Journal of the Korean Institute
of Industrial Engineers
Vol.22, No.4, December, 1996

Heterarchical SFCS를 위한 가공기계의 Planner 모듈 개발*

Development of a planner of processing equipments for heterarchical SFCS

김화진** · 조현보*** · 정무영***

Hwa-Jin Kim** · Hyun-Bo Cho*** · Moo-Young Jung***

Abstract

A common control model used to implement computer integrated manufacturing(CIM) is based on the hierarchical decomposition of the shop floor activities, in which supervisory controllers are responsible for all the interactions among subordinates. Although the hierarchical control philosophy provides for easy understanding of complex systems, an emerging manufacturing paradigm, agile manufacturing, requires a new control structure necessary to accommodate the rapid development of a shop floor controller. This is what is called autonomous agent-based heterarchical control. As computing resources and communication network on the shop floor become increasingly intelligent and powerful, the new control architecture is about to come true in a modern CIM system.

In this paper, heterarchical control is adopted and investigated, in which a controller for a unit of device performs three main functions - planning, scheduling and execution. Attention is paid to the planning function and all the detailed planning activities for heterarchical shop floor control are identified. Interactions with other functions are also addressed. In general, planning determines tasks to be scheduled in the future. In other words, planning analyzes process plans and transforms process plans into detailed plans adequate for shop floor control. Planning is also responsible for updating a process plan and identifying/resolving replanning activities whether they come from scheduling or execution.

Keywords: shop flow control system, heterarchical control, process planning, planner

* 이 논문은 1995년도 한국학술진흥재단의 공모과제 연구비의 일부지원으로 연구되었음.

** LG-EDS

*** 포항공과대학교 산업공학과/제품생산기술연구소

1. 서 론

SFCS(Shop Floor Control System)의 제어 구조(control architecture) 대부분은 계층적(hierarchical) 제어 모델이 적용되고 있는 것이 일반적인 현상이다. 계층적 제어 모델은 생산 시스템의 구조를 상하관계(master-slave relationship)로 구성하여, 생산에 필요한 모든 활동들이 상위 단계(supervisor)에서 계획, 스케줄링되어 하위 단계로 지시된다. 하위 단계에서는 명령의 완성 정도, 생산 현장의 상태 등을 상위 단계로 보고한다. 이러한 계층적 제어 모델은 복잡한 생산현장의 구조를 쉽게 제어할 수 있게 하며 방대한 생산 현장도 계층적으로 분해하여 제어함으로써 단지 몇개의 소프트웨어 모듈만으로도 제어가 가능하도록 한다. 그러나 계층적 제어 모델에서는 중간 계층에 있는 어떤 하나의 소프트웨어 모듈이 작동되지 않게 되면 그 하위에 있는 모든 관련시스템은 사용할 수 없게 되어 전체를 통합하는 데에 어려움이 따르게 된다[3, 5, 9, 10]. 또한, 제어기 및 통신망 기술의 급속한 발달로 인하여 기존의 계층적 제어 모델로서는 이러한 기술의 발전을 효율적으로 활용하기 힘들다는 문제점도 내포하고 있다.

본 연구에서는 생산 시스템의 각 구성 요소를 독립적인 단위로 구성하는 heterarchical 제어 모델을 적용하고자 한다. 이는 계층적 제어 모델의 상대적인 개념으로서, 생산 시스템의 목적 즉 제품을 적기에 생산해 내기 위하여 각 구성 요소들간의 대화 및 협상에 의해서 생산 활동이 이루어지는 제어 모델을 말한다. 생산 현장의 구성요소들은 여러 가지로 분류되며 그 중 직접 가공을 담당하는 processing equipment가 주 관심 대상이 된다. Processing equipment의 제어 모델은 3개의 주요 기능 – planning, scheduling, execution – 을 가지며, 본 연구의 주요 목적은 이 중 planning 기능을 담당하는 planner를 개발하는 데 있다. 부수적으로 heterarchical 제어 구조를 적용하기 위하여 다루어야 할 문제들을 살펴보고, 시스템(shop)을 구성하는 구성 요소(agent)들이 어떻게 분류되어야 하는지를 제안한다. 그리고 생산시스템을 운영하는 데 필수적인 공정계획 표현 모델을 제시하고 그 모델을 토대로 planner가 수행해야 할 역할 및 해법을 제안한다.

2. 기존연구의 고찰

복잡한 생산시스템을 효율적으로 제어하기 위해서 SFCS의 구조에 관한 다양한 연구가 수행되어 왔다. 이러한 연구는 크게 두 개의 범주로 분류될 수 있는데, 첫째는 shop을 이루고 있는 장치(device)들은 하나의 컨트롤러가 부착되어 작동되는데 이들 물리적인 장치들을 구성하는 방법론에 관한 연구들이다. 둘째로는 shop에서 생산 활동이 이루어지는 동안 많은 event들이 발생하며 이들을 담당할 기능들이 필요하게 되고 이 기능들은 다시 몇 개의 그룹으로 묶을 수 있는데 이와 같은 기능적구조에 관한 연구들이다.

2.1 SFCS의 제어 구조

지금까지 SFCS를 위한 다양한 제어 구조들이 연구되고 제시되어 왔는데 그림 1에서 보는 바와 같이 크게 centralized, hierarchical, 그리고 heterarchical 제어 구조로 분류될 수

있다. 그림 1에서 보는 바와 같이 centralized 제어 구조는 주 컴퓨터 한 대가 shop에서 이루어지는 모든 생산 활동을 계획하고 관련된 정보들의 처리 및 유지를 담당하는 시스템을 말한다. Hierarchical 제어 구조로 발전하기 전 단계라고 할 수 있는 이 구조는 (1) 필요한 모든 관련 정보들을 하나의 컨트롤러에서 처리하기 때문에 시스템 전체 차원의 최적화(global optimization)가 가능하고 (2) 다른 제어 구조들에 비해 컨트롤러 비용을 줄일 수 있고 관리가 용이하여 경제적이며 (3) 필요한 시스템의 정보 및 상태를 쉽게 얻을 수 있다는 장점을 가지고 있다[3, 5]. 그러나 이 구조는 시스템의 규모가 커지는 경우, 다른 구조(예를 들면, 분산 제어 시스템)에 비해 상황에 대처하는 능력이 떨어진다는 치명적인 단점을 안고 있다. 즉, 중앙의 주 컴퓨터 한 대로써 모든 정보를 처리하기 때문에 실시간 제어가 어려워진다는 단점이 있다. 또한, 주 컴퓨터에 이상이 생겼을 경우에는 전체 시스템이 중단된다는 문제가 발생한다. 그리고 시스템 내의 모든 관련 데이타를 다루기 때문에 제어 소프트웨어의 유지, 보수 및 확장이 힘들다는 단점도 내포하고 있다.

그림 1의 (b)에서 보는 바와 같이 hierarchical 제어 구조는 시스템을 상부/하부 구조의 여러 계층으로 나누어서 구성한다. 시스템을 구성하는 계층은 일반적으로 shop, workstation, equipment의 세 단계로 나뉘며, 상부 구조에서 의사결정이 이루어진 후 하부 구조로 내려오게 된다. 이 모델이 갖는 장점으로는 (1) 제어 구조가 계층별로 구성되어 있기 때문에 소프트웨어를 개발하는 데 있어 모듈화가 가능하여 관리나 확장이 용이한 편이고 (2) 각각의 계층별로 역할이 분담되어 있기 때문에 상황에 대처하는 반응시간이 빠르다는 장점이 있다[3, 5]. 그러나 (1) 하위 계층은 상위 계층에 의존하게 되므로 어떤 부분에 이상이 생기면 그 하위 단계의 컨트롤러들도 기능이 마비되어 버리고 (2) 시스템을 설계하는 단계에서 이러한 구조가 초기에 확정, 설치되므로 미래에 예측하지 못한 변화에 대처해서 수정, 확장하기 어렵다. 즉, 기존의 이러한 제어 구조가 적용된 시스템에서는 새로운 변화가 발생할 시에 전체 시스템의 구조에 변동이 생기게 되어 유연성이 떨어진다는 단점을 갖고 있다.

한편, 그림 1의 (c)에서 보는 바와 같은 heterarchical 제어 구조는 hierarchical 제어 구조의 단점을 극복하기 위하여 제안되었는데, 앞서 언급한 바와 같이 통신망(communication network), 분산 컴퓨팅 시스템(distributed computing system)분야와 컨트롤러 자체의 기술 발전이 그 배경이 되고 있다. 이 모델의 특징은 시스템 내에 계층적 구조가 존재하지 않는 대신에 시스템을 구성하는 구성원들(controllers)사이의 협력(negotiation)을 통해 원하는 목적을 달성한다는 점에 있다. 즉, 글로벌 정보를 최소화시키고 local 데이터베이스들을 최대한 활용하며, 컨트롤러간의 통신(communication)을 통한 협력에 의해 모든 의사결정이 내려진다[5]. 이 구조에서는 각 컨트롤러들이 독립적으로 작업을 수행하기 때문에 시스템의 확장이 용이하며, 어느 한 컨트롤러의 이상 발생이 전체 시스템에 미치는 영향을 최소화시킬 수 있다. 또한, 현재의 shop 상태를 기반으로 의사결정이 이루어지므로 실시간 제어가 가능하며, 적절한 단위

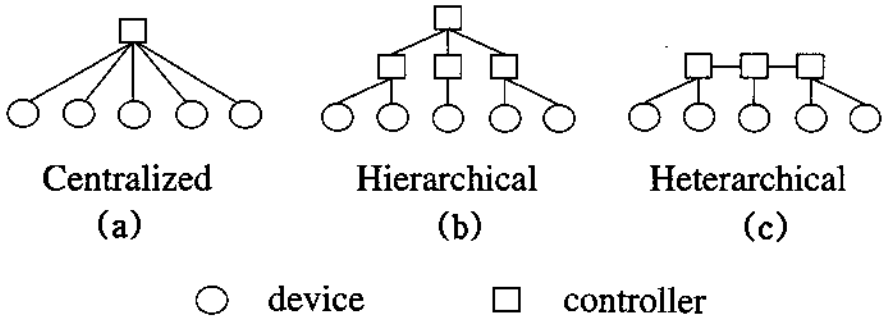

그림 1. SFCS 제어 구조의 분류

로 모듈화됨으로써 제어 소프트웨어의 복잡성이 줄어들어 유지/관리가 용이하다는 장점을 지닌다. 단점으로는 시스템 전체 차원의 최적화가 이루어지지 않을 수 있으며(local optimization), 컨트롤러 등의 기술적 제약이 있을 수 있다[3, 5].

2.2 SFCS의 기능적 구조

SFCS에서 발생하는 복잡한 event들을 담당할 기능들의 분류를 몇 가지로 할 것인가는 시스템 구현을 위해 관련된 전문가나 사용되는 방법론 등에 따라서 달라지게 되며, 각 기능들의 역할 또한 상황에 따라 달리 정의된다. 이들 기능들은 그림 2에서 보는 바와 같이 여러 형태로 분류할 수 있으나 SFCS에서

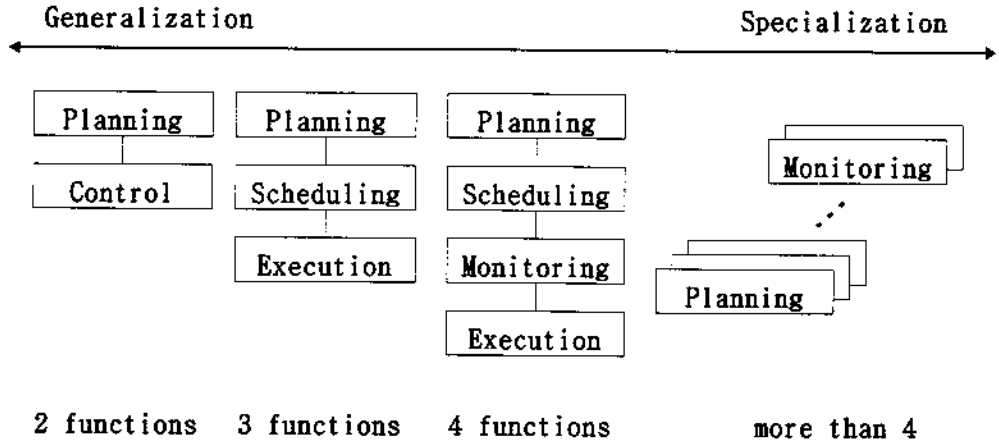

그림 2. 기능 분류 방법

이루어지는 활동은 크게 의사결정(decision-making function)과 결정된 사항을 제대로 수행하는 기능(control function)으로 분류할 수 있다. 따라서 이 두 가지 기능을 어느 정도까지 세분화시키느냐에 따라 시스템을 구성하는 기능의 수가 결정된다[3].

3. Heterarchical SFCS의 개요

3.1 Agent의 정의 및 분류

Heterarchical 제어 모델을 구축하기 위해서는 SFCS를 구성하는 구성원들(이하 본 논문에서는 이를 agent라 칭한다)을 먼저 정의할 필요가 있다. Agent란 의사 결정을 내릴 수 있는 적절한 단위로서 서로 독립적으로 작동하며 통신망을 통해 서로간의 협력으로 생산활동을 수행하는 생산자원의 집합체이다[21]. Agent들은 표 1과 같이 분류한다. 그러나 이러한 agent들의 분류는 고정적인 것이 아니라 상황에 따라 달라진다. 그림 3의 (a)에서와 같이 단순히 부품을 집어서 놓는 기능만 가진 로봇의 경우는 독립된 agent로 간주되지 않지만, (b)의 경우처럼 수치제어 기계들(CNC 1, CNC 2)과 정보 교환을 통해 의사결정이 이루어져야 하는 경우는 하나의 독립된 agent로 분류된다.

3.2 Agent의 기능 및 구성

Agent가 수행하는 기능을 모듈화하는 분류 방법에는 앞서 언급한 바와 같이 여러 경우가 있을 수 있는데, 본 논문에서는 그림 4에서와 같이 planning, scheduling, execution의

표 1. Agent의 분류

종 류	역 할
Processing equipment	소재의 가공을 담당하며 수치제어 기계나 머시닝 센터 등이 해당
AGV(무인 반송차)	물류의 흐름을 담당
컨베이어	물류의 흐름을 담당
로봇	부품의 load/unload 및 이송을 담당
AS/RS(자동 창고)	소재 및 제품을 저장하며 이들에 대한 정보를 관리
공용 버퍼	부품의 임시 저장소 역할
Assembly equipment	부품의 조립을 수행
Inspection equipment	제품의 검사를 담당
InitMon(Initiating & Monitoring) agent	SFCS의 초기화 및 작업 진도 상황에 대한 모니터링을 담당한다. 즉, 부품 가공 정보, 조립 정보 등 다른 agent들이 필요로 하는 정보를 생산 시작 단계에서 분배하는 역할을 하며, 생산 활동 중 agent들로부터 이상 발생이나 작업 진도 상황에 대한 메시지를 전달 받아 관리자에게 표시해 주는 기능을 수행
Resource manager	shop내의 생산 자원 및 생산 자원들 사이의 관계 등에 대한 정보의 유지/관리를 담당

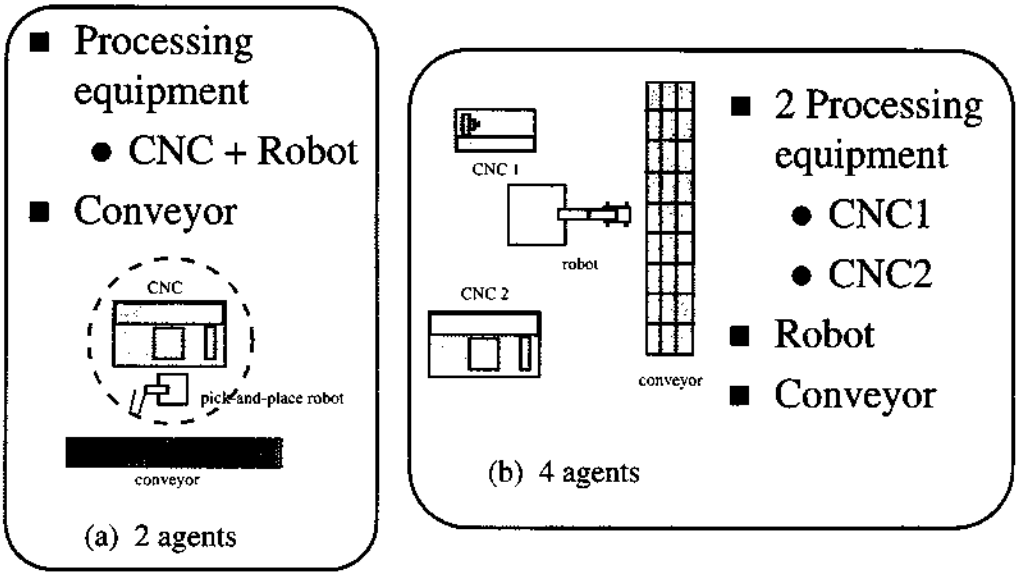

그림 3. Agent의 예

세 가지로 분류한다. 모든 agent들의 경우에 있어서 planning, scheduling, execution의 일반화된 분류 기준을 다음과 같이 언급할 수 있다[21]. Planning 모듈은 SFCS에서 발생하는 의사 결정 문제들을 일부분 해결하여 문제의 범위를 일차적으로 좁히는 역할을 한다. 그런 후, scheduling 모듈이 해결해야 할 문제들을 생성해 냄으로써 결과적으로는 의사결정 기능을 적절히 분배하여 어느 한 모듈에 과도한 부하가 걸리는 것을 방지한다. Scheduling 모듈은 planning 모듈에서 넘어온 문제들을 이용하여 실제로 시스템의 활동이 이루어지기 위하여 execution 모듈이 수행해야 할 작업(task)들을 생성해 낸다. 한편, execution 모듈은 scheduling 모듈에서 생성된 작업들을 수행하기 위하여 시스템 내의 장비들을 제어하며 현재의 작업상태를 모니터링하는 등의 기능을 담당한다.

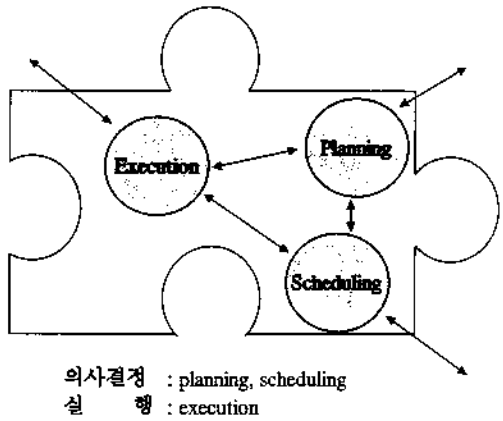

그림 4. Agent의 세가지 기능

3.3 Heterarchical SFCS의 구성

제품을 생산하기 위해서는 우선 공정 계획을 수립해야 하고 이때 작성된 정보는 그림 5에서 보는 것처럼 InitMon agent를 통해 SFCS 내로 전달된다. 즉, InitMon agent는 제품을 생산하기 위하여 SFCS를 초기화시키며

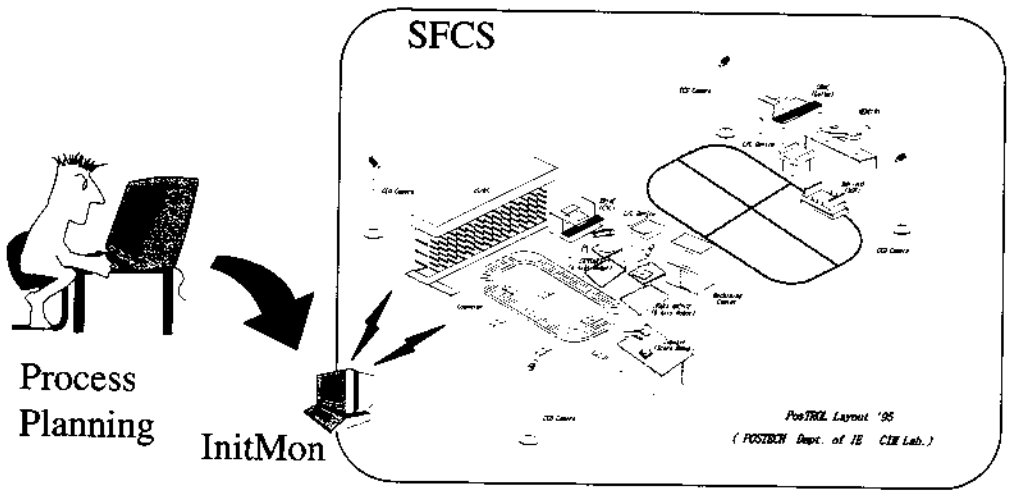

그림 5. 전체적인 구성도

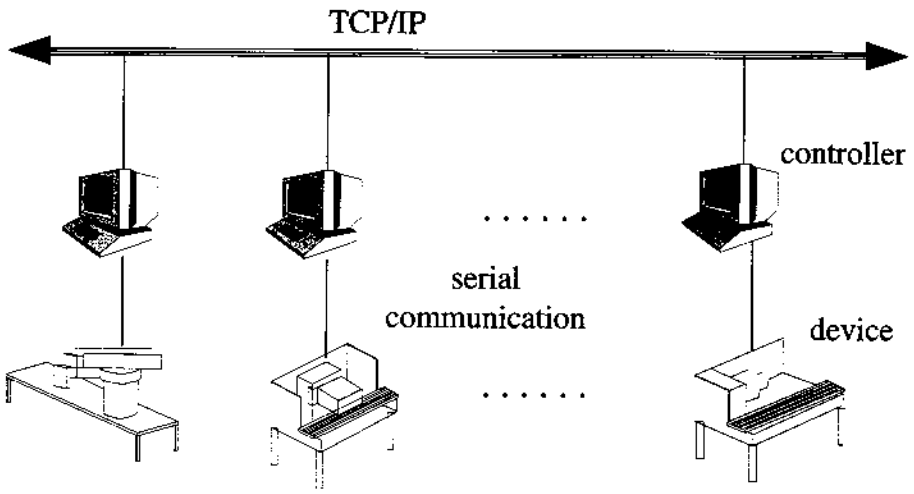

그림 6. SFCS 내의 통신망 구성

시스템 관리자와 SFCS 사이의 교량 역할도 수행한다. InitMon agent로부터 필요한 정보를 전달받은 순간부터 SFCS에서는 본격적인 생산 활동이 이루어지게 된다. 생산 활동이 제대로 수행되기 위해서는 앞에서 언급한 것처럼 각 agent들간의 통신은 필수적인 요인이며 SFCS의 통신망을 어떻게 구현할 것인가 하는 문제는 적용현장에 따라 달라질 수 있다. 본 논문에서는 SFCS의 통신망을 그림 6과 같이 구성하며 agent들간의 통신은 TCP/IP 규약을 통해 이루어지고 실제 장비들의 제어는 직렬(serial) 통신을 통해 수행되도록 한다.

4. 공정 계획 표현 모델

공정 계획(process plan)은 shop floor 제어를 위해 필요한 정보들, 즉 부품의 분기, 가공 순서, 소재에 대한 정보, 가공 자원에 대한 정보 등 많은 양의 정보들을 담고 있다. 과거에는 이러한 정보들을 표현하기 위하여 operation chart나 process routing summary 등을 사용하였으나[12], 이와 같은 표현 방법들은 가공 순서, 가공 자원 등에 대한 대안들(alternatives)을 나타내기 어렵다는 단점을 지니고 있다. 따라서 shop floor 컨트롤러가 쉽게 이해할 수 있도록 공정 계획을 표현할 필요가 생기게 된다.

표준화를 목적으로 하거나 자체적인 사용을 위하여 다음과 같은 표현 모델들이 제안되어 왔는데 ISO 공정계획 모델, ALPS (A Language for Process Specification), AND/OR 그래프 등이 있다[13]. ISO 공정 계획 모델의 기본적 구조는 set에 기반을 두는데 공정 계획 정보는 공정계획 활동들(entities)의 set를 이용하여 재귀적으로 정의된다. ISO 모델은 EXPRESS라는 모델링 언어를 이용하여 STEP 파일로써 표현된다[12]. ALPS는 미국의 NIST (National Institute of Standards and Technology)에서 개발되었는데 방향성 그래프를 기반으로 하는 process specification language이다[2]. 한편, Homem DeMello와 Sanderson[8]은 모든 가능한 조립순서를 나타낼 수 있도록 조립 계획(assembly plan)을 AND/OR 그래프로 표현하였는데 공정 계획을 위한 표현 모델이 그 후 제시되었다. AND/OR 그래프를 이용한 공정 계획 표현 모델에서는 노드(node)는 기계에서의 부품 가공에 대한 정보를 표현하고 아크(arcs)는 그들 간의 AND/OR 선후 관계를 나타낸다[3, 16].

이와 같은 공정 계획 표현 모델은 그 중요성이 최근에 와서야 인식되고 있기 때문에 대부분의 모델들은 현재 계속적인 개발 단계에 있는 실정이다. 공정 계획은 processing equipment의 planning이나 scheduling 모듈이 알아야 할 부품 가공 순서나 가공자원(machine, tool)등에 대한 정보를 담고 있어야 하고, 또한 그에 대한 여러 가지 대안을 표현할 수 있어야 한다. 본 논문에서는 공정 계획을 표현하기 위한 방법으로 AND/OR 그래프 개념을 이용한 MFG Feature 그래프를 사용한다.

4.1 MFG Feature 그래프

MFG Feature 그래프는 SPLIT-AND, JOIN-AND, SPLIT-OR, JOIN-OR, MFG-FEATURE와 같이 5가지 노드로 구성하는데 그림 7은 MFG Feature 그래프에 대한 예를 보여주고 있다. 각 노드들에 대한 설명을 요약해 보면 다음과 같다.

SPLIT-AND, JOIN-AND 노드 : 이 두 노드들 사이의 작업들은 선후 관계가 없으므로 어떠한 순서로든지 모두 수행되어야 한다. 그림 7에서 F8과 F9는 F8, F9혹은 F9, F8의 순서로 두 노드가 모두 수행되어야 함을 의미한다.

SPLIT-OR, JOIN-OR 노드 : 이 노드들은 가능한 가공 경로들에 대한 대안을 표현하며, 그 중 한 가지 경로만이 선택되어야 함을 의미한다. 그림 7의 예에서 F5, F6이나 F7, F8, F9의 두 경로 중에서는 하나만이 선택된다.

MFG-FEATURE 노드 : 이 노드는 공구나 머신 등 셋업의 변화 없이 수행될 수 있는 가공 공정의 집합을 의미하며, 가공을 위한 정보 즉 머신, 공구, 절삭 조건(절삭 깊이, 절삭 속도, 이송률), NC 코드 등의 정보를 표현한다. 그림 7에서 F라는 접두어로 시작하는 노드들(F1, F2, ...)이 MFG-FEATURE 노드에 해당된다.

4.2 MFG Feature 그래프를 위한 자료 구조

MFG Feature 그래프의 각 노드들은 그림 8의 (a)와 같은 자료 구조로서 구성되며, 가공 자원(resource)들에 대한 대안 및 후행 노드(successor)의 표현을 위해 각각 (b), (c)의 자료 구조를 사용한다. 그림 8의 자료 구조를 이용하여 그림 9와 같이 MFG Feature 그래프를 표현할 수 있다.

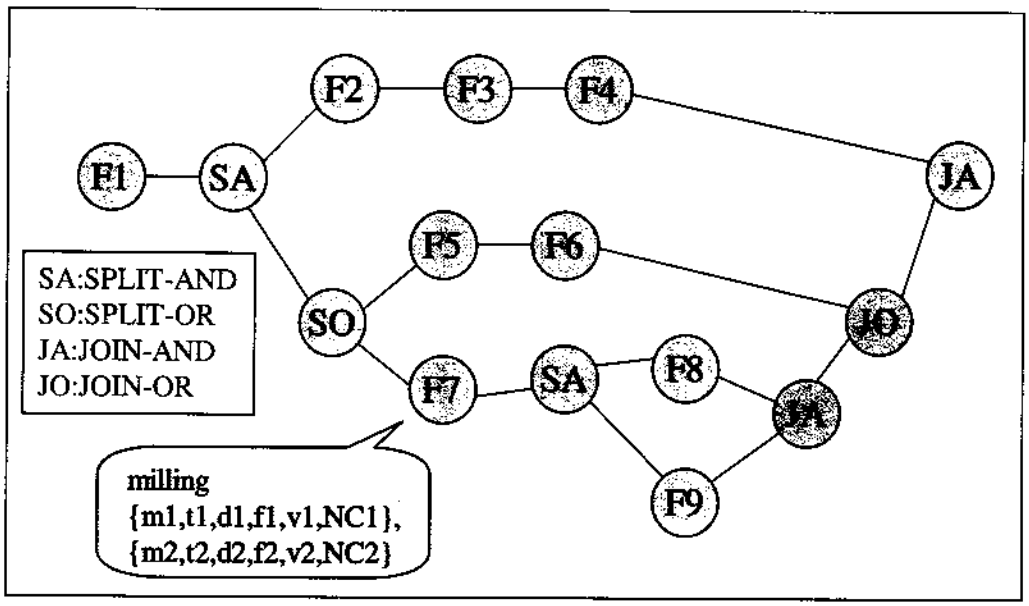

그림 7. MFG Feature 그래프의 예

node ID	*node type	** num.	***pointer to (b)/(c) node	pointer to (a) node

(a) general node

(b) resource list node	resource information	pointer to (b) node

(c) successor list node	pointer to (c) node	pointer to (a) node

* : MFG, AND, OR 중의 하나

*** : if node type = MFG, pointer to (b) node
otherwise, pointer to (c) node

** : 위 ***pointer가 가리키는 리스트의 노드 수

그림 8. MFG Feature 그래프의 자료 구조

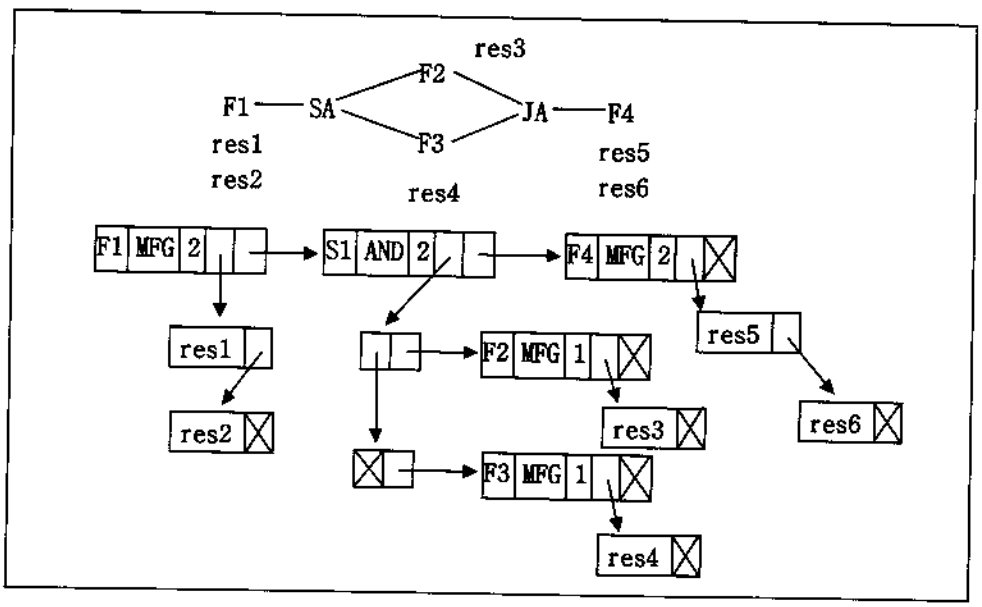

그림 9. MFG Feature 그래프의 자료 구조 표현 예

5. Planner 및 Planning 알고리즘

5.1 Planner의 역할

Heterarchical 제어 구조에서는 각 agent들이 내리는 의사 결정은 planning과 scheduling 모듈에 의해 수행된다. 그 중 planning 모듈을 담당하는 planner는 scheduling 모듈이 앞으로 수행해야 할 일련의 작업들을 결정, 생성하는 역할을 한다. 또한, 단순히 정해진 순서로만 가공을 하도록 이루어진 기존의 공정 계획 정보와는 달리, 가공 자원 혹은 가공 경로에 대한 대안들이 존재하는 공정 계획 정보를 입력받음으로써 현재의 shop 상태를 고려하여 유연성을 가지면서 제품을 생산할 수 있게 한다.

가공 장비의 planner는 InitMon agent(생산 시작 초기) 혹은 다른 가공 장비들로부터 받은 공정 계획 정보를 이용하여 scheduling 문제들을 생성해 내게 된다. 또한 공정 계획 정보를 관리, 갱신하는 기능도 수행하게 된다. 그림 10은 생산이 시작되는 시점에서의 이러한 상황을 나타낸 것인데 미리 준비된 공정 계획, 생산 계획 정보 등이 InitMon agent로 주어지면 실제 생산은 시작된다. InitMon agent는 공정 계획 정보를 이용하여 최초로 가공하게 될 머신을 결정하여 가공에 필요한 정보들을 전달하고, AS/RS agent에 가공을 위한 소재(raw material)를 보내 줄 것을 요청한다. 선택된 머신은 해당 가공 작업을 수행하며, 다음 가공 작업을 수행할 수 있는 머신들과의 통신을 통해 실제 다음 작업을 수행할 머신을 다시 결정하게 된다. 이러한 과정이 부품 가공이 완료될 때까지 계속 반복된다.

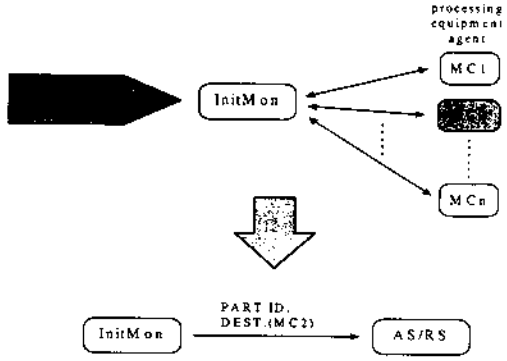

그림 10. 생산 초기의 상황

Planner가 수행하는 기능의 흐름과 다른 agent나 모듈들과의 정보 교환 상태가 그림 11에 나타나 있는데 현재 단계의 머신 agent와 다음 단계의 머신 agent와의 정보 교환이 활발함을 알 수 있다. 그림 11에서의 번호들은 다음에 설명할 planning 알고리즘에서의 과정과의 관계를 보여주기 위해 붙여둔 것이다.

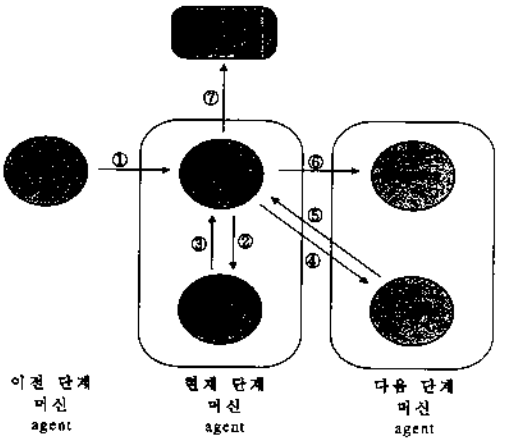

그림 11. Agent 및 모듈들간의 관계

5.2 Planning 알고리즘

5.2.1 Planning 알고리즘의 전체적인 구조

Planning 알고리즘은 현재 가공이 이루어질 머신의 planner가 그 전단계 가공 머신의 planner로부터 공정 계획 정보를 받았을 때부터 다음 단계의 가공머신으로 넘길 때까지의 의사결정을 어떻게 할 것인가를 결정한다. 최초로 가공이 시작되는 단계에서는 전단계 가공 머신은 InitMon이라 간주할 수 있으며, 제품 가공이 완료되었을 때는 완제품은 자동창고로 혹은 경우에 따라서 조립 작업을 담당하는 agent로 보내지게 된다. 그림 12에서 이러한 전반적인 과정을 나타내었고 그림 12에서의 번호는 그림 11과의 관계를 보여주고 있다. 그림 12의 MFG Feature 그래프 탐색에서 ⓐ는 탐색 범위에 포함되어 있는 모든 머신 agent들과의 정보 교환을 의미한다.

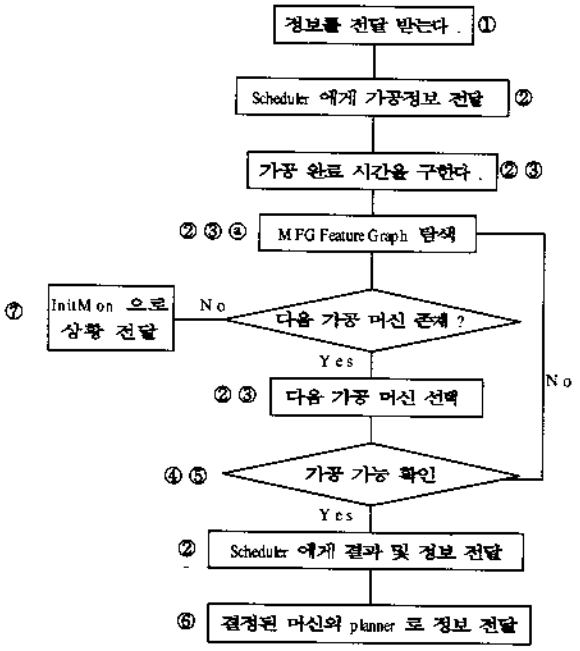

그림 12. Planning 알고리즘

5.2.2 MFG Feature 그래프의 탐색

다음 차례의 가공 머신을 선택하기 위해서는 우선적으로 MFG Feature 그래프에 대한 탐색이 이루어져야 한다. 탐색하는 방법은 다음과 같은 5가지 탐색 단위로 나누어 수행된다.

Single-type

MFG-FEATURE 노드에 할당된 가공 자원이 그림 13에서와 같이 하나만 존재할 경우를 말하며 이 경우 planner는 선택의 여지없이 그 노드와 자원을 선택한다.

Planner selects MC1.

MC1

그림 13. Single-type의 예

Serial-type

그림 14와 같이, MFG-FEATURE 노드들 즉, 가공 순서는 일정하게 정해져 있고 각각의 노드에 할당된 가공 자원들의 대안만이 존재할 경우를 serial-type으로 정의한다. Serial-type의 경우는 다음과 같은 방법을 이용한다.

(1) 우선 각 MFG-FEATURE 노드에 할당된 자원들 중에서 작업이 이루어질 수 없는 것들을 제거하여 경우의 수를 줄인다.

(2) 자원들을 조합하여 가능한 경로들을 생성한다. 그림 14의 경우, 모든 머신들이 작동가능하다면 그림 15와 같은 경로들이 존재

한다.

(3) 생성된 경로들 중 최소의 비용 C를 갖는 것을 선택하며 비용 C는 다음과 같이 정의한다.

$$C=\sum_{j=1}^{n}(U_{j-1}+M_{j-1,j}+L_j+W_j+P_j)$$

여기서 U_j : node j에서의 unloading 시간

$M_{j-1,j}$: node j-1에서 node j로의 이동 시간

L_j : node j에서의 loading 시간

W_j : node j에서 가공 시작까지 기다리는 시간

P_j : node j에서의 가공 시간

을 나타낸다. 그림 15의 경우는 4가지 경로에 대한 비용을 구한 후, 최소값을 갖는 경로를 선택한다. 여기서, 경로에 대한 비용을 구할 때, U_{j-1}, M_{j-1}, j는 노드 j-1을 담당하는 머신의 scheduler를 통해서, 그리고 L_j, W_j는 노드 j를 맡고 있는 머신의 scheduler를 통해서 얻게 된다.

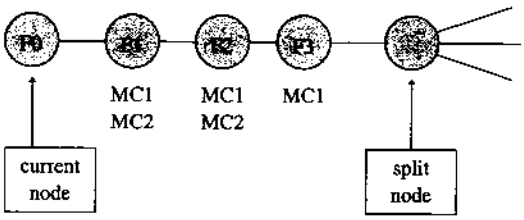

그림 14. Serial-type의 예

그러나 이상과 같은 방법은 노드와 그에 할당된 자원들의 수가 많아질수록 비교 횟수가 크게 증가한다는 단점이 있다. 즉, n개의 노드가 존재하고 각각의 노드에 m개의 자원

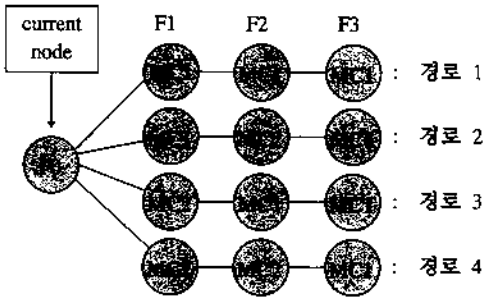

그림 15. 가능 경로

들이 할당되어 있는 경우에는 가능 경로 수가 m^n가지 만큼 존재하게 된다. 따라서 생성되는 경로 수를 줄이기 위하여 적용 상황(시스템의 성능 등)에 따라 일정 상수값을 주고 생성되는 경로 수가 주어진 값보다 작도록 여러 개의 작은 serial-type으로 나누어서 탐색한다.

OR-type

이 형태는 그림 16에서와 같이 가공 자원들에 대한 대안뿐만 아니라 가공 경로에 대한 대안들이 함께 존재하는 경우이다. 각각의 경로별로 비용을 구하여 최소의 비용을 갖는 경로를 선택하는데, 각 경로의 비용을 구할 때는 single-type과 serial-type에서 기술한 방법들이 적용된다. 그림 16의 예는 2개의 serial-type(F1, F2, F3, F4)과 1개의 single-type(F5)으로 나뉘어지므로, 3개의 경로들에 대해 각각 최소의 비용을 갖도록 자원을 선택한다. 그런 후, 3개 경로들의 비용을 비교하여 최소값을 갖는 경로를 선택한다.

Serial-type에서 언급한 비교 횟수에 대한 제약 상수를 이 경우에도 똑같이 적용할 수 있다. 그러나 탐색 범위를 제한하는 serial-type

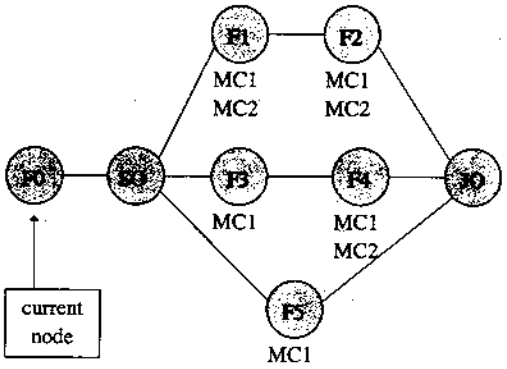

그림 16. OR-type의 예

과는 달리 OR-type의 경우는 비용을 계산하는 범위가 달라진다. 그림 17에서 보는 것처럼 제약 상수보다 작도록 범위를 나눈 후, 각각의 범위에 대한 비용의 합으로써 그 경로에 대한 비용을 구한다.

당된 가공 자원들 중 가장 짧은 가공 시간을 갖는 것을 선택해서 scheduler에게 이 정보를 넘겨준다.

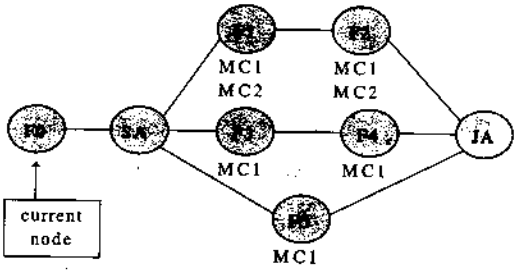

그림 18. AND-type의 예

Composite-type

Composite-type은 위의 OR-type과 AND-type이 함께 존재하는 경우를 말하며, 두 유형의 포함 관계에 따라 다시 두 가지 경우로

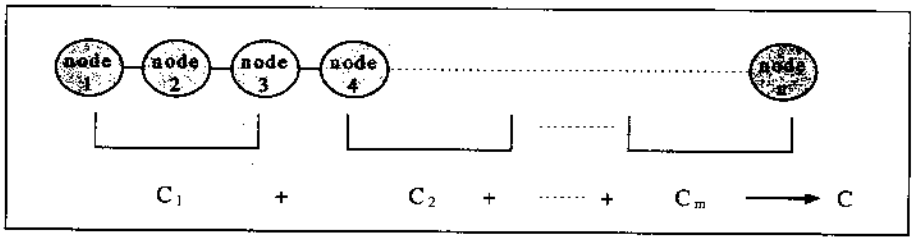

그림 17. 제약 상수에 의한 경로 비용 산출

AND-type

가공 자원 선택 문제와 scheduling 문제(sequencing)가 함께 존재하는 경우를 말하며, 그림 18에서와 같이 SPLIT-AND와 JOIN-AND 노드로 묶여져 있다. 이 경우는 앞에서 언급한 다른 경우들과는 달리 각 노드간의 이동 시간이나 가공 대기 시간을 고려할 수 없다. 따라서 작동이 불가능한 가공 자원들을 먼저 제거한 후, planner는 각 노드에 할

나뉠 수 있다.

(A) AND-type이 다른 type들을 포함하고 있는 경우

포함된 type이 AND-type일 경우에는 상기 AND-type의 경우를 적용한다. 만약 포함된 type이 OR-type인 경우에는 그림 19와 같은 경우로써 OR-type을 먼저 해결한다. 우선 최소 가공 시간을 갖는 자원들을 선택한 후, OR-type 내의 각 경로들의 비용을 구하여 최

소값을 갖는 경로를 선택한다. 경로 비용을 구할 때 가공 대기 시간(W_j)은 고려하지 않으며, 이동 시간($M_{j-1,j}$)은 이동 시점이나 shop 상태를 고려하지 않은 단순 이동 시간으로 대신한다. OR-type이 해결되면 상기 AND-type의 경우를 적용한다. 즉, 그림 19의 예에서 F3, F4, F5의 두 경로 중 하나를 먼저 선택하여 Composite-type을 AND-type으로 변환시킨다.

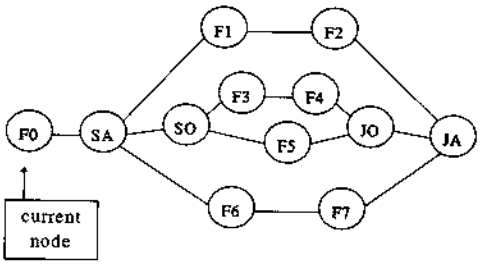

그림 19. AND-type이 OR-type을 포함한 경우

포함된 OR-type이 그림 20과 같이 다시 AND-type을 포함하고 있을 경우가 존재하는데, 이 경우는 AND-type을 하나의 MFG-FEATURE 노드로 간주하여 가공 시간 P_j를 설정한 후 위의 경우와 마찬가지로 해결한다.

(B) OR-type이 다른 type들을 포함하고 있는 경우

앞서 언급한 모든 경우가 복합되어 있는 경우로서 다음과 같이 경로를 선택한다.

(1) 각 경로를 나누어서 고려한다.

(2) 앞서 언급한 모든 type의 경우를 적용하여 각 경로에 대한 비용을 구한다.

(3) 최소의 비용을 갖는 경로를 선택한다.

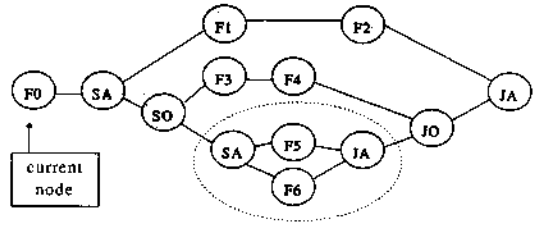

그림 20. AND-type, OR-type, AND-type순으로 포함한 경우

6. 실험 및 결과

6.1 실험 환경

본 연구에서 제안한 heterarchical 제어 모델을 그림 21과 같이, 두 대의 가공기계(machining center), AS/RS, 컨베이어, 로봇으로 구성된 생산시스템에 적용시켜 보았다.

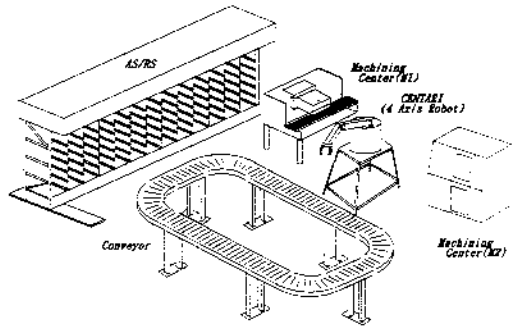

그림 21. 실험시스템 구성도

이 시스템을 agent들과의 통신망을 통해 나타내 보면 그림 22와 같으며, 각 agent는 Windows NT를 탑재한 IBM PC 486/pentium 급 호환 기종으로 구성되었다.

실험에서 사용된 예는 그림 23과 같은 부품을 사용하였고 이에 대한 MFG Feature 그

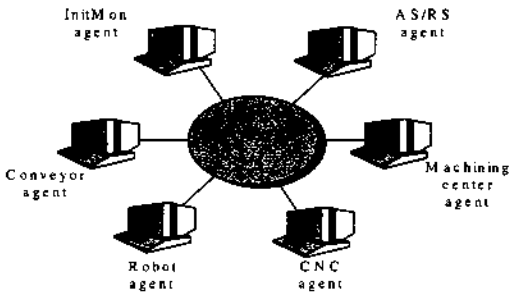

그림 22. Agent들과의 통신망 구성

래프는 그림 24에, MFG-FEATURE 노드의 정보를 표 2에 나타내었다. 두 대의 머신간 이동 시간, 머신에서의 loading/unloading 시간은 각 1분씩으로 가정하였다.

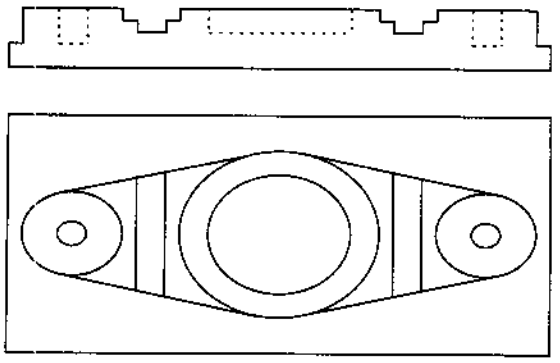

그림 23. 부품 예제

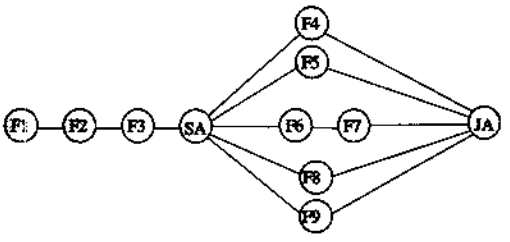

그림 24. 예제에 대한 MFG Feature 그래프

표 2. MFG-FEATURE 노드의 정보

노드	가공 타입	가공 자원	가공 시간 (분)
f1	면삭	(m1, t11)	5
f2	윤곽 가공	(m1, t12) (m2, t21)	4 4
f3	윤곽 가공	(m1, t14)	4
f4	drilling	(m2, t25)	2
f5	drilling	(m2, t25)	2
f6	centering	(m2, t26)	1
f7	pocket	(m1, t13)	4
f8	slot	(m1, t14) (m2, t23)	3 3
f9	slot	(m1, t14) (m2, t23)	3 3

6.2 실험 결과

위 예제와 같은 부품 1,000 개를 가공할 때 걸리는 총 가공 시간 및 부품 1개당 평균 가공 시간은 표 3과 같다. 실험 1은 MFG Feature 그래프 탐색시 노드 하나만을 고려 대상으로 하여 다음 단계 가공 작업이 가능한 머신이 있으면 우선적으로 할당하는 방법을 이용한 결과이다. 실험 2는 본 연구에서 제안한 탐색 단위를 이용하여 현재 상태뿐만 아니라 가까운 미래 상태까지를 고려한 결과이다. Shop의 현재 상태만을 고려한 실험 1의 방법에 비해 본 논문에서 제안한 방법이

표 3. 가공 시간 비교

실 험	총 가공 시간(분)	평균 가공 시간(분)
실험 1	27,000	27.0
실험 2	24,500	24.5

더 효율적임을 알 수 있다.

부품 2개를 가공할 경우, 실험 1과 실험 2의 시간에 따른 가공 과정을 자세히 살펴보면 표 4, 표 5와 같고, 부품에 대한 가공 순서는 그림 25, 그림 26에 각각 나타내었다.

표 4와 표 5를 비교해 보면, 실험 1의 경우는 현재 가공 가능한 머신을 우선적으로 선택함으로써 실험 2에 비해 M1에서 3분간(시간 47-50)의 가공이 이루어지지 않는 시간(idle time)이 발생하게 된다.

표 4. 시간별 가공 과정(실험 1)

시 간	M1	M2	부품이동
0	part1(F1 가공)		
5	part1(F2 가공)		
9	part1(F3 가공)		
13	part1(F8 가공)		
16	part1(F9 가공)		
19	part1(unloading)		
20	part2(loading)		part1->M2
21	part2(F1 가공)	part1(loading)	
22		part1(F4 가공)	
24		part1(F5 가공)	
26	part2(F2 가공)	part1(F6 가공)	
27		part1(unloading)	
28			part1->M1
30	part2(F3 가공)		
34	part2(F8 가공)		
37	part2(F9 가공)		
40	part2(unloading)		
41	part1(loading)		part2->M2
42	part1(F7 가공)	part2(loading)	
43		part2(F4 가공)	
45		part2(F5 가공)	
46	part1(unloading)		
47		part2(F6 가공)	
48		part2(unloading)	
49			part2->M1
50	part2(loading)		
51	part2(F7 가공)		
55	part2(unloading)		

표 5. 시간별 가공 과정(실험 2)

시 간	M1	M2	부품이동
0	part1(F1 가공)		
5	part1(unloading)		
6	part2(loading)		part1->M2
7	part2(F1 가공)	part1(loading)	
8		part1(F2 가공)	
12	part2(F2 가공)	part1(unloading)	
13			part1->M1
16	part2(F3 가공)		
20	part2(unloading)		
21	part1(loading)		part2->M2
22	part1(F3 가공)	part2(loading)	
23		part2(F4 가공)	
25		part2(F5 가공)	
26	part1(F8 가공)		
27		part2(F6 가공)	
28		part2(unloading)	
29	part1(unloading)		part2->M1
30	part2(loading)		part1->M2
31	part2(F7 가공)	part1(loading)	
32		part1(F4 가공)	
34		part1(F5 가공)	
35	part2(F8 가공)		
36		part1(F6 가공)	
37		part1(unloading)	
38	part2(unloading)		part1->M1
39	part1(loading)		part2->M2
40	part1(F7 가공)	part2(loading)	
41		part2(F9 가공)	
44	part1(F9 가공)	part2(unloading)	
47	part1(unloading)		

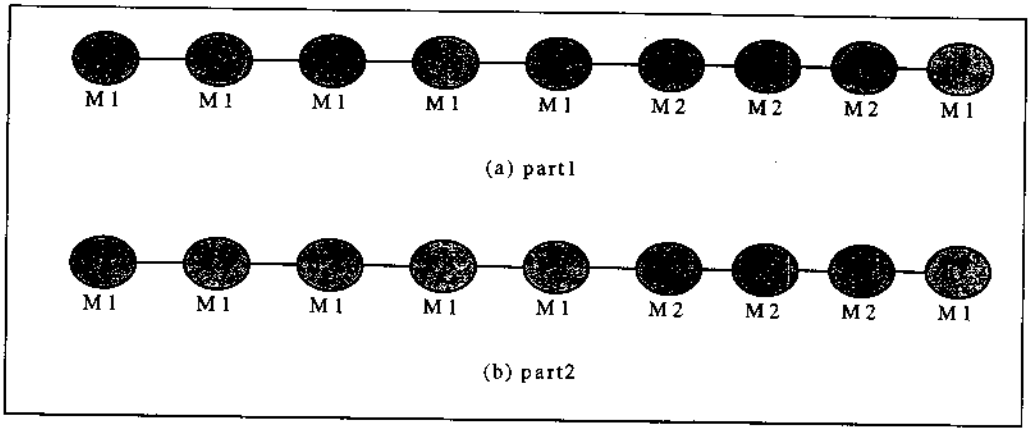

그림 25. 부품별 가공 순서(실험 1)

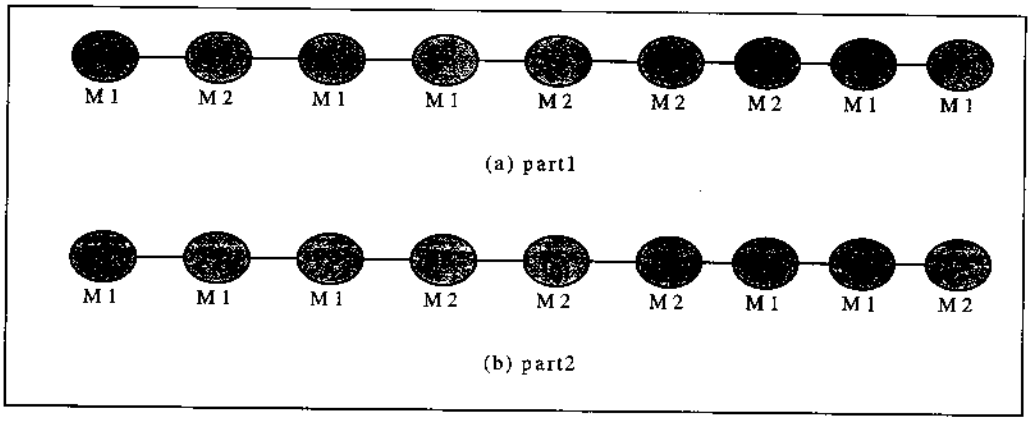

그림 26. 부품별 가공 순서(실험 2)

7. 결론 및 추후 연구 과제

본 논문에서는 급격한 변화에 대처할 필요가 있는 최근의 생산 시스템에 적합한 제어 구조로서 heterarchical 제어 모델을 적용하였다. 이 모델은 독립적인 agent들로 구성되며 agent들간의 정보 교환 및 협력을 통해 생산 활동을 수행한다. SFCS를 구성하는 장비들이 agent별로 모듈화 됨으로써 서로 독립적인 장비들(agent)의 집합체가 하나의 시스템을 이룬다. 즉, 기존의 계층적 제어 모델의 경우는 시스템 레이아웃의 재구성이 용이하지 않은 반면에, agent에 기반을 둔 heterarchical 모델에서는 이식성 및 기존 시스템의 재구성 능력이 뛰어나다. 따라서 변화에 유연하게 대처할 수 있다는 장점을 갖는다.

Heterarchical 제어 모델을 구현하기 위하여 우선적으로 agent의 정의 및 분류 작업을 수행하였으며, agent의 수행 기능을 planning, scheduling, execution의 세 기능으로 나누었

다. 본 연구의 주된 관심사는 가공 장비 agent의 planning 기능에 관한 것이다. Planning은 shop에서 발생하는 의사 결정 문제들에 대한 일차적인 해결 모듈로 볼 수 있으며 공정 계획 정보를 관리하는 책임을 진다. 즉, 공정 계획 정보를 이용하여 가공 자원 및 경로에 대한 대안들을 제거하여 scheduling 문제들을 성성해 내는 역할을 수행한다. 이에 필요한 공정 계획 정보를 MFG Feature 그래프를 사용하여 표현하였으며 이 그래프를 이용한 planning 기능의 수행 과정을 제시하였다.

추후 연구 과제로는: (1) 부품 가공시 발생할 수 있는 re-fixturing과 4축 이상의 CNC 머신에서의 공구 자세 등에 대한 정보를 포함할 수 있는 공정 계획 표현 모델이 연구되어야 한다. (2) 부품을 임시 저장할 수 있는 버퍼의 존재 유무에 따른 의사 결정 알고리즘의 세분화가 필요하다. (3) 교착 상태의 감지 및 발생시의 해결 방법에 대한 모색이 필요하다. (4) Heterarchical SFCS를 구성하는 다른 agent들의 기능 정의가 필요하다.

참 고 문 헌

[1] Bhaskaran, K., "Process plan selection," International Journal of Production Research, 28, 1527-1539 (1990)

[2] Carton, B. A. and Ray, S., "ALPS: A language for process specification," International Journal of Computer Integrated Manufacturing, 4, 2, 105-113 (1991).

[3] Cho, H. and Wysk, R. A., "Intelligent Workstation Controller for Computer-Integrated Manufacturing: Problems and Models," Journal of Manufacturing System 14, 4, 252-263 (1995).

[4] Davis, W. J., Jones, A. T. and Saleh, A., "Generic architecture for intelligent control systems," Computer Integrated Manufacturing Systems 5, 2, 105-113 (1992).

[5] Dilts, D. M., Boyd, N. P. and Whorms, H. H., "The Evolution of Control Architectures for Automated Manufacturing Systems," Journal of Manufacturing Systems 10, 1, 79-93 (1991).

[6] Duffie, N. A. and Piper, R. S., "Nonhierarchical Control of Manufacturing Systems," Journal of Manufacturing Systems 5, 2, 137 - 139 (1986).

[7] Duffie, N. A., "Synthesis of heterarchical manufacturing systems," Computer in Industry 14, 167-174 (1990).

[8] Homen de Mello, L. S.. and Sanderson, A. C., "AND/OR Graph Representation of Assembly Plans," IEEE Trans. Robotics and Automation, 6, 2, 188-199 (1990)

[9] Jones, A. T. and Saleh, A., "A multi-level/multi-layer architecture for intelligent shop floor control," International Journal of Computer Integrated Manufacturing 3, 60-70 (1990).

[10] Joshi, D. S., Wysk, R. and Jones, A., "A scaleable architecture for CIM shop floor control," Proceedings of CIMCON '90, National Institute of Standards and Technology, Gaithersburg, Maryland, 21-33 (May 1990).

[11] Lecocq, P., Guiot, T. and Fang, Q., "An

expert systems application to increase the flexibility and the efficacy of real time flexible manufacturing system controllers," Expert Systems and Intelligent Manufacturing, edited by Oliff, M. D., North-Holland, New York, New York (1988).

[12] Lee, K., Formal Integration of Process Planning and Shop Floor Control for CIM, Ph.D. Dissertation, Pohang University of Science and Technology (1994).

[13] Lee, S., Wysk, R. A. and Smith, J. S., "Process planning interface for a shop floor controlarchitecture for computer-integrated manufacturing." International Journal of Production Research 33, 9, 2415-2435 (1994).

[14] Lin, G., Y-J., and Solberg, J. J., "Integrated shop floor control using autonomous agents," IIE Transactions Volume 24, No. 3, 57-71 (1992).

[15] Maimon, O. Z., "Real-time operational control of flexible manufacturing systems," Journal of Manufacturing Systems 6, 125-136 (1987).

[16] Mettala, E. G. and Joshi, S., "A compact representation of alternative process plans/routings for FMS control activities," Journal of Design and Manufacturing, 3, 91-104 (1993)

[17] O'Grady, P. J., Bao, H. and Lee, K. H., "Issues in intelligent cell control for flexible manufacturing systems," Computers in Industry 9, 25-36 (1987).

[18] O'Grady, P. J. and Lee, K. H., "An intelligent cell control system for automated manufacturing," Knowledge-based Systems in Manufacturing, edited by Kusiak, A., Taylor & Francis, Philadelphia, Pennsylvania (1989).

[19] Ting, J. J., "A cooperative shop-floor control model for computer-integrated manufacturing,"

[20] Wysk, R. A., Mayer, R. J. and Cho, H., "Shop-Floor Scheduling and Control: A System Approach," Proceeding of DGOF/OFSA conference on recent developments and new perspectives OR in manufacturing, Hagen, Germany (June 25-26 1992).

[21] 김화진, 조현보, 정무영, "An intelligent planner of processing equipment for CSCW-based shop floor control in agile manufacturing," 대한산업공학회 춘계학술대회 발표논문집, 185-192 (1995).

96년 6월 최초 접수, 96년 9월 최종 수정