'91 韓國自動制御學術會議論文集 1991. 10.22~24

최적 제어를 위한 Supervisory Control

박 홍 성 *김 면 집 노 갑 선 권 욱 현
서울대학교 공과대학 제어계측공학과

Optimal Supervisory Control

Hong Seong Park, Myeon Jib Kim, Gab Sun No, Wook Hyun Kwon
Dept. of Control & Instrumentation Eng.
Seoul National University

ABSTRACT

This paper presents a framework of a optimal supervisory controller, which consists of decision rules and the supervisory controller proposed by Ramadge and Wonham. From the presented framework, we obtain optimal control patterns minimizing the given cost functions. The properties of the presneted optimal supervisory controller are discussed. Two examples are given to illustrate a designing method of the optimal supervisory controller.

1. 서 론

이산 사건 시스템 (discrete event system)에 대해 제안된 여러가지 모델링 방법과 수식화 방법 중에서 유한 상태 시스템 (finite state machine) 을 확장시킨 관리 제어 (supervisory control)는 Ramadge와 Wonham에 의해서 처음 제안되었다[1].

이 관리 제어 방법은 이산 사건 프로세스를 제어하기 위한 간단한 추상적인 모델로 콘트롤러의 존재 여부와 그 구성등에 대한 정성적인 구조적 특징을 다루는 것을 목적으로 하여 연구되고 있다.

관리 제어에서는 전체 시스템을 연속 변수 시스템에서와 같이 플랜트(plant)와 콘트롤러(controller)로 나누어 생각하게 되는데 각각이 하나의 오토마톤으로 모델링되어 형식적인 언어 (formal language)의 발생기 (generator)와 인식기 (recognizer)로 동작을 하게 된다.

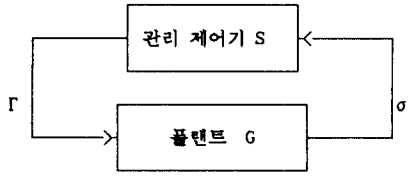


그림 1.1 관리제어를 위한 이산 사건 시스템

제어되는 시스템 즉 플랜트는 형식적인 언어의 발생기로서 다음의 5-쌍 오토마톤 G로 모델링된다.

$$G = (Q, \Sigma, \delta, q_0, Q_m),$$

여기서 Q는 상태(state)의 집합,

Σ는 입력 알파벳(alphabet), 혹은 이산 사건

δ는 전이 관계(transition relation),

q_0는 초기 상태,

Q_m은 목적 상태(marker states)로 불리우는 상태 집합의 부분 집합들을 나타내고 있다.

따라서 오토마톤 G는 초기 상태 q_0에서 시작하여 상태 전이 δ에 따라 사건들의 시이퀀스를 비동기적, 비결정적으로 발생시키는 동적 시스템(dynamic system)으로 해석될 수 있다.

Σ의 원소들로 구성된 유한 시트링(string)의 집합을 Σ^*로 나타내는데 이 Σ^*의 부분집합을 언어(language)라고 한다.

$$\Sigma^* = \{s: s= \sigma_1\sigma_2\sigma_3 \ldots\ldots, \sigma_i \in \Sigma,\ i=1,2,3,4,\ldots\ldots \}.$$

이 관리 제어 방법은 이산 사건 프로세스를 제어하기 위한 간단한 추상적인 모델로 콘트롤로의 존재 여부와 그 구성등에 대한 구조적 특징을 다루는 것을 목적으로 하여 연구되고 있다[1][2].

이 밖에도 관리제어에 대한 여러가지 정성적인 이론들이 있는데 관리제어는 기본적으로 플랜트와 제어기를 분리하여 생각하는 모델이므로 연속 변수 시스템에서 생각하는 제반 성질들을 많이 이용하고 있다.

실제 상황하에서는 관리제어기가 플랜트에서 일어나는 모든 사건들을 관찰할 수가 없을 경우가 생기므로 G와 S 사이에 관찰 단계(observation stage)를 넣어 해석하는 방법이 제시되었다[3]. 이 문제와 관련하여 플랜트 사건 발생을 관찰할 수 있는지를 판별하여 거기에 따르는 관리제어기를 해석하는 연구도 되어 있다[4].

연속 변수 시스템에서 볼 수 있듯이 이산 사건 시스템에서도 분산제어(decentralized control) 개념을 도입하였고[5][6], 관리제어기의 상태수를 줄여서 효과적인 관리제어기를 설계하는 방법[7]도 제시되었다.

관리 제어기를 사용하여 DES의 전체적인 동작을 최

적화 시키는 연구는 거의 없다[8]. [8]에서는 최단 경로를 찾는 시스템을 최적화 시킨것으로 제안된 제어기의 구조가 일반적이지 못하다.

최적 제어 문제가 실제 환경에서 많이 일어나더라도 고전적인 입장에서 DES를 위한 최적 제어 문제의 수식화는 인자들의 미분화된 특성때문에 매우 어렵다. 실제의 플랜트 모델에서는 출발상태에서 어떤 목적 상태까지 갈 수 있는 스트링이 여러개 있는 경우가 대부분이다. 이런 경우에서 여러 스트링중 한 스트링을 선택하기 위해서는 어떤 결정기(decision maker)가 필요하게 되는데 이것이 주어진 성능 지표를 최적화 시키는 스트링을 선택하게 된다.

이 논문에서는 일반적인 환경에서 적용할 수 있는 최적 관리 제어기에 대한 구조를 제안하고 최적 언어(optimal solution)를 가지는 최적 관리 제어기의 존재 여부와 그 성질을 제시한다. 또 그 설계 방법을 예를 통하여 제시한다.

제 2장에서는 관리 제어기를 요약하고 제 3장에서는 최적 관리 제어기의 구조를 제시하여 최적 관리 제어기의 존재 여부에 대해 알아 본다. 제 4장에서는 2개의 예가 제시되어 이 예를 통해서 최적 제어기의 설계 방법을 알아보며 제 5장에서는 결론을 맺는다.

2. 이산 사건 시스템의 관리 제어

플랜트 G에 제어 방법을 접목시키기 위해서 플랜트에서 발생되는 사건들을 제어 가능한 사건(controlled event) Σ_c와 제어 불가능한사건(uncontrolled event) Σ_u로 나눈다. 여기에 관리 제어기에서 만들어지는 제어 형태(control pattern)라는 함수

$$\Gamma : \Sigma \to \{0,1\}$$

을 다음과 같이 정의 한다.

"사건 σ는 Γ에 의해 $\Gamma(\sigma) = 1$ 이면 발생 가능해 지고 $\Gamma(\sigma) = 0$ 이면 그 발생이 불가능해진다."

플랜트 모델 G에 위의 제어 방법을 부가하여 확장된 전이 함수를 정의하면 다음과 같다.

$$\delta_c : \Gamma \times \Sigma \times Q \to Q$$

여기서

$$\delta_c(\Gamma,\sigma,q) = \begin{cases} \delta(\sigma,q) & \text{if } \delta(\sigma,q) \text{ is defined and } \Gamma(\sigma)=1 \\ \text{undefined} & \text{otherwise.} \end{cases}$$

위의 G_c 모델은 외부에서 원하지 않는 사건의 발생을 억제하도록 제어패턴의 시이퀀스를 스위칭해 줌으로써 우리가 원하는 일만을 수행할 수 있는 제어된 이산 사건 프로세스(controlled discrete event process)로 해석될 수 있다.

따라서 우리는 원하는 동작을 하도록 제어 패턴을 스위칭해줄 수 있는 제어기의 설계가 필요하게 되는데 이러한 제어기를 관리 제어기(supervisory controller)라 한다[1]. 관리제어기는 형식적으로 (S,Φ)의 쌍으로 되어있는데 여기서 S는

$$S = (X, \Sigma, Q, x_0, X_m)$$

의 결정적 오토마톤이고

$$\Phi : X \to \Gamma$$

은 관리제어기 상태 X에서 제어 패턴 Γ로의 매핑 함수이다. 많은 경우 X_m=X이고 언제나 억세서블하다고 가정한다.

이 관리제어기 S는 입력 스트링 $w \in \Sigma^*$에 대응하여 상태 전이를 수행하는 디바이스로 해석될 수 있는데 G_c와 되먹임 루프(feedback loop)를 형성하여 G_c의 사건에 의해서 S의 상태가 변하고 이 상태와 함수 Φ로결정되는 연속적인 제어 패턴을 G_c에 가해줌으로써 G_c를 제어하는 역할을 한다.

결국 플랜트 G_c와 제어기 S가 결합된 발생기, S/G_c는

$$S/G_c = Ac\ (X\times Q,\ \Sigma,\ Q\times\delta c,\ (x_0,q_0),\ X_m\times Q_m)$$

이 된다. 형식적으로 위의 모델에 대한 전이 함수를 정의하게 된다.

$$Q \times \delta_c : \Sigma \times X \times Q \to X \times Q$$

여기서

$$(\sigma,X,q) \to (Q(\sigma,X),\ \delta_c(\Phi(X),\sigma,q)).$$

여기서도 형식 언어 $L(S/G_c)$로 결합된 시스템의 동작 상태를 표시하게 되는데 이 언어는 S와 G_c가 결합되어 발생되는 사건들의 모든 가능한 유한 시이퀀스로 해석될 수 있겠다.

3. 최적 관리 제어기

최적관리제어기의 구조를 설명하기 전에 이산 사건 시스템에서 일어날 수 있는 문제에 대해 알아보자. 한 사건의 발생 (q_p,q_d)은 비용 $C(q_p,q_d)$를 가질 수 있으며, 이 때 $q_d=\delta(s,q_p)$이다. 따라서, 비용함수 $C(q_p,q_d)$를 다음과 같이 정의하자.

정의 3.1 (비용함수) 만약 $s_i = \sigma_1\sigma_2 \ldots \sigma_n$이라 하면,

$$C(q_p,q_d)$$
$$= C(q_1,\delta(s_1,q_1))$$
$$= f(\sigma_1,q_p) + f(\sigma_2,\delta(\sigma_1,q_p)) + \ldots + f(\sigma_n,\delta(\sigma_{n-1}\ldots\sigma_1,q_p)).$$

여기서 $f(\sigma_i,q_i)$는 $\delta(\sigma_i,q_i)$의 전이에 의해 생기는 비용이다.

다음의 제안 1은 비용함수 C가 well-defined되었음을 나타낸다.

제안 1)

플랜트 모델 G가 트림(trim)이고 $f(\sigma_i,q_j) < \infty$, $q_d \in Q_m$ 이고 $q_d = \delta(s,q_p)$인 스트링 s에 무한 루프가 존재하지 않으면 C는 well-defined 되어 있다.

증명) G가 억세서블하므로 $\delta(s_1,q_0) = q_d$와 $\delta(s_2,q_0) = q_p$인 스트링 s_1과 s_2가 존재한다. G가 코억세서블이므로 $\delta(s,q_p) = q_d$가 정의되는 finite event sequence인 $s \in L(Tr$

G)가 존재한다.

$s = \sigma_1 \ldots \sigma_m$라 하면 $f(\sigma_i, q_j)$는 유한하고 s는 finite event sequence이므로 $C(q_p, q_d) < \infty$ 이다.

Q.E.D

실제 문제를 생각해 보면 sales traverse와 같은 문제에서 생겨나는데, 목적 상태 q_2 로 가는 사건들 중 최소 비용을 얻는 사건을 선택한다. 또 분산 제어 시스템에서 스케쥴링과 같은 문제에서 생겨난다. 여기서, 한 상태 q_1에서 여러 전이 $\delta(\sigma_i, q_1)$이 일어날 수 있는데, 이러한 전이중 최소 비용을 갖는 전이를 선택하는 문제이다. 이러한 2가지 유형의 문제에 대하여 다음과 같은 성능 지표를 생각할 수 있다.

전자의 경우에는

$$J = \min_{\text{all } s_i} C(q_1, q_2) \text{ where } q_2 = \delta(s_i, q_1)$$

후자의 경우에는

$$J = \min_{\text{all } q_2} C(q_1, q_2) \text{ where } q_2 = \delta(\sigma_i, q_1)$$

들을 생각할 수 있다.

지금까지 이산 사건 시스템에서 사용할 수 있는 비용함수에 대해 살펴 보았다. 비용함수와 기존의 관리제어기가 결합한 구조를 제시하고 최적 관리제어기의 존재여부와 그에 대한 성질을 살펴보자. 먼저 최적 관리제어기의 구조는 그림 3.1에 제시되어 있으며 이 구조의 특성은 기존의 관리제어기의 성질을 그대로 유지하면서 최적제어를 한다는 것이다.

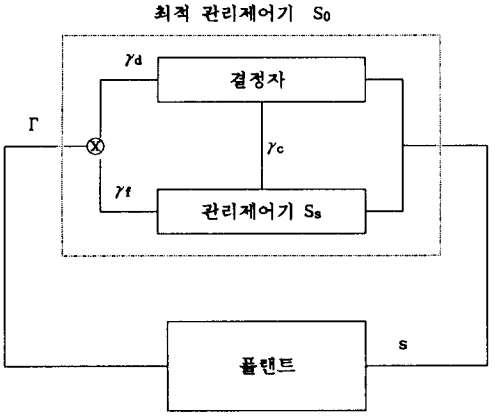


그림 3.1 최적 관리제어기의 구조

이 구조에서 관리제어기 S_s는 기존의 관리제어기이며, 일반적으로 관리제어기 S_s에서 출력되는 제어 형태는 비용과 무관한 것도 존재할 수 있으므로, 이러한 사건과 관련된 제어 형태를 제거한 것이 γ_c이며, 비용과 무관한 제어 형태를 γ_f라 한다. 즉, γ_c, γ_f는 다음과 같다.

γ_c = { 비용에 관련된 사건들과 관련된 제어 형태 }

γ_f = { 비용과 무관한 사건들과 관련된 제어 형태 }

γ_c와 γ_f의 합집합이 기존의 관리제어기에서 출력되는 제어 형태 γ_s이다. 즉,

$$\gamma_s = \gamma_c \cup \gamma_f$$

또, 그림 3.1에서 결정자(decision rule)의 기능을 관리제어기로부터 출력되는 γ_c와 플랜트로부터 들어오는 스트링 s를 사용한 상태에서 최적의 상태로 가도록 γ_c들 중 알맞은 제어 형태 γ_d를 선택하여 출력하게 한다. 따라서 최적 관리제어기에서 출력되는 제어 형태 γ는

$$\gamma = \gamma_d \cup \gamma_f$$

가 된다.

기존의 관리제어기가 존재하는 경우에 최적 관리제어기의 존재 여부는 제안 2에 나타나 있다.

제안 2)

G가 트림이고 $L(S_s/G) = L$인 완전 관리제어기 S_s가 존재하고 $L^* \subset L \subset L(G)$이라하자. 여기서 L^*는 최적 언어(optimal language)이다. 제안 1이 성립하는 비용함수 C가 존재한다고 가정하자. $L(S/G) = L^*$인 완전 관리제어기 S가 존재할 필요 충분조건은 L^* 이 닫혀있고(closed) $(\Sigma_u, L(G))$-invariant이다.

증명) (=>) 비용함수 C가 존재하기때문에 최적 언어 L^*를 선택할 수 있다. $L(S/G)$가 항상 닫혀있기 때문에 L^*는 닫혀있다. 결정자는 Σ_c에 대한 사건만 제어하고 $(\Sigma_u, L(G))$-invariant하고 $L^* \subset L$ 이기 때문에 L^*은 $(\Sigma_u, L(G))$-invariant하다.

(<=) $L^* \subset L$이고 S_s가 완전 관리제어기이기 때문에 이에 대한 것은 당연하다.

Q.E.D

4. 최적 관리제어기의 설계 예

다음 장에서는 제안 2를 사용하여 최적 관리제어기의 설계에 대한 예를 알아 보겠다. 제어 대상인 플랜트로써 그림 4.1과 같이 스테이션의 수가 2이고 버퍼의 수가 2인 분산 처리 시스템을 생각해보자.

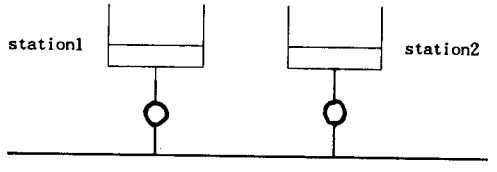
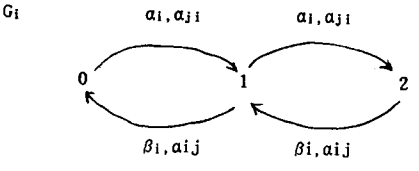


그림 4.1 간단한 분산 처리 시스템

이 시스템에서 각 스테이션의 상태를 나타내는 오토마톤은 그림 4.2와 같다. 여기서 상태를 나타내는 숫자는 그 스테이션에 있는 일의 수를 표시하고 있다.



단. α_i:i번째 프로세서에 일이 들어오는 경우
β_i:i번째 프로세서에서 일을 처리하는 경우
α_{ij}:i번째 프로세서에 일이 j번째 프로세서로 일을 넘기는 경우
α_{ji}:j번째 프로세서에 일이 i번째 프로세서로 일을넘기는 경우

그림 4.2 분산 처리 시스템에서 각 스테이션의 상태도

전체 시스템은 G_1, G_2를 SUFFLE PRODUCT한 것으로 볼 수 있다. 제어 가능한 이벤트의 집합을 Σ_c, 제어 불가능한 이벤트의 집합을 Σ_u 로 나타내면

$$\Sigma_c = \{\alpha_1, \alpha_2, \alpha_{12}, \alpha_{21}, \beta_1, \beta_2\}$$
$$\Sigma_u = \phi$$

이다.

제어 불가능한 이벤트가 없기 때문에 (Σ_u, L(G))-invariant하다. 또한, L이 closed이므로 L(S/G)=L인 complete supervisor가 존재한다. 완전 관측을 하고 있기 때문에 L은 (M, Σ_c, L(G))-controllable하다. 따라서 L은 realizable하다. 완성된 관리제어기 S_s의 오토마톤은 그림 4.3과 같다. 각 상태는 각 스테이션에 있는 일의 수로서 표시되고 있다. 즉, 상태 (1,2)는 스테이션 1에는 1개의 일이 있고 스테이션 2에는 2개의 일이 있는 상태를 나타낸다. 맵핑 함수는 다음과 같다. 여기서 나오는 출력을 γ_d라 하자.

$$\gamma_s = \{ \gamma_i,\ i=0, \ldots, 8 \}$$

$$\phi(z_i) = \begin{cases} \gamma_1, & i=0,1,3, \\ \gamma_2, & i=2 \\ \gamma_3, & i=5 \\ \gamma_4, & i=6 \\ \gamma_5, & i=7 \\ \gamma_6, & i=8 \end{cases}$$

단, $\gamma_1=\{\alpha_1, \alpha_2, \alpha_{12}, \alpha_{21}, \beta_2, \beta_1\}$
$\gamma_2=\{\alpha_1, \alpha_2, \alpha_{12}, \beta_2\}$
$\gamma_3=\{\alpha_1, \alpha_{21}, \beta_1, \beta_2\}$
$\gamma_4=\{\alpha_2, \alpha_{12}, \beta_1\}$
$\gamma_5=\{\alpha_2, \alpha_{12}, \beta_1, \beta_2\}$
$\gamma_6=\{\beta_1, \beta_2\}$

스테이션 i의 일의 갯수가 1만큼 증가하여 (m,n) 상태로 되었을 때 s라는 일을 처리하는데 드는 비용을 C(s,mn)라 하자. 그러면

$$C(s,mn) = \begin{cases} p(s)+I(s,mn)+W(s,mn) & : s가\ \alpha_1, \alpha_2가\ 아닐\ 때 \\ \infty & : s가\ \alpha_1, \alpha_2일\ 때 \end{cases}$$

이 된다.
여기서 p(s)는 스테이션 i에서 s를 처리하는 데 드는 비용을 말하고, I(s,mn)은 다른 스테이션으로 일을 넘겨줄 때에 드는 비용을 말한다. W(s,mn)는 그 스테이션이 다른 일을 수행하고 있을 때에 기다리는 데 드는 비용을 말한다.

$$I(s,mn) = \begin{cases} c & : s가\ \alpha_{12}, \alpha_{21}일\ 때 \\ 0 & : 그\ 밖의\ 경우 \end{cases}$$

$$W(s,mn) = \begin{cases} w & : 한\ 스테이션에\ 일의\ 수가\ 2개일\ 때 \\ 0 & : 기타 \end{cases}$$

를 나타낸다. 여기서 I(s,mn), W(s,mn)는 일의 종류에 관계없이 일정하다고 가정하였다.

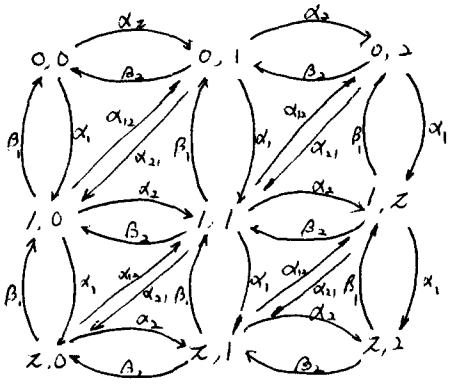


그림 4.3 완성된 관리 제어기의 오토마톤

그림 4.1에서

$$\gamma_c = \gamma_s - \{ \alpha_1, \alpha_2 \}$$
$$\gamma_d = \{ \sigma : \mathrm{Min}\ [C(s,mn)\ ] \}$$
$$\sigma \in \gamma_c$$
$$\gamma_f = \{ \alpha_1, \alpha_2 \} \cap \gamma_c$$

으로 나타난다. 그러면 전체 최적 관리제어기의 출력은

$$\gamma = \gamma_f \cap \gamma_d \quad 이\ 된다.$$

이러한 방법으로 각 상태에서 최소 비용이 드는 제어를 할 수 있도록 최적 관리제어기를 구성할 수 있다.

다음은 sales traverse 시스템을 모델링 한 예로 그림4.4와 같이 동작한다.

이때 원하는 상태 Q_{33}로 도달하면서 비용을 최소화 시키는 최적 관리 제어기를 설계하기 위해 상태 i,j에서 σ 사건으로 발생되는 비용함수 f(σ,(i,j))를 다음과 같다고 가정하자.

f(a1,(0,0)) = 2, f(a2,(0,0)) = 3, f(b1,(1,0)) = 3,
f(b2,(0,1)) = 4, f(b1,(1,1)) = 4, f(b2,(1,1)) = 7,
f(a2,(1,0)) = 7, f(a1,(0,1)) = 3, f(c1,(2,0)) = 6,
f(c2,(0,2)) = 6, f(a2,(2,0)) = 3, f(a1,(0,2)) = 6,
f(c1,(2,1)) = 6, f(c2,(1,2)) = 6, f(b2,(2,1)) = 8,
f(b1,(1,2)) = 5, f(c1,(2,2)) = 7, f(c2,(2,2)) = 9,
f(a2,(3,0)) = 5, f(a1,(0,3)) = 4, f(b2,(3,1)) = 5,
f(b1,(1,3)) = 4, f(c2,(3,2)) = 5, f(c1,(2,3)) = 4

여기서 $Q_{i,j}$에서 마지막 상태 Q_{33}까지 도착하는데

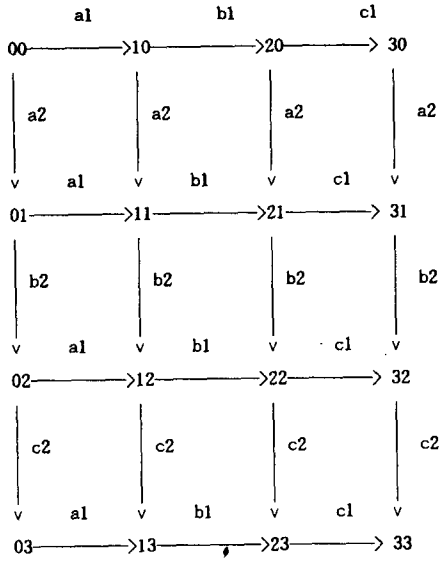


그림 4.4 sales traverse 시스템

드는 비용을 W(i,j)라 하면 최적 관리 제어기를 설계하기 위해서는 W(i,j)가 최소의 값을 갖도록 하여야 한다. 이 최소값을 J(i,j)라 하면

$$J(i,j) = \min W(i,j)$$
$$= \min [\ f(\sigma,i,j)) + W(i+1,j),$$
$$f(\sigma',(i,j) + W(i,j+1)\ ]$$

가 된다.

예제의 비용을 계산해 보면 다음과 같다.

$J(3,2) = f(c2,(3,2)) = 5$ $\gamma_{d32} = c2$

$J(2,3) = f(c1,(2,3)) = 4$ $\gamma_{d23} = c1$

$J(3,1) = f(b2,(3,1)) + J(3,2) = 10$ $\gamma_{d31} = b2$

$J(1,3) = f(b1,(1,3)) + J(2,3) = 8$ $\gamma_{d13} = b1$

$J(3,0) = f(a2,(3,0)) + J(3,1) = 15$ $\gamma_{d30} = a2$

$J(0,3) = f(a1,(0,3)) + J(1,3) = 12$ $\gamma_{d03} = a1$

$J(2,2) = \min [\ f(c1,(2,2)) + J(3,2),$
$f(c2,(2,2)) + J(2,3)\ ] = 12$ $\gamma_{d22} = c1$

$J(2,1) = \min [\ f(c1,(2,1)) + J(3,1),$
$f(b2,(2,1)) + J(2,2)\ ] = 16$ $\gamma_{d21} = c1$

$J(1,2) = \min [\ f(b1,(1,2)) + J(2,2),$
$f(c2,(1,2)) + J(1,3)\ ] = 14$ $\gamma_{d12} = c2$

$J(2,0) = \min [\ f(c1,(2,0)) + J(3,0),$
$f(a2,(2,0)) + J(2,1)\ ] = 19$ $\gamma_{d20} = c1$

$J(0,2) = \min [\ f(c2,(0,2)) + J(0,3),$
$f(a1,(0,2)) + J(1,2)\ ] = 18$ $\gamma_{d02} = c2$

$J(1,1) = \min [\ f(b1,(1,1)) + J(2,1),$
$f(b2,(1,1)) + J(1,2)\ ] = 20$ $\gamma_{d11} = b1$

$J(1,0) = \min [\ f(b1,(1,0)) + J(2,0),$
$f(a2,(1,0)) + J(1,1)\ ] = 22$ $\gamma_{d10} = b1$

$J(0,1) = \min [\ f(b2,(0,1)) + J(0,2),$
$f(a1,(0,1)) + J(1,1)\ ] = 22$ $\gamma_{d01} = b2$

$J(0,0) = \min [\ f(a1,(0,0)) + J(1,0),$
$f(a2,(0,0)) + J(0,1)\ ] = 24$ $\gamma_{d00} = a1$

여기서 $\gamma_f = \emptyset$이기 때문에 $\gamma = \gamma_d$가 되어서 결국 최적 스케쥴(optimal schedule)의 제어 형태 γ은

γ = a1 b1 a2 c1 b2 c2

이 된다.

5. 결론

이 논문에서는 최적 관리제어기의 구조를 제안하였고 그 구조에 대한 성질을 논의하였다.

실제 환경에서는 많은 최적 제어 문제가 있었지만 이산 사건 시스템에서는 알맞은 구조가 없었다. 이 논문에서 제안한 최적 관리제어의 구조를 사용하면 이산 사건 시스템의 최적 제어 문제를 쉽게 해결할 수 있다. 제안한 최적 관리제어기는 결정자와 Ramadge와 Wonham에 의해 제안된 관리제어기 S_S를 사용한다. 그러므로 기존의 관리제어기의 성질을 그대로 유지하면서 최적해 즉, 최적 언어를 얻을 수 있다. 최적 관리제어기가 존재하는 조건을 제시하였고, 예를 통하여 이를 알아보았다. 또, 예를 통하여 최적 관리제어기의 설계 방법을 논의 하였다.

최적 관리제어기를 관측성, 분산제어등에 응용하는 문제, 즉 기존의 관리제어기의 성질 혹은 구조와 연결하는 문제가 남아 있다.

참고 문헌

[1] W. M. Wonham and P.J. Ramadge, "Supervisor control of a class of discrete event processes," *SIAM J. Contr. Optimiz.*, vol.25, no.1, Jan. 1987.

[2] W. M. Wonham and P. J. Ramadge, "On the supremal controllable sublanguage of a given language," *SIAM J. Contr. Optimiz.*, vol.25, no. 5, May 1987.

[3] R. Cieslak, C. Desclaux, A. S. Fawaz and Pravin Varaiya, "Supervisory control of dicrete event processes with partial observations," *IEEE Trans. Automat. Control* AC-33, no.3, Mar. 1988.

[4] F. Lin and W. M. Wonham, "On observability of disrete event systems," *Information Science*, vol.44, 1988.

[5] F. Lin and W.M. Wonham, "Decentralized supervisory control of discrete event systems," *Informatiom Science*, vol.44, 1988.

[6] P. J. Ramadge and W. M. Wonham, "Modular feedback logic for discrete event systems," *SIAM J. Contr.*

Optimiz. vol.25, no.5, Sep. 1987.

[7] A. F. Vaz and W. M. Wonham, "On supervisory reduction in discrete event systems," *Int. J. of Control*, vol.44, no.2, 1986.

[8] A. M. J. Skulimowski, "Optimal Control of Asynchronous Discrete Event Systems," pp243-249