□ 기술해설 □

계층의 구조를 갖는 지능형 유연생산 시스템 시뮬레이션

경남대학교 조대호* · 하경재*

목 차

1. 서 론

인공지능의 많은 연구가와 이 분야에 종사하는 사람들이 인공지능 이론이 공장 내의 제조과정에서 발생하는 문제들의 해결에 주요하게 적용될 수 있을 것이라는 견해를 보였고 특히 인공지능의 전문가시스템은 여러 분야에서 다양한 난이도의 문제를 해결하는데 개발되어 왔다. 인공지능과 전문가시스템이론은 공장 내의 제조과정 전반에 걸쳐 잠재적인 적용범위가 크다. 특히 제조과정 중의 제조계획, 스케줄링, 공정제어, 유지, 고장진단등의 분야에 성공적인 적용 예를 보이고 있다[1, 2, 3]. Matsuo[4]는 컴퓨터인테그레이티드(computer-integrated) 제조환경에서 스테커크레인(stacker crane)을 제어하는 지식베이스를 이용한 지능형 크레인 시스템을 개발하였다. O'Grady[5]는 자동화된 공장의 지능형 셀(cell) 제어시스템을 개발하였다. 그렇지만 인공지능 기법을 이용한 제조도구들에 대한 연구[3, 6, 7, 8, 9, 10, 11, 12, 13]는 많이 이루어지고 있는 반면 실제 현장에서 일상적으로 사용되는 시스템은 아주 적다.

*종신회원

인공지능 스케줄러를 실제 제조환경에서 사용하기 위하여서는 총체적인 시스템 관점에서 문제를 해결하여야 한다. 시뮬레이션은 이미 제조공정의 디자인과 분석에 가장 많이 사용되는 방법 중의 하나이고[14, 15], 인공지능의 기법을 이용한 도구들의 효과와 효능을 실제 시스템에 적용 하기 전에 검증할 때 사용된다. 그러나 규모가 큰 시스템의 연구를 위한 모델들은 아주 복잡한 경향이 있고 이러한 시스템의 시뮬레이션 모델을 구성하는 것은 실로 힘드는 일이다. 이와 같은 사실은 인공지능의 이론를 기반으로 한 도구의 검증을 전체 시스템의 동적인 시뮬레이션 결과로써 검증하려고 할 때에는 더더욱 힘들어진다. 그러므로 현재의 지식베이스를 이용한 제조공정의 시뮬레이션에 있어서 단지 전체 시스템의 일부만을 연구의 대상으로 하는 것은 쉽게 납득할 수 있다. 예를 들면 제조환경에서의 스테커크레인을 제어하는 전문가시스템[4]은 여러 워크스테이션(workstation)을 이동하는 하나의 크레인만을 제어한다. 또 다른 시뮬레이션 모델로써 디스패치룰(dispatch rule)을 검증하기 위한 시스템은 단지 두개의 워크스테이션과 각각의 스테이션에 독립된 큐가 들어 있는 시스템이다[2]. O'Grady[5]의 4 단계

계층(공장, 샵(shop),워크쎌, 기계)의 제어를 위한 블랙보드(blackboard) 구조는 이 인공지능의 기법을 검증하기 위한 제조공정 모델이나 실제의 제조환경 어느 것도 포함하고 있지않다.

이 전의 연구에서는 인공지능의 생산제어 도구를 검증할 목적으로 제조시스템의 모델링과 시뮬레이션을 위한 지식베이스 환경을 개발하였다[16, 17]. 본 논문에서는 이와 같은 시뮬레이션 환경의 유용함을 입증하기 위하여 전문가시스템 모델을 이 환경하에서 구성하고 이 전문가시스템을 이용하여 작업공정에서의 일괄처리 일(batch-job)들의 운송을 담당하는 운송장치의 경로설정(routing)에 적용하여 보여준다. 이 일괄처리 작업들은 우선 부분 일괄처리 일들(sub-batches)로 분할되어 운송 되어지는데 이 분할에 작업중첩(operation overlapping) 알고리듬이 다단계 계층구조를 갖는 유연생산시스템(multi-level hierarchical flexible manufacturing system)에 적용된다. 그리고 시뮬레이션을 통하여 이 작업중첩 알고리듬이 제조리이드타임(manufacturing lead time)과 WIP (work in process)를 줄일 수 있도록 하는 작업수행전략(operation regime)을 찾아낼 수 있다. 먼저, 이러한 중첩수행 사례 예를 명시하는데 필요한 시뮬레이션 환경 개념 및 특징들에 대하여 간단히 살펴보기로 한다.

2. 연구배경

Cho[16, 17]는 전문가시스템기법을 객체지향형 시뮬레이션 환경에 삽입하여 전문가시스템의 기능을 가진 모델을 쉽고 빠르게 생성하는 방법을 제시하였다. 전문가시스템기법의 삽입을 위하여 먼저 객체지향형 프로그래밍 패러다임하에서 전문가시스템 구성을 위한 클래스들과 이에 필요한 프로씨저들을 구성하고 기존의 객체지향형 시뮬레이션 환경 내의 모델클래스와 전문가시스템구성을 위한 클래스로부터 그들 클래스의 기능을 상속받는 하위클래스를 제공함으로써 시뮬레이션 모델의 기능과 전문가시스템의 기능을 동시에 갖는 모델을 이 하위클래스로부터 생성한다. 이 전문가시스템 쉘(shell)은 인공지능과 시스템모델링개념을 결합시킨 지식베이스을 이용한 설계 및 시뮬레이션 환경인 DEVS(discrete-event system specification) scheme[19, 20]하에서 개발되었다. 이 쉘은 인터럽트가 가능한 분산전문가시스템들을 시뮬레이션 모델의 구성요소(component)들로써 정의할 수 있다. 따라서 FMS 및 다른 많은 자율지능시스템(autonomous intelligent system)의 지식베이스를 이용한 시스템 시뮬레이션 모델링을 쉽게 할 수 있다. 더구나 시스템구조를 재귀적 시스템엔티티구조(system entity structure : SES)[16, 19, 21]를 사용하여 나타낼 수 있으므로 재귀적 프루닝(recursive pruning)이라는 확장된 SES 프루닝기법을 사용하여 계층적 구조 군(family)이 생성되도록 전개해 나갈 수가 있다. 따라서 이와 같은 계층구조의 재귀적 생성은 특히 FMS공장의 설계에 적합하다.

2.1 시뮬레이션을 위한 전문가시스템 환경 (DESE)

분산전문가시스템 환경(Distributed Expert System Environment : DESE)이란 전문가시스템모델이 분산된 제어기구(추론엔진)와 지식(규칙과 사실)을 가질 수 있는 환경이다. DESE는 객체클래스, 메소드(method) 및 분산전문가시스템(distributed expert system : DES)들을 생성하기 위한 유틸리티함수들로 구성된다. DESE하에서 생성된 DES들은 포괄적(generic) 클래스들의 인스턴스들이며, 재사용, 모듈화 및 확장 용이성이라고 하는 잘 알려진 객체지향프로그래밍의 이점을 제공한다.

DES의 정의 속에 포함된 객체들은 지식베이스, 추론엔진, 규칙과 사실 클래스들의 인스턴스들이다. 추론엔진 인스턴스는 관련된 지식베이스 인스턴스를 참조하며, 이 지식베이스 인스턴스는 서로 관계를 갖는 규칙과 사실에 대하여 링크들을 갖는다. 따라서, 추론엔진 클래스 혹은 그것의 예속클래스(sub-class)의 추론메소드(inferencing method)들은 포괄적이다 (즉, 여러 클래스들로부터 나온 규칙과 사실에 사용되어 질 수가 있다).

2.2 DESE와 DEVS-scheme의 인터페이스

DESE와 DEVS-scheme은 atomic-expert-model 이라는 클래스를 생성하여 인터페이스 시킬 수 있다. 이 클래스인 기존의 DEVS-scheme의 기본 클래스인 지식베이스와 atomic-model 들로부터 메소드와 변수를 상속받아 증식한다(그림 1). 따라서, 그림 2에서와 같이 atomic-expert-model 클래스의 한 인스턴스는 하나의 DES(expert-core라 한다)와 하나의 atomic-model(DEVS 구성요소라 한다) 모두를 갖는다. 그러한 모델은 동형복사(isomorphic copies)를 만드는 make-copy 메소드를 호출하여 복제시킬 수 있다[25].

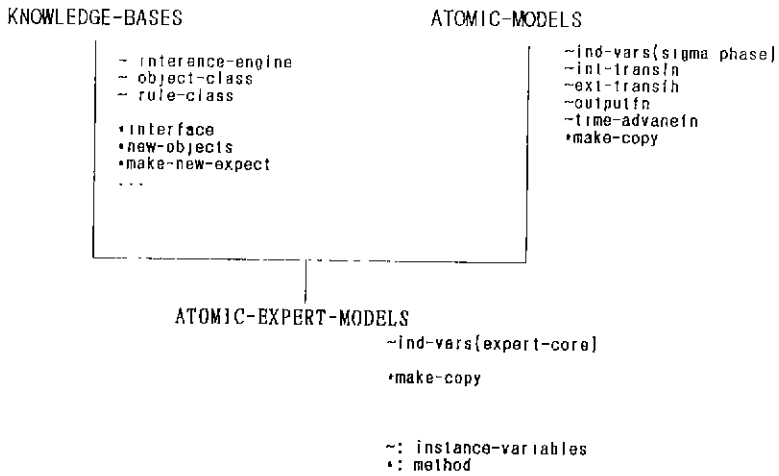


그림 1 DES와 DEVS-Scheme의 인터페이스

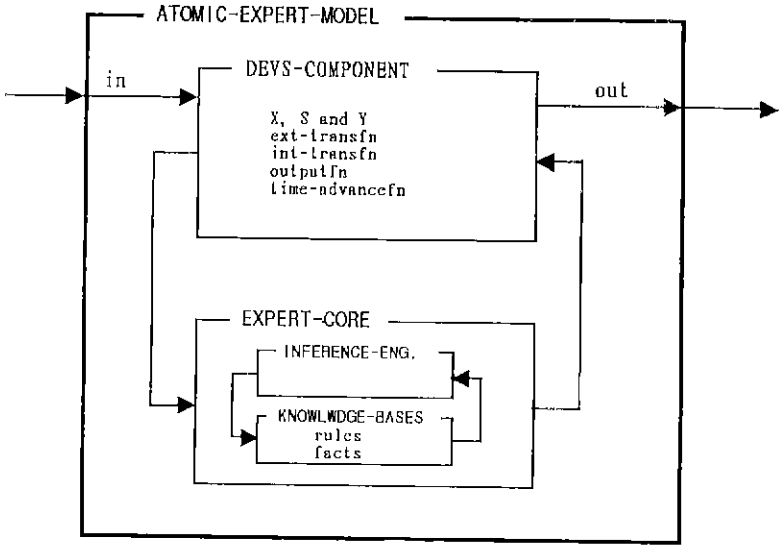


그림 2 Atomic-expert 모델의 구조

상속을 통하여 DESE와 DEVS-scheme을 합병함으로써 나중에 계층적이고 모듈러한 이산-사건모델을 구축하기 위한 분산전문가시스템의 구성요소들의 설계 및 검사가 쉬워진다.

또한 전문가시스템의 구성요소들이 다른 DEVS모델들의 경우와 동등하게 취급되어지기 때문에 시스템엔티티와 모델베이스들이 한번 구축되기만 하면 또 다른 모델아키텍츄어를 생성하기가 상당히 쉽다.

2.3 계층구조의 FMS시스템을 위한 프랙탈 아키텍츄어(Fractal Architecture)

Cho[16, 17]는 재귀적 시스템엔티티 구조를 생성하기 위한 기법을 제시하였으며, 그것을 Tirpak 등[26]에 의해 소개된 FMS시스템의 프랙탈 아키텍츄어를 표현하는 데 적용하였다. 여기서 시스템엔티티구조(SES)란 하나의 모델베이스에 있는 구성요소들로 부터 모델들을 합성한 것을 말한다. SES는 분해(decomposition), 분류(taxonomy) 및 연결관계성(coupling relationship)들을 갖춘 하나의 지식표현체계이다. 재귀적프루닝 이라는 조작기법은 프랙탈과 유사한 성질을 갖는 계층적 모델구조를 생성하기 위해 개발된 것이다.

Tirpak 등[26]은 FMS시스템의 한 모델을 사용하여 재귀적형태로 전체시스템을 만들면 유사한 구조와 제어기능을 갖는 고도의 분리된 장치들을 자연스럽게 계층적으로 분해할 수 있다고 주장하였다. 그러한 구조를 갖도록 하는 목적은 지역적인 기능성(local functionality)을 최대화하고 전역적인 제어(global control)는 최소화 함으로써 FMS시스템의 계층이 갖는 구조적 복잡성(structural complexity)과 코디네이션(coordination)을 관리하기 위한

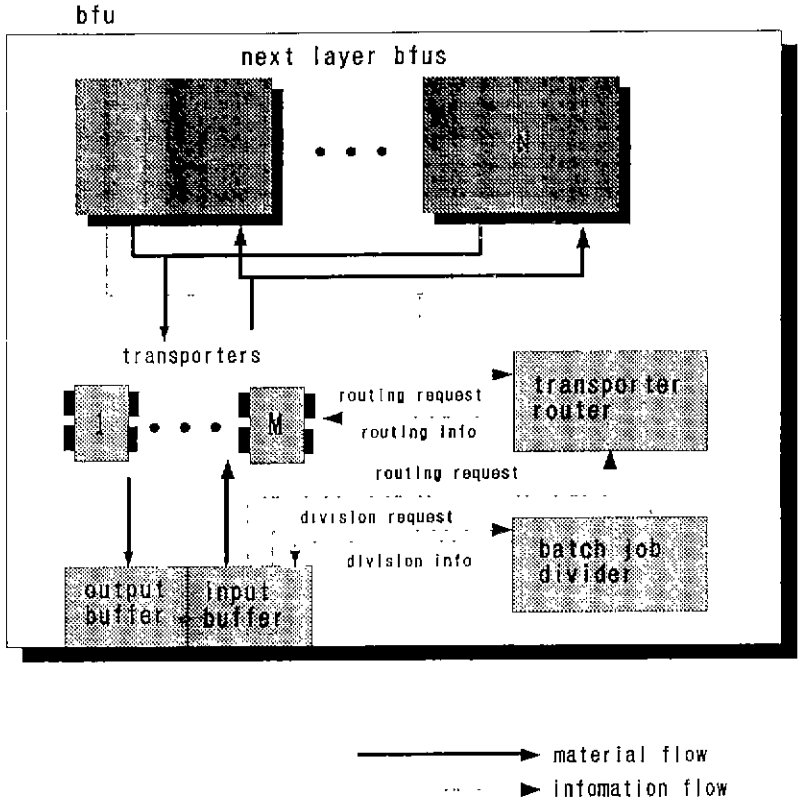


그림 3 기본 프렉탈 유닛(basic fractal unit : BFU)의 구조

것이다. 더구나 반복해서 나타나는 구성요소들은 레벨간의 재사용을 극대화 할 수 있도록 객체지향형 프로그래밍 패러다임하에서 설계될 수가 있다. 따라서 FMS 프랙탈 아키텍쳐어 모델은 기본 프랙탈 유니트(basic fractal unit : BFU)라고 하는 간단한 기본 설계요소들로 구성된 계층적 구조를 나타낸다. BFU를 설계할 때는 계층내의 어떠한 레벨도 완전히 표현할 수 있는 적절한 속성(attribute)집합을 갖도록 한다[26].

그림 3은 모델계층내 어떠한 레벨의 구조도 완전히 기술할 수 있는 요소(element)들이 포함되도록 특별히 설계된 BFU를 보여 준다. 하나의 BFU내에는 내부의 상세한 구조가 숨겨진 하위계층 BFU들의 집합이 포함되어 있다. 경로제어장치(운송장치 경로기 : transporter router)는 이 하위계층의 BFU들을 일괄처리 일들이 이송되어야 할 스테이션들로 간주한다. 또한 하위계층 BFU들의 내부에 있는 운송장치 경로기 또한 인수받은 일괄처리 일에 대하여 계속하여 경로제어를 해야한다.

그림 4는 3레벨 프랙탈 아키텍쳐어를 나타내는 프루닝된 엔티티구조(pruned SES ; PES)이다. 계층구조의 최상부는 하나의 공장을 나타낸다. 공장은 두 개의 샵플로(shop floor)로 구성된다. 첫번째 샵플로(shopfloor 1)는 각각 워크스테이션 1, 워크스테이션 2인 두 워크스테이션을 갖는다. 그러나 두번째 샵플로는 워크스테이션이 없고 두 기계(machine)만을 갖는다. 워크스테이션 1은 두 기계를, 워크스테이션 2는 한 기계만을 갖는다.

그림 5와 그림 6은 BFU를 구성하는 모델의

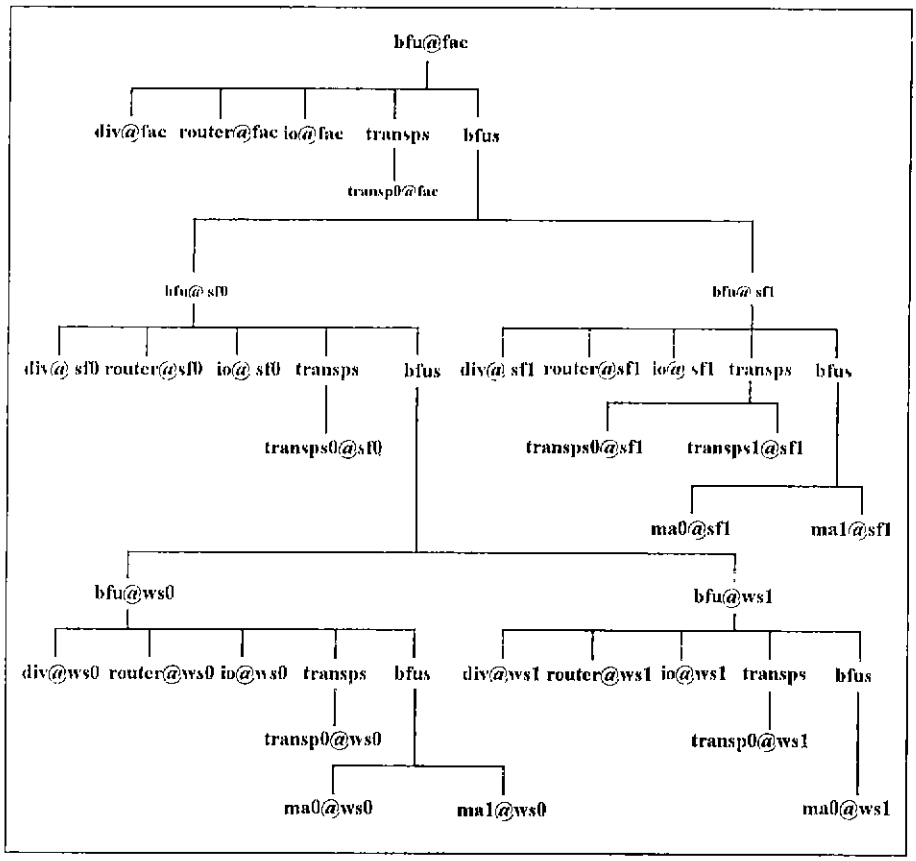


그림 4 다섯개의 BFU를 갖는 PES

결합형태를 보여준다. 시뮬레이션은 제네레이터가 일 식별자(job-id), 일 유형(job-type), 일괄처리 일의 크기(batch-size) 및 작업순서 계획표(sequence schedule)로 구성되어 있는 bfu@fac 모델로 전송 일괄처리 일(transfer batch job)을 보내면서 시작된다. 작업순서 계획표는 그 일에 대하여 순서대로 수행해야 될 작업순서를 갖고 있다. 예를 들면 (a b c d e), (a b c d) 등이며, 여기서 a, b, c, d a 및 e는 서로 다른 유형의 작업들이다. in-b@fac(bfu@fac의 in-b 모델)는 작업순서 계획표를 작업-유형-대-기계테이블에 근거하여 기계리스트로 구성된 순서계획표로 바꾼다. 즉, a는 ma0@ws0, b는 ma1@ws0, c는 ma0@ws1, d는 ma0@sf1 그리고 e는 ma1@sf1으로 각각 바꾼다(그림 4는 시뮬레이션 모델을 생성시켜서 이 기계들이 어디에 속하는 지를 보여주는 프루닝된 엔티티구조이다). 분할기(divider)는 이 전송 일괄처리 일을 분할(fork)한다. 다음 절은 작업의 중첩수행 기법에 의해 주어진 분할로직에 따라 동작하는

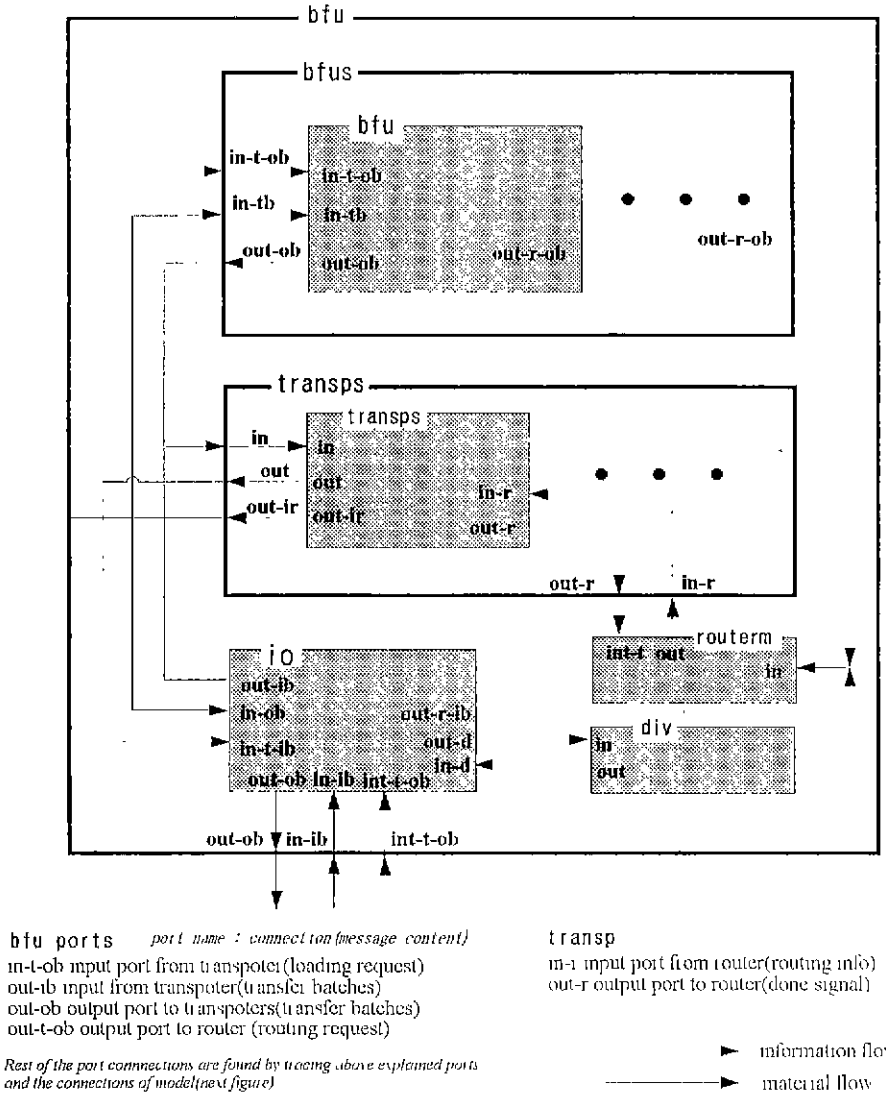
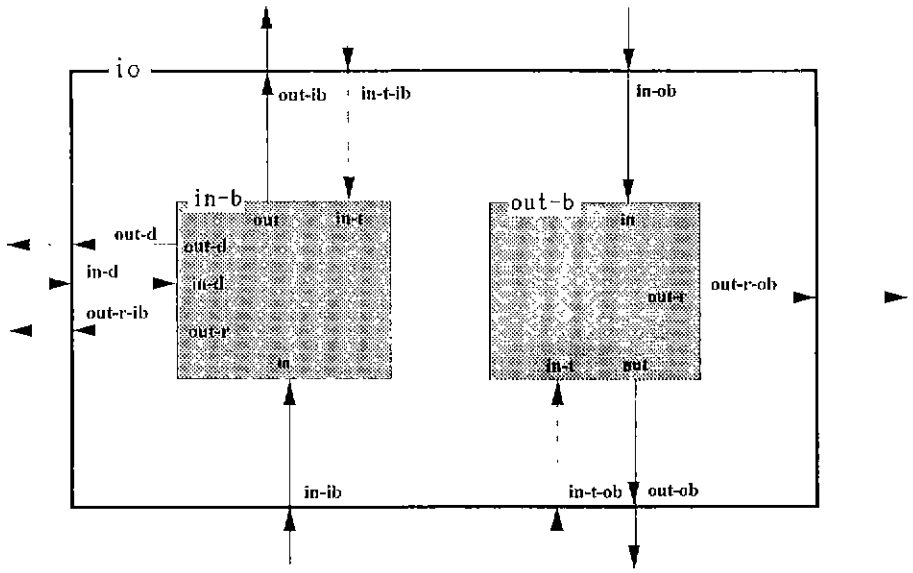


그림 5 BFU의 DEVS 모델

분할기의 성능에 대하여 고찰한다(식 1). 분할된 부분 일괄처리 일들은 운송장치에 의해 작업계획표에 따라 다음 기계로 운송된다. 경로기는 몇가지의 인자(factor)에 근거하여 각 BFU 내에 있는 운송장치의 경로를 설정한다. 분할된 일괄처리 일들은 나중에 그것들이 분할된(forked) 곳과 같은 io모델(io 내의 out-b)상에 모인다(joined). 계획표내의 기계들을 모두 거치게 되면, 일괄처리 일은 bfu@fac 모델(공장레벨 bfu)을 떠나고 bfu@fac의 성능(처리량(throughput), 흐름시간(flow time), WIP 및 평균 도착간격(average inter-arrival time))에 관한 통계치들을 기록하기 위해 측정기(transducer)로 보내어 진다.

전송 일괄처리 일의 크기는 균등분포함수에 의해 결정된다. 제네레이터에서 생성된 최대, 최소 일괄처리 일은 각각 500과 100이다. 일의 유형에 따른 작업 순서계획은 다음 계획들 즉, (a b c d e), (a b d e), (a c d e), (b d e) 및 (a c e) 중에서 같은 확률로 무작위로 선택된다. 표 1은 일의 유형에 따라 기계에 주어진 처리시간(processing time)이 표기되어 있다. 유니트당 처리시간은 2에서 15사이의 범위를 가지며 준비시간(setup time)은 유니트당 처리시간의 10배이다. 나머지 처리시간은 다음과 같다(모든 값들은 임의의 시간단위이다).

- 운송장치 : 100(fac level)



In-b ports *port name port connection(content of message)*
out-d:output to divider(division request)
in-d:input port from divider(divided batches)
out-r:output port to router(routing request)
out:output port to transporter(transfer batches)
in-t:input port from transporter(loading request)
in:input port from higher level transporter(transfer batches)

out-b ports
out-r:output port to router(routing request)
out:output port to higher level transporter(transfer batches)
in-t:input port from higher level transporter(loading request)
in:input port from transporter(transfer batches)

Io ports
port connections and port contents can be obtained from that of in-b and out-b model

: information flow
: material flow

그림 6 IO의 DEVS 모델

80(shop floor level)
30(workstation level)
－경 로 기 : 1
－분 할 기 : 0

표 1 작업종류에 따른 작업시간과 준비시간

Machine name	*job-types*	*setup time*	*processing time per unit*
ma0@ws0	job1	30	3
〃	job2	50	5
〃	job3	60	6
〃	job4	50	5
〃	job5	20	2
ma1@ws0	job1	30	3
〃	job2	100	10
〃	job3	100	10
〃	job4	50	5
〃	job5	40	4
ma0@ws1	job1	30	3
〃	job2	50	5
〃	job3	60	6
〃	job4	50	5
〃	job5	20	2
ma0@sf0	job1	50	5
〃	job2	100	10
〃	job3	100	10
〃	job4	100	10
〃	job5	80	8
ma1@sf1	job1	50	5
〃	job2	150	15
〃	job3	100	10
〃	job4	100	10
〃	job5	50	5

All numbers are in time units.

운송장치 경로기는 각 BFU내에 있는 운송장치의 경로를 제어하는 전문가시스템 모델이다. 이것은 하위계층에 있는 BFU들의 출력버퍼(output buffer)와 입력버퍼(input buffer)로 운송장치의 경로를 설정한다. 이러한 DES 모델은 모두 다섯개(각 BFU에 대하여 하나)가 있으며, 추론은 각 레벨에 있는 BFU내의 여러가지 동적인 조건에 따라 이루어진다.

경로설정 로직은 다음과 같다.

roule 1

IF
continue-list is not empty
and transporter position ids not same as the place condidate request came from
and there was a previious request from the place a transporter exits
and destination of previious request is same as the position of current request
and the position of candidate request is not marked as "continuing" (i.e., it is not expecting the second batch after the first batch have been delivered)
THEN
the candidate request selected for routing
and perform extra -pickup

Rule 2:

IF
candidate request is selected
and continue-list is not empty

THEN

the candidate request selected for routing

Rule 3

IF
there is a single request in the continue-list

THEN

select the request as candidate request

Rule 4.

IF

there are moer than one request in continue-list
and there is a request , among the request in the continue-list, from the place a transporter exists

THEN

select the request from the place transporter is now (if more than one then first in the continue-list) as a candidate request

Rule 5.

IF

there are moer than one request in the continue-list
THEN
select the first request in the list as a candidate request

Rule 6.

IF

candidate request is selected
and continue-list is empty
and destination of candidate request is not marked as "continuing"

THEN

the candidate request selected for routing

Rule 7:

IF

candidate request is selected
and continue-list is empty
and destination of candidate request is not marked as "continuing"

THEN

remove the candidate request from non-continue-list
and mark the candidate request as not selected

Rule 8

IF

continue-list is empty
and there is only one non-continue-list

THEN

select the request as a candidate request

Rule 9:

IF

continue-list is empty
and there are more than one request in non-continue-list
and there is a request, among the request in non-continue-list, from the place a transporter exists

THEN

select the request from the place a transporter exists (if more than one then first in non-continue-list) as a candidate request

Rule 10:

IF

continue-list is empty
and there are more than one request in non-continue-list

THEN

select the first request in non-continue-list as a candidate request

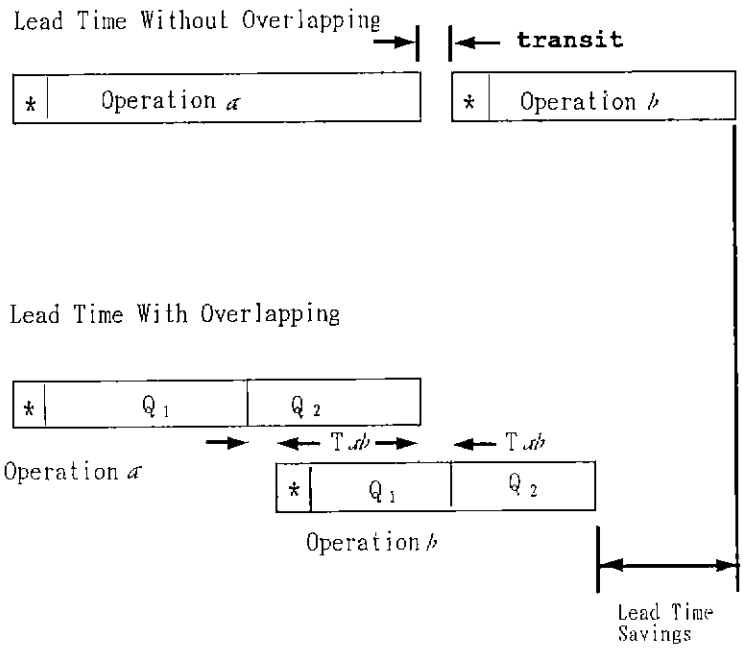
3. 작업의 중첩수행 알고리듬

프랙탈 아키텍츄어의 프레임워크하에서 작업의 중첩수행이라 불리는 간단한 타스크에 대하여 고찰하기로 한다. 작업의 중첩수행은 생산활동관리(production activity control : PAC)의 한 부분이다. PAC의 기능은 생산활동이 계획대로 수행되도록 하는 것이다[27]. 하나의 잡샵(job shop)은 보통 같은 영역에서 유사한 기능들을 수행하는 데 사용되는 설비(예컨대, 재조셀, 워크셀, 워크스테이션)가 묶어져서 물리적으로 배치된다. 전형적으로 플랜트에서 동시에 처리되는 주문들은 많지만 같은 경로를 갖는 것은 비교적 적다. PAC의 첫 단계인 샵플로의 스케줄링은 납기일(due date)을 맞추고 기계의 이용효율을 극대화하며, 준비시간(setup time)을 최소화하고 리이드 타임을 최소화 하거나 이들 사이의 모든 조합을 위해 작업센터(work center)에 일들을 할당한다[27, 13, 28].

그림 7에 나타낸 작업의 중첩수행은 주어진 계획에 따라 전송되어진 일괄처리 일에 대하여 수행되는 하나의 기법이다. 이 기법은 일괄처리 일을 두개 이상의 부분 일괄처리 일들로 나누어 적어도 두개의 연속되는 작업을 직접 연결함으로써 리이드 타임을 줄이는 데 사용된다(한 작업은 다른 하나의 작업이 끝난 즉시 수행된다). 작업의 중첩수행은 준비시간이 요구될 때 제조셀내에서 흔히 적용되는 알고리듬이다. 중첩수행의 장점은 그림 7에 나타낸 바와 같이 개개의 처리 시간을 줄임으로써 전체 제조리이드 타임을 줄일 수 있다는 것이다. 통상적으로 작업의 중첩수행으로 얻을 수 있는 주된 이득은 하나의 일괄처리 일이 작업들의 사이에 있는 큐에서 대기하는 시간-이 시간은 가끔 전체 처리시간의 수배에 달하기도 한다-을 줄일 수 있음으로써 비롯된다. 또한 제조리이드 타임의 감소와 함께 WIP의 감소를 가져오게 되고 제품 납기일을 맞출 수 있는 능력이 향상된다. 반면에 단점으로는 일괄처리 일의 수와 자재의 이동횟수가 두배로 늘어남에 따라 추가 관리비용이 요구된다는 것이다[18, 27].

이제, 두가지의 연속되는 작업 A와 B를 생각하자. 이때 작업의 중첩수행은 다음과 같이 이루어 진다.

1. 한 일괄처리 일은 적어도 두개의 부분 일괄처리 일(전송 일괄처리 일)로 분할한다.



*: setup time

Q = total lot size = $Q_1 + Q_2$

T_{ab} = transit time between Operation a and b

그림 7 작업중첩수행(operation overlapping)

2. 첫번째 부분 일괄처리 일이 작업 A를 마치자 마자 즉각적인 처리를 위해 작업 B로 옮겨진다.
3. 두번째 부분 일괄처리 일에 대하여 작업 A가 수행되는 동안 첫번째 일괄처리 일에 대하여 작업 B가 수행된다.
4. 두번째 부분 일괄처리 일에 대하여 작업 A의 수행이 완료되었을 때는 즉시 작업 B로 옮겨진다.

만약 작업 B가 실질적으로 작업 A보다 피이스당 시간(time per piece)이 작다면, 첫번째 일괄처리 일은 작업 B에서 생기는 유휴시간(idle time)을 없애기 위해 충분히 커야 한다. 따라서 일괄처리 일의 최소크기를 계산하면 다음과 같으며 그림 7에서와 같이 Q_1으로 나타낸다[27].

$$Q_1 >= (QP_a - S_b)/(P_b + P_a) \quad (1)$$

단, Q = 전체 일괄처리 일의 크기 = $Q_1 + Q_2$
Q_1 = 첫번째 일괄처리 일의 최소크기
Q_2 = 두번째 일괄처리 일의 최대크기
S_b = 작업 B의 설정시간
P_a = 유니트당 처리시간, 작업 A
P_b = 유니트당 처리시간, 작업 B

4. 시뮬레이션 결과

두가지 서로 다른 시뮬레이션을 수행하였다. 하나는 작업을 중첩수행시키는 경우이고 또 하나는 그렇지 않은 경우이다. 일의 도착간격 프로세스(job interarrival process)를 위해 지수분포함수를 사용하였다[14]. 일의 도착빈도가 성능에 어떠한 영향을 주는가를 관측하기 위

표 2 FLow time의 90% 신뢰도구간과 평균값

mean interarrival time (exponential distribution)	*w/o overlapping* avg. [conf. interval]	*w/o overlapping* avg. [conf. interval]
1000	21906 [14890, 28912]	21477 [15902, 27033]
2000	11467 [5205, 17728]	12528 [7093, 17962]
3000	7316 [2975, 11656]	9115 [5292, 12937]
4000	5555 [3951, 7158]	7448 [5642, 9253]
5000	4850 [4140, 5559]	6790 [6702, 6877]

All numbers are in time units.

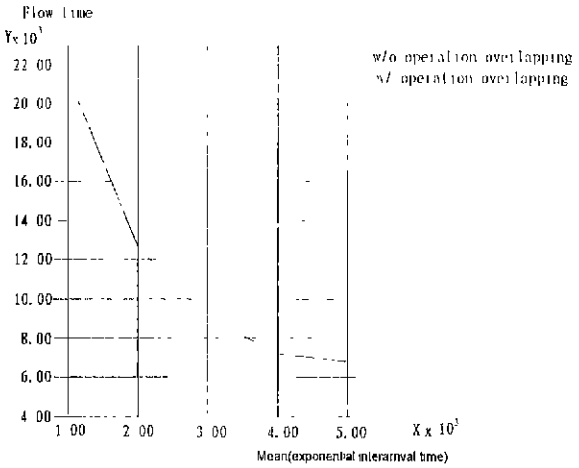


그림 8 Flow time의 비교

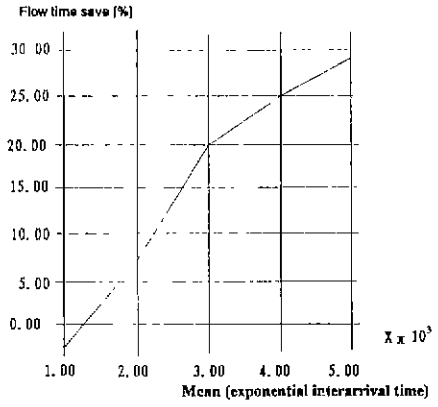


그림 9 Flow time의 단축

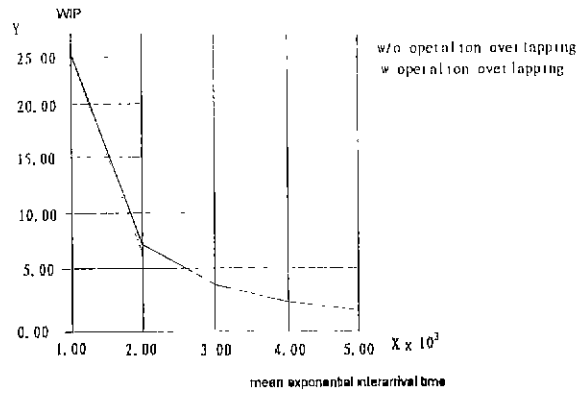


그림 10 WIP의 비교

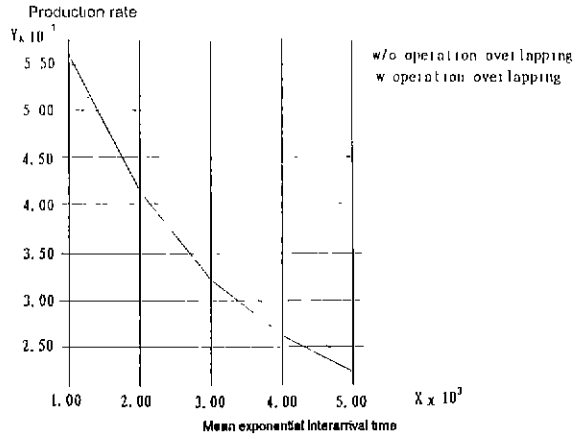


그림 11 생산율의 비교

해 5가지의 평균 도착간격(mean interarrival time) : 1000, 2000, 3000, 4000 및 5000을 사용하였다. 그림 8은 작업의 중첩수행이 있는 경우와 없는 경우에 대하여 관측한 리이드 타임을 비교한 것이다. 90% 신뢰구간은 표 2에 나타내었다.

그림 9는 작업의 중첩수행 기법이 적용될 때 절약되는 흐름시간을 보여준다. 그래프에서 보는 바와 같이 도착간격이 커질 때에는 흐름시간이 약 30% 절약된다는 것을 알 수 있다. 중첩수행을 하는 경우의 흐름시간은 도착빈도가 증가 할수록 중첩수행하지 않는 경우의 리이드 타

임과 더욱 가까워 진다. 이렇게 수렴하는 주된 이유는 도착빈도가 증가 할수록 기계의 사용효율이 증가하여 결국은 증가된 사용효율의 이점이 없어지기 때문이다. WIP와 생산율(production rate)에 대한 비교는 그림 10과 그림 11에 각각 나타내었다.

5. 결 론

본 고에서는 유연생산 시스템내에서의 일괄처리 일들을 작업중첩 알고리듬에 의해 분할하고 이들 분할된 일괄처리 일들을 운송하는 운송장치의 경로기(transporter router)를 전문가시스템으로 구성하여 이들 기법이 전체 시스템의 생산성에 어떠한 영향을 주는가를 측정할 수 있는 시뮬레이션 모델링 예를 제시하였다. 이 예를 통하여 공장내의 운송활동을 제어하기 위해 제안한 전문가 시스템의 정확성(correctness)과 분할 알고리듬을 검증하는데 시뮬레이션이 잘 사용될 수 있음을 설명하였다. 또한 시뮬레이션을 통해 도출한 연구조사가 생산시스템의 성능향상을 기대할 수 있는 측면를 알려줄 수 있다는 것도 예증하였다. 따라서 이러한 지식을 전문가시스템 제어기의 규칙에 반영하고 중첩수행과 같은 생산율에 민감하게 영향을 주는 기법이 도움이 된다고 입증될 경우 실제 현장 투입할 수 있도록 하는 것이 바로 다음에 해야 할 일이다. 심지어 시뮬레이션은 전문가시스템의 제어하에서 온라인으로 수행되어 그 결과들을 현장상황에 바로 반영하여 공장이 필요로 하는 또 다른 유리한 측면들을 얻어 낼 수 있다.

작업의 중첩수행과 이에 따른 적절한 운송제어로 얻을 수 있는 첫번째 이점은 흐름시간의 감소이며 그에 이은 WIP의 감소이다. 프랙탈 아키텍츄어의 경우에 얻을 수 있는 두번째 이점은 자명하다. 즉, 더욱 작은 운송장치들이 사용될 수가 있다는 것이다. 공장에서 샵플로, 워크스테이션 그리고 기계로 계층을 따라 내려가면서 전송 일괄처리 일의 크기가 줄어 들기 때문에 하위 레벨에서는 좀 더 작은 용량의 운송장치가 사용될 수 있다. 작업을 중첩수행 하지 않는 경우, 일괄처리 일들의 크기는 그것들이 처리됨에 따라 변하지 않고 그대로 있다. 그것은 하위레벨에서의 운송장치들도 공장레벨에서와 같이 커야 한다는 의미가 된다. 그러나 각 BFU에서 일괄처리 일들을 분할(fork)하고 결합(join)하기 위한 제어비용이 추가된다는 단점도 있다.

참 고 문 헌

[1] S. D. Wu, "Artificial Intelligence and Scheduling Applications," *Artificial Intelligences ; Manufacturing Theory and Practice (ed. S. T. Kumara), 1989.*

[2] H. Pierreval and H. Ralambondrainy, "A Simulation and Learing Technique for Generating Knowledge about Manufacturing Systems Behavior," *Artificial Intelligence in Operational Research, Macmillan (ed. by G. I. Doukidis and R. J. Paul), 1992.*

[3] A. Kusiak, "Expert Systems and Optimization in Automated Manufacturing Systems," *Artifical Intelligence ; Manufacturing Theory and Practice (ed. by S. T. Kumara), 1989.*

[4] H. Matsuo, J. S. Shang, and R. S. Sullivan, "A Knowledge-Based System for Stacker Crane Control in a Manufacturing Environment," *IEEE Transactions on Systens, Man, and Cybernetice,* vol. 19, no. 5, 1989.

[5] P. J. O'Grady and K. H. Lee, *An Intelligent Cell Control System for auto-mated manufacturing.* Taylor&Francis, 1989.

[6] J. Erschler and P. Esquirol, "Decision-aid in job shop scheduling : A knowl-edge based approach," in *Proceedings of the 1986 IEEE International Conference on Robotics and Automation,* San Francisco, California pp. 1651-1656, 1986.

[7] M. S. Fox, "Constraint-directed search : A case study for job-shop schedul-ing," Ph. D. dissertation, Carnegie-Mellon University, Pittsburgh, Penn., 1983.

[8] S. Subranmanyam and R. G. Askin, "An expert system approch to scheduling in Flexible Manufacturing System," *Flexible*

Manufacturing Systems: Meth-ods and Studies (ed. by Kusiak), North Holland, 1986.

[9] R. J. Paul and G. I. Doukidis, "Operational Research Approach to Artificial Intelligence in Production Planning and Scheduling," *Artificial Intelligence in Operational Research, Macmillan (ed. by G. I. Doukidis and R. J. Paul),* 1992.

[10] P. Duchessi and R. M. O'Keefe, "A knowledge-based Approach to Production Planning," *Artificial Intelligence in Operational Research, Macmillan (ed. by G. I. Doukidis and R. J. Paul),* 1992.

[11] M. S. Steffen, "A Survey of Artificial Intelligence-Based Scheduling System," *in 1986 Fall Industiral Engineering Conference Proceedings,* pp. 395-405, 1986.

[12] K. Kempf, B. Russell, S. Sidhu, and S. Barrett, "AI-Based Schedulers in Manufacturing Practive: Report of a Panel Discussion," *AI Magazine (special issue),* 1990.

[13] A. Kusiac, *Intelligent Manufacturing Systems.* New Jersey : Prentice Hall, 1990.

[14] M. L. Law and W. D. Kelton, *Simulation Modeling & Analysis.* second, McGraw-Hill, inc., 1991.

[15] R. G. Askin and C. R. Standridge, *Modeling and Analysis of Manufacturing System.* John Wiley & Sons, Inc, 1993.

[16] T. H. Cho, "A Hierarchical, Modular Simulation Environment for Flexible Manufacturing System Modeling," Ph. D. dissertation, University of Arizona, Tucson, AZ, 1993.

[17] B. P. Zeigler, T. H. Cho and J. W. Rozenblit, "A Knowledge-based Simulation Environment for Hierarchical Flexible Manufacturing," *IEEE Trans. Sys. Man & Cybernetics,* vol. 25, no. 10, Sep., 1995.

[18] S. A. Melnyk and P. L. Cater, *Production Activity Control.* Homewood, Illinois : DOW JONES-IRWIN, 1987.

[19] B. P. Zeigler, *Object-Oriented Simulation with Hierarchical, Modular Models.* San Diego, CA, USA : Academic Press, 1990.

[20] J. W. Rozenblit, J. W. Hu, T. G. Kim, and B. Zeigler, "Knowledge-based Design and Simulation Environment (KBDSE): Foundation Concepts and Implementation," *Journal of the Operational Research Society,* vol. 41, no. 6, 1990.

[21] B. P. Zeigler, *Multifacetted Modeling and Discrete Event Simulation.* Orlando, FL, USA : Academic Press, 1984.

[22] A. Yonezawa and M. Tokoro, *Object-Oriented Concurrent Programming. Cambridge,* MA : MIT Press, 1987.

[23] S. E. Keene, *Programming in common Lisp Object-Oriented Systems.* Ma : Addison-Wesley, 1988.

[24] S. R. Alpert, S. W. Woyak, H. J. Shrobe, and L. F. Arrowood, "Object-Oriented Programming," *IEEE Expert,* pp. 6-7, 1990.

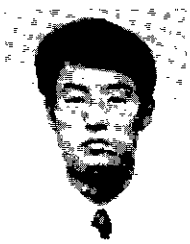
[25] T. G. Kim, "A Knowledge-based environment for hierarchical modeling and simulation," Ph. D. dissertation, University of Arizona, Tucson, AZ, 1988.

[26] T. M. Tirpak, S. M. Daniel, J. D. LaLonde, and W. J. Davis, "A Note on a Fractal Architecture for Modeling and Controlling Flexible Manufacturing System," *IEEE Trans. on Systems, Man and Cybernetics,* vol. 22, pp. 564-567, 1992.

[27] D. W. Fogarty, J. H. Blackstone, Jr, and T. R. Hoffmann, *Production & Inventory Management.* South-Western Pub. Co., second ed., 1991.

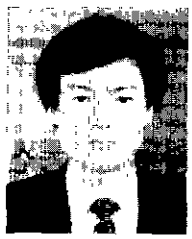
[28] P. Marchall, "The prospects for FMS in UK Industry," in *Proceeding of the 1st International Conference on Flexible Manufacturing System,* Brighton, U. K., pp. 71-76, 1982.

조 대 호



1983 성균관대학교 전자공학과 학사
1987 University of Alabama 전자공학과 석사
1993 University of Arizona 전자 및 컴퓨터 공학과 박사
1993~현재 경남대학교 전자계산학과 전임강사
관심분야 : 모델링 및 시뮬레이션, 인공지능, 지능형 시스템

하 경 재



1980 성균관대학교 전기공학과 학사
1982 성균관대학교 전기공학과 석사
1989 성균관대학교 전기공학과 박사
1980~1984 KIET 연구원
1993~현재 경남대학교 전자계산학과 부교수
관심분야 : 공장자동화, 인공신경망

Future Trends of Distributed Computing Systems

- **일자** : 1995년 8월 28일(월)~30일(수)
- **장소** : 제주프린스호텔
- **주최** : IEEE/CS, KISS 컴퓨터시스템 연구회
- **문의** : 양승민(숭실대 컴퓨터학부) T.02-820-0912
  김문희(건국대 전자계산학과) T.02-450-3859