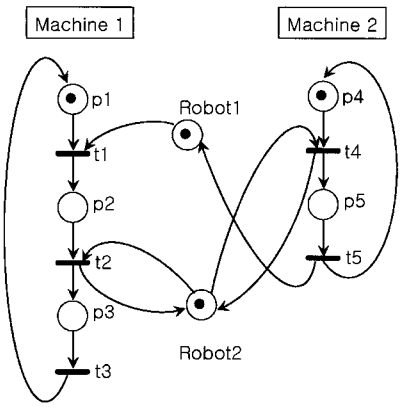


그림 7. DAPN을 사용하여 재구성된 모델.

Fig. 7. Reconfigured model using DAPN.

인 SV2의 자원을 1로 수정하여야 한다. 여기서, SVj는 플랜트 모델의 특성상 제품을 만드는데 필요한 기계나 부품을 의미하는 자원이므로 수정되어서는 안 된다.

앞과 같은 원리를 적용해 수정하면 상태값 테이블은 표 3과 같이 된다.

또한, 계속적인 프로세스 진행을 위한 "boundedness" 속성을 만족하기 위해서는 자원의 초기 상태를 유지해야 하므로, 표 3의 SV6의 robot 1의 값을 1로 수정해야 한다. 이를 적용한 테이블은 표 4와 같다.

위의 상태값 테이블을 이용해 재구성된 $F^{M_{PR2'}}_{M_{PR1'}}$ 와 이를 적용한 플랜트 모델은 그림 6 및 그림 7과 같다.

한편, *DAPN* 과 상태값 테이블을 이용한 교착상태의 탐지와 회피알고리즘은 다음과 같다.

알고리즘 2: 교착상태의 탐지와 회피

```
int main(){
  n=the number of resource sharing places;
  Define value of resource state table, SV, according to the
  definition 6;
  While(exist values of SVi) {
    result=Detection(the values of SVi, n);
    if (result==0){
      i++;
      continue;
    } else
      Avoidance1(the value of SVi, result);
    If (value of initial resource sharing places in table != value of
    final resource sharing places in table){
      Cout << "This net is UNBOUNDEDNESS";
      Avoidance2();
    }
  }
  Detection(the values of SVi, n){
    for(r=0; r<=n; r++)
      if (SVi of the n< 0){
      cout << "This net is deadlock"
      return r;
      }
  }
  Avoidance1(the values of SVi, n){
    #(r, SVi-1)= #(r, SVi-1) - #(r, SVi);
    Compute values of resource state table according to Definition
    6 again;
  }
  Avoidance2() {
    value of final resource sharing places
    = value of initial resource sharing places;
  }
}
```

IV. 재구성된 플랜트 모델과 사용자 요구사항의 통합

먼저 플랜트 모델을 재구성하고 검증하기 위한 패트리넷 제어기를 설계 하기 위해서 다음과 같은 4단계의(그림 8) 설계 단계를 제안한다.

각 단계의 역할은 다음과 같다.

단계 1: DAPN을 이용하여 플랜트 패트리넷 모델을 재구성하고 검증하기.

단계 2: 플랜트 패트리넷 모델과 사용자 요구사항 패트리넷 모델을 정의하기.

단계 3: 재구성된 플랜트 패트리넷 모델과 사용자 요구사항 패트리넷 모델의 병합을 기반으로 도달 가능 식, 사이클 식, 도달 금지식을 계산하기.

단계4: 영역 이론을 이용하여 제어기를 구하고 제어된 시스템 패트리넷 모델 구축하기.

먼저, 단계 1에 의하여 그림 2(a)의 플랜트 모델에 3장의

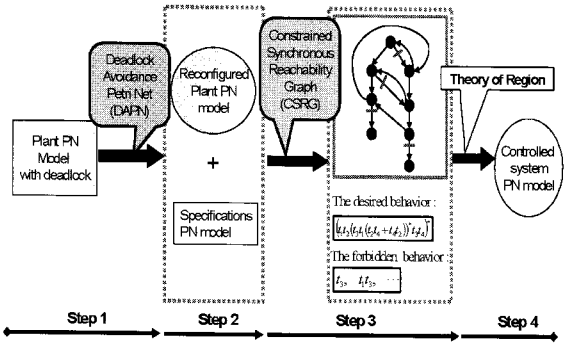
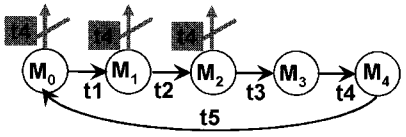


그림 8. 제어기 병합을 위한 4단계.
Fig. 8. 4 steps for synthesize controller.



$$M_0=[1\ 0\ 0\ 1\ 0\ 1\ 1\ 1\ 0]$$
$$M_1=[0\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 0]$$
$$M_2=[0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 0]$$
$$M_3=[1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1]$$
$$M_4=[1\ 0\ 0\ 0\ 1\ 0\ 1\ 1\ 0]$$

그림 9. 그림 7과 그림 2(b)의 동기식도달가능그래프.
Fig. 9. CSRS of Fig. 7 and Fig. 2(b).

DAPN을 적용하여 그림 7에서와 같이 재구성된 플랜트 패트리넷 모델을 생성한다.

단계 2에 의하여 사용자 요구사항 패트리넷 모델인 그림 2(b)를 생성한다.

단계 3에 의하여 [5-6]에서 소개한 순차 상태 전이 금지문제를 해결하기 위한 동기식 도달 가능 그래프 알고리즘을 적용하면 그림 9에 묘사된 동기식 도달 가능 그래프를 구축할 수 있다. 이 때, 이미 DAPN을 사용하여 교착상태를 회피하였으므로 수정된 플랜트 모델에서는 교착상태가 나타나지 않으므로 제어 모델을 재구성할 수 있음을 알 수 있다.

단계 4에 적용하기 위하여 그림 9의도달 가능 식, 사이클 식, 도달 금지식을 생성한다.

도달 가능 식:

$$M_{20}(p_C)\geq 0$$
$$M_{21}(p_C)=M_{20}(p_C)+C(p_C,t_1)\geq 0$$
$$M_{22}(p_C)=M_{20}(p_C)+C(p_C,t_1)+C(p_C,t_2)\geq 0$$
$$M_{23}(p_C)=M_{20}(p_C)+C(p_C,t_1)+C(p_C,t_2)+C(p_C,t_3)\geq 0$$
$$M_{24}(p_C)=M_{20}(p_C)+C(p_C,t_1)+C(p_C,t_2)+C(p_C,t_3)+C(p_C,t_4)\geq 0$$

도달 금지 식:

$$M_{20}(p_{C1})+C(p_{C1},t_4)<0$$
$$M_{20}(p_{C2})+C(p_{C2},t_1)+C(p_{C2},t_4)<0$$
$$M_{20}(p_{C3})+C(p_{C3},t_1)+C(p_{C3},t_2)+C(p_{C3},t_4)<0$$

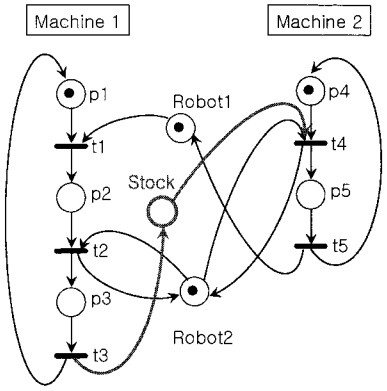
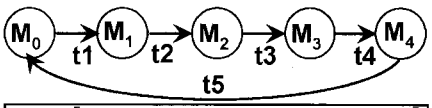


그림 10. 교착상태 없이 사용자 요구사항을 반영한 제어모델.
Fig. 10. Controlled model following specification without deadlock.



$$M_0=[1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0]$$
$$M_1=[0\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 0]$$
$$M_2=[0\ 0\ 1\ 1\ 0\ 0\ 1\ 0\ 1\ 0]$$
$$M_3=[1\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 1]$$
$$M_4=[1\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 1\ 0]$$

그림 11. 재명명된 플레이스를 가진 그림 2(b)와 그림 10의 동기식 도달 가능그래프.
Fig. 11. CSRG of Fig. 2(b) and Fig. 10 with renamed places.

사이클 식:

$$C(p_C,t_1)+C(p_C,t_2)+C(p_C,t_3)+C(p_C,t_4)=0$$

단계 4에 의하여 영역 이론을 적용하여, 위의 수식을 계산하면 $M_{20}(p_{C1})=M_{20}(p_{C2})=M_{20}(p_{C3})=0$ 이고 $C(p_{C1},.)=C(p_{C2},.)=C(p_{C3},.)=(0\ \ 0\ \ 1\ \ -1\ \ 0)$ 인 하나의 제어기에 의하여 세 개의 '도달 금지식'을 모두 금지하는 것이 가능하다는 것을 알 수 있다. 따라서 본 예제 시스템에서 사용자 요구사항에 의한 시스템은 그림 10과 같다.

결과인 그림 10이 사용자 요구사항에 충실한가를 검증하기 위하여, 그림 2(b)의 플레이스 P8은 P9로 P9는 P10으로 다시 이름을 부여하여 사용자 요구사항 모델을 생성하고 그림 10을 플랜트 모델로 하여 그림 11과 같이 동기식 도달 가능 그래프를 구축하였다.

동기식 도달 가능 그래프를 생성한 결과 금지된 순서가 없음을 알 수 있다. 따라서 합성 결과는 사용자 요구사항에 충실하게 제어된 모델임을 알 수 있다.

V. 결론

이 연구에서 우리는 기계, 이동로봇, 부품과 같은 자원을 이용해 다양한 품종을 생산할 수 있는 시스템인 유연생산 시스템에서의 제어기를 합성하기 위한 연구방안을 제시하였으며, 또한 교착상태를 내포하고 있는 플랜트 모델은 사용자 요구사항에 충실한 제어기를 모델링 하기가 불가능한 단점

을 극복하고자 DAPN을 적용하였다. DAPN을 적용하여 먼저 플랜트 모델의 문제점을 제거한 뒤, 제한된 공간으로 플랜트 모델의 동작을 한정하는 사용자 요구사항을 적용한 제어기를 합성하여 최종 제어 모델을 생성하였다.

이 연구를 통해 우리는 기존 연구들에 비해 다음과 같은 장점을 발견할 수 있었다.

1. Ramadge와 Wonham은 유한 오토마타를 기반으로 감독 제어 문제를 연구했으나 유한 오토마타를 기반으로 한 연구는 일반적으로 복잡한 시스템을 모델링 할 때 발생하는 결합 폭발성에 의한 제한성이 있어 병렬처리 및 동기화, 자원공유를 모델링하기 어렵다는 단점이 있다. 반면, 페트리넷은 병렬처리나 동기식 처리가 가능하다. 따라서, 본 연구에서는 오토마타 대신에 병렬처리나 동기화에 대한 모델링을 하는데 있어 결합 폭발성을 줄이는 합성방법이나 분할 방법들이 가능한 패트리넷을 이용함으로써 감독 제어 설계에 보다 더 효율적인 접근을 시도하였다.

2. 자원공유 및 병렬상태를 가진 모델링의 가장 큰 문제점은 교착상태이다. 그러나 이 연구에서는 교착상태의 내포로 인한 제어기 생성의 어려움을 DAPN 알고리즘 적용으로 교착상태를 제거시킴으로써 이를 극복하였다. 특히 교착상태가 발생된 모델링에서는 사용자 요구사항이 충족되게 모델링에 함축 될 수가 없는 점도 예제를 통하여 보였다.

이제 이 연구의 추후 과제로는 대규모 시스템에 적용하기 위한 모듈 형식 응용 방안을 마련하는 것이다.

참고문헌

[1] P. J. Ramadge and W. M. Wonham, " Supervisory control of a class of discrete event processes," *SIAM Journal on Control and Optimization*, vol. 25, no. 1, January 1987.

[2] P. J. Ramadge and W. M. Wonham, "Supervisory control of a class of discrete event processes," *SIAM J. Control and Optimization*, vol. 25, no. 1, January 1987.

[3] A. Ghaffari, N. Rezg, and X. Xie, "Algebraic and geometric characterization of petri net controllers using the theory of regions," *Proceeding of the 2002 Sixth International Workshop on Discrete Event Systems-WODES'02*, reims, France, 2002.

[4] A. Ghaffari, N. Rezg, and X. Xie, "Live and maximally permissive controller synthesis using theory of regions," *in Synthesis and Control of Discrete Event Systems, B. Caillaud et al. (eds). Kluwer Academic Publishers*, pp. 155-166, 2002.

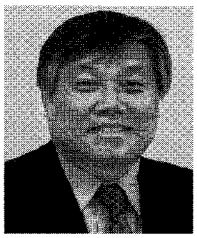
[5] E. J. Lee, A. Toguyéni, and N. Dangoumau, "Petri net based controllers for forbidden sequences of state-transitions in the control of flexible manufacturing systems," *Conceptual Modeling and Simulation Conference-CMS 2005,* Marseilles, France, October 20-22, 2005.

[6] E. J. Lee, A. Toguyéni, and N. Dangoumau, "A petri net based decentralized synthesis approach for the control of flexible manufacturing systems," *4th MultiConference on Computational Engineering in Systems Applications-CESA 2006,* Beijing, China, October 4-6, 2006.

[7] Y. J. Song and J. K. Lee, "The study on the deadlock avoidance using the DAPN and the adjacency matrix," *Journal of the Korea Society for Simulation*, vol. 15, no. 1, March 2006.

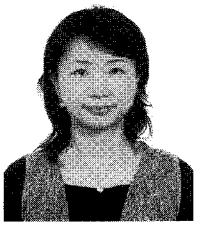
송 유 진

1969년 6월 10일생. 1992년 창원대학교 전자계산학과(이학사). 1995년 창원대학교 컴퓨터공학과(공학석사). 2003년 창원대학교 컴퓨터공학과(공학박사). 2003년~현재 창원대학교 초빙교수. 관심분야는 패트리넷, 정보 보안, 임베디드시스템.



이 종 근

1952년 4월 28일생. 1974년 숭실대학교 전자계산학과 및 동 대학원. 1978년 공학석사 고려대학교 경영대학원. 경영학석사. 1987년~1990년 LSI/Univ. de Montpellier II 연구원 역임. 2002년 LCGI /Ecole Centrale Paris 컴퓨터공학(공학박사). 1983년~현재 창원대학교 컴퓨터공학과 교수. 관심분야는 패트리넷, FMS스케쥴링 분석, 성능분석, 정보보호 관련 연구.



이 은 주

1970년 7월 15일생. 1993년 국립 창원대학교 전자계산학과(이학사). 1999년 국립창원대학교 전자계산학과(이학석사). 2007년 프랑스릴 중앙공대 산업정보 자동화학과(공학박사). 현재 한양대학교 산업공학과 POST-DOC. 관심분야는 Supervisor Control, Petri Net, DES, FMS control, Automatic control, Scheduling.