論文 한국정밀공학회지 제14권 제7호 (1997년 7월)
Journal of the Korean Society of Precision Engineering Vol. 14. No. 7. July. 1997.

공장 자동화를 위한 지능 생산시스템 개발에 관한 연구

최 경 현*

Study of Intelligent Manufacturing System Development for Factory Automation

Kyung-Hyun Choi*

ABSTRACT

This paper describes a task level cell programming environment that deals with difficulties in programming Flexible Manufacturing Cells(FMCs), and consists of the cell programming editor and the automatic generation module. In the cell programming editor, cell programs can be developed with task-oriented cell specifications that reduces the amount of details to be considered by cell programmers. The automatic generation module transforms task specifications into executable programs used by cell constituents. The development tool in designing the environment is an object-oriented approach which provides a simple to use and intuitive user interface, and allows for an easy development of object models associated with the environment. Test results are illustrated in order to demonstrate the applicability of the developed environment.

Key Words: CIM(컴퓨터통합생산), Flexible Manufacturing Cells(유연제조셀), Cell Task Specification Language(셀타스크서술언어), Artificial Intelligence(인공지능)

1. 서 론

제품의 수명 주기(life cycle)가 감소되는 추세에 따라 소비자의 다양한 요구에 신속히 대응할 수 있도록 현대의 제조(manufacturing)시스템은 유연성과 지능을 높이기 위해 유연제조 시스템(Flexible Manufacturing Systems)의 자동화 진행, 제조 관련 지식 베이스의 사용 및 가공 능력의 향상 등이 필요하다. 이러한 진보된 자동화 공장을 실현하려는 노력중의 하나는 높은 수준의 유연성을 가지는 제조 셀(Manufacturing Cells)의 개발에 초점을 맞추고 있다. 기계 가공을 위한 유연제조 셀(Flexible Manufacturing Cells)은 NC 머신, 로봇, 컨베이어 및 주변장치와 제조 셀 내에서 주어진 작업을 원활히 수행하기 위해 머신들간의 유기적인 조화를 관리 및 통제하는 셀 제어기(Cell Controller)로 구성되어 있다.

일반적으로 유연제조 셀을 구성하는 로봇, 머신 툴(machine tool) 및 주변장치들의 통합을 구현하여 실질적인 운용을 하는데 가장 큰 장애요인은 구성 요소들이

* 부산대학교 기계공학부

각각 다른 벤더(multi-vendor)에서 제작되었기 때문에 각각의 다른 해당기계언어(machine programming language)를 실행하여 주어진 작업을 수행하는데 있다. 이는 로봇이나 머신 툴을 제작하는 벤더들이 고유 특성을 부각시킬 수 있는 제어 언어를 사용하며, 다른 벤더들의 머신 툴들과 통합할 수 있는 기능과 제어 언어의 개발에 관심을 갖지 않기 때문이다. 또한 한 벤더가 제조 셀을 구성하는 모든 종류의 로봇 및 머신 툴을 제작할 수 없는 실정이기 때문에 다양한 기계들을 위한 표준 언어가 개발되어 있지 않다.

이러한 문제점을 해결하기 위한 방법으로 모든 벤더들에게 하나의 표준을 따르도록 강요하는 것이 아니라, 벤더들이 추구하려는 개발 경향을 제약하지 않는 소프트웨어 시스템을 개발하는 연구가 많이 진행되었다. 이러한 접근법은 개발된 소프트웨어 시스템이 제조 셀을 설치하고, 셀을 프로그래밍 하는데 사용함으로서, 머신 툴의 형(type)이나 벤더와는 독립적으로 효율적인 제조 셀을 운용 할 수 있는 이점을 부여한다. 따라서 제품 제조 공장은 주어진 제조 환경에 적합한 로봇 및 머신 툴을 구입하여 유연제조 셀을 구성할 수 있으며, 각각의 기계들이 사용하고 있는 제어 언어는 통합 운용에 중요한 판단 기준으로 고려하지 않아도 된다.

카네기 멜론 대학과 Westinghouse의 협력으로 제조 셀에 필요한 데이터와 제어언어정보를 테이블 구조에 기초를 둔 CML이 개발되었으며,[1] Volz는 Ada를 이용하여 로봇을 중심으로 한 제조 셀을 실시간 운용 프로그래밍 하는 방법론에 대하여 논하였다.[2] 또한, Lavas는 소프트웨어 환경인 WADE의 구조와 이를 이용하여 제조 셀을 설계하고 운용에 관한 방법론을 논하였다.[3] 그러나 이 연구들은 셀 제어기에만 연구의 초점을 둔 반면 제조 셀을 구성하는 로봇 및 NC 머신의 프로그램에 관하여서는 고려하지 않았다. 이런 문제를 해결 하기위해 셀 제어기 뿐만 아니라 구성 기계들까지도 단일 언어를 사용하여 프로그램 할 수 있는 UniSet언어가 개발되었다.[4] 그러나, 이 레벨의 언어를 사용하여 실제 제조 공정을 프로그램 하기 위해서는 생산 활동에 관련된 높은 전문 지식을 요구한다. 예를들면, 평범한 셀 프로그래머가 NC머신 작업의 이송량이나 절삭깊이, 또는 출발지점과 목적지점 사이의 로봇의 정확한 궤도 같은 세부사항들의 선정에 어려움이 있다.

이 논문은 타스크 레벨 명령어(task level commands)를 개발하여 유연제조 셀의 프로그래밍과 공정계획에 관련된 어려움을 동시에 해결하는 자동 셀 프로그래밍 시스템을 제안한다. 이 시스템에 관련된 데이터 및 시스템을 모델링하는데 객체 지향과 지식베이스가 기본적인 접근법으로 사용되었으며, 타스크 레벨 명령어로부터 상세한 셀 프로그램을 자동적으로 생성해내는 방법론이 이 논문에서 언급될 것이다.

2. 타스크 레벨 셀 명령어 개발

유연제조 셀에서 주어진 작업을 달성하기 위해서는 필요한 모든 공정들이 해당 기계들의 언어로 상세하게 정의되고, 이 들 기계들간의 유기적인 협조가 이루어져야 한다. 예를 들면, 로봇에 의해 컨베이어로부터 공작물을 집어서 NC 머신에 공작물을 올려놓는 공정은 로봇에게 필요한 명령어들을 다운로드하여 실행하여야 하며, 또한 다른 행위들이 컨베이어와 NC 머신에 의해서 실행되어져야 한다. 컨베이어와 로봇의 상호 협조 및 로봇과 NC 머신의 상호 협조를 제어하기 위하여 연동신호(interlock signal)가 이들 제어기들 사이에 교환되어져야 할 것이다. 이러한 모든 상세한 정보 및 명령어의 실행 순서들은 셀 프로그래머에 의해서 미리 정의되어져야 한다.

이를 위하여 셀 프로그래머는 제조 셀의 생산 활동에 관련된 모든 정보 및 지식을 가지고 있어야 하며, 또한 수행 해야할 작업에는 일반적으로 다양한 기계들이 포함되어 해당 기계언어를 모두 습득해야 한다는 사실이 더욱 셀 프로그래머의 작업을 어렵게 만든다. 만약 소프트웨어 시스템이 유연제조 셀에서의 모든 생산활동에 필요한 상세한 정보 및 지식을 갖춘 전문가 시스템이며, 타스크레벨 셀 명령어가 도입된다면, 셀 프로그래머에게 남겨진 작업은 제품 생산에 필요한 작업을 타스크 레벨 명령어를 사용하여 공정 순서를 정의하는 일이다.

각각의 타스크 레벨 명령어는 공정을 달성하는데 필요한 기계들의 상세한 행위들이 함축되어 있으며, 공정에 필요한 모든 정보를 포함하고 있다. 이러한 개념에 기초를 둔 타스크 레벨 명령어를 객체 지향언어인 Smalltalk을 사용하여 개발하였다. 객체는 특성을 나타내는 상태와 객체의 행위를 서술하는 절차(procedure)로 이루어져 있다. 유연제조 환경에 포함되는 모든 실재 물과 정보를 객체로 정의할 수 있으며, 다중성, 상속성, 추상, 및 데이터 함축성 등의 객체 지향 특성들이 타스크 레벨 명령어 개발에 중요한 개념으로 사용되었다.

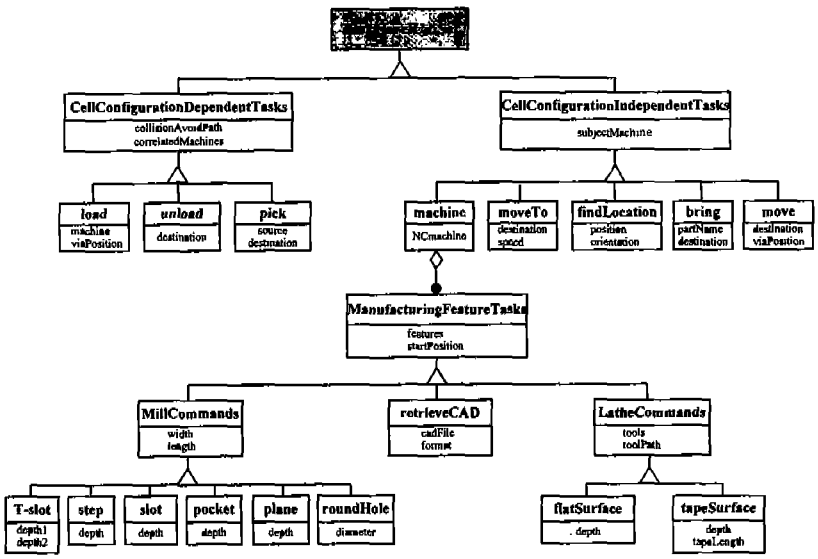

Fig. 1 Hierarchy of classes defined in the Task Level Instructions

본 연구에서 개발된 타스크 레벨 명령어의 클래스 계층 구조가 OMT[5]심벌을 사용하여 Fig. 1에 보여진다. 명령어는 크게 두개의 그룹 즉, 셀 구성에 의존하는 공정을 위한 명령어와 셀 구성에 독립적인 공정을 위한 명령어로 구분되어 진다. 셀 구성에 의존하는 공정을 위한 명령어는 이의 수행을 위하여 셀 구성 기계간의 유기적인 협조가 요구된다. 예를 들면, 공정 "Robot load part to Machine"에서 load라는 명령어는 로봇과 NC 머신과의 유기적인 연동 신호의 교환을 요구한다. 셀 구성과 독립적인 공정을 위한 명령어는 단지 해당 기계만의 구체적인 행위가 필요하다. 공정 "Robot move to Machine" 에서 move to를 의미하는 moveTo라는 명령어는 셀 구성 기계들과의 아무런 연동 신호의 교환 없이 독립적으로 로봇에 의해서만 이루어진다. 또한, 공정 "Mill machine Part"에서 machine이라는 명령어는 생산형상(Manufacturing feature)의 인스턴스들중의 하나에 의해 실행되어지는데, 이 명령어는 명령어 Mill, Lathe 및 retrieveCAD로 분류되어진다. 처음 두개의 명령어들은 슬롯과 tapeSurface라는 더 구체적인 명령어로 더 분해되어 질 수 있다. 명령어 retrieveCAD를 사용하여 CAD 시스템과 포스트 프로세서(post-Processor)에 의해서 생성되어진 공구 경로 파일을 불러 사용할 수 있다.

3. 자동 셀 프로그래밍 시스템

유연제조 셀의 효율적인 프로그램밍을 위한 자동 셀 프로래밍 시스템은 객체지향과 지식베이스 시스템으로 설계되어졌다. Fig. 2에서 보는바와 같이, 시스템은 셀 프로그래머가 타스크레벨 명령어를 사용하여 공정을 프로그램할 수 있는 셀 프로그래밍 에디터와 자동 생성 모듈로 구성되어 있다. 타스크레벨 명령어는 의미있는 명령어

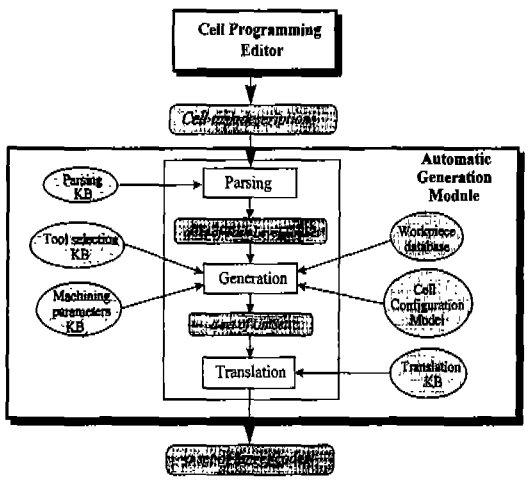

Fig. 2. Structure of Automatic Programming System

와 변수로 구성되어 있기 때문에 타스크레벨 명령어로 작성된 셀 프로그램은 평범한 셀 작업자에 의해서도 쉽게 이해될 뿐만 아니라, 셀 제어 및 프로그래밍에 관련된 깊은 지식이 요구되지 않고 단일 언어로 셀 전체를 쉽게 프로그램할 수 있다.

자동 생성 모듈은 타스크서술을 셀 구성 기계에 의해 실행될 수 있는 명령어들을 생성해 낸다. 이러한 임무를 원활하게 수행하기 위하여 각각의 기능을 가진 서브-모듈들이 포함되어 있는데, 이 들 모듈들은 다양한 지식 표현법과 추론 메카니즘의 도입에 의해 설계되어졌다. IF-Then형태의 생산 규칙은 forward chaining 추론 메카니즘과 함께 최적의 툴 선정에사용되었으며, 신경망 접근법은 경제적인 가공조건의 선정에 적용되었다. Prolog베이스의 지식 표현법은 언어 변환 작업에 도입되었다. 이러한 방법론의 도입으로 자동 생성 모듈이 더욱더 진보된 기능을 갖도록 하였다.

타스크레벨 명령어를 사용하여 유연제조 셀의 프로그램을 작성하기 위한 그래픽 사용자 인터페이스(Graphic User Interface)가 Fig. 3에 보여진다. 이는 셀 프로그래밍 작업에 필요한 정보의 양과 종류 및 이러한 정보를 셀 프로그래머에게 가장 효율적으로 전달하는 방식 등에 관한 분석 결과로 pane의 배열과 각 pane이 담고있는 정보들을 결정하였다. Programming Editor pane에는 제품 생산을 위한 셀의 구성 기계들이 수행해야 하는 공정들이 순차적으로 서술되는데, 그림에서 보는바와 같이, 먼저 해당 기계의 이름을 쓰고 수행하는 공정들이 타스크레벨 명령어를 사용하여 작성된다. 오른쪽 상단에 위치한 Machine Icon pane에 배열되어 있는 아이콘들은 실제 유연제조 셀을 구성하는 로봇, NC 머신, 콘베어등을 의미하는 심벌인데, 마우스로 원하는 아이콘을 선택하면 해당하는 기계가 실행할 수 있는 타스크 레벨 명령어들이 바로 아래 있는 Command Icons pane에 아이콘의 형태로 나타난다. 필요한 명령어 아이콘을 선택하면 dialog가 열리면서 셀 프로그래머와 적절한 대화를 통하여 공정을 서술한다. 이러한 접근법은 셀 프로그래머의 입력 에러를 줄일 수 있는 이점을 준다.

4. 로봇/NC머신 프로그램 자동 생성

타스크 레벨 명령어의 도입으로 유연제조 셀의 효율적인 운영을 위하여, Fig. 4에서 보여지는바와 같이 제품의 생산을 위하여 필요한 제조 셀의 공정들을 타스크 레벨 명령어로 작성하고, 각 해당 기계가 실행해야하는 구체적인 행위들이 중립코드의 형태로 자동 생성되어진 후 실제 셀 작업에 앞서 해당기계가 사용할 수 있는 제어 언어 형식으로 변환되어야한다. 이러한 자동 생성 및 번역은 공

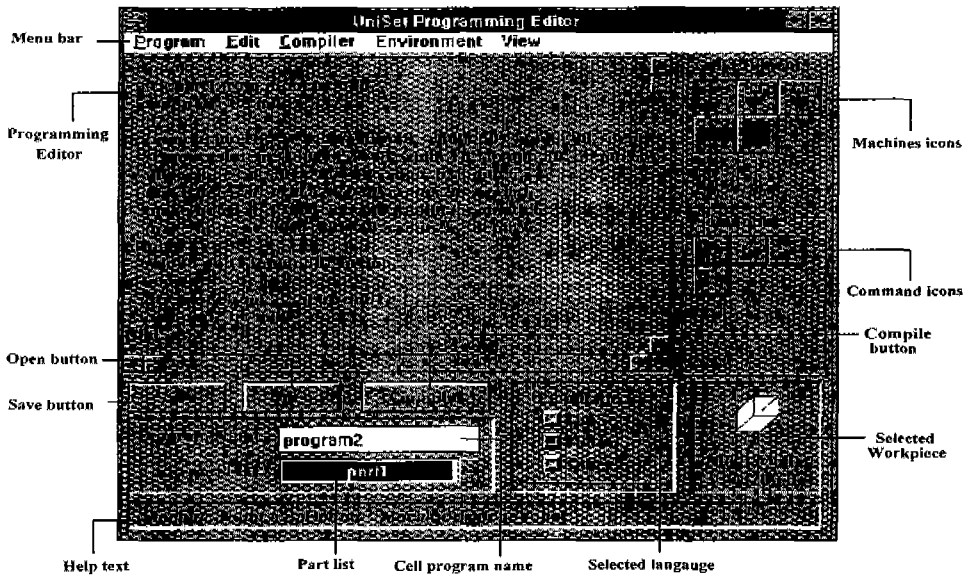

Fig. 3 Graphic User Interface for Task level Cell Program

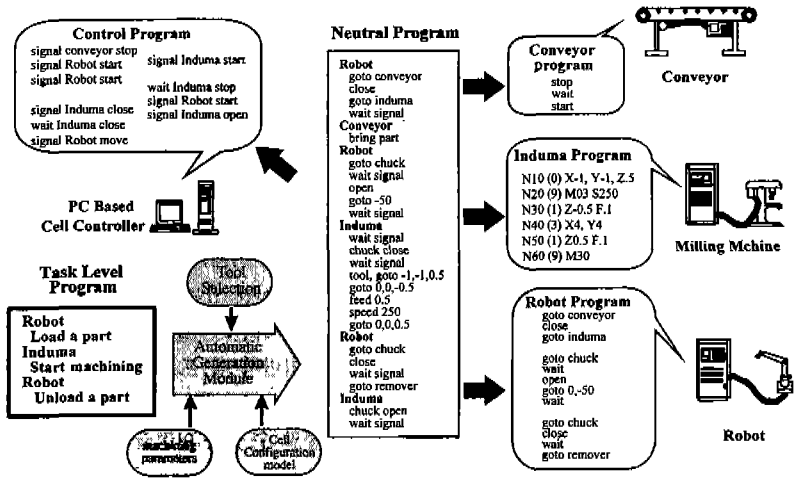

Fig. 4 Task level cell program and automatic generation mechanism

구 선정, 가공 조건 선정, 번역 를 및 유연제조 셀의 월드 모델링 등의 모듈들에 의해 수행되어진다.

4.1 최적 공구의 선정

효율적인 절삭 가공 공정을 위한 가장 적절한 절삭 공구의 선정은 가공형상(Manufacturing feature)에 의해서 좌우된다. 최적의 공구 선정 규칙은 데이터베이스 질의(Query)를 생성하는데 사용되며 질의는 가능성 있는 툴들의 집합을 생성한다. 공구 데이터베이스는 객체지향 접근법으로 만들어졌다. 공구는 크게 두 가지 그룹으로 분류한다. 현재 공구 매거진에 있는 공구와 공구 저장소에 있는 사용 가능한 공구로 분류한다. 공구들이 양 그룹에서 선정될 때는 가공공정에서 리드시간을 감소시키기 위해서는 공구 준비 시간을 최소화 하기 위해 공구 매거진에 속해있는 공구에 우선 순위가 주어진다. 만약 적합한 공구를 찾을 수 없다면 저장소에 있는 최적의 공구를 선정한다.

공구 선정 모듈은 다양한 형상(features)을 가공할 때에도 동일한 공구를 사용하려 할 것이다. 비록 하나의 공구가 모든 형상을 가공하는데 가장 알맞은 공구가 아닐 수도 있지만 공구의 교환 횟수를 감소시키고 가공에 필요한 공구의 수를 감소시키는 효과를 가져온다. 머신 샵(machine shop)에서 만약 각각의 형상에 가장 적절한 공구만을 사용한다면 수백 개의 공구를 구입하고 관리를 해야한다. 이런 경우 많은 자본 투자뿐만 아니라 필요 없는 설치와 재고 유지 문제를 발생시킬 수 있다. 또한 공구 매거진이 어떤 주어진 시간에 단지 한정된 수의 공구를 보관할 수 있기 때문에 부가적인 공구의 장착과 분리를 하는데 상당한 유휴 시간이 발생된다.

실제 공구 선정에 있어서 다음 사항을 결정하는 것이 중요하다. 공구 재질(예:HSS...), 공구 형상구조(예:Rake...), 공구 유형(예:milling....), 공구 치수(예: 길이...) 등이다. 이중 공구의 유형은 가공 공정의 종류에 의해서 결정되며 공구 형상구조는 가공하려는 형상, 공작물 재질, 그리고 공구 재질에 의해서 결정된다. 예를 들면, 공구의 flute 길이가 실제 가공에 필요한 길이보다 더 길도록 선택되어져야 하며, 공구 지름은 형상 치수에 의해서 결정되는데 구멍을 뚫을 때는 지름은 구멍의 치수와 일치해야 함으로 쉽게 정해진다. 슬롯(slots)을 가공하기 위해서는 형상의 폭이 공구 지름의 최대한계이다. 비록 공구 지름이 슬롯의 폭 길이와 일치하는 것이 바람직하지만 실제로 그런 공구를 사용하기가 어렵기 때문에, 공구 지름은 슬롯의 폭보다 더 작은 것 중에서 가장 큰 공구를 선택해야 한다.

주어진 가공 형상에 적절한 공구 선정은 생산 규칙(production rule)을 이용하였다. 이러한 규칙들은 중요한 세 가지 파라메터를 결정을 할 수 있는 정보를 포함하고 있다. 이 들 파라메터는 공구 재질, 형상구조, 공구 유형이며, 공구 선정 지식 베이스는 forward chaining 메커니즘을 사용하여 추론을 하며 공구 선정 규칙의 결과는 데이터베이스 관리 시스템에 보내져 질의(Query)로부터

```
Expert add:(ToolRules
    number:1
        condition:[ #(process drill) isRight]
        action:[#(cutType drill) doTool]
        description:'process is drilling').
Expert add:(ToolRules
    number:2
        condition:[ #(process turn) isRight]
        action:[#(cutType turn) doTool]
        description:'process is turning').
Expert add:(ToolRules
    number:3
        condition: [#(process profiling) isRight]
        action:[#(cutType endMill) doTool]
        description:'process is profiling').
Expert add:(ToolRules
    number:4
        condition: [#(process facing) isRight]
        action:[#(cutType faceMill) doTool]
        description:'process is facing').
" Rules for tool material selection based on the raw stock
material and its hardness."
Expert add:(ToolRules
    number:5
        condition:[ #(material 1) isRight]
        action:[#(toolMaterial 1) doTool]
        description:'material is #1').


    Expert add:(ToolRules
        number:33
        condition: [#(process profilling) isRight & #(width d) isAssume ]
        action:[#(maxToolDiameter d) doAssign]
        description:'profiling process and width d').! !
```

Fig. 5 Production rules for the tool selection

가장 적절한 공구가 선택된다. 지금까지의 연구에서 35개의 공구 선정 규칙이 개발되었으며, 그 중의 일부 규칙이 Fig. 5에 나열되어 있다. 그림에서 주어진 규칙 번호 33을 예를 들어 규칙을 해석하면 다음과 같다.

″공정이 profiling이고 폭이 d이면, 공구의 최대 지름은 d이다.″

만약 규칙 베이스에 의해서 여러 개의 사용 가능한 공구가 선택된다면, 최적의 공구를 정하기 위한 많은 휴리스틱 접근이 가능하다. 이러한 휴리스틱 방법 중에서 본 연구에서는 최소비용의 선택법, 수명이 긴 공구 선택법, 현재 보유하고 있는 가장 많은 공구 선택법 등이 사용되었다.

4.2 가공조건의 선정

절삭 가공중 표면 거칠기나 비용 등은 이송률과 절삭속도 및 절삭깊이 등의 가공조건으로부터 직접적인 영향을 받게 된다. 이러한 조건들은 임의적으로 선정되는 것이 아닐뿐만 아니라, 다양한 가공공정에 대해 일률적으로 동일한 조건을 적용할 수 없다. 적절한 절삭 가공조건은 최적 기법 및 머신닝 핸드북을 이용하거나, 가공 작업자의 경험에 의하여 결정되는 것이 일반적이다. 최적 기법은 원하는 결과를 최적화하려는 수학적인 관계식에 의하여 가공조건을 수치화 하는 것인데, 가공조건의 수학적인 관계식이 너무 복잡하여 실제작업에서는 그리 널리 사용되지 않는다. 가공작업자들은 오래된 경험을 바탕으로 가공조건들을 결정할 수 있지만, 경험들의 다양성 때문에 다양한 가공 환경에 적용 가능한 형식으로 나타내기는 어렵다. 실험에 의하여 구한 자료를 근거한 핸드북은 효과적인 절삭 가공을 위한 적절한 가공 조건들 포함하고 있다. 이러한 핸드북에는 방대한 데이터들을 담고 있기때문에 데이터베이스를 구축하는데 힘들뿐만 아니라, 적절한 가공조건을 탐색으로 찾아야 하는 어려움을 가지고 있다.

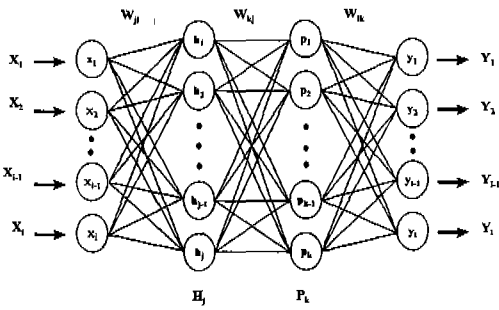

Fig. 6 Back-Propagation Model for machining parameters selection

본 연구에서는 최적의 가공조건을 선정하기 위해서 신경망 접근법을 사용하였다. Fig. 6 나타난 것처럼 다층 역전파(backpropagation) 신경망은 공작물의 재질 및 공구의 재질 등 작업의 조건을 나타내는 입력층(X_i)과 주어진 작업 조건에 적절한 가공조건을 계산한 후 보여지는 출력층(Y_i) 및 은닉층(H_j,P_k)으로 구성되어 있다. 학습이 진행되는 중에 추정데이터와 참고데이터의 오차를 감소시키기 위해 이 들 층간의 노드를 연결하는 가중치(W_{ji}, W_{kj}, W_{lk})와 각 노드의 바이어스(bias)값들이 반복적으로 갱신된다. 핸드북 데이터[6]를 참조로 하여 학습을 시킨 밀링과 드릴링 작업 및 선반작업에 대한 3개의 신경망이 구축되어 시스템에 내재되어 있다.

Ahmad는 학습에 필요한 입력 벡터의 수를 결정하는데 입력 변수의 수(k)에 대하여 $k^{2/3}$의 order에 해당하는 수가 충분하다고 제안하였다.[7] 드릴링 작업의 경우 공작물 재질, 경도, 공구 재질 및 구멍의 지름이 입력(k=4)이기

Table 1 Training input data vectors for drilling

Input vector	Input Variables				Referenced output variables	
	Material	Hardness (BHN)	Hole diameter (mm)	Tool material	Speed (mpm)	Feed (mpr)
1	1212	125	12.7	M10	38.1	0.254
2	1213	175	25.4	M10	38.1	0.457
3	1212	125	50.8	M10	38.1	0.635
4	1215	125	19.05	M7	39.6	0.381
5	1108	175	25.4	M1	36.6	0.457
6	11L13	225	38.1	M10	41.2	0.508
7	11L14	125	19.05	M10	39.6	0.381
8	12L13	125	50.8	M7	30.5	0.635
9	1005	150	6.35	M1	27.4	0.127
10	1012	150	38.1	M1	27.4	0.457
11	1019	200	12.7	M7	24.4	0.229
12	1021	200	38.1	M10	19.8	0.381
13	1030	200	50.8	M10	22.9	0.559
14	1040	250	19.05	M7	18.3	0.254
15	1049	300	25.4	M33	15.2	0.305

때문에 k2/3=2.5198이며, 첫 번째 값인 2의 의미가 입력 벡터의 개수가 2자리 order의 값(10~99)을 가짐을 의미한다. 본 연구에서는 입력 샘플링 크기를 90개로 결정하였으며, 그 중 학습에 사용된 벡터의 일부가 Table 1에 보여진다. 은닉층의 설계는 2개의 은닉층을 사용하였고, 각각의 은닉층에 사용된 노드의 수는 15×15이고, 학습비(learning rate)는 0.5와 모멘텀비(momentum rate)는 0.6을 선정하였다. 선반 및 밀링 작업에 대해서도 동일한 노드 수와 조건을 적용하였다. 학습된 신경망이 가공 조건을 얼마나 정확히 예측하는지를 실질 조건을 도입하여 검증을 실시하였는데, Fig. 7에 보여지는바와 같이 드릴링 작업에 있어서의 이송률 및 절삭 속도에 대하여 학습된 신경망으로부터 구한 결과와 참고 값의 비가 10%와 15% 이내로 신뢰성이 높음을 보여주었다. 이러한 접근법은 시간과 저장 공간을 줄여주고 새로운 데이터의 추가시 쉽게 확장해 나갈 수 있다.

4.3 중립 명령어 생성

각각의 타스크레벨 명령어는 관련된 NC 머신 및 로봇이 수행해야 하는 상세한 동작들을 포함하고 있으며, 이들은 접근, 실행 및 귀환의 3가지 연속적인 행위로 정의되었다. 예를 들면, NC 머신에 로봇을 이용하여 가공물을 로드하는 공정에서, 접근 작업에서는 로봇은 공작물을 NC 머신의 척크안에 위치시키기 위하여 로봇의 엔드 에펙터를 충돌을 회피하여 목표 위치까지 이동한다. 실행 행위는 로봇이 공작물을 놓기 전에 척크가 확실히 닫혀있는 지를 확인하고, 엔드 에펙터를 열어 공작물을 NC 머신에 로드를 마친다. 마지막으로 귀환 행위는 로봇이 다음 공정을 준비하기 위해 원 위치로 이동을 하는데, 이때에도 충돌을 회피하는 경로가 선정되어야한다.

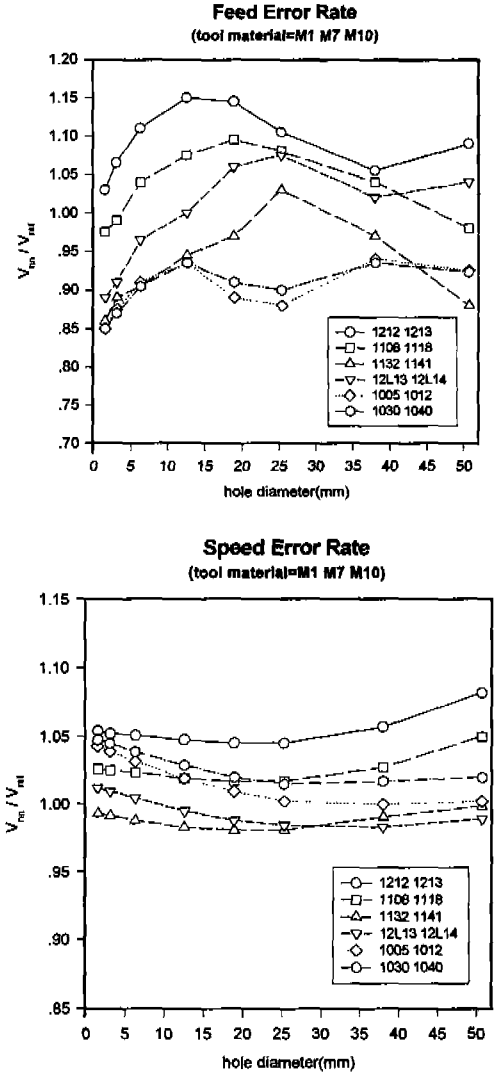

Fig. 7 Comparison between actual set data and back-propagation results for parameter speed

이들 머신/로봇의 움직임에 관한 구체적인 서술이 중립 명령어로 변환된다. 중립 명령어의 도입은 다양한 해당 기계언어가 포함되어 있어도 타스크 명령어로부터 단 하나의 생성 검파일러만 필요로 하는 이점을 준다. Table 2에 보여지는바와 같이, 중립 명령어는 motion, configuration, communication 등 NC 머신/로봇 및 셀 제어기의 필요한 기본적인 기능을 나타내는 명령어들을 포함하고 있다.[4]

타스크 레벨 명령어에 함축된 모든 NC 머신/로봇의 움

Table 2 Neutral Instruction Set

Instruction Description		Neutral Instructions
Joint Interpolation motion	Absolute	Goto,rapid,<*location*>
	Relative	Go,rapid:<*location*>
Linear Interpolation motion	Absolute	Goto:<*location*>
	Relative	Go:<*location*>
Spindle Control		Spindle,{off|cw|ccw}:{*number*}
Coolant Control		Coolant,{on|off|flood|mist|tapping}
Gripper Control		{open|close}[:*number*]
Time Delay		Dealy:<*number*>
Communication Control		{Wait|Signal},<*line number*>

직임에 필요한 위치등의 정보들을 유연 셀의 작업 환경 모델(Cell Configuration Model)로부터 얻을 수 있다. 이 객체 모델은 유연셀을 셋업(set-up)하는 단계에서 만들어 지며, 중요한 각 위치들을 포함하는 객체 데이터 모델이 Fig. 8에 보는바와 같이 개발되었다. 여기서 중요한 위치들은 로봇의 기준 좌표, 경유 위치, NC 머신의 기준 좌표 및 정지위치 등인데, 이들은 모두 월드 좌표계로 정의되며 동차(homogeneous)행렬을 이용하여 나타낸다. 경유 위치란 로봇의 엔드 에펙터가 충돌을 회피하기 위해 멈춤 없이 지나가는 경로상의 위치를 의미하며, 정지 위치는 로봇의 엔드 에펙터가 NC 머신 근처에서 완전히 정지한 후 다음의 명령어를 기다리는 위치이다. 이 들 위치들은 그래픽 시뮬레이터를 이용하거나, 실제 로봇을 온라인으로 움직여 정의할 수 있다. 본 연구에서는 그래픽 시뮬레이터를 사용하여 충돌을 회피하는 적절한 위치들을 정의하였다. 이러한 모든 움직임들이 로봇 좌표계에 기준을 둔 엔드 이펙트의 위치를 변환 행렬로부터 구하여, 필요한 중립 명령어들의 변수로 사용된다.

타스크레벨의 명령어로 부터 셀 컨트롤러의 프로그램은 타스크 명령어속에 함축되어 있는 로봇과 NC머신 및 주변장치와의 동기화에 대한 정보를 바탕으로 자동 생성되어 지는데, 각 타스크 실행에 포함되어 있는 3가지 연속 행위의 각 행위가 끝나는 시점마다 동기화의 명령어가 생성되어 셀 프로그램에 첨가된다.

5. 해당 기계언어의 명령어로 번역

번역 모듈은 자동 생성된 중립 명령어들이 해당기계에 다운로드 하기전 실행할 수 있는 언어의 명령어 형태로 변환시킨다. 명령어의 변환은 인공지능 언어인 prolog를 이용하여 이루어지는데, 중립 명령어와 해당 기계 명령어 사이의 일대일 대응을 나타내는 서식들이 predicates의 변수들로 표현되는데, Fig. 9은 predicate에 의한 번역의 규칙들이 보여진다.

하나의 지시어만 포함하는 형태의 중립 명령어는 해당 기계 언어의 명령어와 상대적으로 단순한 일대일 대응으로 번역 과정이 진행된다. 예를 들면, 로봇이 gripper를 닫는 동작을 정의하는 중립 명령어 Close는 VAL II 명령어 Close로 직접 번역된다. 이에 반해, 한정어를 가지는

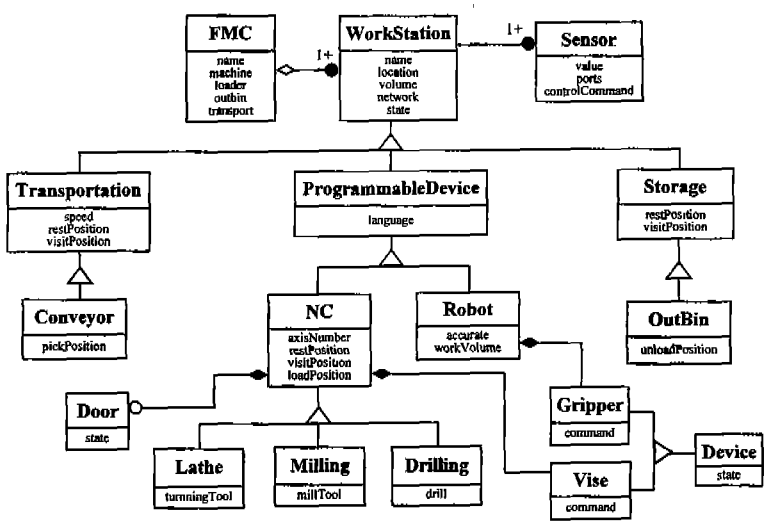

Fig. 8 Architecture of Cell Configuration Model Abstract

```
!ExpertUniTranslator methods !

translation('goto','rapid:','MOVE','VAL II').
translation('goto','rapid:','G90;G00','WORD ADDRESS').
translation('goto:','POS','ASEA')
translation('go:','G91;G01','WORD ADDRESS').
translation('go:','GODLTA/','apt').
translation('go:','MOVES HERE').
translation('go:','POS OFFSET','ASEA').
translation('gocircle','G17G02','WORD ADDRESS').
translation('spindle','ccw:','M04S','WORD ADDRESS').
translation('spindle','ccw:','SPINDL/,CCW','APT').
translation('coolant','off','COOLNT/OFF','APT').
translation('coolant','off','M09','WORD ADDRESS').
translation('coolant','on','M08','WORD ADDRESS').
---
translation('close','M22','WORD ADDRESS').
translation('close','Close','VAL II').
translation('close','GRIPPER','ASEA').
translation('close:','close','VAL II').
translation('tool:','G10H<num>Z<num>','WORD ADDRESS').
```

Fig. 9 Prolog based Translation Rules

명령어들은 한정어의 문맥에 종속되어 번역되어진다. 예를 들면, 중립 명령어 coolant를 Word Address 명령어로의 번역은 한정어 on과 off에 종속되는데, 한정어 on에 대응한 명령어는 M08인 반면, 한정어 off에 대응하여 명령어 M09가 선택된다.

변수들의 변환은 일괄적인 번역 메카니즘에 의해 수행되기보다는 번역된 각각의 명령어가 요구하는 변수의 형식을 정의한 객체에 의해 수행되어진다. 이 방법론은 명령어와 관련된 각 변수들이 중립 명령어 객체에 포함된 지식을 바탕으로 자동적으로 번역하도록 한다. 중립 명령어에서는 변수를 위치(position)와 자세(orientation)의 조합으로 나타내는데, VAL II, APT 그리고 Word Address에서도 거의 동일한 변수의 형식을 요구하기 때문에 이들 사이의 변수 변환은 상대적으로 간단하다. 그러나, ASEA에서는 엔드 에펙터의 자세(orientation)를 정의하기 위해 Quaternion을 사용하는 반면, 중립 명령어에서는 로봇 좌표 계를 기준으로한 오일러 각도를 사용하기 때문에 오일러 각도로부터 Quatenion으로의 변환 알고리즘이 Smalltalk에 정의된 객체의 메소드로 정의되어 있다.

6. 타스크 레벨 셀 프로그래밍 접근법의 실용성

이 절에서는 자동적으로 생성된 셀 프로그램과 해당기계 언어(target machine language)를 사용하여 개발된 셀 프로그램을 비교 분석함으로써 실제 유연제조 환경에서의 실용 가능성을 검토하였다. Fig. 10과 같이 주어진 유연제조 셀의 설치 환경에서 수행할 작업 공정은 로봇을 이용하여 공작물을 컨베이어에서 집어서 NC 머신에 투입하는 "pick/load" 공정이며, 사용된 로봇은 PUMA이고 제어 언어로 VALⅡ를 사용하고 있다. 모든 중요한 위치와 좌표계는 동차 행렬로 표시되었으며, 길이가 16cm, 넓이가 10cm, 높이가 5cm인 직육면체 형상의 공작물이 로봇에 의해 NC 머신으로 로드된다.

Fig. 11에는 이 작업을 위하여 타스크 레벨 명령어를 사용한 프로그램과 시스템에 의해 자동 생성된 중립 명령어로 된 프로그램이 보여진다. 타스크 레벨 명령어를 사용하여 셀을 프로그램 하는 단계에서는 모든 수치적인 정보는 함축되어 있으며, 자동 생성 모듈은 로봇의 엔드 에펙터의 경로에 따른 위치를 변환 방정식을 이용하여 로봇의 좌표계로 생성하여 위치를 수치적인 변수로 할당한다.

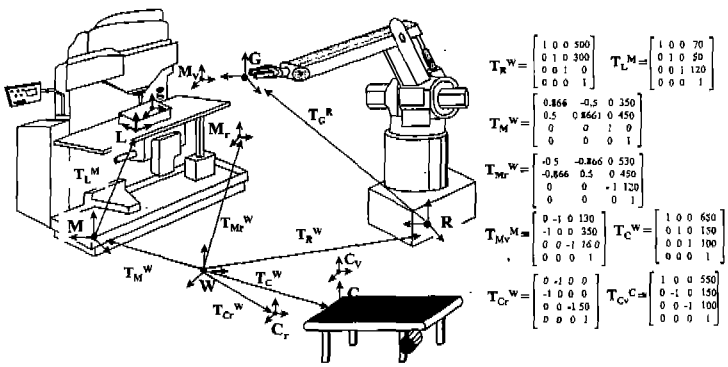

Fig. 10 Coordinate for the task pick/load

```
{Task Level Instruction}
pick part from:conveyor to:Mill.          ;Pick a part from conveyor to a mill
load part to:Mill.                        ;Load a part to mill
{Neutral codes}
Go,rapid:155@158@105,90@-180@90,50.       ;Move to the top of a part on a conveyor
Go:155@158@105.                           ;Move to a part
Close.                                    ;Grasp a part
Go:50.                                    ;Back off
Goto:100@150@120,90@-180@90.              ;Return to conveyor rest location
Goto:30@150@120.                          ;Move to mill rest location

Goto:-112@259@160,0@-180@-60.             ;Move to via location
Go,rapid:-159@237@110,30.                 ;Approach to load position
Go:-159@237@110.                          ;Move to load position
Wait.                                     ;Wait signal from mill
Open.                                     ;Release a part
Go:30.                                    ;Back off
Goto:-112@259@160,0@-180@-60              ;Move to via location
Goto:30@150@120.                          ;Return to mill rest location
```

Fig. 11 Task level cell program and generated neutral code for pick/load task

자동 생성된 코드와 VAL II를 사용하여 독립적으로 개발된 코드 사이의 비교는 Table 3에 나타나 있다. VAL II로 이 작업을 프로그램 하는 작업자는 로봇이 작업 수행을 위해 필요한 Mill, pick, load 등 중요한 위치들은 미리 온-라인으로 정의하여야 한다. 일반적으로 이러한 위치 변수들을 정의하는 편리한 방법은 로봇을 파워 리드(power lead)나 수동 리드(manual lead)를 사용하여 원하는 지점에 위치시키고 그 위치를 변수에 할당하고 메모리에 기록하는데, VAL II에서는 HERE Mill 이라는 명령어가 로봇의 현 위치를 변수 명에 할당하기 위해 사용된다. 그러나 온-라인 접근법은 작업의 변경시 현재 진행되는 작업들을 모두 정지시키고 새로운 지점들을 로봇에게 학습시켜야 하기 때문에 리드 시간을 증가시키는 결과를 초래할 수 있다.

본 연구에서는 월드 모델링 개념을 도입한 Cell Configuration Model에서 정의되어 생산 활동의 중단 없이 새로운 작업을 위한 프로그램밍을 할 수 있기 때문에 생산성을 높일 수 있다. 이들 목표 위치들은 제조셀 환경에 따라 계산된 수치적인 형식으로 구해진다. 첫 세 개의 숫자들은 엔드 에펙터의 위치를 표시하고, 마지막 세 개수는 자세(orientation)를 나타낸다. 이러한 특정 형식들은 번역모듈에 의해 UniSet 코드들로부터 번역되어진다.

Table에 의하면, 라인 번호 1과 8에서 VAL II를 이용한 프로그래머는 명령어 APPRO를 사용하는 반면, 자동

Table 3 Comparison between generated code and user developed code in VAL II

No	Generated Codes	Developed Codes	Comments
1	MOVES(155,158,155,90,-180,90)	APPRO *pick*,50	approach the *part*
2	MOVES(155,158,105)	MOVES *pick*	move to the *part*
3	CLOSE	CLOSE	grasp the *part*
4	DEPART 50	DEPART 50	back off
5	MOVES(100,150,120,90,-180,90)		move to *conveyor rest location*
6	MOVES(30,150,120)	MOVES *Mill*	move to the *mill*
7	MOVES(-112,259,160,0,-180,-60)	MOVES *via*	move to *via location*
8	MOVES(-159,237,140	APPRO *load*,30	approach to the *load location*
9	MOVES(-159,237,110)	MOVES *load*	move to the *load location*
10	WAIT	WAIT	wait signal from *mill*
11	OPEN	OPEN	Release the *part*
12	DEPART 30	DEPART 30	Back off
13	MOVES(-112,259,160,0,-180,-60)	MOVES *via*	move to the *load location*
14	MOVES(30,150,120)	MOVES *Mill*	Return to the previous location

생성 모듈은 명령어 MOVES를 생성한다. 이것은 생성 모듈에서 로봇이 NC 머신 에 접근하는데 충돌을 회피하는 모듈에 의해 최적 화된 경로를 정의하기 때문에 직선 움직임을 정의하는데 편리한 명령어 MOVES를 사용한다. 명령어 MOVES와 APPRO는 정확히 같은 로봇의 움직임을 명령한다. 라인 번호 5에서 VAL II 프로그래머는 명령어를 사용하지 않는 반면에 생성 모듈은 명령어 MOVES를 생성시킨다. 이 과정에서도 충돌 회피 모듈은 제조 셀을 구성하는 기계들 사이를 로봇이 충돌 없이 움직이기 위해 미리 정의된 경유 지점들을 통과하도록 하였다.

7. 결 론

본 논문에서는 선행된 연구에서 개발한 UniSet 언어를 중립언어로 도입한 타스크 레벨 셀 프로그래밍 명령어를 개발과 유연제조 셀에서 변종변량의 제품을 효율적으로 생산하기 위한 소프트웨어 시스템의 개발에 관하여 논하였다. 이 시스템은 크게 두 개의 모듈로 구성되어 있다. 첫 번째 모듈은 그래픽 사용자 인터페이스(GUI)로서 셀 프로그래머가 유연 셀을 프로그래밍 하는데 셀 구성요소들에 대하여 필요한 세부사항을 고려하지 않아도 되며, 오직 하나의 셀 명령어를 사용하여 필요한 공정들의 실행 순서를 서술하는데 집중할 수 있다. 이는 셀 프로그램과 공정계획의 어려운 점을 동시에 해결해주는 이점을 준다.

두 번째 모듈인 자동 생성 모듈은 타스크 레벨 명령어

를 사용하여 개발된 셀 프로그램을 중립 명령어의 형태로 각 구성 기계에 대하여 상세한 프로그램을 생성한 후, 기계 제어기에서 실행되기 전에 해당 기계 언어의 명령어 형태로 변환시킨다. 이러한 기능을 수행에 필요한 실제 생산 활동에 관련된 정보와 전문적인 지식들인 공구 선택 및 가공 조건 선정 규칙 베이스, prolog베이스 규칙, 유연 셀의 작업 환경 모델 등을 인공지능과 객체지향 접근법의 도입으로 효율적으로 시스템에 포함시켰다.

지금까지 개발된 타스크 레벨 명령어는 셀 구성요소들에 의해 수행되는 기본적인 공정만을 다루고 있으므로, 계속적인 연구로 더 많은 셀의 공정들이 정의되면 시스템의 적용성이 높아질 것으로 여겨진다.

참 고 문 헌

1. Wright, P.K., Bourne, D.A., Manufacturing Intelligence, Addison-Welsey Publishing Company, Don Mills, Ontario, 1989.
2. Volz, R.A., Mudge, T.M., Gal,D.A., Using Ada as a Programming language for robot-based manufacturing cells. IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-14, No. 6, pp. 863-878, 1984.
3. Levas, A., Jayaraman, R., WADE: an object-oriented environment for modeling and simulation of workcell applications. IEEE Transaction on robotics and automation, Vol.5, No. 3, pp. 324-335, 1989.
4. 최경현, 김성청 "UniSet개발을 통한 공장자동화에 관한 연구", 한국정밀공학회 논문집, Vol. 14, No. 2, pp.84-91, 1997.
5. Rumbaugh, J., Blaha, M., Premerlan, W., Eddy, F., Loresen, W., Object-Oriented Modeling and Design; Prentice-Hall, 1991.
6. Machinability Data Center, Machining Data Handbook second Edition; Metcut Research Associates, Inc., 1972.
7. Ahmad, S., Tesauro, G., "Scaling and Generalization in Neural Network: A case study", Advances in Neural Information Processing System 1, Morgan Kaufman, Palo Alto, CA. pp.160-168, 1989.