

FMS를 위한 Conveyor System의 이산구조 모델링

신 옥 근[†]

요 약

본 논문에서는 자동화된 생산 시스템과 조립 시스템에서 반제품을 생산기기를 사이로 운반하기 위해 가장 많이 이용되고 있는 컨베이어 (conveyor) 시스템의 이산구조 모델을 제안한다. 조립라인과 같이 조립 기기를 사이에 반제품을 운반하는 데 필요한 시간이 단위공정을 수행하는 데 걸리는 시간보다 클 경우, 조립기기의 배치와 컨베이어로 연결되는 조립기들 사이의 거리는 전체 조립시스템의 성능에 중요한 영향을 미치게 되므로 컨베이어 시스템의 특성을 정량적으로 분석하기 위한 모델이 필요하다. 본 논문에서 제안하는 모델은 팔레트 (pallet)를 이용하는 컨베이어 시스템에서 각각의 컨베이어의 길이가 팔레트 길이의 정수 배이며 컨베이어의 속도가 일정하다는 가정을 전제로 한다. 이런 가정 하에서 컨베이어 운동의 관측 시간과 컨베이어의 길이는 양자화 될 수 있으므로 각 샘플링 시각에서의 컨베이어 시스템의 상태는 양자화된 컨베이어상의 팔레트 유무와 이 팔레트의 이동도 (mobility)를 나타내는 두 가지의 Boolean 변수로 간단히 표시될 수 있다. 이 두 변수를 바탕으로 전체 컨베이어 시스템은 생산기들 사이의 통로인 branch와 branch의 끝점인 knot의 조합으로 구성된 network으로 모델링된다. 이 모델링 방법으로 여러 가지 모양의 컨베이어 시스템을 모델링해 본 결과 복잡한 토폴로지의 컨베이어 시스템도 간단히 모델링 및 시뮬레이션될 수 있었으며 공장설계, 공정제어 등의 시뮬레이션에 유용하게 이용될 수 있음을 보여 주었다.

A Discrete Model of Conveyor Systems for FMS

Ok-Keun Shin[†]

ABSTRACT

In this paper, we propose a discrete model of conveyor systems, which is frequently used in flexible manufacturing systems to transfer work-in-process (WIP) between manipulators. In the cases where the time required for transferring WIP's between manipulators are greater than that of manufacturing itself, as in many flexible assembly lines, the quantitative model of the transfer system is needed to analyze the behavior and productivity of the whole manufacturing system. The proposed model is based upon the assumptions that the length of any unit conveyor component is integer multiple of the length of a pallet and that the transferring speed of the conveyor is constant. Under these assumptions, the observation moments and the length of the conveyor can be quantized. Hence, the state of a conveyor can be represented by two kinds of Boolean variables: one representing the presence of a pallet on each quantized conveyor length and the other representing the mobility of this pallet. The whole conveyor system can be modelled as a network composed of branches and knots based on these two Boolean variables. The proposed modelling method was tested with various conveyor system configurations and showed that the model can be adopted successfully for the simulation of transfer systems and of the piloting of manufacturing processes.

[†] 정 회 원: 한국해양대학교

논문접수: 1996년 1월 30일, 심사완료: 1996년 6월 5일

1. 서 론

자동화된 생산공정에서 컨베이어 시스템 (conveyor system)은 반제품 운반에 많이 이용되며 특히 유연한 조립 시스템 (Flexible Assembly Lines, FAL)의 유연성(flexibility)은 다관절 로봇과 컨베이어에 크게 의존한다. 이런 환경에서 하나의 조립기에서 작업이 끝난 반제품 (pallet)을 다음 작업을 위해 다른 조립기기로 운반하는 데 필요한 시간이 단위 조립 공정에 필요한 시간 보다 훨씬 큰 경우가 많으므로 반제품 운반의 제어는 전체 조립라인의 성능을 결정하는 중요한 요소 중의 하나이다. 또 다품종 소량 생산을 위한 FAL의 경우, 하나의 FAL 내에 두 가지 이상의 반제품이 동시에 조립될 수도 있으므로 컨베이어 시스템 상의 모든 팔레트를 구별하여 제어할 필요가 있다.

기존의 생산공정의 성능 분석에 이용되어 왔던 모델들은 대부분이 생산 시스템의 전반적인 성능 분석을 위한 방법으로 queueing theory를 바탕으로 한 통계적인 방법을 이용하거나 [1, 4] 다이나믹 프로그래밍 기법 [3]등을 이용하여 왔으나 계산량이 많고 복잡하며 순간 순간의 컨베이어의 상태를 파악, 분석하기 위한 모델링 도구로는 적합치 않다. 또 Petri net을 이용하여 모델링할 경우, colored Petri net을 이용하여 [2, 5] 각 팔레트를 개별화하여 인식할 수 있으나 복잡해지는 단점이 있다. 본고에서는 간단한 이산 구조 모델을 통하여 컨베이어 시스템을 모델링하는 방법을 제안한다.

다음의 2장에 컨베이어 시스템을 모델링하기 위한 두 가지 Boolean 변수를 정의하고 난 다음 컨베이어 시스템의 요소들인 branch, corner, T-intersection 및 crossroad를 각각 정의한다. 3장과 4장에서는 branch의 모델링과 knot의 모델링에 관해 서술하고 5장에서는 3장과 4장에서 모델링한 branch와 knot의 조합으로 컨베이어 시스템 전체를 모델링한다.

2. 컨베이어 시스템의 요소

2.1 정의 및 가정

본 모델에서 컨베이어 시스템은 branch와 이 branch들이 서로 접하는 부분인 knot으로 구성된 것으로 간주하며 이들을 모델링하기 위한 상태변수는 샘플링

된 시간 및 공간에서 정의한다. 먼저 본고에서 다루고자 하는 컨베이어 시스템은 다음과 같은 특성을 갖는 것으로 가정한다.

- 조립 기기들 사이에 반제품들이 운반될 수 있게 하는 컨베이어의 부분을 branch라 할 때 모든 branch는 한쪽 방향으로, 동일한 속도 W 로 팔레트를 운반한다.
- 모든 branch의 길이는 팔레트의 길이의 정수배이다, 즉 L 을 어떤 branch의 길이이라 하고 l 을 팔레트의 길이이라 하면 $L=N \cdot l$ (N 은 자연수)의 조건을 만족한다.
- 각각의 branch는 l 의 길이를 갖는 등간격 구간 b_1, b_2, \dots, b_n 으로 샘플링 된다.
- U 를 branch의 입구 혹은 출구에 위치한 stopper에 주어지는 명령이라고 하고 다음과 같이 정의한다:

$$U=0, \text{ if the order of blockage is given,}$$

$$U=1, \text{ otherwise.}$$
- 샘플링 시간 간격을 δt 라 하면 입구의 샘플링 시간 t_k 는 $k \cdot \delta t$, 혹은 간단히 k 로 나타낼 수 있으며 시간 간격 δt 와 branch의 단위 샘플 거리 l 은 다음과 같은 관계를 갖는다: $\delta t=l/W$.

2.2 상태변수의 정의

두 가지 종류의 이진 상태변수를 다음과 같이 정의한다.

2.2.1 팔레트의 유무

샘플링 시간 k 에서 branch의 i 번째 구간의 팔레트 존재 유무는 다음과 같은 이진 상태 변수 P_i 로 나타낼 수 있다(그림 1 참조).

$P_i(k)=1$, 샘플링 시간 k 에서 branch의 i 번째 구간에 팔레트가 존재할 경우,

$P_i(k)=0$, 팔레트가 없을 경우.

2.2.2 팔레트의 이동도

샘플링 시간 k 에서 branch의 i 번째 구간에 팔레트가 존재할 경우 이 팔레트의 이동도 (mobility)는 다음과 같이 정의된다(그림 1 참조).

$V_i(k)=1$, 샘플링 시간 k 에서 branch의 i 번째 구간에

팔레트가 존재하고 이 팔레트가 시간 구간 $k \leq t(k+1)$ 에 branch의 다음 구간으로 진행할 수 있을 경우,

$V_i(k)=0$, 팔레트가 없거나 진행할 수 없을 경우.

2.3 Branch와 Knot

Branch는 컨베이어의 직선 혹은 곡선 형태의 팔레트 운반 통로이며 조립 기기와 조립 기기 사이, 조립 기기와 path controller 사이를 연결하며 knot는 하나의 branch를 다른 branch 또는 knot와 연결시켜 주는 역할을 한다. 일반적으로 knot에는 팔레트의 운반을 제어하기 위한 path controller (stopper 혹은 tractor)가 존재한다.

2.3.1 Branch

직선형 branch와 곡선형 branch의 2가지의 branch를 정의한다. 곡선형(curvilinear) branch의 경우 내반경의 길이가 R_i 이라고 할때 길이가 $\pi * R_i$ 인 직선형 branch로 모델링할 수 있다((그림 2) 참조).

2.3.2 Knots

Knot은 인접하는 branch의 수 및 상대적인 위치에 따라 다음과 같이 세 가지로 나누어 정의한다. Knot을 통과하는 팔레트의 진행방향은 knot의 종류 및 제처에 따라 바뀌어진다.

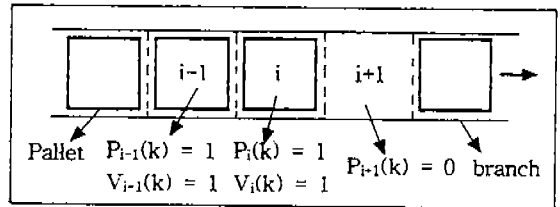
- Corner: 두개의 branch가 직각으로 만나는 곳으로 첫 branch를 통해 들어온 팔레트가 둘째 branch로 전달되는 knot이다 ((그림 4) 참조).
- T-intersection: 세개의 branch가 직각으로 만나는 곳으로 입력 branch와 출력 branch의 상대적인 위치에 따라 다음과 같이 4가지 종류의 T-intersection을 정의한다: Symmetric Convergent T-intersection (SCT), Asymmetric Convergent T-intersection (ACT), Symmetric Divergent T-intersection (SDT), Asymmetric Divergent T-intersection (ADT) ((그림 5) 참조).
- Crossroad: 네개의 branch가 만나는 곳으로 다음과 같이 4가지 종류로 나눈다: Symmetric Crossroad (SC), Asymmetric Crossroad (AC), Convergent Crossroad (CC), Divergent Crossroad (DC) ((그림 7) 참조).

3. Branch의 모델링

3.1 직선형 branch

다음 (그림 1)에 보인 바와 같이 구간 i 에 있는 팔레트가 시간구간 $k \leq t(k+1)$ 사이에 이동도의 값이 1 (true)이 되기 위해서는 다음의 두 가지 조건중 하나를 만족해야 한다:

- a. 샘플링 시각 k 에 구간 $(i+1)$ 이 비어 있을 경우
- b. 샘플링 시각 k 에 구간 $(i+1)$ 에 팔레트가 있고 이 팔레트의 이동도가 1인 경우



(그림 1) Linear branch의 모델링
(Fig. 1) Modelization of a linear branch

따라서 팔레트의 이동도는 다음과 같은 수식으로 표현할 수 있다.

$$V_i(k) = P_i(k) * [P_{i+1}(k) + V_{i+1}(k)], 1 \leq i \leq N-1 \text{ Eq(1)}$$

또 시각 $(k+1)$ 의 순간, 구간 i 의 팔레트 유무를 나타내는 $P_i(k+1)$ 는 다음 두가지 경우 중 하나를 만족할 때 그 값이 1이 된다:

- a. 구간 i 에 시간 $k \leq t(k+1)$ 동안 이동도의 값이 0인 팔레트가 있을 때,
- b. 구간 $(i-1)$ 에 시간 $k \leq t(k+1)$ 동안 이동도의 값이 1인 팔레트가 있을 때.

따라서 샘플링 시각 $(k+1)$ 의 순간, 구간 i 의 팔레트 유무를 나타내는 $P_i(k+1)$ 는 다음과 같이 나타낼 수 있다:

$$P_i(k+1) = P_i(k) * V_i(k) + P_{i-1}(k) * V_{i-1}(k), 2 \leq i \leq N \text{ Eq(2)}$$

이 식은 Eq(1)에서 얻어질 수 있는 $V_{i-1}(k)$ 을 대입

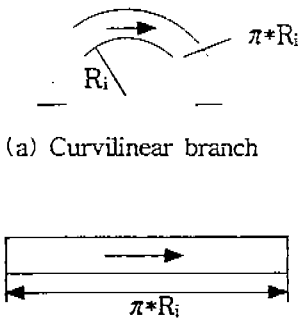
하여 간단히 표현할 수 있다. 즉,

$$P_i(k+1) = P_i(k) * V_i(k)' + P_{i-1}(k) \quad \text{Eq(2)}$$

Eq(1)은 구간 $1 \leq i \leq N-1$ 에서 유효하며 branch의 마지막 구간인 bN 에서의 이동도는 이 branch에 접해 있는 knot의 상태에 의해 결정된다. 마찬가지로 Eq(2)는 $2 \leq i \leq N$ 의 구간에서 유효하며 구간 $i=1$ 의 다음 샘플링 순간인 $(k+1)$ 에서의 팔레트의 유무는 이 branch의 입력단에 연결된 knot의 상태에 의해 결정된다. 이들 경계조건은 다음장의 knot 모델링에서 다루기로 한다.

3.2. 곡선형 branch

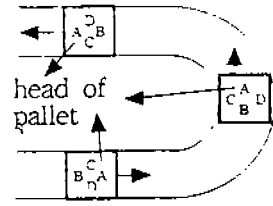
곡선형 branch는 길이가 같은 직선형 branch로 대체할 수 있으며 아래의 (그림 2(a))에 곡선형 branch를, 그리고 (그림 2(b))에 이와 등가인 직선형 branch를 보인다.



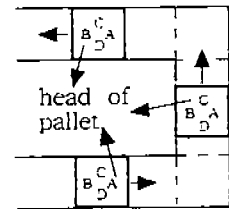
(그림 2) 곡선형 branch와 이와 등가인 직선형 branch
(Fig. 2) Cuvilinear branch and its equivalent

대부분의 팔레트를 이용한 조립 시스템에서 컨베이어 시스템은 폐회로로 구성되며 폐회로를 구성하는 방법은 (그림 3)에 그 일부분을 보인 것 같이 두 가지로 생각할 수 있다.

(그림 3 (a))에 보는 것처럼 곡선형 branch를 이용한 경우, 팔레트의 진행 방향은 항상 일정하게 유지된다. 반면에 (그림 3 (b))와 같이 두개의 corner와 한개의 직선형 branch를 이용하여 폐회로를 구성할 경우에는 팔레트의 진행 방향이 달라지게 된다.



(a) Curvilinear branch



(b) Two corners and three branches

(그림 3) 곡선형 branch의 특성
(Fig. 3) Characteristics of a Curvilinear branch

4. Knot의 모델링

Knot은 두개 이상의 branch들을 서로 연결시켜 주는 역할과 팔레트의 방향전환 혹은 경로제어의 역할을 수행한다. 일반적으로 knot의 입구와 출구에는 tractor나 stopper가 있어 이들의 제어를 통해서 팔레트의 운반을 제어할 수 있다. 본고에서는 각 knot의 한 변의 길이가 팔레트의 길이와 같다고 가정한다.

4.1 Corner

Corner는 입력 branch와 출력 branch가 직각으로 만나는 knot이다. 여기서는 corner를 제어하기 위한 두 종류의 제어 명령을 corner의 입구와 출구에 하나씩 정의한 다음 corner와 이에 맞닿은 두 branch에서의 팔레트의 유무와 이동도를 모델링한다.

- Corner의 입력 제어명령: $U_E(k) = 1$ if input enabled, 0 otherwise,
- Corner의 출력 제어명령: $U_S(k) = 1$ if output enabled, 0 otherwise.

4.1.1 입력 branch의 모델링

(그림 4)에 보인 것처럼 입력 branch의 마지막 구간에 있는 팔레트의 이동도는 다음의 조건을 모두 만족시킬 경우에 1의 값을 갖는다.

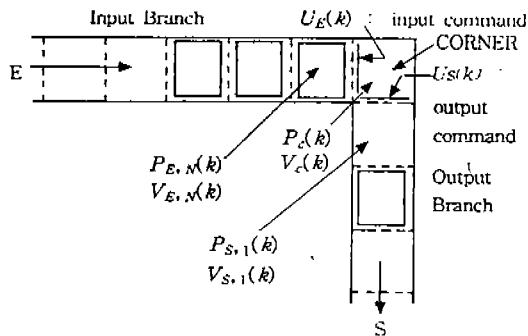
- (a) 입력 branch의 마지막 구간에 팔레트가 존재하고 ($P_{E,N}(k)=1$),
- (b) corner의 입력 명령이 주어지고 ($U_E(k)=1$),
- (c) corner에 팔레트가 없을 경우 ($P_C(k)=1$).

위의 조건 (c)는 corner가 갖는 특징중의 하나로 corner에 팔레트가 존재할 경우 그 팔레트의 이동도에 관계없이 입력 branch의 팔레트는 corner에 진입할 수 없다. 즉, 입력 branch에 연속적으로 팔레트가 공급된다 하더라도 출력 팔레트는 한 구간씩 건너뛰어 출력된다.

또 이 구간의 다음 샘플링 순간에서의 팔레트의 유무 및 $P_{E,N}(k+1)$ 는 Eq (3)와 같은 형식으로 표현되며 corner에 접한 입력 branch의 마지막 구간에서의 이동도는 Eq(4)와 같이 표현된다.

$$P_{E,N}(k+1) = P_{E,N}(k) * V_{E,N}(k)' + P_{E,N-1}(k) \quad \text{Eq(3)}$$

$$V_{E,N}(k) = P_{E,N}(k) * U_E(k)' + P_C(k)' \quad \text{Eq(4)}$$



(그림 4) Corner
(Fig. 4) Corner

4.1.2 Corner의 모델링

위의 branch와 마찬가지로 방법으로 corner에서의 팔레트의 유무와 그 이동도는 다음의 식들과 같이 표현된다.

$$P_C(k+1) = P_C(k) * V_C(k)' + P_{E,N}(k) * V_{E,N}(k) \quad \text{Eq(5)}$$

$$V_C(k) = P_C(k) * U_S(k) * [P_{S,1}(k)]' + V_{S,1}(k) \quad \text{Eq(6)}$$

이상의 Eq(3)-Eq(6)에서 PC 와 VC는 각각 corner의 팔레트의 유무와 corner에 있는 팔레트의 이동도를, 그리고 $P_{S,1}(k)$ 와 $V_{S,1}(k)$ 는 출력 branch의 첫 구간의 팔레트 유무와 이동도를 나타낸다.

4.1.3 출력 branch의 모델링

마찬가지 방법으로 출력 branch의 첫 구간의 모델링은 다음과 같이 표현된다.

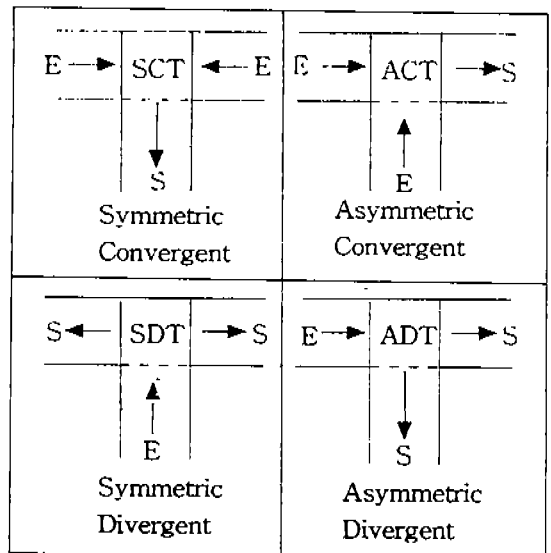
$$P_{S,1}(k+1) = P_{S,1}(k) * V_{S,1}(k)' + P_C(k) * V_C(k) \quad \text{Eq(7)}$$

$$V_{S,1}(k) = P_{S,1}(k) * [P_{S,2}(k)]' + [P_{S,2}(k)] \quad \text{Eq(8)}$$

위의 식 Eq(4)와 Eq(8)를 비교하면 입력 branch의 이동도와 출력 branch의 이동도의 차이를 알 수 있는데 이는 corner의 특성을 반영한 것이다.

4.2 T-Intersection

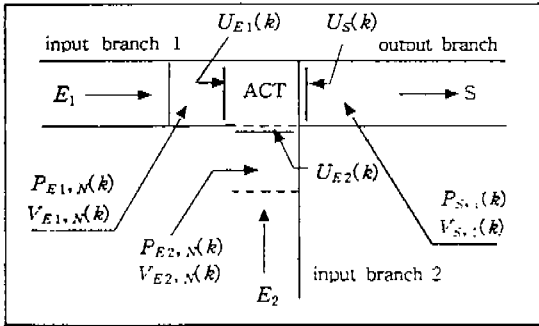
2.3.2에서 정의한 네 가지 T-intersection을 (그림 5)에 보인다. 본고에서는 T-intersection의 모델링을 설명하기 위해 ACT (Asymmetric Convergent T-intersection)를 예로 들어 모델링한다.



(그림 5) T-Intersections
(Fig. 5) T-intersections

(그림 6)에 보인 것처럼 ACT에는 $U_{E_1}(k)$, $U_{E_2}(k)$ 와 $U_S(k)$ 의 세가지 제어명령을 정의한다. 단 ACT에서는 두개의 입력 branch에서 들어오는 팔레트가 서로 충돌할 수 있으므로 $U_{E_1}(k)$ 과 $U_{E_2}(k)$ 는 다음과 같은 조건을 만족하여 동시에 두개의 입구가 개방되지 않도록 해야 한다. : $U_{E_1}(k) * U_{E_2}(k) = 0$.

ACT와 이에 연결된 세개의 branch의 마지막 구간은 다음과 같이 모델링된다.



(그림 6) 비대칭 Convergent T-Intersection
(Fig. 6) Asymmetric Convergent T-intersection

- 입력 branch 1의 모델링

$$P_{E1,N}(k+1) = P_{E1,N}(k) * V_{E1,N}(k)' + P_{E1,N-1}(k) \text{ Eq(9)}$$

$$V_{E1,N}(k) = P_{E1,N}(k) * U_{E1,N}(k) * [P_T(k)' + P_T(k) V_T(k)] \\ = P_{E1,N}(k) * U_{E1,N}(k) * [P_T(k)' + V_T(k)] \text{ Eq(10)}$$

- 입력 branch 2의 모델링

$$P_{E2,N}(k+1) = P_{E2,N}(k) * V_{E2,N}(k)' + P_{E2,N-1}(k) \text{ Eq(11)}$$

$$V_{E2,N}(k) = P_{E2,N}(k) * U_{E2,N}(k) * P_T(k)' \text{ Eq(12)}$$

- ACT의 모델링

$$P_T(k+1) = P_T(k) * V_T(k)' + P_{E1,N}(k) * V_{E1,N}(k) \\ + P_{E2,N}(k) * V_{E2,N}(k) \text{ Eq(13)}$$

$$V_T(k) = P_T(k) * U_{S1}(k) * [P_{S,1}(k)' + V_{S,1}(k)] \text{ Eq(14)}$$

- 출력 branch의 모델링

$$P_{S,1}(k+1) = P_{S,1}(k) * V_{S,1}(k)' + P_T(k) * V_T(k) \text{ Eq(15)}$$

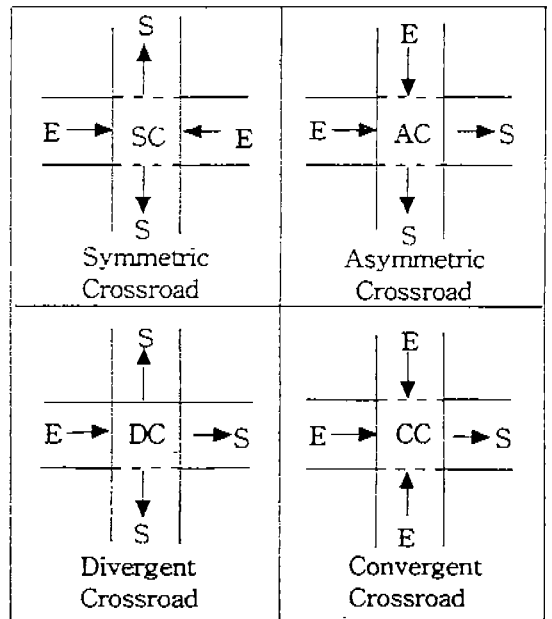
$$V_{S,1}(k) = P_{S,1}(k) * [P_{S,2}(k)' + V_{S,2}(k)] \text{ Eq(16)}$$

두개의 입력 branch의 이동도인 Eq(10)과 Eq(12)의

차이는 출력 branch에 대한 두 입력 branch의 상대적인 위치가 서로 다르기 때문이다. 여기서 다루지 않은 나머지 세 가지의 T-intersection에 관한 모델링은 ACT와 유사한 방법으로 얻어질 수 있다.

4.3 Crossroad

(그림 7)에 보인 것과 같은 네 가지 종류의 crossroad를 생각할 수 있다. 이들 중 AC (Asymmetric Crossroad)에 관한 모델링을 예로 든다.



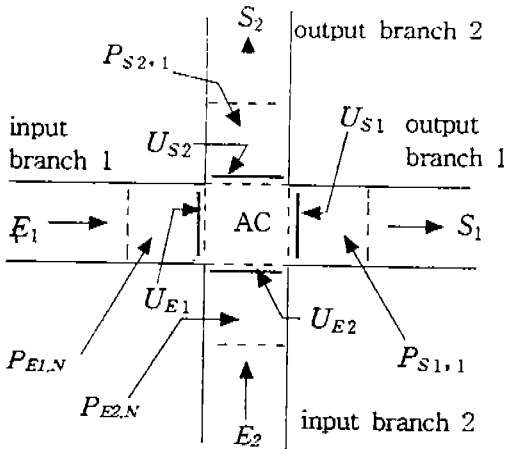
(그림 7) Crossroads
(Fig. 7) Crossroads

(그림 8)에 보이는 AC는 두개의 입력 branch와 두개의 출력 branch로 구성되어 있으며 입력 branch 두개와 출력 branch 두개가 서로 인접해 있다. 이 branch들의 입력과 출력을 제어하기 위한 명령을 각각 UE1과 UE2, US1과 US2 라 하면 branch간의 충돌을 피하기 위해 다음과 같은 조건이 필요하다.

$$U_{E1} * U_{E2} = 0 \text{ and}$$

$$U_{S1} * U_{S2} = 0.$$

- 출력 branch 1의 모델링



(그림 8) 비대칭 Crossroad
(Fig. 8) Asymmetric Crossroad

$$P_{S1,1}(k+1) = P_{S1,1}(k) \cdot V_{S1,1}(k)' + P_C(k) \cdot V_{S1}(k) \quad \text{Eq(17)}$$

$$V_{S1}(k) = P_C(k) \cdot U_{S1}(k) \cdot [P_{S1,1}(k)' + V_{S1,1}(k)] \quad \text{Eq(18)}$$

여기서 $V_{S1}(k)$ 은 crossroad내에 있는 팔레트에 대한 출력 branch 1의 첫 구간으로 향하는 이동도를, 그리고 $V_{S1,1}(k)$ 은 출력 branch 1의 첫 구간에 있는 팔레트의 둘째 구간으로 향하는 이동도를 나타낸다.

- 출력 branch 2의 모델링

$$P_{S2,1}(k+1) = P_{S2,1}(k) \cdot V_{S2,1}(k)' + P_C(k) \cdot V_{S2}(k) \quad \text{Eq(19)}$$

$$V_{S2}(k) = P_C(k) \cdot U_{S2}(k) \cdot [P_{S2,1}(k)' + V_{S2,1}(k)] \quad \text{Eq(20)}$$

- 입력 branch 1의 모델링

$$P_{E1,N}(k+1) = P_{E1,N}(k) \cdot V_{E1,N}(k)' + P_{E1,N-1}(k) \quad \text{Eq(21)}$$

$$V_{E1,N}(k) = P_{E1,N}(k) \cdot U_{E1}(k) \cdot [P_C(k)' + V_{S1}(k)] \quad \text{Eq(22)}$$

- 입력 branch 2의 모델링

$$P_{E2,N}(k+1) = P_{E2,N}(k) \cdot V_{E2,N}(k)' + P_C(k) \cdot V_{E2}(k) \quad \text{Eq(23)}$$

$$V_{E2,N}(k) = P_{E2,N}(k) \cdot U_{E2}(k) \cdot [P_C(k)' + V_{S2}(k)] \quad \text{Eq(24)}$$

- Asymmetric crossroad의 모델링

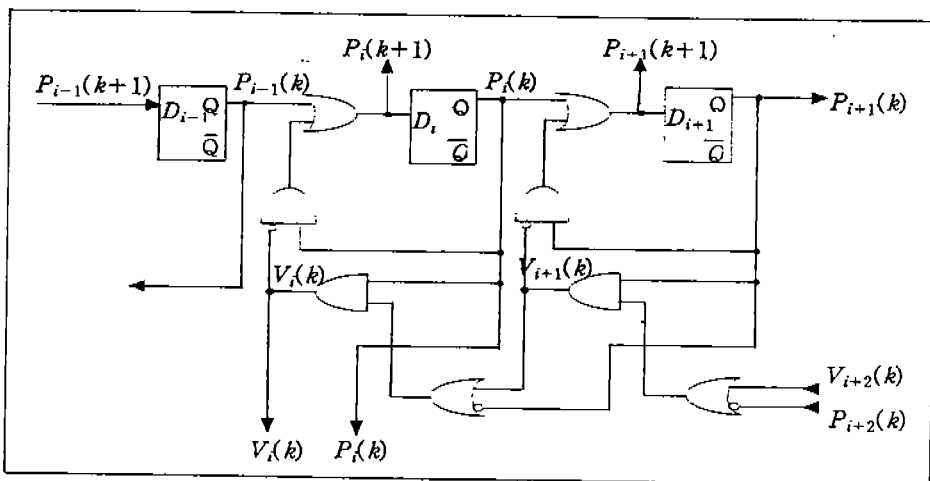
$$P_C(k+1) = P_C(k) \cdot V_C(k)' + P_{E1,N}(k) \cdot V_{E1,N}(k) + P_{E2,N}(k) \cdot V_{E2,N}(k) \quad \text{Eq(25)}$$

$$V_C(k) = V_{S1}(k) + V_{S2}(k) \quad \text{Eq(26)}$$

Eq(26)에 나타난 AC의 이동도는 출력 branch가 두 개이기 때문에 각 출력 branch의 이동도의 합으로 나타내어진다. 다른 세가지의 crossroad도 같은 방법으로 모델링될 수 있다.

5. Transfer System의 모델링

앞장에서 서술한 컨베이어 시스템의 각 요소들에



(그림 9) D flip-flop과 로직 게이트를 이용한 시뮬레이션의 예
(Fig. 9) Simulation by D flip-flop and gates

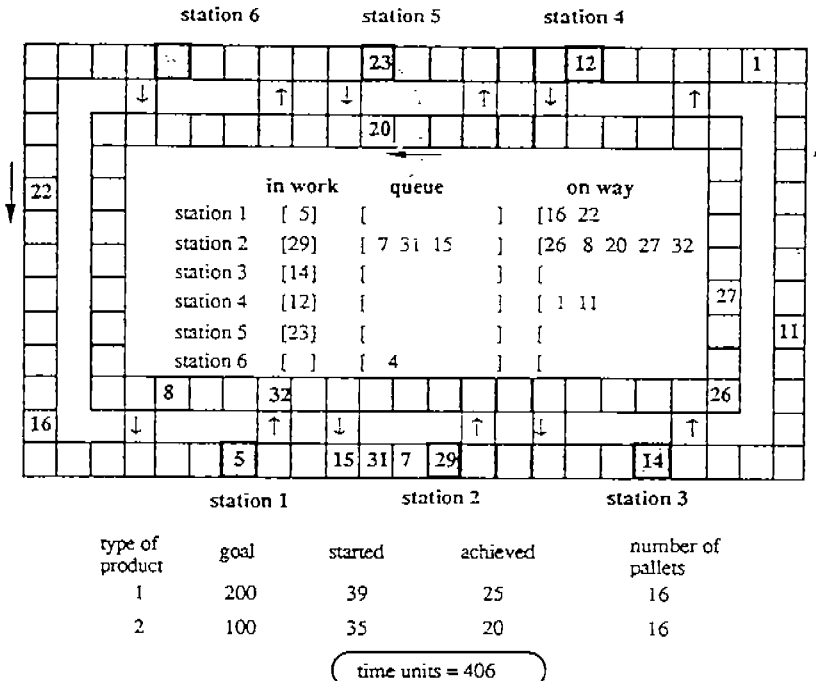
대한 모델링을 이용하여 일반적인 컨베이어 시스템, 혹은 transfer system들은 다음과 같은 순서에 의해 branch와 knot의 조합으로 모델링할 수 있다:

1. 주어진 컨베이어 시스템을 knot과 branch로 구분한 다음 각 knot을 입력력 branch의 수와 방향에 따라 corner, T-intersection 및 crossroad로 세분한다.
2. 각각의 branch를 팔레트의 길이와 같은 크기의 등간격 구간으로 나눈다.
3. 컨베이어 시스템의 topology를 고려하여 knot과 branch의 각 구간에 상응하는 이동도 $V_i(k)$ 및 팔레트의 유무 $P_i(k)$ 를 나타내는 식을 할당한다. 이때 $V_i(k)$ 및 $P_i(k)$ 에 각 knot의 입출력을 제어하는 stopper (tractor)의 명령 $U_i(k)$ 를 포함시킨다.

이렇게 얻어진 컨베이어시스템의 모델은 소프트웨

어나 (그림 9)에 보이는 것처럼 latch와 logic gate로 구성하여 시뮬레이션할 수 있다. (그림 9)에 보인 회로는 linear branch의 한 부분을 모델링한 것이다.

(그림 10)에 실제의 컨베이어 시스템을 본고에서 제안하는 모델링 기법을 이용하여 소프트웨어로 시뮬레이션한 예를 보인다. 이 예에서 보인 컨베이어 시스템은 두개의 컨베이어 loop와 여섯 개의 로봇 매뉴플레이터로 구성된 조립라인이며, 제안된 시뮬레이션 기법을 시스템의 파일로팅 및 라우팅 알고리즘의 성능을 분석하기 위해 이용했다. 이 알고리즘은 다품종 소량 생산을 위한 FAL의 라우팅과 파일로팅을 실시간 제어함으로써 FAL의 효율을 높이는 동시에 복잡한 단기 생산계획을 생략할 수 있는 알고리즘이다[6, 7]. (그림 10)에 보인 각각의 격자는 branch의 한 구간, 또는 하나의 knot에 해당하며 각각의 격자 i 에 팔레트의 유무 $P_i(k)$ 와 팔레트의 이동도 $V_i(k)$ 가 할당된다. 각 격자에 할당된 $V_i(k)$ 와 $P_i(k)$ 는 Eq(1)과 Eq(2)에서 정의한 것처럼 구간 $(i-1)$ 과 구간 (i) 의 P, V



(그림 10) 이중 루프 컨베이어 시스템의 시뮬레이션 예
(Fig. 10) A Simulation of a Double-Loop Conveyor System

및 U 의 함수이므로 전체 컨베이어는 일련의 recursive Boolean 방정식으로 표현될 수 있다. 따라서 팔레트의 초기 분산 정보가 주어지면 이로부터 시간 변화에 따른 팔레트의 위치변화를 시뮬레이션할 수 있다. 또 격자가 knot일 경우 팔레트의 운반을 제어하는 stopper (혹은 tractor)가 부착될 수 있으며 이 경우 knot에 부착되는 stopper (혹은 tractor)의 수와 같은 수의 제어 명령 $U(k)$ 가 할당되어 팔레트의 라우팅을 제어하는 수단을 제공한다.

본 연구에서 구현한 시뮬레이션 소프트웨어는 (그림 10)에 보인 것처럼 매 샘플링 주기마다 팔레트가 컨베이어에 의해 운반되는 상태를 컴퓨터 화면을 통하여 볼 수 있게 하였다. 격자 안의 숫자는 팔레트의 식별번호를 나타내며 작은 화살표는 내부의 컨베이어와 각 매니플레이터 사이에 팔레트가 반입/반출될 수 있게 하는 stopper/tractor를 표시하며 이를 통하여 팔레트의 라우팅을 제어한다.

제한한 모델링 기법을 이용함으로써 다음과 같은 FAL의 동적 특성을 고려한 모델링 및 시뮬레이션이 가능하였다:

- 매니플레이터들 사이의 컨베이어의 길이,
- 시간에 따른 팔레트의 위치 변화,
- 매니플레이터의 상호 위치
- 컨베이어/매니플레이터의 고장으로 인한 transfer 장애 등.

FAL의 이러한 동적 특성은 FAL의 성능에 중요한 영향을 미치며ダイ나믹 파일로팅과 반제품(팔레트)의 라우팅 등에 대한 결정을 실시간으로 내리기 위해 매순간 고려되어야 하는 데 queueing theory를 바탕으로 하는 통계적 방법[1,4]이나 다이내믹 프로그래밍[3]을 이용하는 기존의 성능 분석 방법으로는 다루기 어려웠던 부분이다.

6. 결 론

본고에서는 유연한 생산 시스템과 조립 시스템에서 조립 기기들 사이의 반제품 운반에 많이 이용되는 컨베이어 시스템을 위한 모델을 제안하였다. 이 모델은 생산 시스템 내에서 반제품운반에 소요되는 시간이 생산 작업 자체에 소요되는 시간보다 길어서 반제품 운반시스템의 영향을 무시할 수 없는 조립 시스템

의 성능분석, 파일로팅 시뮬레이션, 부품의 라우팅등에 이용될 수 있다. 본고에서는 컨베이어 시스템의 각 branch와 knot의 길이가 팔레트 길이의 정수 배이며 컨베이어의 속도가 일정하다는 가정 하에 컨베이어 시스템의 이산구조 모델을 제안하였다. 이런 가정 하에 컨베이어운동의 관측시간과 컨베이어의 길이를 양자화하고, 각 샘플링 시각에서 단위 길이당 컨베이어의 팔레트 유류와 팔레트의 이동도를 각각 Boolean 변수로 표시하고 이들의 관계를 Boolean 방정식으로 표현하였다. 또 전체 컨베이어 시스템은 이들 두 변수를 바탕으로 한 branch와 knot의 조합으로 모델링되며, 이를 위해 필요한 branch와 knot의 모델링에 대하여 기술하였다. 이 모델을 이용하여 특정 컨베이어 시스템을 소프트웨어로 시뮬레이션 하였으며 그 결과 공장설계, 공정제어, 생산시스템의 성능분석 등에 유용하게 이용될 수 있음을 보였다[6, 7].

참 고 문 헌

- [1] D. P. Heyman and M. J. Sobel "Stochastic models in operations Research," Vol. 1. New York: MacGraw-Hill, 1982.
- [2] Y. Narahari & Viswanadham "A Petri Net Approach to the Modeling and Analysis of Flexible Manufacturing Systems", Annals of Operations Research 3, 1985 pp. 449-472.
- [3] Stanley B. Gershwin, "Hierarchical Flow Control :A Framework for Scheduling and Planning Discrete Events in Manufacturing Systems", Proc. IEEE, Jan. 1989, pp. 195-209.
- [4] S. P. Sethi, M. Taksar & Q. Zhang "Capacity and Production decisions in Stochastic Manufacturing Systems:An Asymptotic Optimal Hierarchical Approach," Produc. Op. Management, Vol. 1, pp. 367-392, 1992.
- [5] G. Brault, "Mod lisation, commande et Simulation d' lot flexible d'assemblage", Th se de doctorat de l'Univ. de Franche-Comt. Dec. 1986.
- [6] 신옥근, "유연한 조립 시스템의 단기 생산 스케줄링과 라우팅에 관한 연구", 정보처리 논문지, 제2권, 제6호, pp. 823-830.

[7] J. P. Bourrieres, O. K. Shin, F. Lhote, "Real Time Scheduling and Dynamic Parts Routing for Flexible Assembly Lines", Intl. Journal of Systems Automation: Research and Applications vol. 1, No. 4, pp. 387 -396. 1991.



신 옥 근

1981년 서강대학교 전자공학과 졸업 (학사)

1983년 부산대학교 전자공학과 (공학석사)

1989년 프랑스 Université de Franche-Comté (공학박사)

1983년~1995년 한국전자통신연구소 선임연구원

1995년~현재 한국해양대학교 컴퓨터공학과 전임강사
관심분야: Factory Automation, 멀티미디어 시스템