

Process Planning Using An Integrated Expert System And Neural Network Approach¹

Mark Wilhelm, Alice E. Smith² and Bopaya Bidanda

Department of Industrial Engineering
University of Pittsburgh
1048 Benedum Hall
Pittsburgh, Pennsylvania 15261 USA

A Chapter in *Hybrid Intelligent System Applications* (Jay Leibowitz, Editor)

¹ Part of this work was supported by the National Science Foundation under grant DDM-9209424.

² Corresponding author.

Process Planning Using An Integrated Expert System And Neural Network Approach³

Mark Wilhelm, Alice E. Smith⁴ and Bopaya Bidanda

Department of Industrial Engineering
University of Pittsburgh
1048 Benedum Hall
Pittsburgh, Pennsylvania 15261 USA

Abstract. This chapter presents a unique computer aided process planner for metal furniture assembly, welding and painting using a rule based expert system integrated with an artificial neural network. The process planner creates parts lists, process plans and estimates standard times for individual product variations using input of product number or selection of product features. Although fundamentally a variant process planner, the rules and neural network allows some generalization capability to new products. This demonstrates that a composite intelligent approach can be useful for process planning in real manufacturing situations.

1. Introduction

The development of process plans and the determination of standard processing times are essential functions for many manufacturing organizations. These functions are time consuming and require significant skill and/or a great deal of experiential knowledge. To fully or partially automate these functions would certainly provide very tangible benefits to many organizations. This study deals with the use of neural network and expert system computing tools in an integrated fashion to effectively perform process planning and process time prediction activities. The study used a metal furniture manufacturing environment as a test bed for the prototype system.

1.1. Process Planning Basics

Process planning is the activity of taking the product design or specification, which is defined in terms of the design engineer's terminology (e.g., size, shape, tolerances, finish, and material properties/treatment), and transforming it into a detailed list of manufacturing instructions which are stated in terms that are useful to manufacturing personnel (e.g., specifications for materials, processes, sequences and machining parameters). The process plan is the recipe that is followed by all shop floor personnel to produce the desired end product; as such the process plan plays a key role in the manufacture of high quality and cost efficient products. For many products there is no unique manufacturing method and many variations of the process plan could provide an acceptable end product. However, the process plan chosen can have a significant impact on the manufacturability of the product. Developing a good process plan given the current manufacturing system constraints requires extensive experiential knowledge and is a time consuming process.

³ Part of this work was supported by the National Science Foundation under grant DDM-9209424.

⁴ Corresponding author.

Computer-aided process planning (CAPP) systems have been devised to help simplify, improve, and provide consistency within the process planning function. Computer-aided systems have the potential to capture and retain the experiential knowledge of expert process planners which may have taken years to develop. Furthermore, capturing this expertise in a knowledge base provides the ability to easily replicate process plans and expertise. There are two basic types of CAPP systems, variant and generative. The variant approach groups parts into families based on geometric similarities and stores standard process plans for each family. The standard plans can be retrieved automatically and annotated to conform to the specific product of interest. The generative approach uses a detailed feature description and decision logic to synthesize process plans, optimizing for each individual part and tool.

1.2. Standard Process Time Basics

A standard process time for a given manufacturing process is the average amount of time for the average worker to complete the operation. Standard process times are typically determined via the use of detailed time studies of the operation. The time studies involve repetitively timing the operation with various operators and making some subjective assessments about the operator's performance. Time studies are time consuming and tedious, however, the availability of standard process times is a valuable tool for many manufacturing organizations. Standard process times can be used to balance the flow of products through a manufacturing cell, estimate production costs, predict throughput times, and evaluate operator performance.

When the members of a product family vary significantly in the amount of time required to complete certain manufacturing operations the effort required to determine standard process times increases dramatically. This is because a single standard time does not apply to all members of the family, and thus a standard time for each specific product variation must be determined. Since determining a standard process time involves repetitive timing cycles, the determination of standard times for all products in the family can quickly become impractical. The ability to generalize the time study data taken from a small sample of the product family to all other members of the product family would provide a significant reduction in the effort required to develop good standard times. This is the approach which is taken in this chapter.

1.3. Neural Network Basics

Artificial neural networks are massively parallel computing mechanisms modeled after the biological brain. They consist of nodes, linked by weighted connections. Neural networks are built hierarchically in layers with connections between layers, and sometimes within a layer. Neural networks learn relationships between input and output vectors by iteratively changing interconnecting weight values until the outputs over the problem domain represent the desired relationship. This empirical learning is an attractive alternative to the slow process of knowledge acquisition for expert systems or the explicit coding of algorithmic systems. The resulting "trained" network consists of fixed weights and parameters which will then produce outputs for new inputs. This process of translating new inputs into outputs is referred to as recall. Neural networks are noted for their ability to perform with incomplete and noisy data, and their ability to generalize their learned representations to handle new circumstances. This last quality makes neural networks an attractive tool for process planning and process time prediction activities, where each new product is likely to differ somewhat from past ones.

1.4. Expert System Basics

Expert systems capture knowledge about a given domain in the form of a knowledge base of rules and facts. In this way expert systems have the potential to duplicate and standardize human expertise which may have taken years of experience to develop. Expert systems are quite different from traditional algorithmic computer programs in the way in which knowledge is organized. Expert systems have three distinct modules of information 1) rules, 2) facts, and 3) control strategy. This organization allows for the ready adaptation of the expert system architecture to various problem domains. These intelligent systems have proven effective in many problem domains. However, there are well established drawbacks of expert systems including the difficulty and effort involved during the knowledge acquisition phase, and the brittleness of the rule base in regard to handling new or modified features.

The control strategy for an expert system is the means by which attribute values are pursued. The control strategy may be forward chaining, backward chaining, or a mixture of both. Forward and backward chaining refer to the method in which IF-Then rules are processed. During forward chaining whenever an attribute changes value all rules which have that attribute as the antecedent will be evaluated. For any of these rules, if the IF portion of the rule evaluates as true then the rule will fire and conclude one or more attribute values. These recently changed attributes will similarly be evaluated in all other rules in which they appear in the antecedent and so on in a chained fashion. During backward chaining an attribute value is pursued via locating all rules in the knowledge base which conclude that attribute. Each of these rules is then evaluated by examining the antecedent of the rule. The values of these attributes are then determined by evaluating all other rules which conclude these attributes. This process is followed in a chained fashion until the chain reaches a known value.

2. Previous Related Research

The automation of the process planning function using various software approaches has been and continues to be an active research area. Expert systems have been studied in depth and implemented to perform some process planning functions. Gupta and Ghosh [1] provided a survey of expert systems in manufacturing and process planning. This paper presents various applications of expert systems to the problem domain of industrial and manufacturing engineering. Each system is discussed in terms of the problem definition, implementation scheme, and special features. Another survey of expert system approaches to process planning was published by Gupta [2] which discusses available process planning expert systems. The important features and limitations of the various systems are presented, with respect to their part design input scheme, knowledge base representation, and control strategy. Other approaches to the automation of process planning have also been studied including using a relational data base management approach. Wang and Walker [3] presented a framework for the creation of an intelligent process planning system within the relational data base management system.

Automating the CAD / CAM link has been the topic of considerable research and development efforts. Madurai and Lin [4] provide a discussion on the automatic extraction and recognition of part features directly from a CAD database. The objective of the study was to develop an intelligent feature extraction methodology and to implement it using currently available CAD software. Wang and Wysk [5] consider that an expert system technique for CAPP may replace traditional generative CAPP methods for various reasons. These include 1) the organization of knowledge in an expert system (i.e., facts, rules, and control strategy) provides for much easier modification when compared to traditional

computer systems, 2) decision trees and decision tables, which are often used in traditional generative CAPP systems, work effectively only for simple decision making processes, and 3) expert CAPP systems may be designed so that knowledge can be accumulated as time passes, which is not a capability of current variant and generative systems. Wang and Wysk [5] discuss an intelligent process planning system called Turbo-CAPP which has the ability to 1) interpret and extract surface features in a wireframe-based CAD database, and 2) perform intelligent reasoning for process planning.

Automated assembly planning is an important activity in implementing an integrated manufacturing system. As such, Lin and Chang [6] presented a algorithmic methodology for automatic generation of assembly plans for three-dimensional mechanical products. The intent was to develop an efficient method to describe and plan the assembly of mechanical products. The method uses a two state planning scheme which considers only the geometric constraints first and then considers sequence constraints derived from non-geometric properties. The system infers mating and collision information from the assembly solid models and uses a frame-based representation scheme to explicitly represent the assembly non-geometric information. using expert system rules.

As discussed above the vast majority of research in the area of automated process planning (including machine parameter and sequence planning and assembly sequence planning) has focused on the use of expert systems, traditional generative process planning techniques, or other approaches, such as the use of relational database technology. Recently some emphasis has been placed on implementing neural network technology in the development of automated process planning systems. Knapp and Wang [7] formulated an approach for the automated acquisition of process selection and within-feature process sequencing knowledge using neural networks. The authors in this reference cite several reasons why the use of neural networks are a preferable alternative to expert systems for process planning: 1) the expert system knowledge acquisition process is time consuming, costly and error prone, 2) the systems can not adapt to change in manufacturing practice and technology, 3) the expert systems are brittle in nature (i.e., with sharp boundaries of application), and 4) expert systems cannot generalize from past experience to handle new cases. Knapp and Wang [7] used a back-propagation network to map spur gear attributes or features into feasible machining operations, and the selection of only one machining operation at a time was forced using a MAXNET network. The network used a recurrent input to keep track of the previous machining operation. Chen and LeClair [8] presented another neural network approach to process planning. In this reference the authors discussed the use of an unsupervised neural network in machine setup generation. Intersecting and non-intersecting features within a setup are identified and classified using an associative memory.

Chen and Pao [9] presented an approach to the design and planning of mechanical assemblies using an integrated neural network and rule-based system. The system first uses an unsupervised neural network learning algorithm in the design stage to cluster similar conceptual designs which provides the input interface to the design cluster memory. The rule-based portion of the system then accesses this design cluster memory as required while performing its role in the generation of assembly plans (i.e., performing the functions of pre-processor, liaison detection, obstruction detection, and plan formulation). The system attempts to utilize the strengths of both neural network and rule-based modules to provide a more powerful single system, which is similar to approach taken in work presented herein.

3. Project Objectives and Scope

The objective of this project was to develop a tool which could perform process planning and process time prediction functions automatically. The result was intended to be a user

friendly software package which utilizes expert system and neural network computing methods in a fashion which is transparent to the user. The prototype system was developed for a metal furniture manufacturing environment to ensure that the problems and difficulties associated with a real manufacturing environment would be factored into the system and to illustrate that an integrated system is implementable.

3.1. Project Scope and Manufacturer

Haskell of Pittsburgh was selected as the test case for the prototype system. Haskell produces metal office furniture including pedestals, desks, file cabinets etc. and is considered representative because of its mainstream product line. Due to the extensive line of products produced at Haskell and the significant product variations possible, the prototype system was limited to only a small subset of the products manufactured there. It was determined that the Cube Pedestal product line would provide a good test case for the prototype system. This product line provided a manageable subset of products while still introducing significant variation, resulting in a non-trivial yet manageable problem.

The Cube Pedestal is a stand alone, desk high, filing cabinet. There are significant variations available within this product line including two different heights (executive and typing), three depths (20, 24, and 30 inches), various combinations of drawer types (four drawer types are available: tray, box, file and EDP) and configurations (i.e., eight standard drawer configurations for a executive height pedestal and three standard drawer configurations for a typing height pedestal), various drawer handle and filing accessory options, several lock and support options, and many paint color choices.

Within the Cube Pedestal product line additional product variations are feasible which are not currently available and which would not require changes to the basic components which make up the assembly. Specifically, many different drawer combinations which are currently not a standard option are feasible via altering only the quantity of parts required and the assembly process. Altering the process plan to accommodate these special order cases is a good test of the process planner's ability to generalize.

The scope of this project was further limited to the final assembly area which includes the weld, paint, assembly and packaging processes. The final assembly processing sequence is divided into 14 sub-processes as shown in Figure 1 and is briefly described below. There is only limited flexibility in the sequencing of these major sub-processes due to the inherent nature of the assembly process (i.e., certain processes must necessarily precede others). Haskell was interested in development work in this area for two reasons; 1) no written process plans currently exist for the final assembly area and 2) only limited time studies have been completed in this area.

3.2. General Approach

Once the project scope had been sufficiently limited and defined the following five step general approach was followed:

3.2.1. Document The Current As-Is Process

As stated above, no written process plans exist for the final assembly area. Currently the specific processing steps completed for each of the 14 major sub-processes are determined by an experienced operator. A brief description of each of these 14 sub-processes follows:

MIG WELD NUTS TO BOTTOM CHANNEL - Two bottom channels form the support base for the Cube Pedestal. In this process two nuts are MIG welded to each bottom channel. These nuts are used to attach the glides to the bottom of the pedestal. Three lengths of bottom channel exist to support the three available pedestal depths.

FORM WRAPPERS - In this process the pre-cut sheet metal blank is bent to form the outer surface or covering for the pedestal (i.e., the sides and back of pedestal). The tabs which are used to weld the top cap are also formed in this process. Five different sheet metal blank sizes are available to support the different pedestal sizes.

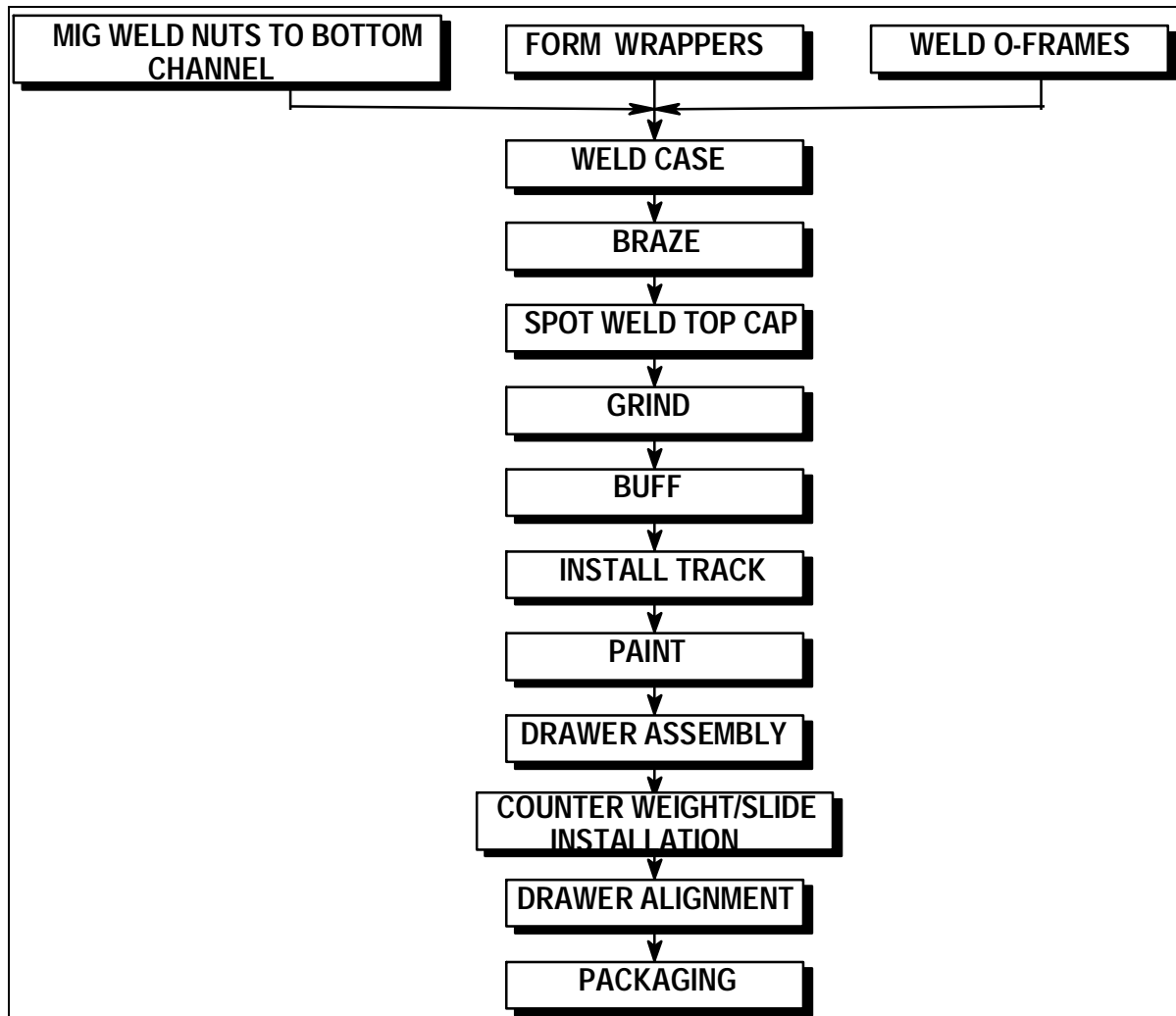


Figure 1. Overview of Cube Pedestal Process.

WELD O-FRAMES - The o-frames provide the inner support structure for the pedestal. There are two o-frames in each Cube Pedestal assembly, one near the front and the other near the back. The o-frame is a rectangular structure formed by spot welding two vertical stiffeners to two horizontal stiffeners. All four stiffeners are loaded into an adjustable fixture and the welds are made simultaneously by machine. Two sizes of o-frame are available to support the two pedestal heights.

WELD CASE - In this process the wrapper, o-frames, bottom channels, and other stiffeners and plates are welded together to form the shell of the Cube Pedestal. An appropriate

assembly fixture is used to obtain the proper alignment between the piece parts for the particular product variation being built.

BRAZE - In this process the top cap (i.e., top of the pedestal) is positioned on the pedestal case and brazed in all four corners. Some additional brazing of support structures and the bottom front corners of the pedestal is also completed.

SPOT WELD TOP CAP - In this process the top cap is spot welded along both sides and the back of the pedestal shell. The quantity of welds depends on the particular variation in size of the product being built.

GRIND - The grind operation removes excess braze from all the externally visible brazed locations. This is a cosmetic operation to ensure the final pedestal surfaces are smooth.

BUFF - The buff operation is also a cosmetic operation to prepare the surfaces, which have been spot welded, for the painting operation.

INSTALL TRACK - In this operation the runners (i.e., which support the pedestal drawers) and the lock bar(s) are installed. This installation requires proper positioning of the parts based on the product variation being built, making hook and slot connections, and installing holding screws.

PAINT - The pedestal case and drawers are painted in this process. Due to the inaccessibility of this process it was not documented in detail and is not supported by the process planning and standard time prediction system.

DRAWER ASSEMBLY - When the drawers reach this process they are in the general configuration of a drawer, i.e. they are built elsewhere. In this operation various options are added to the drawers, such as the hanging mechanisms for file folders. Also the drawers are cosmetically touched-up to remove excess paint and to touch-up areas missing paint.

DRAWER SLIDE AND DRAWER INSTALLATION - In this operation the casters or glides, drawer counter weight, drawer roller mechanisms, and various bumpers and sound deadeners are installed. Subsequently, the drawers are installed.

DRAWER ALIGNMENT - In this operation various adjustments are made to ensure that the pedestal shell is square, that proper spacing exists around all edges of the drawers, and that the drawers move in and out smoothly.

PACKAGING - The completed product receives a final inspection and is packaged for shipment in this operation.

Each of the processes was documented via detailed interviews with the operators and observation of the process. It was observed that the operators based their decisions concerning which processing steps were required on their past experience and on basic written instruction detailing the type and quantity of final product required. During this documentation phase it became evident that each of the 14 major sub-processes consisted of a core or standard process, which was the same for all variations of the product line, plus a small number of key processing steps, which were dependent on the specific product variation selected. The specific processing steps and the reasoning or rationale behind the decision to perform that particular processing step was documented for all the major sub-

processes. For example, in the Weld Top Cap process the operator determines the required number of spot welds to make on each side of the top cap based on observing the depth of the pedestal (i.e., 20, 24, or 30 inches) or by referring to a computer output for the pedestal depth. Knowing the depth of the pedestal is sufficient information to determine the required number of spot welds. This type of reasoning/action process was extracted from the operator interviews.

3.2.2. Develop Standard or Core Process Plans

Based on the documentation acquired from the operator interviews a set of core process plans was developed, one for each of the 14 major sub-processes. These core process plans consisted of the detailed processing steps common to all product variations within the Cube Pedestal product line plus key words which identified where the process plan required modification for specific product variations. The types of information that is specific to the product variation selected includes machine settings, appropriate fixtures, and number of welds. Each of these core process plans were stored in a separate text file. The word *VARIABLE* were used as key words for a process and indicate that a manufacturing step must be inserted in these locations which is specific to the product variation under consideration.

3.2.3. Develop Knowledge Base Rules

A set of rules was developed which conclude the appropriate processing steps to be inserted into the core process plan for a specific product variation. In general the core process plan is updated with the specific processing steps appropriate to the product variation under consideration. The rules determine the correct processing step to be inserted based on the attributes of the product variation under consideration. It was found that simple rules were sufficient to correctly select the product specific processing steps. Rules were developed which could correctly determine these product specific processing steps for two cases, 1) when the product selected was a standard or currently available product variation and 2) when the product selected was a non-standard or currently unavailable product variation.

It must be noted that new products (i.e., non-standard or currently unavailable products) must be within the physical constraints of the available piece parts. Specifically, any new combination of drawers could be selected with the constraint that the overall dimensions of the pedestal are not changed. For example an executive height Cube Pedestal with 8 tray drawers, which is a non-standard product variation that is not currently available, could be generalized by the system because the 8 tray drawers would fit in the executive height Cube Pedestal. However, specifying a typing height Cube Pedestal with 8 tray drawers is not acceptable to the system because the 8 tray drawers physically will not fit into the typing height pedestal. Thus any product variation can be generalized by the system provided it does not alter the basic piece parts required for the assembly and is physically feasible.

3.2.4. Evaluate Standard Processing Times

Recent time studies completed at Haskell provided timing data for 8 of the 14 sub-processes. Evaluation of this data and discussions with Haskell personnel indicated that most of these sub-processes had relatively static processing times across all product variations. Specifically, there is minimal or no variation in the time required to complete the processes as a result of the specific product variation being considered. The variation in the time data is attributable to such things as operator, equipment and part variability. Thus,

for most of the sub-processes the standard processing time is applicable to all product variations. For these cases the development and use of standard processing times is a straight forward look up table.

In general, standard processing times do not behave as nicely as those noted above. The standard processing time required to complete a given operation is typically highly dependent on the product variation selected. This is the case for the Install Track sub-process of the Cube Pedestal assembly operation. Depending on which product variation is selected, the type, quantity, and installation positions of the piece parts involved in this process are highly variable and thus the associated time required to complete the process is highly variable. Haskell had not performed time studies on this process for this reason. Timing studies performed on one variation of the product line are not in general applicable to other product variations. Thus if accurate time data is desired, then time studies of all the product variations would be required. When significant product variations are involved this can become a very time consuming and impractical task.

Use of a neural network to generalize the standard time data from a small subset of the possible Cube Pedestal products to all the remaining Cube Pedestal products could greatly simplify the determination of standard process times for the Install Track process. This method would require that time studies be performed on only a limited number of representative product variations. Also for non-standard products the neural network could be used to estimate standard process times for products which had not yet been built. This was the approach taken in this study. As discussed in more detail below, a neural network model was developed and tested utilizing a training and test set of simulated data. A similar neural network model is planned using actual data collected by Haskell.

3.2.5. Develop Software System

A software system was developed which consists of an interactive expert system that acts as user interface, procedural data base, inference engine, and system integrator. Exterior to the expert system are the neural network component and external data files. Figure 2 provides the system architecture. The expert system utilizes both forward and backward chaining reasoning together with developer designed procedures to accomplish the specified system tasks. The expert system development software used was Level 5 Object version 2.5, running under Windows 3.0, and the neural network development software used was BrainMaker, version 2.0.

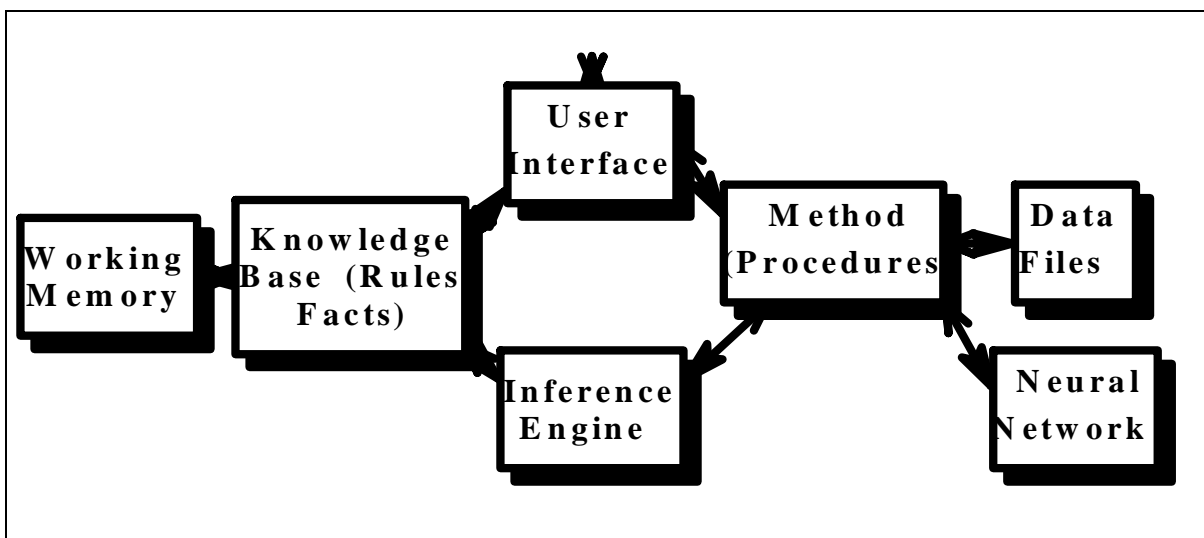


Figure 2. System Architecture.

4. System Description

4.1. System Architecture

The system has three primary functions 1) develop detailed process plans, 2) develop parts lists, and 3) determine standard process times for each major sub-process. Each of these functions are completed for the specific product variation selected. To perform these tasks the system utilizes an expert system, a neural network, and external data files. The User Interface, which is a user friendly set of input screens, can set current facts in the knowledge base or activate methods. Facts are simply attributes for which the current values are known with certainty. For example, when the attribute "height" of the Cube Pedestal class is selected by the user to be 24 inches this is stored as a fact in the working memory. The fact can then be accessed throughout the consultation when the value of that attribute is required. A method is a developer designed procedure which can perform various tasks and/or set attribute values based on a series of If-Then clauses. Methods may have loops and nested Ifs which make them different from basic rules. A simple example of a method is shown below.

```
WHEN CHANGED
  BEGIN
    IF drawer option OF cube pedestal = "Other" THEN
      ASK drawer option choice display
    ELSE
      ASK Cube Ped Process Flow display
  END
```

This example shows a WHEN CHANGED method. WHEN CHANGED methods are attached (or associated with) a given attribute; whenever that attribute changes value the WHEN CHANGED method is activated and it completes its procedure. The above method simply checks the value of a certain attribute to determine if it equals the string value "Other" and activates the appropriate display screen.

The Methods in this system perform four primary functions 1) access and read/write to external data files, 2) access, provide input vector, and run the neural network in recall mode, 3) activate the expert system Inference Engine, and 4) perform various tasks required by the system, e.g., print files etc. The Inference Engine is either activated by a Method to pursue a given attributes value via backward chaining or is activated in a forward chaining mode when certain attribute values are changed by the user. The knowledge base consists of a set of backward chaining rules (called Rules) and a set of forward chaining rules (called Demons). Finally the working memory keeps track of the current values for all attributes.

4.2. Detailed System Description

Figure 3 shows the program flow diagram for the system. Upon starting a consultation the user is greeted with an introduction screen and is provided access to a system information screen which briefly describes the system. When the user continues, a prompt to select a product line is displayed. After the Cube Pedestal product line is selected a product specification screen is displayed. This screen is specific to the Cube Pedestal line and provides all its attributes with the corresponding possible values. These attributes are:

- Series Option
- Drawer Pull Option
- Case Height Option
- Case Depth Option

- Lock Option 1
- Support Option
- Drawer Option
- Paint Color Option
- Filing Accessories Option
- Lock Option 2

Via the use of radio buttons the user can select the desired value for each option. If incompatible options are selected for the various attributes, an error message is displayed and the user is required to modify the selection. The attribute values selected in this screen are set for this consultation in the system's Working Memory via firing a set of backward chaining rules. This product specification screen can be used for both standard products (i.e., currently available product variations) and non-standard products (i.e., special product variations which are not currently available).

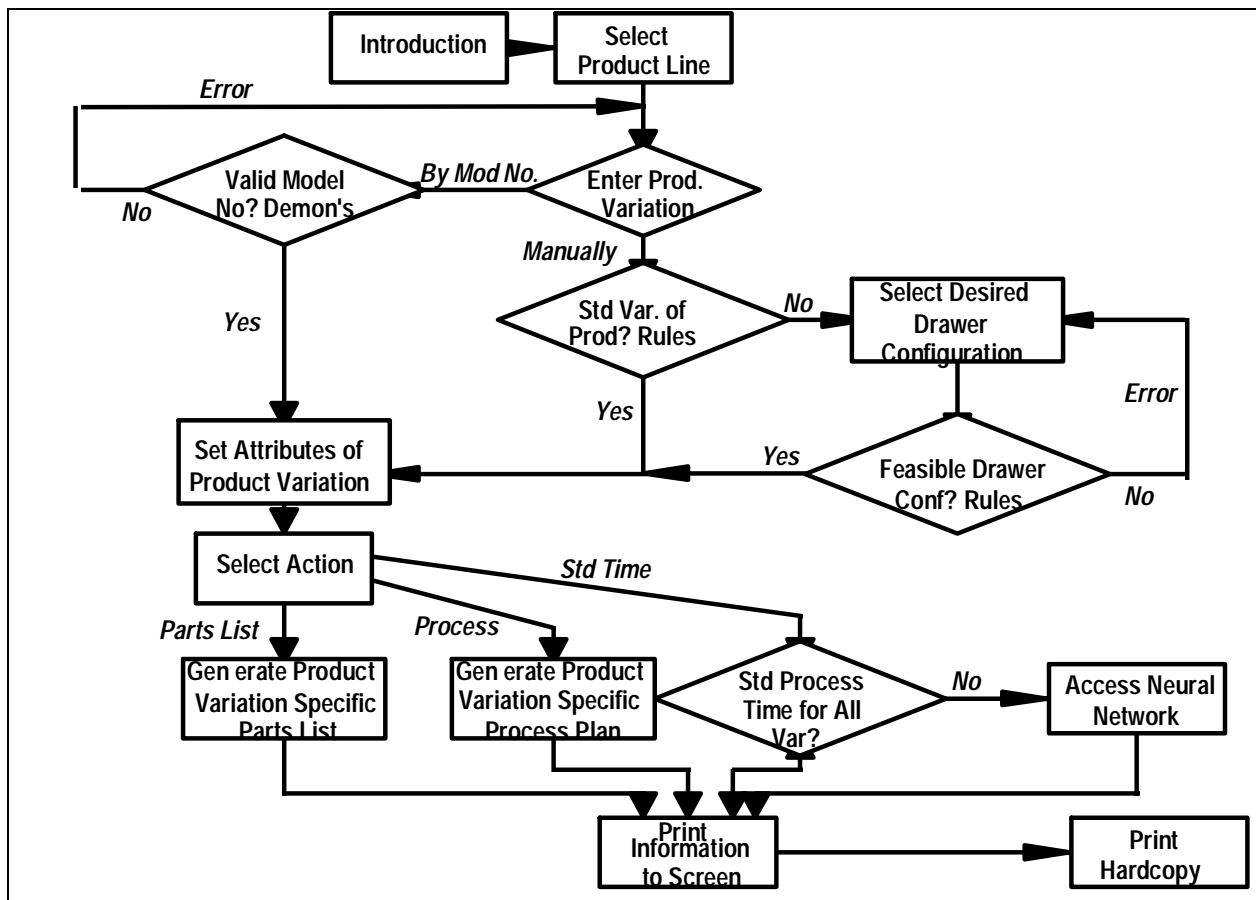


Figure 3. Flow Chart of User Consultation.

An alternative approach to specify the product variation is to directly input the product's serial number. This option is obviously only available for standard product variations. This selection calls another input screen which accepts the product's serial number. When the product's serial number is input a series of checks are run by the system to determine if the input is a valid serial number; if it is not valid, an error message is displayed and the user is prompted to modify the serial number input. When a valid serial number is input, a set of forward chaining demons is fired which translates the serial number input to the appropriate attribute values and sets these values in the system's Working Memory similar to the above.

If a non-standard product variation is selected in the product specification screen, two additional input screens are required to be completed by the user. The first screen requests the User to specify the quantity of tray, box, file, and EDP drawers that are desired in the non-standard product. Upon completing this screen a series of checks is performed by the system to determine if the chosen number of drawers is feasible. If the quantity of drawers selected is infeasible, an error message is displayed and the user is required to re-enter the input. The second screen uses a series of radio buttons to allow the user to select the desired configuration of the drawers. A series of checks is also run on this input to ensure the input form has been completed and that the configuration selected is consistent with the quantity of drawers specified in the previous screen.

All the above discussed screens perform functions required to set the initial conditions for the current consultation (i.e., defining the product variation under consideration and setting the values of appropriate variables in the Working Memory). After these necessary steps have been completed the system is prepared to perform its three primary functions. The system provides the user with a process flow diagram display which shows each of the 14 major sub-processes in their appropriate sequence. This screen is for information only and provides no other function. Upon continuing from this screen the main action screen of the system is displayed. From the main action screen the user can activate the three primary system functions one at a time:

- 1) Develop detailed process plan
- 2) Develop detailed parts list
- 3) Determine standard process time

The part type and quantity attributes are determined via backward chaining through the knowledge base. The output screen has two functions, a) the user can view the parts list and b) the user can print a hard copy of the parts list.

Prior to performing the other two system functions (i.e., determining a detailed process plan or determining standard process times) the user must specify which of the major sub-processes is of interest. The user makes this selection via a radio button group which is part of the main action screen display. The default selection is ALL SUB-PROCESSES. After making the selection the process plan and processes time prediction functions will, when executed by the user, pertain to this sub-process.

Selecting the "Detailed Process Plan" push button fires a method which

- 1) checks to see which sub-process has been selected,
- 2) opens the external data file which holds the core process plan for this sub-process,
- 3) reads the core process plan file line-by-line,
- 4) if the line is standard (i.e., applicable to all product variations) it is written to a text box,
- 5) if the line read contains a key word then the inference engine is activated in a backward chaining mode to determine which processing step should be inserted at this location,
- 6) the correct processing step is written to the text box,
- 7) an output screen display is activated which shows the complete process plan for the sub-process selected.

Selecting the "Standard Time" push button fires a method which:

- 1) Opens the external data file which holds information about the Cube Pedestal assembly standard process times.
- 2) Reads the standard time file line-by-line and stores the data as attribute values within the knowledge base. This data is stored in the order which corresponds to the major sub-processes. The data on file can be one of three types: a) the actual standard process time (i.e., for those sub-processes which have a constant standard time for all product variations), b) a code which designates that no standard time information is currently available for that sub-process, and c) a code which indicates that the standard process time for that sub-process must be estimated by a neural network.

- 3) Checks to see which sub-process has been selected and displays the appropriate information retrieved from the standard time file to a value box in the main action screen.
- 4) A forward chaining demon is attached to the value box which performs a check each time the value in the value box is changed. If the value is a constant standard time or is the code which indicates no standard time data is available then no additional action is taken. If the value is the code that indicates that a neural network is required to estimate the standard time then another Method is fired to generate the input vector for the neural network and to activate the network. See 4a below.
- 4a) A Method is executed which generates an external text file that holds the input vector for the neural network. This input vector is a 28 element binary vector. The values of this input vector are set by backward chaining through a series of rules which conclude portions of the input vector based on the attributes of the product variation selected.
- 5) The neural network is executed, taking the input vector stored in the text file and processing it in recall mode to estimate a standard process time for the particular product variation under consideration. See the neural network discussion below for further details. The output of the neural network is stored in an external text file.
- 6) Reads the contents of the neural network output file (i.e., the estimated standard process time) and displays the value in the main action screen display.

4.3. Expert System

As discussed above the interactive expert system acts as user interface, procedural data base, inference engine, and system integrator. The actual code for the expert system is written in LEVEL5's Production Rule Language (PRL) which is analogous to the source code for a FORTRAN or C program. As can be seen the expert system PRL file consists of the following major elements:

71 DEMONS - Demons are forward chaining rules which are fired each time the antecedent of the demon changes value. An example demon which is used to translate a portion of the Cube Pedestal serial number into a specific Cube Pedestal attribute value is:

```

DEMON model number conversion 017
IF UPCASE(drawer option OF cube ped serial numbers) = "L"
THEN drawer option OF cube pedestal := "4 Box"

```

A series of similar demons are used to translate a product serial number into specific attributes. These demons are only required when the serial number input option is used.

127 RULES - Backward chaining rules are fired whenever the rule's concluding attribute corresponds to the attribute currently being pursued directly or in a chained fashion. An example rule is:

```

RULE qty glides 1
IF series OF cube pedestal = "Haskell"
OR series OF cube pedestal = "System One"
AND support option OF cube pedestal = "Glide"
THEN qty glides OF cube ped parts := 4
AND glide OF cube ped parts := "35181"
ELSE qty glides OF cube ped parts := 0
AND glide OF cube ped parts := "35181"

```

This rule determines the part number and quantity of glides required based on the attributes selected for the particular product variation selected.

36 WHEN CHANGED METHODS - When changed methods are developer designed procedures which execute each time the attribute attached to (or associated with) the method changes value. An example when changed method is:

```
WHEN CHANGED
BEGIN
  finished := FALSE
  WHILE (finished = FALSE)
  BEGIN
    filename OF file := "d:brainRTS.out"
    action Of file IS open old := TRUE
    read line OF file := TRUE
    neural network time OF cube pedestal time window := TO NUMERIC(current line OF file)
    IF neural network time OF cube pedestal time window <> 0
    THEN finished := TRUE
    ELSE
      action OF file IS close := TRUE
    END
    action OF file IS close := TRUE
  END
```

Whenever the attribute associated with this method is changed the method opens the output file which contains the neural network output. The method continually opens and reads the file using a loop, until a value is placed in the output file by the network. In this way the expert system pauses until the neural network has completed its processing.

9 WHEN NEEDED METHODS - When needed methods are developer designed procedures which execute each time the attribute attached to the method is being pursued by the inference engine. An example when needed method is:

```
WHEN NEEDED
BEGIN
  FOR (count := 2 TO 9)
  IF run position[count] OF other drawer conf = 1 THEN
    pos1 OF network input := 1
  END
```

This when needed method is executed each time the attribute "pos1" (i.e., of the class called network input) is being pursued. The method loops through all eight elements of the array attribute called run position[] to determine if any of its elements has the value of 1. If so the value of attribute pos1 is set equal to 1.

26 USER OBJECTS - User objects are used to define all aspects of the expert system application. Each object has a set of attributes which define the object. The attributes may be numeric, simple, compound, multi-compound, or string valued. An example definition of a user object is:

```
CLASS cube ped process
WITH macro process COMPOUND
  MIG Weld,
  Form Wrappers,
  Weld OFrames,
  Weld Case,
  Braze,
  Weld Top Cap,
  Grind,
  Buff,
  Install Track,
```

Paint,
Drawer Assembly,
Slide Assembly,
Drawer Alignment,
Packaging,
All Macro Processes
DEFAULT All Macro Processes

This definition defines an object or class called cube ped process which has one compound attribute called macro process. The value of this compound attribute can only be one of those listed in the definition. Also for this definition the default value of the macro process attribute is set to All Macro Processes.

223 SYSTEM OBJECT INSTANCES - System objects are defined within LEVEL 5. New attributes for the system objects may not be defined, however, new instances of the system objects can be defined to develop the system (e.g., text boxes, push buttons, and value boxes which make up the display screens are all instances of system objects). An example definition for an instance of a system object is:

```
INSTANCE promptbox 10 ISA promptbox
WITH location := 90,50,130,80
WITH justify IS left
WITH frame := TRUE
WITH show current := TRUE
WITH attachment := drawer pull OF cube ped serial numbers
```

This is a definition for the prompt box instance called promptbox 10. This prompt box prompts the user for the value of the attribute drawer pull (i.e., of the user defined object called cube ped serial numbers). The location, justification, and other characteristics of how the prompt box should appear on the computer screen are also defined here.

For this particular application the expert system was designed such that the inference engine was activated on a when needed basis. Specifically, the inference engine is activated in a forward chaining mode whenever the antecedent of a demon changes value or in a backward chaining mode whenever a Method pursues the value of an attribute. This approach was considered appropriate for this application due to the various functions performed by the system and the various methods the user can use to input information to the system (e.g., via entering the product serial number or by selecting specific attributes).

4.4. Neural Network Component

A neural network was developed which is used to predict the standard process time for the Install Track sub-process. The processing time required to complete this operation varies significantly with the specific product variation selected. If the other sub-processes in the Cube Pedestal assembly operation were similarly variable then a neural network for each sub-process could be developed in the same way. A separate network for each major sub-process is required because the product variation selected may impact the process times differently for each sub-process. The function of the neural network model is to capture the relationship between the product variation selected and the associated processing time required. Figure 4 shows the neural network architecture.

A six step approach was followed to develop, train, and test the neural network.

STEP 1: Determine All Variables Which Affect The Processing Time

A review of the Install Track process was completed to determine which variables could have an impact on the time required to complete that process. Based on operator interviews, observations of the Install Track process, and engineering judgment the following product specific variables were determined to have a potential effect on processing time.

- HEIGHT ----- Height of the pedestal
- DEPTH ----- Depth of the pedestal
- LOCK ----- The lock option selected
- RATIO ----- The mixture of Box/Tray runners and File/EDP runners to be installed
- POS1-POS8 ---- The relative positions of the runners in the vertical stiffener
- QTY B/T RUN - Quantity of Box/Tray Runners required
- QTY F/E RUN - Quantity of File/EDP Runners required

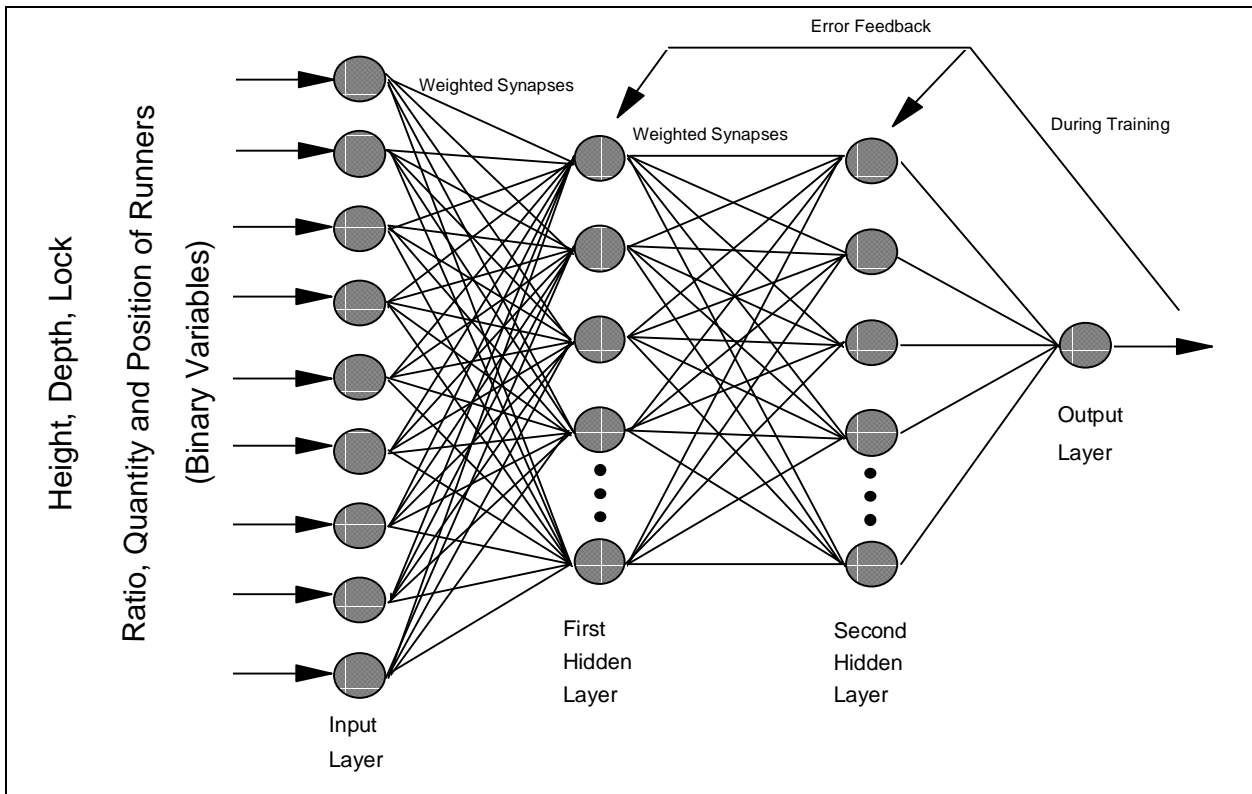


Figure 4. Neural Network for Process Time Estimation.

STEP 2: Encode Network Input

The above identified input variables were encoded into binary (0-1) values. A binary encoding approach was sufficient for this problem domain because the possible values for the input variables were finite and relatively small. Specifically, because the problem domain is defined as all the Cube Pedestal product variations which can be made with the currently available piece parts only, the number of feasible product variations is finite. The neural network can predict standard process times for any of these feasible combinations.

The HEIGHT variable has two possible values, the DEPTH variable has three, the LOCK variable has three, the RATIO variable has two, the QTY B/T RUN variable has nine, the QTY F/E RUN variable has three, and each of the POS1 through POS8 variables have two possible values. Together this results in a neural network input vector with 28 bits. The general form of the input vector has fourteen variables:

VECTOR = {HEIGHT, DEPTH, LOCK, RATIO, QTY T/B RUN, QTY F/E RUN, POS1, POS2, POS3, POS4, POS5, POS6, POS7, POS8}

Thus with the specific example of:

HEIGHT = Executive

DEPTH = 24

LOCK = No Lock

RATIO = All Same

Product Variation = 3 Box and 2 Tray drawers (i.e., this in turn determines the variables QTY B/T RUN, QTY F/E RUN, and POS1 thru POS8)

the input vector to the neural network would be:

VECTOR = {0,010,001,0,000001000,100,1,1,0,1,0,1,0,1}

The output from the neural network is a single continuous variable which represents the processing time in minutes.

STEP 3: Determine Number Of Possible Product Variations

Not all of the product options have an impact on the Install Track process (e.g., the color of paint option has no impact on this process). It was determined that only the pedestal height and depth, lock, and drawer configuration options would impact the Install Track process. Evaluating the possible combinations of these options resulted in a total of 72 product variations that could impact the Install Track process. Therefore there are potentially 72 different standard process times for this process. This only includes the currently available product variations. If the feasible yet non-standard product variations are considered the total number of product variations increases dramatically.

This case illustrates the potential benefit of using a neural network to predict standard process times. In theory a neural network model could be developed which can generalize the timing data collected on only a small sample of the total possible product variations to all the remaining variations. This could eliminate the need to perform time studies on all product variations while still providing accurate standard time data for each individual product. Furthermore, standard process times for new products could be predicted with the system prior to building the new product.

STEP 4: Collect Data

For this study two cases were planned, one in which the timing data was simulated and one with actual time study data. To develop realistic data for the simulated data case a set of assumptions was developed. These assumptions attempt to capture the effect of selecting each Cube Pedestal option on the Install Track processing time required. For example the particular drawer option selected determines the specific positions in which drawer runners need to be installed. The level of difficulty and thus time required to install drawer runners is dependent on the position the runner is installed. Therefore, a set of penalty times were developed for each runner position. A similar set of assumptions and penalty times were developed that correspond to the other available Cube Pedestal options.

The above simulated time factors were used to calculate a standard time for all the 72 product variations discussed above. This data set was used as the training and test set for the simulated data case. For this case the neural network's function was to learn the assumed relationship between the product options selected and the standard processing time.

The actual time study case has not been completed to date. The process of collecting standard time data has not yet been completed due to production priorities at Haskell. The sample of data to be collected is anticipated to be spread over the possible product variations.

STEP 5: Develop And Train Network

All networks developed for this study used fully connected multi-layer perceptrons trained with the error back-propagation algorithm. The following network architectures were attempted using the simulated data:

CASE 1	28-10-10-1
CASE 2	28-10-5-1
CASE 3	28-5-5-1
CASE 4	28-4-3-1

The sigmoid transfer function was used exclusively. In each case 20 percent of the data points were randomly withheld for testing purposes. For each case the learning rate for each layer was set to 1 and the learning tolerance was set initially to 0.10 and adjusted down in 20% increments to a final value of 0.02. The adjustments in the learning tolerance were made each time 95% of the training points were predicted "correctly" (i.e., within the current tolerance limit) by the network. The number of iterations of training varied for each architecture, however, no more than 700,000 iterations were completed for each case.

The same data sets (i.e., training and test) were used for each architecture so that an equivalent comparison of the architectures could be made. As noted above the complete set of simulated data consisted of 72 data points of which 20%, or 14 data points, were withheld for testing. Three metrics were used to evaluate the performance of the network architectures, 1) average absolute error, 2) maximum absolute error, and 3) number of test points that were predicted with errors exceeding 0.333 minutes. These three metrics provide a good overall measure of network performance. The average error is a good indication of the consistency of performance of the network, maximum error gives insight into worst case predictions of the network, and the number of errors which exceed an established limit gives some indication of the amount of confidence the user can place in the network predictions. The limit of 0.333 minutes for the third metric was established based on judgment. The limit corresponds to a network standard time prediction error of 20 seconds. For the simulated data the processing times for the members of the product family ranged from 1.80 to 6.73 minutes. A 20 second error in prediction of standard times is considered acceptable for this range because this level of potential error could easily be masked by other system variables such as particular operator, etc.

Using the three metrics discussed above the four cases performed as shown in Table 1 over the test set.

Table 1. Test Case Error Summary

CASE	MAX ERROR (minutes)	AVG ERROR (minutes)	# ERR > 0.33
1	1.171	0.427	6
2	1.184	0.479	6
3	0.950	0.195	2
4	2.330	0.470	5

As shown, case 3 (the 28-5-5-1 architecture) was found to out perform the other architectures for all three measures. The results show that the larger networks, case 1 and case 2, performed poorly in all three metrics. A significant improvement in performance was achieved for all three measures when the size of the network was reduce to 28-5-5-1. This indicates that the larger networks over-fit the data set and were attempting to memorize the training data, thus reducing their ability to generalize to the test data. However, it was found that further reduction

of the network size (to case 4) significantly decreased the network's performance, particularly for the maximum error criteria.

Figure 6 shows a X-Y plot for case 3. The X-Y graphs plot the estimated and actual process time for each of the 14 test points. The graphs provide a good visual indication of how well the network's estimated process times track the actual process times. A similar approach could be followed to train and test a network on actual data. For the actual data case the best network architecture (i.e., 28-5-5-1) from the simulated data studies would be used. It is considered that this architecture would also perform best with the actual data case since the size of the problem had not changed.

STEP 6: Integrate Neural Network And Expert System

The expert system takes the options selected for the Cube Pedestal product variation during the current system consultation and encodes this into the 28 bit binary input vector. This vector is then stored in a text file that is accessible to the neural network program. The encoding is performed using a set of rules which conclude portions of the 28 bit input vector. An example rule:

```

RULE net input set depth 1
IF case depth OF cube pedestal = 20
THEN depth1 OF network input := 1
AND depth2 OF network input := 0
AND depth3 OF network input := 0
    
```

This rule encodes the portion of the input vector which represents the pedestal depth. Because three pedestal depths are available, three rules were developed for this encode action. The particular rule that fires will depend on the pedestal depth selected by the user.

After the input vector is defined and stored in the text file the expert system activates the neural network program and provides the location of the file holding the input vector and the location of the file which will hold the network output value. The neural network then performs its recall operation independent of the expert system and concludes by storing the output value in the appropriate file. The contents of the output file are read by the expert system using a Method.

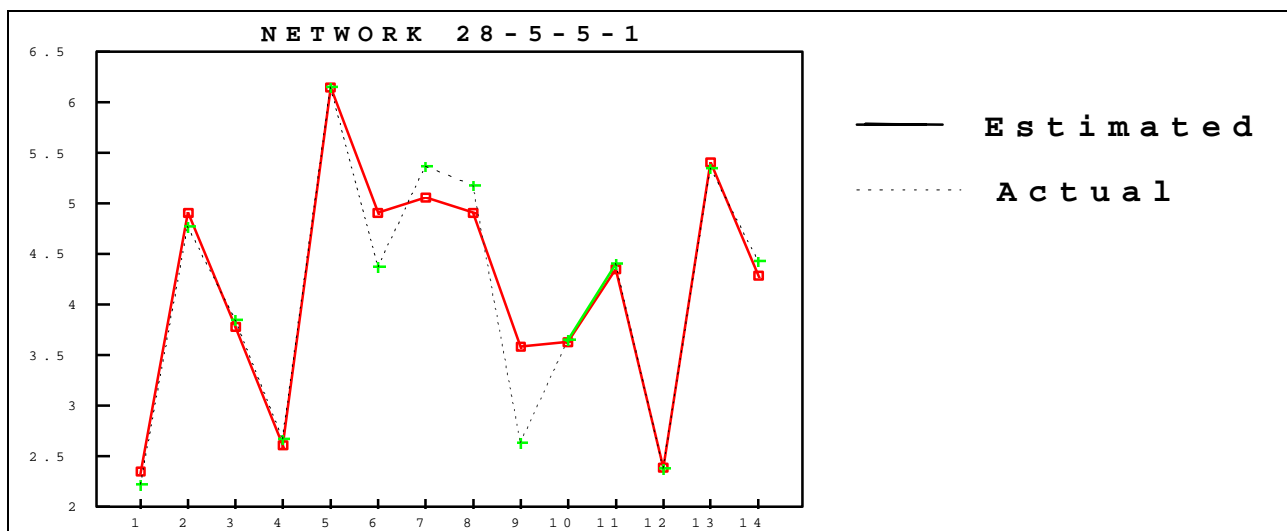


Figure 5. Neural Network Estimations and Actuals.

5. Conclusions

The following conclusions have been drawn from the work presented:

- 1) Neural networks and expert systems are complementary computing tools for the problem domain of automated process planning and process standard time prediction.
- 2) An integrated neural network and expert system approach to automated process planning and process standard time prediction has the potential to provide a system with capabilities that exceed the capabilities of either approach taken independently.
- 3) The use of a neural network to estimate standard process times is effective and can significantly reduce the time invested in determining standard times. This is particularly true when the product family of interest is large and the product family members have highly variable process time requirements.
- 4) Expert system rules can generalize process plan knowledge to new product variations for simple problems like the office furniture assembly process studied. However, for more complex problems like the job shop process planning task the use of neural networks is considered to be potentially more effective. Extension of the current effort to this problem could determine the full potential of an integrated neural network and expert system approach.
- 5) The integrated approach was shown to be feasible in an actual manufacturing application. The system performed well for all test cases tried as evidenced by:
 - Consistent prediction of standard process times for both standard and non-standard products to within a 20 second tolerance.
 - Consistently providing correct process plans and parts lists for both standard and non-standard products.
- 6) The computations and approaches of the system described herein are transparent to the user. To use the program as a tool requires only a standard personal computer, and no user knowledge of the system specifics is required.

References

- [1] Tarun Gupta and Biman K. Ghosh, A Survey of Expert Systems in Manufacturing and Process Planning, *Computers in Industry* **11** (1988) 195-204.
- [2] Tarun Gupta, An Expert System Approach In Process Planning: Current Development and Its Future, *Computers and Industrial Engineering* **18** (1990) 69-80.
- [3] Ming Wang and H. Walker, Creation of an Intelligent Process Planning System within the Relational DBMS Software Environment, *Computers in Industry* **13** (1989) 215-228.
- [4] Sprinivasakuman S. Madurai and Li Lin, Rule-Based Automated Part Feature Extraction and Recognition From CAD Data, *Computers and Industrial Engineering* **22** (1992) 49-62.
- [5] Hsu-Pin (Ben) Wang and Richard A. Wysk, Intelligent Reasoning for Process Planning, *Computers in Industry* **8** (1987) 293-309.
- [6] A. C. Lin. and T. C. Chang, An Integrated Approach to Automated Assembly Planning for Three-Dimensional Mechanical Products, *International Journal of Production Research* **31** (1993) 1201-1227.

- [7] Gerald M. Knapp and Hsu-Pin (Ben) Wang, Acquiring, Storing, and Utilizing Process Planning Knowledge Using Neural Networks, *Journal of Intelligent Manufacturing* **11** (1992) 333-344.
- [8] Philip C. L. Chen and Steven R. LeClair, Unsupervised Neural Learning Algorithm for Setup Generation in Process Planning. In: C. Dagli, L. Burke, B. Fernandez and J. Ghosh (Eds), *Intelligent Engineering Systems Through Artificial Neural Networks Vol. 3*. ASME Press, New York, 1993, 663-668.
- [9] Philip C. L. Chen and Yoh-Han Pao, An Integration of Neural Network and Rule Based Systems for Designs and Planning of Mechanical Assemblies, *IEEE Transactions on Systems, Man, and Cybernetics* **23** (1993) 1359-1371.