INFORMATION TO USERS

This reproduction was made from a copy of a manuscript sent to us for publication and microfilming. While the most advanced technology has been used to photograph and reproduce this manuscript, the quality of the reproduction is heavily dependent upon the quality of the material submitted. Pages in any manuscript may have indistinct print. In all cases the best available copy has been filmed.

The following explanation of techniques is provided to help clarify notations which may appear on this reproduction.

1. Manuscripts may not always be complete. When it is not possible to obtain missing pages, a note appears to indicate this.

2. When copyrighted materials are removed from the manuscript, a note appears to indicate this.

3. Oversize materials (maps, drawings, and charts) are photographed by sectioning the original, beginning at the upper left hand corner and continuing from left to right in equal sections with small overlaps. Each oversize page is also filmed as one exposure and is available, for an additional charge, as a standard 35mm slide or in black and white paper format.*

4. Most photographs reproduce acceptably on positive microfilm or microfiche but lack clarity on xerographic copies made from the microfilm. For an additional charge, all photographs are available in black and white standard 35mm slide format.*

*For more information about black and white slides or enlarged paper reproductions, please contact the Dissertations Customer Services Department.

U·M·I Dissertation Information Service

University Microfilms International
A Bell & Howell Information Company
300 N. Zeeb Road, Ann Arbor, Michigan 48106
Sevenler, Korhan

KNOWLEDGE-BASED SYSTEMS APPROACH TO FORMING SEQUENCE DESIGN FOR COLD FORGING

The Ohio State University

University Microfilms International

Copyright 1986 by Sevenler, Korhan

All Rights Reserved
KNOWLEDGE-BASED SYSTEMS APPROACH
TO FORMING SEQUENCE DESIGN
FOR COLD FORGING

DISSERTATION

Presented in Partial Fulfillment of the Requirements for
the Degree Doctor of Philosophy in the Graduate
School of the Ohio State University

By

Korhan Sevenler, B.S., M.S.

* * * * *

The Ohio State University
1986

Dissertation Committee:
R. A. Miller
T. Altan
G. P. Maul

Approved by

R. A. Miller
Adviser
Department of Industrial and Systems Engineering
©Copyright 1986
By
Korhan Sevenler
Dedicated to

My mother, Süheyla Sevenler
and
my father, Turgut Sevenler

for all the sacrifices they have made,
and the encouragement they have given
for my education.
ACKNOWLEDGEMENTS

I wish to express my sincerest gratitude to my advisor Dr. Richard A. Miller for the inspiration and guidance he had provided throughout my studies at the Ohio State University.

I am also deeply indebted to Dr. Taylan Altan for his advice and assistance throughout the research. Special thanks to Dr. P. S. Raghupathi for his continuing counsel in the acquisition of rules and guidelines for cold forming. I also wish to thank Dr. Gary P. Maul for his interest and valuable suggestions in this study.

Messrs. Larry Wilson, Robert Black and Robert J. Schilling of National Machinery, Tiffin, Ohio, were kind enough to provide their valuable knowledge in forming sequence design for the multistage cold forming. The work was financially supported by the National Science Foundation (Grant No. MEA-8411956) and by internal funds of Battelle Columbus Division.
**VITA**

<table>
<thead>
<tr>
<th>Date</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>September 16, 1956</td>
<td>Born - Istanbul, Turkey</td>
</tr>
<tr>
<td>1978</td>
<td>B.S., Mechanical Engineering Department, Bosphorus University, Istanbul, Turkey</td>
</tr>
<tr>
<td>1978-1980</td>
<td>Graduate Research Assistant, Department of Industrial and Systems Engineering, The Ohio State University, Columbus, Ohio</td>
</tr>
<tr>
<td>1980</td>
<td>M.S., Department of Industrial and Systems Engineering, The Ohio State University, Columbus, Ohio</td>
</tr>
<tr>
<td>1980-1982</td>
<td>Researcher, Engineering and Manufacturing Technology Department, Battelle Columbus Laboratories, Columbus, Ohio</td>
</tr>
<tr>
<td>1982-1983</td>
<td>Graduate Teaching Assistant, Department of Industrial and Systems Engineering, The Ohio State University, Columbus, Ohio</td>
</tr>
<tr>
<td>1983-Present</td>
<td>Researcher, Engineering and Manufacturing Technology Department, Battelle Columbus Laboratories, Columbus, Ohio</td>
</tr>
</tbody>
</table>

**PUBLICATIONS**


FIELDS OF STUDY

Major Field: Computer Aided Design/Computer Aided Manufacturing (CAD/CAM)
   Faculty member - Dr. Richard A. Miller
   Dr. Taylan Altan

Minor Field: Computer Science
   Faculty member - Dr. Edwin Tripp

Minor Field: Systems Theory
   Faculty member - Dr. Gary P. Maul
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>ACKNOWLEDGEMENTS</th>
<th>iii</th>
</tr>
</thead>
<tbody>
<tr>
<td>VITA</td>
<td>iv</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>vii</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>vii</td>
</tr>
</tbody>
</table>

## CHAPTER

### 1 INTRODUCTION ................................................................. 1

### 2 FORMING SEQUENCE DESIGN FOR COLD FORGING WITHOUT FLASH .......... 7

#### 2.1 COLD FORGING WITHOUT FLASH ........................................ 7

##### 2.1.1 Definition ............................................................. 7
##### 2.1.2 History .................................................................... 8
##### 2.1.3 Advantages ............................................................ 9
##### 2.1.4 Limitations ............................................................ 13
##### 2.1.5 Applications .......................................................... 14
##### 2.1.6 Materials for Cold Forging ........................................ 16
##### 2.1.7 Presses for Cold Forging .......................................... 18

#### 2.2 COLD FORGING PROCESSES CONSIDERED IN THIS STUDY .......... 20

##### 2.2.1 Upsetting Processes ............................................... 21
##### 2.2.2 Extrusion Processes ................................................ 22

#### 2.3 FORMING SEQUENCE DESIGN FOR COLD FORGING ................. 28

##### 2.3.1 The Art of Forming Sequence Design ........................... 28

2.3.1.1 Design for Cold Forging ............................................ 29
2.3.1.2 Forming Sequence Design ......................................... 31

##### 2.3.2 The State of the Art ................................................ 35

### 3 KNOWLEDGE-BASED SYSTEMS APPROACH FOR ENGINEERING DESIGN AND MANUFACTURING ................................................................. 38

#### 3.1 ARTIFICIAL INTELLIGENCE ............................................. 38

##### 3.1.1 Definition ............................................................. 38
##### 3.1.2 History .................................................................... 39
# TABLE OF CONTENTS
(Continued)

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.2 KNOWLEDGE-BASED SYSTEMS</td>
<td>42</td>
</tr>
<tr>
<td>3.2.1 Definition</td>
<td>42</td>
</tr>
<tr>
<td>3.2.2 History</td>
<td>43</td>
</tr>
<tr>
<td>3.2.3 Rule-Based Systems</td>
<td>47</td>
</tr>
<tr>
<td>3.2.4 Introduction to Prolog</td>
<td>52</td>
</tr>
<tr>
<td>3.2.4.1 Facts</td>
<td>53</td>
</tr>
<tr>
<td>3.2.4.2 Querying the Database</td>
<td>54</td>
</tr>
<tr>
<td>3.2.4.3 Conjunctions</td>
<td>54</td>
</tr>
<tr>
<td>3.2.4.4 Rules</td>
<td>55</td>
</tr>
<tr>
<td>3.2.5 Logic Programming: History, Advantages, Limitations, and Future</td>
<td>56</td>
</tr>
<tr>
<td>3.3 KNOWLEDGE-BASED SYSTEMS FOR ENGINEERING DESIGN</td>
<td>63</td>
</tr>
<tr>
<td>4 KNOWLEDGE-BASED SYSTEMS APPROACH TO FORMING SEQUENCE DESIGN</td>
<td>68</td>
</tr>
<tr>
<td>4.1 FORMING SEQUENCE DESIGN AS A DESIGN AND PROCESS PLANNING TASK</td>
<td>68</td>
</tr>
<tr>
<td>4.2 IMPORTANT CHARACTERISTICS OF THE TASK</td>
<td>73</td>
</tr>
<tr>
<td>4.3 FUNDAMENTALS OF THE SYSTEM</td>
<td>76</td>
</tr>
<tr>
<td>4.3.1 Production Systems Approach</td>
<td>76</td>
</tr>
<tr>
<td>4.3.2 Decomposition of the Task</td>
<td>80</td>
</tr>
<tr>
<td>4.3.3 Classification of Parts - State of the Art</td>
<td>82</td>
</tr>
<tr>
<td>4.3.4 Classification of Parts - Present Approach</td>
<td>85</td>
</tr>
<tr>
<td>4.3.5 Internal Geometry Representation</td>
<td>87</td>
</tr>
<tr>
<td>4.3.6 Choosing the Billet Dimensions</td>
<td>94</td>
</tr>
<tr>
<td>4.3.7 Finding the Extrusion Sequence</td>
<td>96</td>
</tr>
<tr>
<td>4.3.8 Finding Upsetting Sequence</td>
<td>103</td>
</tr>
<tr>
<td>4.3.9 Combining and Sequencing Extrusion and Upsetting Operations</td>
<td>109</td>
</tr>
<tr>
<td>4.3.10 Load Estimations for Cold Forging</td>
<td>110</td>
</tr>
<tr>
<td>5 IMPLEMENTATION AND RESULTS</td>
<td>116</td>
</tr>
<tr>
<td>5.1 INPUTS</td>
<td>116</td>
</tr>
<tr>
<td>5.2 OUTPUTS</td>
<td>121</td>
</tr>
</tbody>
</table>
TABLE OF CONTENTS
(Continued)

5.3 MENU OPTIONS ................................................................. 125
5.4 WARNING MESSAGES ......................................................... 133
5.5 EXAMPLE FORMING SEQUENCES .......................................... 135
6 CONCLUSION AND RECOMMENDATIONS ............................... 151
   6.1 THE CONTRIBUTION ....................................................... 151
   6.2 THE FORMEX SYSTEM .................................................. 153
   6.3 RECOMMENDATIONS .................................................... 155
REFERENCES .............................................................................. 161
APPENDIX A. SAMPLE FORMEX RULES ................................. 167

LIST OF TABLES
Table 1. Examples of Cold-Forged Parts ................................... 15

LIST OF FIGURES
Figure 1. An Example Forming Sequence for Cold Forging of a Gear Blank ........................................ 3
Figure 2. Different Techniques for Heading or Upsetting ............. 21
Figure 3. Various Types of Extrusion Processes ......................... 23
Figure 4. Open-die and Closed-die (trapped) Extrusion ............... 25
Figure 5. Obtaining the Cold-Formed Part Geometry from the Finished Part Geometry ......................... 29
Figure 6. Illustration of the Forming Sequence Design for the Hypothetical Part in Figure 5 ..................... 32
Figure 7. Dimensioned Cold Forming Sequence for the Hypothetical Part in Figure 5 ............................. 33
# LIST OF FIGURES

(Continued)

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>Two Similar Parts with Different Intermediate Upsetting Stages</td>
<td>36</td>
</tr>
<tr>
<td>9</td>
<td>Structure of the FORMEX System</td>
<td>78</td>
</tr>
<tr>
<td>10</td>
<td>An Example of Classification of Parts for Cold Forming</td>
<td>83</td>
</tr>
<tr>
<td>11</td>
<td>Block Diagram Showing Conversions to and from Internal Geometry Representation</td>
<td>89</td>
</tr>
<tr>
<td>12</td>
<td>Construction of 2-D and 3-D Internal Geometry Representations from Input Coordinates</td>
<td>90</td>
</tr>
<tr>
<td>13</td>
<td>Guidelines for Open Extrusion</td>
<td>97</td>
</tr>
<tr>
<td>14</td>
<td>Guidelines for Trapped Extrusion</td>
<td>99</td>
</tr>
<tr>
<td>15</td>
<td>Double Reduction Open Forward Extrusion in One Pass</td>
<td>102</td>
</tr>
<tr>
<td>16</td>
<td>Definition of Unsupported Stock Length in Upsetting</td>
<td>103</td>
</tr>
<tr>
<td>17</td>
<td>Design of Taper Die for Upsetting Stock Length Equal to 3.25 Times Billet Diameter</td>
<td>105</td>
</tr>
<tr>
<td>18</td>
<td>Upsetting Before Open Forward Extrusion</td>
<td>111</td>
</tr>
<tr>
<td>19</td>
<td>Open Forward Extrusion Before Upsetting</td>
<td>111</td>
</tr>
<tr>
<td>20</td>
<td>Example Part Drawing</td>
<td>117</td>
</tr>
<tr>
<td>21</td>
<td>Input Data File for the Example Part in Figure 20</td>
<td>118</td>
</tr>
<tr>
<td>22</td>
<td>Portion of the FORMEX Material File</td>
<td>119</td>
</tr>
<tr>
<td>23</td>
<td>Forming Sequence Plot for the Example Part in Figure 20</td>
<td>122</td>
</tr>
<tr>
<td>24</td>
<td>Output File for the Example Part in Figure 20</td>
<td>123</td>
</tr>
</tbody>
</table>
**LIST OF FIGURES**

(Continued)

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 25.</td>
<td>Enlarged Plot of Station 3 from Figure 23</td>
<td>126</td>
</tr>
<tr>
<td>Figure 26.</td>
<td>Forming Sequence Plot for the Example Part in Figure 20 with 2 Upsetting Blows</td>
<td>137</td>
</tr>
<tr>
<td>Figure 27.</td>
<td>Example Part with 3 Upsetting Zones</td>
<td>138</td>
</tr>
<tr>
<td>Figure 28.</td>
<td>Input Data File for the Example Part in Figure 27</td>
<td>139</td>
</tr>
<tr>
<td>Figure 29.</td>
<td>Forming Sequence Plot for the Example Part in Figure 27 with 2 Upsetting Blows</td>
<td>140</td>
</tr>
<tr>
<td>Figure 30.</td>
<td>Forming Sequence Plot for the Example Part in Figure 27 with 3 Upsetting Blows</td>
<td>141</td>
</tr>
<tr>
<td>Figure 31.</td>
<td>Forming Sequence Plot for the Example Part in Figure 27 with 4 Upsetting Blows</td>
<td>142</td>
</tr>
<tr>
<td>Figure 32.</td>
<td>Example Part</td>
<td>144</td>
</tr>
<tr>
<td>Figure 33.</td>
<td>Input Data File for the Example Part in Figure 32</td>
<td>145</td>
</tr>
<tr>
<td>Figure 34.</td>
<td>Forming Sequence Plot for the Example Part in Figure 32</td>
<td>146</td>
</tr>
<tr>
<td>Figure 35.</td>
<td>Example Part</td>
<td>147</td>
</tr>
<tr>
<td>Figure 36.</td>
<td>Input Data File for the Example Part in Figure 35</td>
<td>148</td>
</tr>
<tr>
<td>Figure 37.</td>
<td>Forming Sequence Plot for the Example Part in Figure 35</td>
<td>149</td>
</tr>
<tr>
<td>Figure A-1.</td>
<td>Sample FORMEX Rules for Constructing the Internal Geometry Representation</td>
<td>169</td>
</tr>
<tr>
<td>Figure A-2.</td>
<td>Sample FORMEX Rules for Forward Extrusion</td>
<td>172</td>
</tr>
<tr>
<td>Figure A-3.</td>
<td>Sample FORMEX Rules for Upsetting</td>
<td>175</td>
</tr>
</tbody>
</table>
CHAPTER 1
INTRODUCTION

Cold Forming is a broad term that covers various metal forming operations commonly performed in the mass production of axisymmetric parts from steel bars, wire rods and wire at room temperature. Principle cold forming operations are upsetting (or heading), forward extrusion and backward extrusion. These operations are performed in multi-stage presses to produce parts of various geometric complexity.

The application of cold forging for manufacturing of various parts is steadily increasing. Components for automobiles constitute a major percentage (about 85 percent) of the parts made by cold forming. With ever-increasing material and production costs, forming is frequently the more economical way to make profiled steel or non-ferrous workpieces than by metal cutting. The increasing cost of energy will further promote the development of the cold forming technology.

Because of the high stresses and loads needed to cold form steel, cold forging processes are best suited for small parts. Most of the early markets were for mass produced components, with simple shapes weighing less than 1 kg. In recent years, larger more complex parts have been produced; that change led to economical applications of cold forging for fabricating components required in smaller lot quantities. A number of parts are being made from billets up to 75 mm in diameter and weighing several kilograms. A few large parts, such as axles, are
being made by a combination of hot upsetting and cold extrusion, with weight from 4 to 8 or more kilograms.

Ideal workpieces for cold forming are solid or hollow axisymmetric parts. Typical advantages of the cold forming method are: higher strength properties gained by work-hardening, consistently high profile and dimensional accuracy and high-surface finish combined with increased resistance to corrosion.

The major advances in this field are expected to be in new forming sequence and tool design, new machines, and flexible automation, i.e. in making automated cold forming economical even for relatively small lots of production.

Cold forming requires several "preforming" operations (Figure 1) to transform the initial simple billet geometry into more complex product geometry without any surface and internal defects. Given a final product geometry, the die designer is faced with the problem of determining the optimum forming sequence (preforming operations) to forge the part. Establishing the forming sequence is an art and so far it has been mainly achieved by using experience and trial and error. Die designers, who have been in this business for many years, use some well-established design rules and guidelines as well as "rules of thumb" in finding cold forming sequences.

The design of a forming sequence starts with the final cold formed part geometry and proceeds backwards to find the shapes of the dies for each preforming station in that sequence (Figure 1). The designer assigns principal forming operations such as upsetting, forward extrusion and backward extrusion to every station using his expertise in
cold forming and die design. There are some well-established design rules and guidelines for each of these principal forming operations. These rules have to consider process parameters such as final part shape and tolerances, number of parts to be produced, billet material, billet coating, lubrication and machine specifications.

All of the knowledge and reasoning that the die designer uses in forming sequence design, is not well formalized into design rules and guidelines. In fact, most of it is accumulated through his many years of training and experience in die design and may not even be recorded anywhere except his mind.

So, the first task of this project was to study the processes of forming sequence design carefully and try to acquire and formalize the knowledge used for forming sequenced design. The next task was to
use the results of the first task and develop a knowledge-based system that can assist less experienced die designers in forming sequence design.

A knowledge-based system is a computer program whose performance depends more on the explicit presence of a large body of knowledge rather than on the possession of ingenuous computational procedures. More "knowledge" and "reasoning" than "computation" intensive nature of forming sequence design process makes it suitable for knowledge-based systems approach.

In the first phase of this study, the forming sequence design task was studied in detail. Besides an extensive literature review, several die designers were interviewed and a 2-day seminar on tool design for multi-die cold forming was attended. Important characteristics of this design task were abstracted. These include the ability to decompose the forming sequence design task into several subtasks which are partially independent. Another important property of the forming sequence design task is the classification of axisymmetric parts into major groups which have different design rules and guidelines (such as solid parts versus hollow parts). Another property of forming sequence design is the intensity of geometry manipulation during this design task which makes a powerful internal geometry representation scheme crucial for the success of the knowledge-based system. Several other important properties of this task were discovered during this phase of the study.

The results were then used to develop a knowledge-based system to assist die designers in forming sequence design. This system, called FORMEX (for FORMing EXpert), was implemented as a production system
using the Prolog language. FORMEX finds and plots the forming sequence for solid parts (a class of axisymmetric parts) given the final part geometry, material type and press type. Initial results show that FORMEX can be a useful tool for the less experienced die designers in tool design for multi-die cold forming.

This study is presented here in six chapters. Chapter 2 describes cold forming, its advantages, limitations, applications, materials for cold forming, presses used and principle cold forming operations. It also explains the process of forming sequence design in detail. Chapter 2 ends with the discussion of the state of the art in forming sequence design for cold forging.

Chapter 3 starts with a review of artificial intelligence and knowledge-based systems. It continues with a detail discussion of rule-based systems (production systems), logic programming and Prolog. Chapter 3 concludes with the discussion of the application of knowledge-based systems in engineering design.

Chapter 4 describes the fundamentals of the approach. In the beginning of this chapter the task is defined in detail. This is followed by a description of the important characteristics of the task and the formalization of the forming sequence design knowledge. Fundamentals of the system are explained next, including production systems approach, decomposition of the task, classification of parts, internal geometry representation, choosing the billet dimensions and finding the extrusion and upsetting sequences.

Chapter 5 describes the implementation of the system using the Prolog language. The details of FORMEX such as inputs, output and menu
items are explained. Forming sequences designed by FORMEX, for some example parts, are discussed.

Chapter 6 contains the conclusion and the recommendations. It starts with a summary of the contribution of this study. The second section of the chapter briefly mentions the highlights of the FORMEX system. The last section of Chapter 6 contains the recommendations for further studies in forming sequence design and knowledge-based systems applications for manufacturing in general.

An appendix is included to show some example rules from the FORMEX system in Prolog clauses. The purpose of the appendix is not to explain the structure of FORMEX thoroughly, but only to give an idea how some cold forming rules can be represented as Prolog clauses.
CHAPTER 2

FORMING SEQUENCE DESIGN FOR COLD FORGING WITHOUT FLASH

2.1 COLD FORGING WITHOUT FLASH

2.1.1 Definition

Cold forging is a special type of forging process where metal at room temperature is forced to flow plastically under compressive force into a variety of shapes [1]. Most of the cold-forged parts are axisymmetric with relatively small nonaxisymmetrical features. Cold forging produces very little or no flash, contrary to impression die forging. Other terms used interchangeably for cold forging are "cold forming", "cold extrusion", and "impact machining" [2]. Cold forging, as a general term, includes processes such as extrusion, upsetting, coining, ironing, and swaging used in metal forming at room temperature. Several forming stations are used successively to form a relatively complex part, starting with a billet of simple shape as shown in Figure 1 [3].

In cold forging, the billets are at room temperature, whereas in warm forging, they are heated to temperatures below the recrystallization temperature in order to reduce the flow stress and the forging pressures. The aim of warm forging is to capitalize on the advantages of both hot and cold forging [4]. The forging temperature affects both
the behavior of the material during deformation and the properties of the finished part. Warm forging requires lower flow stress and enables production of larger and more complex parts. The optimal temperature for a particular warm forging operation always represents a compromise: its lower limit is set by the force which can be produced by the forging press and by the forgeability of the material; the upper limit is determined by the amount of oxidation—and thus the amount of scale—which can be tolerated. Warm forging temperature range for steel is 450 to 900 C (842 to 1652 F).

In cold forging, the temperature of the workpiece and the tooling is increased due to friction and plastic deformation. However, it never reaches the recrystallization temperature (about 1100 to 1300 F for steel) and work hardening always occurs. The improved physical properties resulting from cold working are retained in the finished parts unless the parts are subsequently heat treated for further machining. This is one of the principle differences between cold and hot forging. In hot forging, recrystallization eliminates the effects of work hardening unless the part is rapidly cooled. Cold extrusion also differs from other metalworking processes since the metal is nearly always formed under compression and seldom in tension. Therefore, larger amount of deformation can be achieved without cracking or tearing.

2.1.2 History

Even though principles of cold forging have been used for more than 100 years in the forming of soft metals, modern day cold extrusion
of steel had its beginning in Europe during the mid-1930s [2]. A shortage of copper and zinc for brass led to military demands for cartridge cases, fuse bases, and other ordnance components produced from steel. Phosphate coating of blanks and the cold extrusion of steel were investigated in the Neumeyer Cable and Metalworks in Nuremberg, and the Collis Metalworks in Westhausen, Germany, and German patents were granted in 1937.

After World War II, a Technical Industrial Intelligence Committee sponsored by the U.S. military and the Department of Commerce visited Germany and learned of these developments [2]. Subsequent U.S. Army Ordnance contracts led to the successful cold extrusion of 20-millimeter steel shells, 3.5-inch steel rockets, and an experimental 75-millimeter projectile. The primary reason of these programs were the conservation of steel and critical alloys, and the reduction of machining time, which would be crucial in a period of national emergency. About the same time, a few manufacturers investigated the possibility of using cold forging for the production of nonmilitary items. One of the first examples of cold forging in commercial use was the production of automotive hydraulic valve lifters in 1955. [2]

2.1.3 Advantages

Principal advantages of cold forging are [2,5]:

a. Material savings - Since almost all of the metal in each billet is formed into a finished or nearly finished part,
there is little or no loss of material in the form of chips or flash. In general, material savings in cold forging are 40 to 75 percent and more compared to machining. When parts must be made from more expensive alloys, the savings from using less material become more substantial. In some applications, stock required for cold forging is less than one-fourth of the stock required for machining.

b. Reduction or elimination of additional machining requirements - In most cases, parts produced by cold forging require little subsequent machining because of the exactness of the shapes produced, closer tolerances maintained, and smooth surface finishes obtained. In other cases, machining times are reduced due to the accuracy and consistency of cold forging dimensions. Savings resulting from the reduction or elimination of machining generally increase with the complexity of the part.

c. Improvements in mechanical properties - Cold forging produces continuous contoured grain flow lines in the material. Cold-forged parts also have smooth surface finishes and no stress inducing tool marks. These features of cold forging improve the resistance to shock and fatigue failure of the part.
During the plastic deformation of steel below its recrystallization temperatures, the ferrite grains and carbide structures are elongated in the direction of the principle plastic flow. Such a structure has higher tensile and yield strengths. Even though this is an advantage, it makes subsequent operations more difficult due to the increased strength and decreased ductility. Cold forging of low- and medium-carbon steels generally increases the tensile strength from 30 to 130 percent, the yield strength from 100 to 300 percent, and the hardness from 60 to 150 percent depending on the amount of deformation.

d. Possible use of less costly material - Cold forging makes possible the use of lower cost raw material than if the parts were to be produced by ordinary machining. Some parts that are conventionally produced from hexagonal or other specially-shaped rolled, drawn, or extruded stock can be made from standard, less costly, round bar stock by cold forging. Sometimes cold forging makes it possible to use hot-rolled bar instead of more expensive cold-drawn material, or to substitute less costly low- or medium-carbon steels for high carbon or alloy steels.

e. Replacement of assemblies with single components - Cold forging can produce a single part instead of a previous assembly or subassembly, thus reducing or eliminating the
cost of producing individual parts and assembling them. By combining several components into one part, cold forging provides additional strength by eliminating local stresses created by fasteners or welding.

f. Special shapes - Some parts which are difficult or impractical to machine can be incorporated in a cold forging. Parts that could not be produced economically by any other method can be designed for cold forging.

g. Dimensional accuracy - Close tolerances can be obtained in cold forging. Scrap losses and inspection costs are reduced by the consistent repeatability of the tolerances from part to part. Other advantages of cold forging are smooth surface finish and scale-free surfaces. The surface finish depends on the quality of the finish on the dies used. Since the finish on the tools deteriorates with use, the finish required on parts is a factor in establishing the usable life of the tools.

h. High production rates, reduced material handling, and no preheating - All these characteristics of cold forging can be favorable compared with other processes such as hot forging, casting, sintering, and conventional machining.
2.1.4 Limitations

The main limitation is the cost of cold forging a particular part and material [2]. High-carbon and alloy steel parts are sometimes too expensive to produce in this way because of the higher pressures required and the need for several press operations with intermediate annealing treatments.

Cold forging is generally limited to the forming of cylindrical, square, hexagonal, or similar symmetrical shapes having solid or hollow cross sections. Generally, cold forging is not recommended for nonconcentric parts having nonuniform wall thickness due to the non-uniform pressures exerted on the tooling.

Cold forging requires high initial equipment cost. Production requirements must be sufficient to justify this cost. The economic advantage of cold forging over other processes increases with the quantity of the particular part to be produced. However, cold forging is not limited to mass production needs. Low-volume production is sometimes feasible depending on the economics of the particular situation, the strength requirements of the part, and other factors.

Generally, 10,000 to 50,000 large or multiple purpose parts per month are sufficient to make cold forging economically justifiable. For smaller parts, the minimum number of pieces necessary is approximately 100,000 per month [2]. These are rough numbers and other factors such as the shape of the part, material, surface finish, and tolerances required should also be considered in each particular economic justification.
2.1.5 Applications

Due to the substantial savings in steel and critical alloys, and the great reduction in machining time, cold forging was popular during World War II and the Korean War for producing cartridge cases, shells, rocket heads, and other military requirements. Production of commercial items by cold forging has increased rapidly since the early 1950s. Some examples of parts that are cold forged are listed in Table 1 [2].

Cold forging is suitable for forming circular, square, hexagonal, or other symmetrical cross sections. Axisymmetrical, nonrotational parts can only be produced if the imbalance is not too severe: Typical characteristics of the cold-forged components are [5]:

- stepped or tapered diameter shafts,
- thin or thick wall,
- bottom thicker than wall,
- flanges at their tops,
- opened or closed end,
- flat, conical or hemispherical closed end,
- tapered or parallel sidewalls,
- web at one end of the part,
- longitudinal projections or depressions either internal or external such as flutes, grooves, ribs, splines.
## Table 1. Examples of Cold-Forged Parts [2]

<table>
<thead>
<tr>
<th>Ordnance Industry</th>
<th>Aircraft Industry</th>
<th>Electrical and Electronic Industries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Steering ball joints</td>
<td>Bearing races</td>
<td>Motor and generator housings</td>
</tr>
<tr>
<td>Cartridge cases</td>
<td>Hydraulic brake pistons</td>
<td>Pole shoes</td>
</tr>
<tr>
<td>Shells</td>
<td>Oil filter cases</td>
<td>Switch housings</td>
</tr>
<tr>
<td>Rocket heads</td>
<td>Hose couplings</td>
<td>Door-latch rotors</td>
</tr>
<tr>
<td>Missile components</td>
<td>Air conditioner accumulators</td>
<td></td>
</tr>
<tr>
<td>Satellite parts</td>
<td>Motor housings and mounts</td>
<td></td>
</tr>
<tr>
<td>Gears</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Hardware Industry</strong></td>
<td><strong>Electrical and Electronic Industries</strong></td>
<td></td>
</tr>
<tr>
<td>Socket wrenches</td>
<td>Bearing races</td>
<td>Motor and generator housings</td>
</tr>
<tr>
<td>Door-check cylinders</td>
<td>Alternator and generator shells</td>
<td>Pole shoes</td>
</tr>
<tr>
<td>Grease-gun reservoirs</td>
<td>Switch housings</td>
<td>Switch housings</td>
</tr>
<tr>
<td>Flashlight and battery cases</td>
<td>Door-latch rotors</td>
<td>Tube cases</td>
</tr>
<tr>
<td>Vacuum-bottle holders</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Toys and giftware items</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Automotive Industry</strong></td>
<td><strong>Air Conditioning &amp; Heating Industries</strong></td>
<td></td>
</tr>
<tr>
<td>Piston (wrist) pins</td>
<td>Bearing races</td>
<td>Heat exchangers</td>
</tr>
<tr>
<td>Spark-plug shells</td>
<td>Alternator and generator shells</td>
<td>Pressure vessels</td>
</tr>
<tr>
<td>Transmission shafts</td>
<td>Switch housings</td>
<td>Pump components</td>
</tr>
<tr>
<td>Shock-absorber tubes</td>
<td>Door-latch rotors</td>
<td>Cylinders and pistons</td>
</tr>
<tr>
<td>Bolts and nuts</td>
<td></td>
<td>Manifolds and fittings</td>
</tr>
<tr>
<td>Hydraulic valve lifters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Driveshafts</td>
<td></td>
<td>Miscellaneous</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Food and beverage cans</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Home appliance components</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Business machine parts</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2.1.6 Materials for Cold Forging

All materials can be plastically deformed to varying degrees, providing enough pressure can be applied and the tooling is sufficiently strong. Higher pressures can be obtained by increasing the press size, but the real restriction is the strength of the tooling. So, the materials that can be cold forged are limited by the capabilities of the tooling.

The pressures required for cold forging must exceed the yield strength of the material. It must also overcome the frictional resistance at the interfaces of the material, lubricant, die and punch. Even low-carbon steels require pressures exceeding 350,000 psi in cold extruding with high reductions in area [2]. Tooling made from carbide can withstand ultimate compressive values of well over 500,000 psi. However, maximum loading limit with a reasonable fatigue life would be about 250,000 psi for tool steel and 375,000 psi for carbide tooling. So, the maximum pressure the tools can stand is a restriction for choosing the material to cold form.

Steels with lowest yield strength and the greatest range between yield and ultimate strengths are the best for cold forging [2]. Low-yield strength decreases the starting pressure required for extrusion, thus reducing initial loading on the tooling and increasing the tool life. Also, greater deformations can take place at lower pressures due to decreased rate of work hardening.

The ease of cold forging a material also depends on the hardness value. Although an ideal starting hardness for plain carbon steels
containing up to 0.15 percent carbon is 65 Rockwell B or lower, this is
often impractical and uneconomical [2]. So, steels having a hardness
value of 85 Rockwell or higher are commonly used for cold forging. When
steels with hardness value higher than 85 Rockwell are to be cold
forged, it requires more careful tool design, smaller reductions in
area, and less penetration in backward extrusion. Sometimes, harder
steels can be forward extruded more easily. If the hardness after cold
forging increases up to 100 Rockwell B, the part may require annealing
before further operations.

Besides copper and aluminum alloys, steels having very differ-
ent chemical compositions and mechanical properties can be cold forged,
such as [5]:

1. Low-carbon unalloyed steels

   C ≤ 0.23%
   Si ≤ 0.15%
   Mn 0.30-0.60%
   P,S ≤ 0.04%
   N ≤ 0.008%
   Al ≤ 0.04%

2. Case hardening steels

   plain carbon types
   alloyed types with Cr, Mo, Ni

3. Steels for quenching and tempering

   not alloyed steels
   carbon content up to 0.55%

   alloyed types

   Cr alloyed types, Cr ≤ 1.2%
   Cr Mo alloyed type, Cr ≤ 1.2%-Mo ≤ 0.3%
   Cr Ni Mo alloyed types, Cr ≤ 1.2%-Ni ≤ 1.0%-Mo ≤ 0.3%
4. Stainless steels

ferritic steels
austenitic steels

2.1.7 Presses for Cold Forging

Both mechanical and hydraulic presses are used for cold forging. When these two types of presses are compared, no differences have been found in the grain structure or quality of parts produced on either type press. [2] The decision on the type of press depends on a complete analysis of the production process, the size and shapes of the parts to be made, the material, reduction in area required, tonnage necessary, cost of the equipment, and the production rate.

Generally, mechanical presses are preferred for extruding smaller and shorter parts, where the production requirements are higher and the forces are lower. Mechanical presses offer greater production rates and lower initial cost especially for lower-tonnage, shorter stroke requirements. They are also easy to adapt for automation such as automatic loading, feeding, and unloading. Automation equipment is directly controlled from the press drive and, with the rigidly fixed stroking of mechanical drives, synchronization becomes easy.

The disadvantage of mechanical presses for cold forging is that the velocity of the slide is constantly changing. This velocity is maximum at mid-stroke and decreases to zero at the top and bottom of the stroke. So, the punch makes the first contact with the workpiece at a high velocity and slows down during the extruding cycle. This initial
high-contact velocity causes heavy impact of the punch with the workpiece which directly affects the tooling. To increase die life, it is preferred that the punch contacts the workpiece slowly and accelerate after the contact. Specially designed cold forging presses can supply this type of a cycle by providing maximum tonnage at various positions on the stroke [2]. Generally, these presses have oversize main drives which include extra-heavy gears, shafts, clutches, and motors.

Hydraulic presses are generally better suited for extruding larger, longer, or heavier parts with higher tonnage requirements. They are usually recommended when the total length of stroke required is greater than 24 inches or the working stroke is continuous over 5 inches or more [2]. Generally, parts 4 inches or more in length need a minimum tooling stroke of 12 inches for hand-fed operations, and 18 inches for automatic feed.

One disadvantage of hydraulic presses is their limited production rates since they are slower than mechanical presses. They cannot also counteract unbalanced loading as well as mechanical presses. This may create a problem in critical multiple operations when tooling is sensitive to tilting of the slide and nonparallelism of the slide with the bed.

Cold forging presses have horizontal and vertical versions, as well as single- and multi-stage variations. Single-stage presses are preferred for the processes with characteristics such as (1) concentrated high axial loads, (2) proportionally long strokes (for solid and hollow forward extrusion much longer than for backward can extrusion), (3) high work capacity requirements, and (4) sufficient forces for ejecting the workpiece [2].
Generally, exact loading on the center of multi-stage presses is not possible. Horizontal and vertical multi-stage presses are available with either mechanical or hydraulic drives. Large batch sizes and workpieces between 3 g (0.1 oz.) and 3 kg (6.8 lb) are more appropriate for multi-stage cold forging presses. If the diameter of the billet is 30 mm (1.2 in.) or less, the material is fed from a wire coil, otherwise it is fed in the form of bars or rods. Four to seven stages may be available in multi-stage presses after shearing.

Single-stage machines are generally hand-fed for low production requirements. For single-stage presses, when the production size is large, automation is preferred. For this, mechanical presses with automatic loading, feeding, and ejection equipment may be used. For multiple-stage machines, even for low-production requirements, automating may be preferred since hand-fed vibratory feeders and tools generally make it necessary to provide inter-stage annealing and relubrication. [2] The need for reannealing and relubricating may be eliminated by automating.

2.2 COLD FORGING PROCESSES CONSIDERED IN THIS STUDY

There are three principle operations in cold forging: (1) upsetting, (2) forward extrusion, and (3) backward extrusion. A very large number of different parts can be formed by combining these basic cold forging operations.
2.2.1 Upsetting Processes

Upsetting is defined as "free forming, by which a workpiece segment is reduced in dimensions between usually plane, parallel platens" [4]. It is a metal forming process where the cross-sectional area of a portion or all of the stock is increased. Figure 2 shows the different techniques for upsetting or heading. In many cases, by upsetting, a head is formed on top of a shank by increasing the starting diameter and shortening the length. This is why sometimes upsetting is also called "heading".

Figure 2. Different Techniques for Heading or Upsetting [6].
Figure 2.a shows upsetting a bar with a flat heading tool. Here it is important that the length of the unsupported stock between the die and the punch is not too great, otherwise the stock may buckle during upsetting. If the unsupported portion of stock is long, it must be guided by either the die (Figure 2.b) or by the punch (Figure 2.c), or by both the die and the punch (Figure 2.d).

In cold upsetting of axisymmetric steel parts, there is a limit on the total strain, or deformation, which is independent of the number of upsetting stages. [4] If the total deformation required is larger than this limit, surface cracks may occur. For head shapes that are not axisymmetric, this limiting strain is not a criteria since local transverse strains depend upon the shape of the head. There are two ways of overcoming this limitation [4]. One method is intermediate annealing or hot upsetting to eliminate the effects of strain hardening. The other approach is to modify the head shape to create an additional radial pressure, thus decreasing the value of the tensile stresses on the free surface of the part.

Generally, hot rolled stock is used for cold upsetting. Sometimes cold drawn stock is preferred due to smaller variations in diameter and volume.

2.2.2 Extrusion Processes

In cold extrusion, the billet is pressed through a die by a punch. Cold extrusion processes are classified depending on the material flow in relation to the movement of the punch. In forward
extrusion (Figure 3.a and b), the punch compresses the billet so that the billet material flows through a die in the same direction as the movement of the punch. In backward extrusion (Figure 3.c) the punch applies a steady pressure to a slug confined in a closed die, and forces the metal to flow around the punch in a direction opposite the direction of the punch movement [1].

Figure 3. Various Types of Extrusion Processes (P = punch, C = container, W = workpiece, E = ejector) [7].
Extrusion processes can further be classified by the geometry of the components formed. In forward rod (solid) extrusion (Figure 3.a), the final product is a solid workpiece with a profile, and the shape of the die opening is determined by the die only [4]. In forward cup (hollow) extrusion (Figure 3.b) the final product is a hollow workpiece with reduced wall thickness and the shape of the die opening is determined by both the die and the punch. In backward cup (hollow) extrusion (Figure 3.c), the final product is a cup-shape workpiece with a profile, and the shape of the die opening is determined by both the die and the punch.

Sometimes some of the principle extrusion operations described above can be combined in a single stage. Figure 3.d shows forward rod extrusion and backward cup extrusion combined in one die. Another example is combined forward and backward cup extrusion shown in Figure 3.e. In combined extrusion metal may be forced to flow both in the same and in the opposite directions to the punch movement. Reduction in area in combined extrusion is less than that obtained with either forward or backward extrusion alone. Complex parts, that cannot be made with either method alone, can be formed by combined extrusion.

Forward rod extrusion can further be classified as open-die extrusion and closed-die (trapped) extrusion. In open-die extrusion (Figure 4.a), the undeformed portion of the workpiece is not supported by either the punch or the die. In closed-die (trapped) extrusion (Figure 4.b), the undeformed portion of the workpiece is confined in the die. In open-die extrusion, the unsupported length should not upset or buckle during extrusion.
In forward extrusion, a slug is placed in a die cavity having a diameter slightly lower than that of the slug and pressure is applied to the top of the slug by a punch. The metal is forced to flow ahead of the punch through an orifice immediately under the die cavity, so the diameter of the final product is smaller than the diameter of the slug. Forward extrusion parts can have tapered or parallel sidewalls and flanges at their ends. Shapes other than cylindrical, such as hexagonal and square shapes, can also be extruded.
Generally, forward extrusion requires less pressure than backward extrusion and smaller presses can be used for producing forward extruded parts of the equivalent size [2]. Due to the self alignment of the punches in the dies, closer tolerance can be obtained in forward extrusion. Shapes that can be produced by forward extrusion are more limited than by backward extrusion. Special attention is required to design the forward extrusion dies so that the workpiece can be removed from the die easily after extrusion. In extrusion, maximum possible reduction in cross-sectional area is an important limit to consider. Reduction in area, \( R \) (in percent), is defined as:

\[
R = \frac{A - a}{A} \times 100
\]

where "\( A \)" is the original cross-sectional area of the slug and "\( a \)" is the cross-sectional area of the extruded part.

The limit on reduction in area in forward extrusion depends on the material being extruded, the lubricant, and the shape of the die. Certain aluminum alloys can be extruded up to 80 percent reductions in area and low-carbon steel can be reduced up to 60 percent under production conditions. However, high-carbon steels usually require very high pressures for extruding with a reduction in area above 50 percent [2].

Another criteria in forward extrusion is the ratio of the length of extruded part to original slug diameter. This ratio can exceed 200 to 1 for certain aluminum alloys [2]. However, in cold extrusion of steel, the length to diameter ratio is limited to 8 to 1 in a single operation.
In backward cup extrusion, the slug is placed in a stationary die cavity, and the pressure is applied to the top of the slug by means of a press-ram actuated punch (Figure 3.c). The metal is displaced upward through the annular opening between punch and die (orifice) and rises along the punch. The configuration and the size of punch defines the internal shape and the dimensions of the part. The cavity of the die controls the external contour of the part. Circular, oval, square, rectangular, and other shapes can be formed by backward extrusion.

Even though some soft aluminum alloys can be backward extruded with very high reductions in area, generally very high pressures are required to achieve more than 75 percent reduction in area. Similarly, low carbon steels can be backward extruded with higher than 75 percent reduction, but high-carbon and alloy steels require very high pressures when reduction in area exceeds 50 percent [2].

Proper lubrication is most important for backward extrusion. The only lubricant available for the entire inner surface of the part is that on one end of the slug. Under production conditions, heat generated by rapidly repeated operations will tend to break down the lubricant. So, depending on the reductions in area and the materials, the depths of penetration of the punch is limited to four to five times the punch diameters.
2.3 FORMING SEQUENCE DESIGN FOR COLD FORGING

2.3.1 The Art of Forming Sequence Design

Cold forging requires several "preforming" operations (Figure 1) to transform the initial simple billet geometry into a more complex product geometry without any surface or internal defects. Given a final finished geometry, the manufacturing engineer is faced with the problem of determining the forming sequence (preforming operations) for the part. In some cases, the part may be produced in a single blow. But, due to the limitations on the principle cold forging processes described in Section 2.2, multiple forming stations are required for most complex parts.

The task of designing the forming sequence is best described with an example. A hypothetical part shown in Figure 5 is chosen as an example [8]. It is assumed that this part is to be formed on a six-station five-die cold former. The task of the designer can be divided into three subtasks as:

1. finding the cold formable part geometry from the final machine part geometry (design for cold forging),
2. establishing the cold forming sequence (forming sequence design), and
3. designing the cold forging dies for each stage.
2.3.1.1 Design for Cold Forging. In the first task, the designer has to decide if the final part can be directly produced by cold forging. Since cold forging can achieve very good tolerances and surface finish, sometimes it is possible to produce the final part by cold forging. There are important savings in this case, since further operations such as machining and grinding are eliminated. But, cold
forging the final part is not always possible. The designer carefully analyzes the final part geometry and checks several points. He must check the tolerances on all dimensions and decide if it is practical to cold form these tolerances. He also checks for sharp corners, undercuts, and chamfers that are not possible to cold form. The overall geometry of the part is also checked to see if this part has a shape that can be cold formed. The finished part drawing is analyzed for concentricity, flatness, and parallelism that may effect the cold-formed part [8]. The designers also check to be sure whether surface finish, material, and mechanical properties of the part are suitable for cold forging.

After checking for all these conditions, the designer draws what the cold-formed part will look like. A machined final part and the cold-formed part derived from it are shown in Figure 5. The cold-formed part has the same tolerances for the outside diameters as the machined part. But, the tolerance for the inner diameter on the machined part cannot be obtained by cold forging, so the inside diameter is undersize (Figure 5). The undercuts on each side of the collar are replaced by radii. Due to backward extrusion, ends of the part will be slightly concave, so the part is made longer and it must be machined on each end to bring it within the original length tolerances. The small steps on the large end of the part are omitted on the cold-formed part (Figure 5). It may be possible to cold form these steps, but it will require very complex and delicate tooling so they are left for machining after the cold forming.
The task of developing the cold formed part shape from the final machined part drawing, which is described above, is not studied in detail in this project. This task is also suitable for the application of knowledge-based systems methodologies. Rules and guidelines on cold formability should be acquired to develop such a system. These rules and guidelines should include the constraints on tolerances, surface finish, undercuts, and chamfers that can be achieved by cold forging. Such a system can work in conjunction with the system developed in this study for forming sequence design.

2.3.1.2 Forming Sequence Design. After designing the cold-formed part from the drawings of the final machined part, as seen in Figure 5, the second task is to establish to cold forming sequence to forge this part. Given the cold formed part drawing in Figure 5, the designer finds the forming sequence shown in Figure 6. The same forming sequence is shown in Figure 7 with the proper dimensions. The arrows at the top of the forming sequence designate the transfer of the workpiece from one station to the next. A curved arrow means that the workpiece is rotated during the transfer. The decisions of the designer to achieve the cold forming sequence in Figure 7 from the cold-formed part drawing in Figure 5 can be briefly described as follows.

First, the billet diameter is selected. In this case, a billet diameter of 0.730 inch (Figure 7) is chosen based on the 0.750-inch finished diameter of the main body of the part (Figure 5). The volume of the billet is calculated from the blank prior to piercing in Station 5. To do this, the blank is divided into simple geometric
ESTABLISH COLD FORMING SEQUENCE

1. USE .750 DIA. AS BASIS FOR PICKING WIRE SIZE AT APPROX. .730 DIA.
2. FORM .500 DIA. BY FORWARD EXTRUSION FROM .730 DIA. WIRE (1ST DIE) .730 TO .500 = 53% RA
3. BACKWARD EXTRUDE HOLE AT APPROX. .368 DIA. FROM .745 DIA. (2ND DIE) .745φ TO .368φ = 24% RA
   .625 HOLE DEPTH = 1.7 DIA.'S DEEP
4. BACKWARD EXTRUDE HOLE AT APPROX. .365 DIA. FROM .500 DIA. (3RD DIE) .500φ TO .365φ = 53% RA
5. UPSET 1.000φ FLANGE (4TH DIE)
6. PIERCE CENTER WEB (5TH DIE)

Figure 6. Illustration of the Forming Sequence Design for the Hypothetical Part in Figure 5 [8].
Figure 7. Dimensioned Cold Forming Sequence for the Hypothetical Part in Figure 5 [8].

shapes such as cylinders and conical shapes, volumes of each individual shapes are calculated, and the summation of these gives the total volume of the part. In this case, it is found out to be 0.399 cubic inch [8]. The cutoff length can then be calculated from the volume of the part and the diameter of the wire. It turns out to be 31/32 inches. In the first die, it is decided to trap forward extrude 0.500 inch from 0.730 inch. The reduction in area is 53 percent and within the limits for the trapped forward extrusion. In the second die, 0.368-inch diameter hole in the large body of the part is backward extruded. The reduction in area is 20 percent. Even though this is close to the
minimum reduction in area for backward extrusion it is still acceptable. The depth of extrusion in this station is 1.7 diameters which is within the backward extrusion limits.

In the third die, a 0.365-inch diameter hole is backward extruded at the small end of the part. The backward extrusion reduction in area is 53 percent and well within the limits.

In the fourth die (Figures 6 and 7), 1-inch diameter flange is upset. The upset length should be checked at this station. Since the flange is very thin and the starting diameter (0.750 inch) is large, buckling during upsetting is not possible. The center hole is pierced in the fifth die, and this completes the forming of the part.

The third task is to design the cold forming dies to produce this part. The dimensioned blank sizes shown in Figure 7 are used to design the dies for each station.

The topic of this study is the second task described above, namely the forming sequence design. It is the process of establishing the cold forming sequence (Figures 6 and 7) from the cold-formed part drawing (Figure 5). The designer uses his experience and intuition as well as some basic forming rules to establish the forming sequence. Forming sequence design is often a trial and error process. Even though there are some principle guidelines for forming sequence design [8], there is not a well-developed methodology for it. There are some very experienced die designers performing this task. Besides some common basic principles, they all have their own styles of forming sequence design. Forming sequence design for cold forging is an art, and different artists (die designers) have different styles.
The task of this study is to analyze the forming sequence design process and to develop a CAD system to assist the designer in this complex planning/design problem. Previous attempts to develop computerized tools for forming sequence designs are briefly reviewed in the next section.

2.3.2 The State of the Art

Currently, forming sequences for cold forging are established by experienced die designers who have been doing this job for many years. They use the knowledge and experience that they have acquired through these years. One of the tools they can use to ease their job is computer-aided drafting systems. These systems may help them to plot the final blank dimensions faster and the drawings of the previous jobs can be stored in the files. When a new forming sequence is to be developed for a given part, these files can be checked to display forming sequences for similar parts that were designed before. But, for each case, they may not expect to find a forming sequence designed for a similar part. Even if they are lucky to find one, they still need the experienced designers to make the necessary modification on these drawings for the new forming sequence. It is important to realize that, in cold forging, two very similar parts may have different forming sequences. Figure 8 shows an example of this where two parts, with the same overall upsetting ratios, require different intermediate upsetting stages, because the smaller upset diameter must be produced in the first blow in one case but not in the second. Therefore, well classified
computer archives of forming sequences does not eliminate the need for knowledgeable, experienced designers.

Several studies have been made to develop tools to assist designers in forming sequence design for round parts [10-17]. In some of them, computer programs are developed to speed up the process of
forming sequence design. The use of computers in this area has been very similar to the use of computers in the rest of the mechanical design discipline—they are used as number crunchers to speed up mathematical calculations or as drafting tools using computer graphics capabilities. The computer programs developed for cold forming sequence design helps the designer to establish forming sequences faster, but they have not eliminated the need for a very experienced and knowledgeable designer to do the job.

Can the use of computers go one step further in forming sequence design; can they capture the knowledge of experienced designers to some extent [18,19]? If this is possible, not only the design process will be faster, but less experienced and less knowledgeable designers will be able to perform as well as the experienced ones with the help of these computer programs.

The recent developments in artificial intelligence are promising, and the knowledge-based systems approach seems suitable for forming sequence design.
CHAPTER 3

KNOWLEDGE-BASED SYSTEMS APPROACH FOR ENGINEERING DESIGN AND MANUFACTURING

This chapter gives a background on artificial intelligence (AI), knowledge-based systems (KBS) and the application of KBS in engineering design and manufacturing.

3.1 ARTIFICIAL INTELLIGENCE

3.1.1 Definition

One popular way of defining AI is "getting machines to do things that people would agree require intelligence" [20], or "the study of ideas that enable computers to be intelligent". [21] Here the definition of intelligence is carefully avoided because, even though we know that it has something to do with the ability to reason, the ability to acquire and apply knowledge and the ability to perceive and manipulate things in the physical world, it is very difficult to define "intelligence". Another definition of AI is "the study of mental facilities through the use of computational methods". [22] The two central goals of the field of AI are to understand the principles that make intelligence possible and to make computers more useful. [21]
3.1.2 History

The history of AI may be traced all the way back to 1840s when Babbage was putting together the ideas of the first computer. Around this time people were already talking about whether it would be possible to make machines intelligent or not. The two important forces for the evolution of AI in 1930s and 1940s were mathematical logic which had been under rapid development since the end of the 19th century, and the new ideas about computation. [23] Frege, Whitehead and Russell, Tarski and others tried to show that some aspects of reasoning could be formalized by logical systems. Even before there were computers, mathematical formalization of logical reasoning was thought to explain the relation between computation and intelligence.

The abstract conception of computation as "symbol processing" by Church and Turing was very important. Early computers were seen as numerical calculators. But Church and Turing had thought even before computers that numbers were not an essential aspect of computation but only one way of interpreting the internal states of the machine. Turing invented a simple, universal, and nonnumerical model of computation and also claimed that there is the possibility that computational mechanisms could behave as "intelligent".

The development of computers accelerated the study on machine intelligence. People started to write programs to solve puzzles, play chess, and translate text from one language to another. These are accepted as the first AI programs. Several ideas about computing relevant to AI emerged from the early computer designs, such as ideas about
memories and processors, about systems and control, and about levels of languages and programs. [23]

Major developments in the field of AI started by 1960. By 1965, the basic ideas that initiated today's start-up companies were in place. [24] Early work on AI was focused around several areas. Problem solving was one of them. First successes in AI were programs that could solve puzzles and play games like chess.

Another area close to problem solving was logical reasoning. Programs were written to prove assertions by manipulating a set of facts similar to the formulas in mathematical logic. Logical reasoning is now a major area in AI and is described further under logic programming.

Language is another area which attracted lot of interest by early AI researchers. Programs were developed to translate sentences from one language to another, to follow instructions given in English, and that acquire knowledge by reading textual material and building an internal database. [23] The failures as well as successes in language understanding research have focused attentions on such principle issues as the importance of vast amounts of general commonsense world knowledge and the role of expectations based on the subject matter when interpreting sentences.

Learning is another area where there were several interesting attempts in early AI research. Programs were developed which could "learn" from examples, from their own performance and from being told. But, not much success can be claimed so far in this area.

Although the early results supported the theoretical possibility of machine intelligence, they could not provide a basis for
constructing programs that could solve complex practical problems. The early hope that a relatively small number of powerful general mechanisms would be sufficient to generate intelligent behavior has disappeared. [25] General problem-solving techniques could not handle imprecisely stated "problems" and uncertain "facts". When significant problems were addressed the problem-independent heuristics methods could not handle the sheer combinatorial complexity of the problem. AI research went through some changes with the growing recognition of the many kinds of knowledge required for high performance reasoning systems. The focus of AI shifted from the identification of a few powerful techniques to how to represent large amounts of knowledge in a fashion that permits their effective use and interaction. This is also described as a shift from "power based" strategy to a "knowledge-based" approach. [25] One of the results of this shift is the development of programs called knowledge-based systems, which is discussed in the next section.

This section concludes with some remarks about the current status and the future of AI. There are very different opinions about the developments in AI in the last decade. Some (such as A. Sloman, H. Dreyfus, M. Boden and T. Winograd) believe that AI has not developed much in the last decade. [26] Even though the number of people and the amount of money committed to AI have increased rapidly, they believe that the central problems of AI, and the theoretical basis of its achievements, have remained essentially the same. Others (such as S. Amarel, L. McCarthy and R. Shank) admit that many problems are much harder than was thought but also believe that there was significant
progress recently. Others (such as J. Brown, J. Feldman and A. Newell) are more optimistic and see AI as in the process of becoming scientifically mature. They believe that the future progress may appear to be slower than in the past but will probably be more substantial. Currently the most significant topics in AI are qualitative reasoning, learning, connectionisms (or massive parallelisms), and expert systems. [26]

While almost everyone agrees on the bright commercial future of expert systems, some scientist believe that most of the work done in this area can not be considered basic AI research. Some (like M. Minsky) believe that none of the artificial intelligence groups in industry is doing really basic research. [27] On the other hand, currently the important contribution of expert systems research may be not the development of high performance programs but the systematisation and codification of knowledge previously thought unsuited for formal organization. New approaches on formalizing and managing knowledge are certainly important for a variety of scientific and economic endeavors. [25]

3.2 KNOWLEDGE-BASED SYSTEMS

3.2.1 Definition

A knowledge-based system is defined as an AI program whose performance depends more on the explicit presence of a large body of knowledge than on the possession of ingenious computational procedures. [25] Knowledge-based systems are also called expert systems
(ES) when their performance is intended to be as good as human experts. According to these definitions of KBS and ES, the system developed in this study could be called both a KBS or an ES as discussed in Chapter 4. Another definition of ES given by Feigenbaum is "an intelligent computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require significant human expertise for their solution." [28]

Sometimes the distinction between knowledge-based systems and expert systems is not clear. In expert systems a set of domain experts' knowledge is emphasized whereas in knowledge-based systems the knowledge representation formalism such as rule, frame, etc., is emphasized. The knowledge of human experts, which is often heuristic or rule of thumb, is not necessarily essential in some knowledge-based systems. [29] The purpose of expert systems is to perform an intelligent task which has formerly been carried out by a human expert.

3.2.2 History

As mentioned in Section 3.1.2, in 1970s, there was a shift in AI research from "power based" strategy to a "knowledge-based" approach. The development of knowledge-based programs is one of the results of this shift. By mid-1970s several expert systems have been developed. Some scientists (such as Winograd, Minsky and Brachman) who believed in the important role of knowledge in these systems worked to develop comprehensive knowledge representation theories and general-purpose systems.
In 1977, at the Joint International Conference an Artificial Intelligence, Feigenbaum presented important ideas on AI which is considered by some researchers as the key insight to expert systems. He stated that the power of an expert system derives from the knowledge it possesses, not from the particular formalisms and inference schemes it employs. This was a major change in the perspectives of many researchers in AI. [30] After this, "knowledge is power" became the slogan for some AI researchers.

To give an overview in the developments of knowledge-based systems, some of the well known systems will be briefly described.

One of the earlier projects on KBS was the DENDRAL project which started in 1965. [31,32] The main idea was manipulating large amounts of expert, heuristic knowledge into a form that a program can use to help solve difficult problems. The task was to infer molecular structures from mass spectrographic information. Since the number of all possible molecular structures was huge, an exhaustive search would take too long. Heuristic knowledge of chemists is represented as rules which controls the search in DENDRAL. The concept of data-directed search control was first used by DENDRAL. Given the mass spectrographic data, the rules in DENDRAL can prune the search space by reducing the number of possible molecular structures.

CASNET (Casual Associational Network Program) which was developed in the early 70s, is used in the diagnosis and therapy of glaucoma. [33] CASNET has a general framework for modeling diseases. This project provided a general framework for building expert systems. A general tool, called EXPERT, was developed and applied to rheumatology
and endocrinology. CASNET uses a causal network to model diseases. It can also accommodate probabilistic rules and can give alternative diagnoses by referring to different expert opinions.

The early algebraic manipulation programs for general mathematical simplications led to a program called MACSYMA. MACSYMA performs symbolic computations associated with applied analysis. Currently it is being used daily by several mathematicians.

About the same time as CASNET was developed, another important program MYCIN was developed to give consultative advice on diagnosis and therapy for infectious diseases. In MYCIN, medical knowledge is represented as production rules with certainty factors. MYCIN is a hypothesis driven system and it uses backward-chaining control strategy. MYCIN can give the reasons for its decisions in terms of its rules. It has approximately 400 rules relating possible conditions to associated interpretations. A domain independent version of MYCIN, called EMYCIN was produced at Stanford. EMYCIN is the same as MYCIN except its knowledge of infectious diseases. It is a more general framework for building expert systems and several other diagnostic systems, such as PUFF, were developed using it.

TEIRESIAS was developed for automatic knowledge acquisition for the MYCIN system. It uses knowledge about how MYCIN knowledge is presented and used (metaknowledge). It has rule models and the rules about the structure of rules in MYCIN. It can detect faulty rules as they enter the system. TEIRESIAS fills in much of a new rule given the context of the dialogues and its own knowledge of what a rule should look like. Another important feature of TEIRESIAS is its better
explanatory ability since it can present its understanding of a rule at various levels of detail.

In 1970s several speech-understanding systems were developed. The most important one was HEARSAY-II. An important idea in HEARSAY II is the "blackboard" concept which is a global working memory in which different types and levels of information are integrated into a uniform structure. The advantages of this idea are the elimination of unnecessary recalculations and the simpler access to the global view of the current state of the solution. Different knowledge sources can use the blackboard. Each knowledge source has a different type of knowledge that can be applied to the problem. This idea of independent knowledge modules increases the ability to modify and acquire knowledge.

CADUCEUS, previously called INTERNIST, is another medical consultation system that attempts to make a diagnosis in the domain of internal medicine. It has one of the largest knowledge-bases among expert systems. Similar to CASNET, CADUCEUS represents its knowledge in a disease tree structure. The program creates disease models as it goes and dynamically partitions the disease tree into disease areas corresponding to the current case. It uses both data-directed and hypothesis-directed reasoning. First patient's data is used to predict hypotheses, and these are then used to predict other facts that must either be confirmed or used to change the hypotheses. CADUCEUS has about 500 diseases, 350 disease manifestations, and 100,000 symptomatic associations. Its explanation facilities are minimal.
3.2.3 Rule-Based Systems

An important class of expert systems, called rule-based systems, will be briefly discussed in this section since the system developed in this study falls into this category.

Rule-based systems were first studied by Post [42] in early 1940s. Since then they have gone through major developments in AI such that the current rule-based systems are very different than Post's formulation. Rule-based systems are so dominate in the ES area, some authors [22] will use "rule-based systems" as a synonym for "expert systems". In fact, rule-based systems are only a class of expert systems which are based on one very general idea, called production rules or just productions. This is why rule-based systems are sometimes called "production systems".

Currently, most successful expert systems are rule-based systems. Most of the rule-based systems perform analysis tasks such as medical diagnosis, electronic troubleshooting, or data interpretation. Current rule-based systems are specialist in very narrow areas and have very limited (but some) abilities to acquire new knowledge or explain their reasoning. [43]

A rule-based system mainly consists of three parts: [23] (a) a "rule base" composed of a set of production rules, (b) a special buffer like data structure which is called "context" or "short-term memory buffer" and (c) an "interpreter" which controls the systems activity.

A production rule is a condition action pair in the form "If this condition holds, then this action is appropriate. The "if" part of
the rule is called production part or the left-hand side and it states
the conditions that must be present for the rule to be applicable. The
"then" part of the rule is called the action part or right-hand side and
gives the appropriate action to take. For example,

IF stop light is red AND you have stopped

THEN right turn is OK.

When all the conditions of a rule are satisfied the rule is said to be
"triggered". [21] When the actions are executed, then the rule is said
to be fired. Therefore, triggering does not always mean firing, since
the conditions of several rules may be satisfied simultaneously. When
several rules are triggered, a conflict resolution scheme decides which
rule to fire.

Context is the working memory where production rules are try-
ing to match their conditions with. For a production rule to be fired,
all of its conditions must be present in the context data structure.
Firing production rules can change the context by its actions.

The interpreter controls the execution of the program. It
decides what to do next. The main task of the interpreter is to decide
which rule to fire next. In a classical production system the inter-
preter executes rules in a "recognize-act" cycle. The rule interpreter
cycles through the condition parts of the rules, looking for one that
matches the current context and executing the associated actions for
rules that do match. If there are more than one rule that is triggered,
rule interpreter uses a conflict-resolution strategy to choose the one
to fire. Some conflict-resolution strategies are: [21]
• Specificity ordering: If the conditions of one triggering rule are the superset of the conditions of another triggering rule, use the rule with the superset since it is more specialized for the current context.

• Rule ordering: Arrange all rules in a list with decreasing priorities. The triggering rule appearing earliest in the list fires, all the others are ignored. This is the strategy used by the Prolog interpreters.

• Data ordering: Arrange all possible aspects of the situation (all the conditions in the rules) in one long priority list. The triggering rule which has the highest priority fires.

• Size ordering: The triggering rule having the longest list of constraining conditions fires.

• Recency ordering: The triggering rule which is most recently used fires. The designer may instead want to give the highest priority to the least recent rule.

• Context limiting: Separate the rules into groups and let only some of them be active at a given time. This will decrease the possibility of conflicts since the number of active rules is reduced. A procedure activates or deactivates the groups.

No simple conflict-resolution strategy can be completely satisfactory. Rule-based systems use different combinations of these simple strategies for conflict-resolution. For example, XCON, [44] a rule based system that configures VAX-11 computers uses both the
specificity ordering and context limiting strategies for conflict resolution. In some cases conflict-resolution strategies may become quite complicated scheduling algorithms as in HEARSAY-II. [40]

Among the three phases of each production-system cycle (matching, conflict resolution and action), the matching process uses by far the most computational time. [23] This becomes a more important issue as the production systems get bigger and more complex. To increase efficiency rule-bases and contexts are built as more complex data structures. In order to determine the rules that are applicable in a given situation without checking through all of the rule base, the rules are sometimes indexed or partitioned according to conditions that will make them fire. Complex context data structures are used in some rule based systems (such as context tree in MYCIN, blackboard in HEARSAY II and semantic net in PROSPECTOR), to increase efficiency and to be able to represent complex knowledge.

Deduction-oriented rule-based systems use two different methods of inference: forward chaining or backward chaining. Forward chaining systems which are also called data driven, event driven or bottom-up, work from known facts to new, deduced facts. Backward chaining systems which are also called goal driven, expectation driven or top-down, hypothesize a conclusion and use the antecedent-consequent rules to work backward toward the hypothesis-supporting facts.

Rule-based systems are more suitable for some domains than others. More appropriate domains for rule-based systems are suggested as: [45]
1. domains in which the knowledge is diffuse, consisting of many facts (e.g., clinical medicine), as opposed to domains in which there is a concise, unified theory (physics);

2. domains in which processes can be represented as a set of independent actions (a medical patient-monitoring system), as opposed to domains with dependent subprocesses (a payroll program);

3. domains in which knowledge can be easily separated from the manner in which it is to be used (a classificatory taxonomy, like those used in biology), as opposed to cases in which representation and control are merged (a recipe).

There are several advantages and some disadvantages of rule based systems. An important characteristic of rule-based systems is modularity. [23] The rules in the rule-base are independent of each other to some degree. Individual rules can be added, deleted or changed without effecting the rest of the system too much. However, there are indications that it is not easy to maintain modularity as the systems get larger and more complex. [23]

The uniform structure of the rule-base is another advantage for rule-based systems. Since the knowledge is represented in a rigid structure as production rules, it is easy to understand by another person or another part of the system itself. The rigid structure of rule-based systems make them easier to understand than other representation schemes such as semantic nets and procedural systems.
Another important advantage of rule based systems is the ease with which one can express certain important kinds of knowledge. [23] Statements about what to do in predetermined situations can be naturally represented in production rules. Most of the time experts use these kind of statements to explain how they perform their tasks.

One disadvantage of rule-based systems is inefficiency of program execution. The modularity and uniformity of the rules may cause high overheads. Since production systems perform every action by means of the match-action cycle, it may be difficult to respond efficiently to predetermined sequences of situations. [23] The Prolog interpreter solves this problem by the backtracking control scheme.

Another disadvantage of rule-based systems is the difficulty of expressing algorithmic knowledge. It is difficult to follow the flow of control, since the rules are isolated from each other and the size of the rules are uniform. Again logic programing and Prolog offers some solution to this problem as discussed in the next section.

3.2.4 Introduction to Prolog

Since the result of this study includes a system which is implemented in Prolog language, an introduction to the Prolog is presented in this section with several examples. This section is intended to give a "feel" for what it is like to program in Prolog and most of the material is taken from the excellent book [46] by Clocksin and Mellish on "Programming in Prolog" and the tutorial notes from IJCAI 9 [47]. A more detail discussion of logic programming, its
history, advantages, limitations, and expected developments in this area are presented in the next section.

Prolog is a programming language that is used for solving problems that involve objects and the relationships between objects. [46] For example, when we say "John owns the book", we are declaring that a relationship, ownership, exists between one object "John" and another individual object "the book".

Programming in Prolog consists of [46]:

- declaring some facts about objects and their relationship,
- defining rules about objects, and their relationships, and
- asking questions about objects and their relationships.

Important concepts in Prolog can be explained by introducing some fundamentals, such as facts, conjunctions, and rules.

3.2.4.1 Fact. Suppose we want to tell Prolog that "Fred teaches English". This fact consists of two objects, called Fred and English and a relationship, called "teaches". In Prolog, this fact is written in a standard form, as:

\[
teaches(fred,english).
\]

There are three important points: [46]

- The names of all the relationships and objects must begin with a lower-case letter.
- The relationship is written first, and the objects are written separated by commas, and the objects are enclosed by a pair of round brackets.
• The full stop character "." must come at the end of a fact.

Following are some examples of Prolog facts and their explanations in English: [47]

- teaches(john,cemistry). "John teaches chemistry"
- teaches(laura,mathematics). "Laura teaches mathematics"
- teaches(fred,english). "Fred teaches english"
- science(chemistry). "Chemistry is a science subject"
- science(mathematics). "Mathematics is a science subject"
- humanities(english). "English is a humanities subject"

3.2.4.2 Querying the Database. In Prolog questions can be asked about the facts in the database. A question looks just like a fact, except that we put a special symbol before it. The special symbol consists of a question mark and a hyphen. Given the facts in the previous section, following are the answers of Prolog to some questions:

- ?-teaches(fred,english). "Does Fred teach English?"
  yes
- ?-teaches(fred,mathematics). "Does Fred teach mathematics?"
  no
- ?-teaches(john,X). "What does John teach?"
  X=chemistry
- ?-teaches(W,mathematics). "Who teaches mathematics?"
  W=laura

3.2.4.3 Conjunctions. Questions about more complicated relations can be asked using conjunctions. Suppose we want to ask the question
"Who teaches a humanities subject?"

This can be represented in Prolog, as:

```
teaches(P,H),humanities(H).
```

The comma stands for "and", and it serves to separate any number of different goals that have to be satisfied in order to answer a question.

Given the facts in section 3.2.4.1, Prolog answers this question as follows:

```
?-teaches(P,H),humanities(H).
P=fred
H=english
```

3.2.4.4 Rules. In Prolog, rules are used when you want to say that a fact depends on a group of other facts. A rule consists of a head and a body. The head and body are connected by the symbol ":-" which is made up of a colon and a hyphen. The ":-" is pronounced "if". For example the Prolog rule

```
science_teacher(P):-teaches(P,S),
   science(S).
```

means

P is a science_teacher if P teaches subject S and S is a science subject.

Following are some example questions and the answers of Prolog for the above rule and the facts in section 3.2.4.1:

```
?-science_teacher(fred).    "Is Fred a science teacher?"
no
?-science_teacher(W).      "Who is a science teacher?"
```
3.2.5 Logic Programming: History, Advantages, Limitations, and Future

After the brief introduction to the Prolog language in the previous section, logic programming is discussed in this section. This section concentrates on the history, advantages, limitations, and the future of logic programming.

Mathematicians have often worked on systematization of mathematical proof. The combination of a set of inference rules with a strategy for applying them makes an algorithm called a proof procedure. [48] A proof procedure can be coded as a computer program. Such a program can be executed to generate logical inferences from logic sentences supplied as data. This is called automatic theorem proving. The work on programmed proof procedures were first concentrated on mathematical theorem proving. Computers were thought to accelerate the pace of mathematical discovery by providing proofs of significant theorems whose proofs were too lengthy.

Automatic theorem proving has also played an important role in artificial intelligence in the manipulation of knowledge by logical inference. It has been applied to such tasks as question answering, game playing, and state-space problem solving. The main reasons for its success are the sufficient expressiveness of logic for representing some kinds of knowledge and the power of logical inference for processing it. [48]
Many researchers tried to find systematic and efficient proof procedures for first-order logic. A major step was the discovery of the resolution rule by Robinson. Resolution is a generalization of modus ponens with a powerful pattern-matching operation called unification. Resolution has the important properties of soundness and completeness. Resolution and a simplification step called factoring are the only inference steps required to build a complete inference system for predicate logic. Resolution applies on sentences known as clauses. Every set of sentences in first-order logic can be converted to a set of clauses having identical satisfiability properties. Sentences consisting only of denials, assertions, and implications form a subclass of clausal forms and are called Horn clauses. Sometimes logic programming language is referred to as "the Horn clause subset of logic". Logic programming has originated from the advances in automatic theorem proving and the development of the resolution principle. The adaptation of theorem-proving concepts to the computational techniques already understood by programmers was possible by the procedural interpretation formulated by Kowalski. A Horn clause

\[ B \leftarrow A_1, \ldots, A_m \quad m \geq 0 \]

is interpreted as a procedure whose body \( \{A_1, \ldots, A_m\} \) is a set of procedure calls \( A_i \). Generation of a new goal statement from an old one by matching the selected procedure call with the name \( B \) of a procedure

\[ B \leftarrow A_1, \ldots, A_m \]

is called procedure invocation. Several advances in implementation technology have also contributed to the presentation of logic as a practical formalism. The
first interpreter was developed by Roussel, Colmerauer, and others at the University of Aix-Marseille in 1972 and was called Prolog. This system has strongly influenced the design of the later systems. Among the several later more practical implementations, Warren's DEC-10 Prolog [52] developed at the University of Edinburgh is the most notable. Since then there has been several Prolog implementations covering a wide range of design philosophies, host machines, and application environments. [48]

Sometimes the terms "logic programming" and "Prolog" are used interchangeably but Prolog's default strategy is not the only one available for executing logic programs. Another strategy developed by Kowalski underlies the connection graph proof procedure and operates using a special scheme of link activation applied to the links in a graph connecting calls to responding procedures.

Prolog systems also differ from each other since they provide different additional facilities for enriching the programmer's resources. In most interpreters, the user can modify the control strategy. Such enhancements to the standard strategy may give rise to computations which cannot be wholly justified in terms of logical inference and these kinds of interpreters are said to be "impure".

Logic programming requires the programmer to have a clear understanding of the relations pertinent to the problem. The programmer expresses these relations by stating their relevant logical properties in the simple language of assertions and implications. The problem itself can be viewed as a query about the contents of the relations.
After the assumptions and the query are formulated, they are submitted as an input to a programed resolution system implemented on the computer. If the query is answerable on the basis of the knowledge presented as the input assumptions, then the answer is inevitably discoverable by the system. Any solution computed has the desirable property of being logically implied by the given assumptions and the entire computation is directly comprehensible as a process of rational and systematic reasoning. [48] Very few existing programming formalisms have these properties.

Data structures in Prolog are general trees which are constructed from various types of record. An unlimited number of different types may be used and they do not need to be declared. [52] Records can have any number of fields and there are no type restrictions on the fields of a record.

In Prolog there is no distinction between procedures and what would conventionally be regarded as tables or files of data. Program and data can be mixed together and they are accessed the same way. In general, a Prolog procedure consists of a mixture of explicit facts and rules for computing further "virtual" data. Because of these properties, Prolog has interesting potential for a query language for a relational database. Examples of Prolog rules and facts are given in the previous section.

A Prolog variable stands for a particular unchangeable data item. The actual value of the variable need not be specified immediately and may remain unspecified for as long as is required. The programmer need not be concerned whether or not the variable has been given
a value at a particular point during execution. This is entirely a consequence of constraints arising from logic. [52]

The variable in most other languages is a name for a machine storage location. The programer assigns values to the variables and in many situations he must guarantee that the variable is not left unassigned. This is also true for the variables used in the Planner family of pattern-matching languages. In these languages each occurrence of a variable in a pattern has to be given a prefix to indicate the status (assigned or unassigned) of the variable at that point. Using these languages, the programer has to understand the details of the implementation and sequencing of the pattern-matching process; whereas, in Prolog the unification process is a black box for the user.

A Prolog procedure may return as output a "complete" data structure containing variables whose values have not yet been specified. [52] These "free" variables can be "filled in" later by other procedures. This is achieved during the normal matching process, but has much the same effect as explicit assignments to the fields of a data structure. When two variables are matched together, they become linked as one, that is, a reference to one variable is assigned to the cell of the other. These references are completely invisible to the user and all the dereferencing necessary is done automatically by the interpreter.

A unique and fundamental property of Prolog is that it can be interpreted declaratively as well as procedurally. This makes Prolog an easier language to use. In most programming languages, a program is simply a description of a process. To understand the program and check
whether it is correct or not, the user either runs the program on a computer or in his mind. When considering efficiency, this can also be done for Prolog programs. It is said that, like other languages, Prolog has a procedural semantics, one which determines the sequence of states passed through when executing a program. [52]

A Prolog program can also be interpreted declaratively, as a set of descriptive statements about a problem domain. In this case the statements of the program are a convenient shorthand for ordinary natural language sentences. Each clause is a statement which makes sense in isolation. The program is correct if each clause is true. For example: [47]

\[
P : - P1, P2, P3.
\]

has a procedural reading

"To prove P, prove P1, then prove P2, then prove P3."

and a declarative reading

"P is true if P1 is true, and P2 is true, and P3 is true."

The natural declarative reading is possible because the procedural semantics of Prolog is governed by an additional declarative semantics which is inherited from logic. [52] The statements of Prolog programs are, in fact, actually statements of logic. The facts that can be inferred from these clauses are defined by the declarative semantic. When programing in Prolog, one can initially ignore procedural details and concentrate on the (declarative) essentials of the algorithm. Decomposing the problem into small independently meaningful units makes it much easier to understand. This inherent modularity may also reduce the interfacing problems when several programers are working on the same project.
Prolog is a transparent and easy-to-use language and very suitable for knowledge-based systems applications. It is easy to understand mainly because it is formulated in small units which have a natural declarative reading. Prolog also allows a wide range of problems to be solved without resort to machine- or implementation-oriented concepts. The logical variable and "iteration through backtracking" go a long way towards removing any need for assignment in a program.

In 1981 a research project known as Fifth Generation Computer Systems (FGCS) was started in Japan for research and development of the next generation of computers. [53] The results of the first phase were reported at an international conference in Tokyo in November 1984. The hardware results include a workstation known as personal sequential inference machine (PSI) and a relational database machine known as the Delta machine. Software systems demonstrated at the conference included Kernel Language 0 (KLO) the Prolog derivative used as the machine language for the PSI, and Extended Self-Contained Prolog (ESP), an object-oriented extension of KLO used for system programming on the PSI. [54].

The initial announcement of Fifth General Computer Systems Project in the Fall of 1981 attracted lots of attention on logic programming. The results of this project may be even more convincing to show the power of logic programming for knowledge-based systems applications.
3.3 KNOWLEDGE-BASED SYSTEMS FOR ENGINEERING DESIGN

Computer Aided Design (CAD) systems have been used in mechanical design in different ways, such as drawing, checking the motions of mechanisms, generating NC machinery data and finite-element methods. Designers have saved considerable time using computers for mostly number crunching tasks. But, in general, designers do not have creative or intelligent support from computers as yet.

Design tasks can be divided in five stages: [55]

1. Conceptual design - decide the basic method and basic structure of the design solution,
2. Basic design - decide the layout and the structure of the design solution,
3. Detail design - decide the minute specifications of the parts,
4. Production design - generate the necessary data for production,
5. Prototyping and test - trial manufacturing and the test.

In reality there are many loops within the various stages when some requirement are found being unsatisfied.

Currently CAD is mostly used in "detail design" stage which is the least time-consuming stage in many cases. [55] The use of computers in the other design stages will be drastically increased if computers can provide creative and intelligent support to designers.

The knowledge-based systems approach described earlier seems promising for developing creative and intelligent assistants for
designer. KBS's is a new approach for implementing so-called knowledge as it is. For instance, it is fairly difficult to implement knowledge which is written in mathematically exact logic on a computer using a conventional procedural computer language. Logic programming can provide a flexible solution. Machine design includes knowledge that cannot be described in a procedural way but can be described only in a declarative or illustrative way. [55] Engineering design also requires knowledge that comes from skilled designers. The task of forming sequence design for cold forging discussed in Chapter 2 is a good example where heuristic methods of problem solving are effective. Therefore KBS's approach is important and useful for engineering design and manufacturing applications.

Although there are many attempts to develop KBSs in areas such as interpretation, diagnosis, and prediction, the attempts in engineering design and planning are much less (surveys of KBSs in manufacturing is given in references [56] and [57]). The difficulty of developing KBSs for design and manufacturing lies in the requirements and the characteristics of the design tasks.

There are several requirements on applying KBSs to design tasks. [58] A designer must accomplish the design without consuming excessive resources or violating constraints. If goals conflict, the designer establishes priorities. If design requirements are not fully known or change, the the designer must be flexible and opportunistic.

The key problems on applying KBSs to design tasks can be summarized as: [58] (1) In large tasks the designer cannot immediately assess the consequences of design decisions. He may need to explore
design possibilities tentatively. (2) In large tasks a designer must cope with the system complexity by factoring the design into subtasks which are seldom independent. (3) When the task is large, it is easy to forget the reasons for some design decisions and hard to assess the impact of a change to a part of a design. This suggests that a design system should record justifications for design decisions and be able to use these justifications to explain decisions later. This is especially important when subsystems are designed by different designers. (4) In modifying a design it is important to be able to reconsider the design possibilities. In redesign, designers need to be able to see the task at a higher level in order to escape from points in the design space that are only locally optimal. (5) A very important problem is that many of the design problems require reasoning about spatial relationships. Reasoning about distance, shapes and contours demands considerable computational resources and powerful representation schemes. Currently there are no good ways to reason approximately or qualitatively about shape and spatial relationships.

In design and planning tasks the number of reasonable solutions is usually a very small fraction of a very large number of possible solutions. The size and characterization of the solution space is an important organizational parameter.

Design and planning systems require the ability to undo the effects of assumptions since during design and planning processes it may often be discovered that some assumptions are unwarranted. In almost
any design and planning task, some of the assumptions will fail, so there is an incentive to employ methods that facilitate the reworking of assumptions and trade-offs during iterations of the design process.

During engineering design, the designer uses various abilities such as intuition, creation, association, induction, abduction, recognition, deduction, as well as relatively low-level functions such as computation, searching and retrieving, pattern matching, etc. [59] Among these only a few functions; computation, searching and retrieving, pattern matching, and deduction have been analyzed in detail so that algorithms to achieve these functions are obtained. We can now implement these functions by a machine. The other functions remain unanalyzed. We cannot represent these functions in a machine because we do not know how these functions are performed. In the near future computers may not perform functions higher than deductive inference. There are several studies toward the higher-level intelligent functions such as non-monotonic logic, reasoning by analogy, inductive inference, probabilistic reasoning, etc. [59] But it will require some time to analyze these functions precisely and implement in computers.

Engineering design is one of the most intelligent activities of man. There are several attempts to formalize design process and develop a general model for engineering design. [55,59-64] This is a very difficult task and currently there are no adequate models of the general design process. It is recommended that current attempts to develop KBSs in engineering design should select a specific class of design problem. [65]
Given the above difficulties of more general design systems, this study was concentrated on a rather specific task. Even though there were some more general design issues studied, this study was focused on a more specific task of forming sequence design for cold forging as described in the next chapter.
Forming sequence design for cold forging was described in Chapter 2. It was explained that this task is reasoning intensive rather than computation intensive and there is a need for a CAD system to assist the less experienced designers in this task. The developments in AI, knowledge-based systems and their applications in engineering design were summarized in Chapter 3. This chapter describes an attempt to apply the AI methodologies summarized in Chapter 3 to the domain of forming sequence design explained in Chapter 2.

4.1 FORMING SEQUENCE DESIGN AS A DESIGN AND PROCESS PLANNING TASK

The task of forming sequence design was defined in detail in section 2.3. It will again be briefly explained for the sake of completeness. The designer is given (1) the cold form part geometry which is obtained from the the final finished part geometry (Figure 5), (2) the possible materials from which this part is to be formed and (3) the available press specifications. He is asked to establish the forming sequence (or "preforming" steps) to produce this part (Figure 6). This task can both be considered to be a process planning task or a design task.
Figure 5. Obtaining the Cold-Formed Part Geometry from the Finished Part Geometry [8].

Forming sequence design is similar to process planning used in manufacturing via machining. Process planning includes phases such as:

1. selection of processes and tools,
2. selection of machine tools,
3. sequencing the operations,
4. grouping of operations,
5. selection of workpiece holding devices and datum surfaces, etc.

In process planning for machining, the different processes, such as drilling, milling, grinding, etc. must be selected first. Similarly in forming sequence design, the designer must select the processes, such as upsetting, forward extrusion and backward extrusion.

The next phase in process planning is to select the machines, such as milling machines, drilling machines or FMS's that will be used to produce the part. Similarly, in forming sequence design, the designer selects the press, such as horizontal transfer presses vs. vertical presses, to forge the part. In forging, the options of the designer are limited since he usually must use the presses he has on hand.

The third phase above, sequencing the operations for metal cutting is to put operations selected into a definite order to guarantee certain precedence relationships which are determined by tolerance considerations and technological considerations. Similarly, in forming sequence design, cold forging operations are ordered one after the other. One difference is that, in machining, the operations ordered may be performed in different machines but in forming sequence design they are done generally on the same machine in different stations (section 2.3.1).

The fourth phase above, grouping of operations in machining, is done respecting the order of precedence and taking into account relative positional tolerances (associated surfaces). By grouping, several operations may be performed on the same machine without reclamping the
workpiece. Similarly, in forming sequence design, principal cold forming operations are grouped together and performed in the same station. In the first station after cutoff in Figure 1, forward extrusion and backward extrusion operations are combined. The main purpose for combining operations in cold forming is to reduce the number of stations required.

Figure 1. An Example Forming Sequence for Cold Forging of a Gear Blank [3].

As explained above, the task of forming sequence design is similar to process planning for machining parts. Process planning is a suitable domain for application of KBSs and there are numerous studies [67-75] applying AI methodologies to process planning for the metal cutting industry. There are also some studies applying AI methodologies for process planning in sheet metal forming [76,77].
Forming sequence design can be classified as a process planning task, as described above, but it can also be classified as a "design" task. The manufacturing engineer, considering several technical and economical factors, designs the preforming shapes before the finishing blow. In process planning for machining, the initial and final geometry are defined; starting from the initial geometry, several operations are performed to reach the final shape. Each operation removes more material and the shape looks more and more like the finished geometry. Since material is being removed, the size of the workpiece can only get smaller. However, in forming sequence design there may be an intermediate stage where the length of the workpiece is larger than the final shape. The number of possible preform shapes for cold forming are infinitely large. In cold forging, the manufacturing engineer "designs" the preforming shapes. An example of this is the design of taper upsets where there are infinitely many different taper shapes.

In cold forming, the designer establishes the forming sequence and designs the dies. There is a substantial manipulation of geometric information in this process as in most of engineering design tasks. Whereas in machining, it is mostly ordering cutting processes, selecting machines, grouping operations, etc. and it can be classified as a planning task.

Therefore, forming sequence design can be classified both as a "planning" task and a "design" task. The requirement and the key problems of planning and design tasks described in section 3.3 apply to forming sequence design for cold forging.
4.2 IMPORTANT CHARACTERISTICS OF THE TASK

In the earlier stages of this study, cold/warm forging of at the axisymmetric parts was studied in detail. The literature on forming sequence design was reviewed carefully and the important ones [2,3,5, 9-19] for this project are studied in detail. Several forging companies were visited and several die designers were interviewed. The main sources of the forming sequence design knowledge were National Machinery Co. of Tiffin, Ohio, the Metalworking Section of Battelle Columbus Division and the literature on forming sequence design mentioned above. The author has attended the seminar on "Tool Design and Part Shape Development for Multi-Die Cold Forming" at the National Machinery Co., Tiffin, Ohio, on April 23-24, 1985.

The initial study of the domain was necessary to identify the important characteristics of this task. The knowledge based system was structured according to these properties of the domain. The key features of the forming sequence design task are:
- decomposable into subtasks,
- importance of classification of parts,
- importance of geometry manipulation, and
- non-unique acceptable solutions.

Careful study of the preform design for cold forging shows that the process can be divided into 7 subtasks. These subtasks are quite independent from each other and are explained in section 4.3.2.

Another important characteristic of forming sequence design is the classification of cold formed parts. Most of the cold form products
are axisymmetric. Axisymmetric parts can be classified further, such as "solid parts", "parts with a cavity on one end" and "hollow parts". The designers use such rough classifications to store previous examples of forming sequence drawings. Similarly, rules and guidelines can be grouped for similar parts in the same category as described in section 4.3.3.

The forming sequence design task requires manipulation of part geometries. The designer starts with the finished part geometry and goes backwards to find the forming sequence design as explained in section 2.3.1. He has a picture of the part in his mind and he keeps modifying this picture to find the preforming steps. He sometimes draws sketches of these figures [8]. Most of the rules and guidelines for forming sequence design are related to the geometry of the part. This suggests that a powerful geometry representation scheme is crucial for the success of the KBS to be developed. A 2-D and a 3-D internal representational schemes are suggested as described in section 4.3.4.

Finally, different designers may have different approaches and it is very probable that there is more than one acceptable forming sequences for a given part. Even the same designer may establish two different forming sequences for the same part at different times due to the changing constraints and requirements. Therefore any system developed for this purpose should be flexible and not insist on a unique solution. The user should be able to override some of the decisions of the system. For example, the billet dimensions can be selected by the program, but the user, for some reason (maybe due to already existing stock), may want to use different billet dimensions. The program should
be flexible enough to accept the new billet size and continue with the design process. Similarly, the user may want more (or less) taper upsetting stations than recommended by the program. So he should be able to override the decision of the program and the program should continue to design the taper dies (stations) as many as the user wants.

This property is not unique to this domain but holds for most design tasks. When a design task is automated by a program, the program may not consider all the "exceptions", and it needs to be flexible enough to let the user override its decisions at some points. A practical system should be equipped with all the interactive drafting tools so the user may design as he wishes. It should also advise the user on the solutions for sub-problems during the design process. Otherwise, completely automatic systems, without any user interaction, are not very practical in the real world environment. So, on one extreme is a computer aided drafting system with no specific support on forming sequence design and on the other extreme a totally automatic program which produces the forming sequence design without any interaction with the user. This study was directed in such a way that the program automates the process of the forming sequence design but it also gives the user the flexibility to override the decisions of the program at some crucial points.
4.3 FUNDAMENTALS OF THE SYSTEM

4.3.1 Production Systems Approach

A production system (or "rule-based system") was developed for the process of establishing forming sequences for cold forging. Production systems were described in section 3.2.3. The system was implemented using the Prolog language and was named FORMEX for "FORMing EXPert".

The process of forming sequence design can be decomposed in steps which we call subtasks (or context), such as choosing the billet dimensions and finding the upsetting sequence. Each subtask produces several decisions. When all the decisions in a subtask are made, higher level control moves to the next subtask.

The structure of the FORMEX system is shown in Figure 9. The inputs to the system are the final part geometry, the machine type and the material type. The output of the system is the drawing of the forming sequence (Figure 9) as well as the operations to be performed in each stage with the relevant dimensions. Machine and material data files are consulted to get the specifications for the machine and material type given in the input. Details of inputs, outputs and the machine and material files are explained in Chapter 5.

An important aspect of the system is the separation of specific forming rules files from the FORMEX Shell. The FORMEX Shell is composed of rules for basic functions of the system, such as input/output (including graphics for plotting), recognizing part geometry, consulting
appropriate specific forming rules files and the top level control. As mentioned in the previous section, an important characteristic of forming sequence design is that the approach is different for different classes of parts. Cold formed parts can be classified (Figure 9) as solid parts with decreasing diameters on one end, solid parts with decreasing diameters on both ends, hollow parts with a cylindrical cavity and different diameters outside, etc. This classification is further explained in section 4.3.3.

Given the input part geometry FORMEX Shell recognizes the shape and decides in which shape group it belongs. Next the specific forming rule file for that shape group is consulted. "Consulting", in Prolog terms, is reading the rules from a file and placing them in memory. So, at any given time, only forming rules related to that group of parts are in the knowledge-base. This has two main advantages: (1) since the number of clauses the interpreter has to search through is decreased, the execution time is decreased, and (2) since the number of clauses in the knowledge-base are decreased, the possibility of having conflicting rules has also decreased.

As described in section 3.2.3, a conflict may occur in a KBS if more than one rule is triggered at a time. If this ever happens, there has to be a mechanism to choose one of the rules to fire among the triggered rules. This is called conflict-resolution. Several conflict-resolution schemes are explained in section 3.2.3. Two of these schemes are used in FORMEX. One of them is context limiting. This is to
Figure 9. Structure of the FORMEX System.

Solid parts with decreasing diameters on one end

Solid parts with decreasing diameters on both ends

Hollow parts with a cylindrical cavity and different diameters outside
separate rules into groups (contexts) and let only some of them be active at a given time. This decreases the possibility of conflicts since the number of active rules is reduced. The context limiting is achieved as described above by only consulting the forming rules related to a part family. For example if the input part geometry is a solid part, only the rules for solid parts are consulted. These do not contain any rules for backward extrusion process since backward extrusion is not applicable to solid parts.

The other conflict resolution scheme used is called rule ordering. The rules are arranged in a list with decreasing priorities. The triggering rule appearing earliest in the list fires and all the others are ignored. This is very suitable for Prolog implementation because the Prolog control scheme works in a similar way. It goes top-down and fires the first rule that is triggered. So the only condition the user has to satisfy is to order the rules in each specific forming rules file in a priority list.

FORMEX is a forward-chaining (or data-driven) system. It works on the given data and tries to deduce new facts from the known facts. For example, given the final shape of the part and the billet diameter, it finds the diameters to be forward extruded. Once diameters to be forward extruded are found, it uses forward extrusion rules to decide in which order the extrusions should take place.
4.3.2 Decomposition of the Task

As mentioned in section 4.2, an important characteristic of the process of forming sequence design is that it can be divided into a sequence of subtasks. In the initial phases of this study, cold forming was studied in detail and it was concluded that the FORMEX system can be structured as seven top level subtasks which are made of subtasks themselves. The seven top level subtasks are:

Subtask 1: Check for formability of a given part.
Subtask 2: Recognize (classify) part geometry.
Subtask 3: Convert input part geometry to an internal geometry representation.
Subtask 4: Choose billet diameter and cutoff length.
Subtask 5: Find forward extrusion sequence.
Subtask 6: Find upsetting sequence.
Subtask 7: Combine and sequence upsetting and forward extrusion operations.

This sequence of subtasks corresponds to the steps the die designer takes when he establishes forming sequences. The first task is to check whether the given part can be cold formed using the given material type and the machine type. Cold formabilities of different materials are different as explained in Chapter 2. Parts that can be cold formed from some materials may not be cold formed from others. Some parts cannot be cold formed at all, such as; lumpy shapes, eccentric shapes, and asymmetric shapes. Other geometric properties which make cold forming impossible are: sharp edges, deep (or too steep
or too narrow) undercuts and thin (or too deep) borings. The die designer first checks for these properties and decides whether this part can be cold formed or not and modifies the part geometry if required (Figure 5). Similarly the first subtask of the system should be to check the cold formability of the input part geometry. A separate KBS can be developed just to perform this task. Given a final part shape, this system can modify the geometry so that it is formable by cold forging (see section 2.3.1.1). Then the output of this system, which is the cold formed part geometry, can be an input to the FORMEX system.

The other top level subtasks (Subtask 2 through Subtask 7) are explained in the next sections. It is important to notice that subtasks are quite independent of each other and they do not overlap. The control moves from Subtask 1 to Subtask 7 in sequential order. The decisions taken in a top level subtask can only affect the subtasks below it, they do not affect the decision taken in the previous subtasks. For example the decisions about the forward extrusion sequence (Subtask 5) will affect the decisions in Subtask 7 but they cannot affect the decisions already taken in Subtasks 1 through 4.

At the end of a session (after Subtask 7) the user may want to go back and change the billet dimensions and try again. In this case the control goes back to Subtask 4 and forming sequence is redesigned. This is the only backward looping allowed in the design session for a given part. In this case Subtasks 4 through 7 are active one after the other and all the previous decisions of these subtasks are ignored. So again, even this backward looping, the decisions of a subtasks can only affect the decisions to be taken by the subtasks that follow it but it cannot change the decisions of the subtasks before it.
Each of the seven top level subtasks are made of subtasks themselves. At this level, subtasks of the top level subtasks, do not follow the rule described above. That is, they may have backward loops. For example in Subtask 5, when the forward extrusion sequence is being found, there are several backward loops among subtasks due to the iterative method for finding the forward extrusion sequence. This iterative method is explained in section 4.3.6.

4.3.3 Classification of Parts - State of the Art

Classification of parts into part families is widely used in production planning for machining parts. Similarly parts have been classified into groups in some studies for forming sequence design [15,17,78]. Figure 10 shows a classification of round parts for cold forging often used in German literature [15-17].

A detailed part classification scheme for forming sequence design for upsetting is given by Gokler et al. [10,78] This classification scheme is used to create a handbook and a computer program to develop forming sequences for upsetting. In this approach parts are classified by the basic shape features and then rated according to the material used, so that the relative difficulty of the various shape/material combination for forming is found. The reason behind this is that the major factors which influence forming difficulty in a particular case is the complexity of the part and the material used. These two factors are strongly interrelated since a shape that is relatively easy to forge in one material may be difficult or impossible in another.
| CLASS 3 | Longitudinal shape, solid body | 13 | Basic shape with stepped shaft |
| Top surface | Without auxiliary shape | With cavity (hollow body) | With boss | With cavity and boss |
| Shell surface | Single | Double | Single | Double | Single | Double |
| 13 | Basic shape with stepped shaft | 23.3 | 23.4 | 23.5 | 23.6 | 23.7 | 23.8 |

Figure 10. An Example of Classification of Parts for Cold Forming. [4]
<table>
<thead>
<tr>
<th>CLASS 4</th>
<th>Top surface shell without auxiliary shape Single</th>
<th>Double</th>
<th>With cavity (hollow body) Single</th>
<th>Double</th>
<th>With boss Single</th>
<th>Double</th>
<th>With cavity and boss Single</th>
<th>Double</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Basic shape with straight inner and outer surfaces</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.2</td>
<td>Basic shape stepped outer and straight inner surfaces</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.3</td>
<td>Basic shape stepped inner and straight outer surfaces</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.4</td>
<td>Basic shape outer and inner stepped surfaces</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.5</td>
<td>Basic shape with through bore</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.6</td>
<td>Basic shape with curved, profiled, or conical shell lines</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 10. An Example of Classification of Parts for Cold Forming. [4] (continued)
material. They suggest a detailed classification scheme which first divides the parts in three main groups: compact parts (with similar overall dimensions in each orthogonal direction), flat parts (disc, etc.) and long parts [78]. These main groups are further divided into different groups. Compact parts are further grouped relative to the position and orientation of projections and depressions since these factors determine the parting line requirements and the necessity for multiple action dies, etc. Flat parts are further grouped by the presence of losses, thin ribs and through holes which will determine the forging difficulty. Long parts are further divided into groups by the orientation of supplementary futures such as asymmetric projections, holes or eyes, together with their relationship to the plane of bending [78]. The classification continues this way into several other levels of detail.

4.3.4 Classification of Parts - Present Approach

The major problem of using a very detailed classification scheme to find forming sequences is that no matter how detailed the classification is, it is not detailed enough to give the exact forming sequence of a given part. Two very similar parts may require completely different forming sequences even though they may seem to fall into the same group. An example to this is shown in Figure 8. Another disadvantage of using such detailed schemes is that when a new part shape is to be added to the existing system, the rigid classification scheme may be too difficult to charge or modify.
Our approach does not depend on a detailed classification of parts to find the forming sequence design. Instead, the detailed knowledge about forming sequence design is represented in the production rules. The rules may be as specific as necessary to separate similar part geometries from each other. New rules may be added and the old ones can be changed easier than changing the whole classification scheme.

Even though our system does not use a detailed classification scheme, it still takes advantage of classification of parts in the initial stages of design. Rules for finding forming sequences for "solid parts" are quite different than rules for finding forming sequences for "hollow parts". A very broad grouping of parts can simplify the process of forming sequence design and save time. Because of this, the system is structured as in Figure 9, where rules specific to some broad part classes are grouped together. These groups are: solid parts with decreasing diameters on one end, solid parts with decreasing diameters on both ends, hollow parts with a cylindrical cavity and different diameters outside, etc. (Figure 9). Each group differs from the others in some basic way. For example, the first two groups are solid parts and the third one is hollow parts. Therefore, the first two groups need only upsetting and forward extrusion operations, but the third group will also need backward extrusion and piercing operations. Currently FORMEX has rules and guidelines for solid parts only.

The second top level subtask of the system is to "recognize" the part shape and "consult" the corresponding forming rules. If the input is a solid part with decreasing diameters on one end, only rules
for that broad group of parts will be brought in the knowledge base. Advantages of this approach as mentioned in section 4.3.1 are that (1) the execution time is decreased and (2) the possibility of having conflicting rules is also decreased since there are fewer active rules in the knowledge base at a given time. Another advantage of structuring the system as shown in Figure 9 is that different companies working with different part families can develop their own specific forming rules files and use them with the FORMEX Shell. Our experience with different companies shows that they work with different types of parts and each part has some "tricks" or "exceptions". This structure allows them to include the tricks they have in their specific forming rules files (Figure 9).

The disadvantage of dividing rules into some main groups like this is that there will be some repetition among the rules in these groups. That is, some rules may be repeated in each of the groups and if all the groups are gathered as one file, these repetitions would be eliminated. The total size of the different specific forming rules files is greater than the size of a single file if they were grouped all together. This was not a major concern in our study and it is believed that the above mentioned advantages of the structure of the system overcomes this disadvantage.

4.3.5 Internal Geometry Representation

Forming sequence design for cold forging involves a lot of geometry manipulation. The die designer looks at the drawings of the
final part geometry, represents it in his mind in 3-D and designs the forming sequence backwards from this image. Humans have a very powerful way of understanding geometries. The die designer uses this all through the forming sequence design. The example in section 2.3.1 (Figures 5 and 6) illustrates how this design task evolves around the final part shape.

When a designer looks at the final part drawings, he constructs an image of this part in his mind. Similarly, when the design system reads the input part geometry, it must represent this in a "meaningful" way so that this information will be easily used by the inference rules of the system when required. So, initially, the input data about the final part geometry must be converted to an internal geometry representation (Figure 11). Forming sequence is established on this internal geometry representation. Finally the forming sequence is converted to the output representation (plots, dimensions) as shown in Figure 11.

As mentioned in Chapter 2 most of the parts produced by cold forming are round parts, and only round parts are considered in this study. Round parts can be represented fully by a cross section along their axis. Figure 12 shows the construction of 2-D and 3-D internal representations from the input geometry data on an example. Since the cross section of the part is adequate to represent the part, the part geometry data is input to the system as the (x,y) coordinate pairs of this cross section. In the example drawing in Figure 12, only half of the cross section on the positive x and positive y quadrant of
Figure 11. Block Diagram Showing Conversions to and from Internal Geometry Representation.
Figure 12. Construction of 2-D and 3-D Internal Geometry Representations from Input Coordinates.
the cartesian plane is shown. This half of the cross section is enough to represent it since it is symmetric around the y axis. This example is a solid part and the cartesian coordinates of the six points, (3,6), (3,5) (2,5), (2,4), (1,3) and (1,0) are sufficient to represent the part given that we know the convention used. For parts with cavities, the coordinates of the corners of the cavity are also given.

For the current version of FORMEX, the coordinate pairs for the final part shape are read from an input file as described in Chapter 5. The input file is created by the user. Instead of creating the input file using the editor, FORMEX should be interfaced with the CAD system of the company. One method of achieving this is to develop a preprocessor that can extract the coordinate pair data from IGES files and a postprocessor program that can create IGES files for intermediate preform shapes from the output of FORMEX. Another method is to interface FORMEX directly with a specific geometry modelling system. Currently, it is being considered to interface FORMEX with the ANVIL 4000 system.

The cross section of the example part in Figure 12 has sharp corners. There may be radii associated with these corners in the drawings of the final part geometry. These radii are not entered to FORMEX since, generally, they are not crucial for the forming sequence design. Currently, it is the user's task to create a suitable file, without radii, given the final part drawings. As the system is interfaced to a CAD system, the preprocessor mentioned above can handle this task and produce the FORMEX input without radii.
Given the \((x,y)\) coordinate pairs of the cross section of the part as input to the system, the Subtask 3, mentioned in section 4.3.2, constructs a 2-D and a 3-D internal representation of the part geometry (See Appendix A). This is analogous to the designer looking at a blueprint and constructing the image of the part in his mind. The system has knowledge about the possible part shapes (solid, hollow, etc.) and it adds additional information to the input \((x,y)\) coordinates while constructing the internal representations.

In the 2-D representation scheme, the cross section of the part is divided into "zones" which are either rectangles or parallelograms. In Figure 12, the example cross section is divided into four zones and the zones are named from 1 to 4. Each zone is made of four edges. In Figure 12, zone 1 is made of edge 1, edge 2, edge 3, and edge 4. Every edge is made of two vertices which are the endpoints of that edge. In Figure 12, edge 1 is made of vertex 1 and vertex 2. Finally vertices are made of two cartesian coordinates. In Figure 12, the vertex 1 is made of \(x_1\) and \(y_1\). Due to space limitations only some branches of the tree diagrams are shown in Figure 12. Zones are numbered top down as shown by the example in Figure 12.

Input coordinate pairs of the cross section are also used to construct a 3-D representation of the part. In this case the part is divided into "shape elements" instead of zones. Shape elements are 3-D objects and they can be either cylinders or conical shapes. The example in Figure 12 is made of four shape elements. Shape elements are also numbered top down as zones. All the shape elements of the example are cylinders except the shape element 3 which is a conical shape element.
Shape elements are described with their dimensions as shown in Figure 12. They all have three dimensions, the height and two diameters. If the two diameters are the same, then, that is a cylindrical shape element. If the two diameters are different, it is a conical shape element. The 3-D internal representation scheme is very similar to the constructive geometry representation in some geometry modelling systems. The 3-D internal geometry representation of FORMEX is a small constructive geometry scheme which has only a couple of primitive elements and can represent only axisymmetric shapes.

In the bottom half of Figure 12, the 2-D and 3-D internal representations are shown as Prolog facts. There are part, zone, edge, vertex and shape_dimension predicates. The first argument of each fact is the name of the object that the fact is about, and the rest of the arguments are the attributes of that object. For example in the fact

\[ \text{edge}(\text{edgel}, \text{vertex1}, \text{vertex2}). \]

edge is the predicate, edgel is the name of the specific edge we are dealing with, vertex1 and vertex2 are the names of the two vertices at the ends of this edge, which define this edge. Similarly in

\[ \text{shape_dimensions}(\text{shape3}, 1, 4, 2). \]

the predicate is shape_dimensions, the object is shape3 (shape number 3), the height of the shape is 1, the upper diameter is 4 and the lower diameter of the shape is 2. Since the upper and lower diameter of this shape are different, it is a conical shape (see Figure 12).

This method of dividing the part geometry into zones or shape elements with different diameters is similar to what the designer does at the initial stages of forming sequence design [8]. He divides the
part into shape elements and calculates their volumes and adds them up to find the part volume. Similar steps are taken in the FORMEX system at the initial stages of design (Appendix A).

Forming sequence design is finding the shape elements of the part after each blow (or station). When the system establishes an operation, say forward extrusion, new geometries of the shape elements are computed and stored with the blow number. So, the above mentioned example represents only the final part geometry. Similar information is produced about the intermediate part shapes during the design process.

The above mentioned 2-D and 3-D geometry representation schemes are quite suitable and adequate for solid parts. But they are not necessarily suitable for parts with cavities or hollow parts. For forming hollow parts, forward and backward extrusion is often performed simultaneously in a single station. The material volume deforms in a more complex way and the boundaries of the shape elements mentioned above may not exist anymore. Different types of properties, such as depth of the extruded hole over the diameter of the hole or bottom thickness over wall thickness ratios, become important in backward extrusion. Therefore, a different internal geometry representation scheme which takes backward extrusion into account is more suitable for hollow parts.

4.3.6 Choosing the Billet Dimensions

As mentioned in section 4.3.2, the fourth top level subtask is to choose the dimensions of the billet. Finding the most suitable
billet size for cold forming a part is not a trivial task. There are technical and economical considerations involved. Wires and rods do not come at every diameter, so the availability of the stock is a factor.

In practice, the billet diameter for solid parts is selected to be between the largest diameter and the smallest diameter of the part. If it is closer to the largest diameter, there is less upsetting required but higher reduction ratios are necessary for extruding the smaller diameters. If the billet diameter is smaller, then the reduction ratios are smaller, but the upsetting ratios become higher. Therefore, for a given solid part, there is an optimum billet diameter which minimizes the total deformation required for the upsettings and extrusions.

A simple algorithm is developed for choosing the billet diameter for solid parts. The steps of this algorithm are as follows:

1. Find the zones of the part with length larger than diameter.
2. From these zones pick the one with the largest diameter, $d_m$.
3. Check the table of available wire sizes for that specific material, and pick the wire or rod diameter equal or just larger than $d_m$.
4. Find the maximum number of extrusions ($n_e$) for extruding the smallest diameter of the part from this billet diameter.
5. If $n_e \leq 3$, go to Step 6. If $n_e > 3$, pick the next smaller wire diameter from the table as the billet diameter and go to Step 4.

6. Prompt the user the choice of the billet diameter and ask if he wants to change it. If the answer is "yes", then have the user input the billet diameter.

After choosing the billet diameter, cutoff length is to be calculated. The volume of the part is found by adding the volumes of the shape elements described in the previous section. If there is no trimming or piercing, the billet cutoff length is found simply by dividing this volume by the area of the base of the billet.

This is a simple algorithm to find the billet dimensions for cold forming solid parts. To find the billet dimensions for hollow parts is a little more difficult. But since the number of possible billet diameters is not very high due to commercial availability, some set of rules developed from experience can be used to find the billet diameter for hollow parts.

4.3.7 Finding the Extrusion Sequence

The fifth top level subtask in cold forming sequence design is finding the forward extrusion sequence (section 4.3.2). Forward extrusion is one of the principal cold forging operations and was discussed in detail in section 2.2.2.

The solid part geometry is represented internally as a series of zones (or shape elements) as discussed in section 4.3.5. Once the
billet diameter is chosen, the zones that are to be extruded are selected. The zones that are to be forward extruded are simply the zones whose diameters are smaller than the billet diameter. So, given the zones to be forward extruded, the task of this fifth top level subtask is to (1) decide in which order these extrusion will take place, (2) decide whether each zone is to be extruded in one blow or in several blows, (3) decide if zones are open extruded or trapped extruded and (4) decide if double open forward extrusion is possible for any zones (See Appendix A).

There are several rules and guidelines for forward extrusion. Figure 13 shows some of the rules for open die extrusion. A major rule

**Figure 13. Guidelines for Open Extrusion [9].**
is that maximum reduction in area (see Figure 13) should not exceed 30-35 percent for mild steel. This rule is very much dependent on the material type. For different types of the materials, the maximum reduction in area for open extrusion (as well as for trapped extrusion and the maximum length of the unsupported billet length in upsetting) are represented as Prolog facts in the knowledge base. Given the material type, the proper limit for maximum reduction in open extrusion is known from these facts and any attempt to open extrude a zone is checked using the rule for the limit.

After deciding for a specific extrusion operation, the load required for this operation can be calculated for double-checking this decision. Formulas for calculating load in forward rod extrusion and backward cup extrusion are compared and summarized by Altan, T., et al [1] (Chapter 15). Even though the current version of FORMEX does not include any load calculations, work is underway to add load calculations to the system in order to check the load requirement for the recommended operations. To illustrate the parametric formulas for load calculations in cold forging, an example is given in section 4.3.10.

There are similar rules for trapped extrusion as open extrusion. Figure 14 shows some of these rules and guidelines. In trapped extrusion, since the billet is completely enclosed by the die, much higher reductions in area are possible. Again, the maximum reduction in area is very dependent on the material type. Similar to the open extrusion case explained above, the maximum reduction in area for the given material is obtained from the knowledge-base before applying this rule.
MAX. REDUCTION IN AREA FOR TRAPPED EXTRUSION - 70-75%

TO CALCULATE REDUCTION IN AREA:
\[
\text{AREAD} \times 100 = \% \text{R.A.}
\]
\[
\text{AREAD} = \text{AREA D} - \text{AREA d (NORMAL)}
\]
\[
d = \text{EXTRUDE DIAMETER}
\]
\[
H = \text{BLANK LENGTH BEFORE EXTRUSION}
\]

CONCENTRICITY OF D, d AND d₁ ARE CRITICAL.
FINISH ON D, A, d, d₁ AND E MUST BE SMOOTH AND FREE OF TOOL MARKS.

<table>
<thead>
<tr>
<th>ITEM</th>
<th>FOR MOST COMMON EXTRUSION SIZES</th>
<th>FOR UNUSUAL EXTRUSION SIZES</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>BLANK DIA. (+.010) (+.012)</td>
<td>1.03 x BLK. DIA. UP TO 3/8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.022 x BLK. DIA. UP TO 7/8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.015 x BLK. DIA. UP TO 2&quot;</td>
</tr>
<tr>
<td>d₁</td>
<td>d + .004 APPROX.</td>
<td>1.006 x d</td>
</tr>
<tr>
<td>B</td>
<td>1/32 to 1/16</td>
<td>1/64 to 1/8</td>
</tr>
<tr>
<td>C</td>
<td>3/16 to 3/8</td>
<td>TO SUIT &quot;E&quot; BELOW</td>
</tr>
<tr>
<td>E</td>
<td>TO SUIT &quot;C&quot; ABOVE</td>
<td>1/2° MAX.</td>
</tr>
</tbody>
</table>

ABOVE DATA MAY NEED TO BE ALTERED TO SUIT FOLLOWING FACTORS:
1. BLANK MATERIAL AND COATING
2. LUBRICATION
3. DIE FINISH
4. END SHAPE ON EXTRUDED TIP
5. FINAL SHAPE OF SHOULDER ON BLANK
6. FINAL PART TOLERANCE
7. SHOP PRACTICE

Figure 14. Guidelines for Trapped Extrusion [9].

Another important rule is that one should not extrude over the same diameter more than 3-4 times. To achieve minimum number of extrusions over the same zone, the smallest diameter zones should be extruded first. But sometimes it is not possible to extrude the smallest diameter first because the reduction in area might be too high and violate the rule discussed above. In this case the rule on the maximum reduction in area is more powerful than the "extrude the smallest diameter first" rule. So the extrusion of the smallest diameter is not possible and the next smallest diameter is the next candidate to be extruded.
In order to have an idea about how these rules are represented as Horn clauses in Prolog, let's look at an example. Following is a rule from FORMEX for single-reduction forward rod extrusion:

```
rule(extrusion,20,single_open) :-
    pick_smallest_zone,
    area_before_extrusion(Area0),
    area_after_extrusion(Areal), !,
    material(M,N),
    material(M,N,R,_),
    (Area0 - Areal) * 100 / Area0 < R,
    current_zone(Z1),
    assert(single_open_extrusion(Z1)).
```

This is a Prolog rule (see section 3.2.4.4) with 7 conditions in its body. This rule will be satisfied only if all these seven conditions are satisfied. The first condition "pick_smallest_zone" is to find the smallest diameter zone which is not yet extruded. We want to extrude the smallest diameter first because we do not want to extrude over the same diameter more than 3-4 times as mentioned above. The next two conditions in the above rule

```
area_before_extrusion(Area0), and
area_after_extrusion(Areal)
```

are for calculating the areas of the zone (that was picked) before and after extrusion. Area0 is the area of that zone before extrusion and Areal is its area after extrusion. The next two conditions

```
material(M,N), and
material(M,N,R,_,_)
```

are for finding the limit on maximum reduction in area for open forward extrusion for the specific material. R will be instantiated to the maximum reduction in area for the material class M and material type N. The next condition
\[ \frac{(\text{AreaO} - \text{Area}) \times 100}{\text{AreaO}} < R \]

checks if the required percent reduction in area to extrude this zone is less than the limit (R) for this material. If this condition is satisfied, the last 2 conditions are for adding the fact

\text{single_open_extrusion(Z1)}

into the knowledge base. This fact can be read as "Zone number Z1 is open extruded in a single reduction die".

In cold forming it is preferred to open extrude instead of trapped extrude if possible. This rule is also implemented in the system such that it tries to open extrude first and trapped extrusion is assigned if the reduction in area is too high for open extrusion.

Sometimes double reduction open forward extrusion in one pass (Figure 15) is used in cold forming. This process reduces the need for two stations for two extrusions to a single station. It may be used when the number of forming stations to forge a part is more than the stations available on the machine. Figure 15 shows an example where 0.670 inch and 0.625 inch diameters are open extruded from a 0.750 inch billet in one blow. There are two main restrictions to the use of double reduction open forward extrusion in one pass. One condition is that the distance between the two extrusions must be at least as much as the initial diameter (Figure 15). The second condition is that the maximum reduction in area should not exceed 30 percent for mild steels (Figure 15). Similar rules for double open forward extrusion in one pass exist in the system developed in this study, and this method is used wherever it is suitable.
Distance between extrusions should be 1 x initial diameter or more.

.750 \( \phi \) to .825 \( \phi \) = 30\% RA = approx. max. reduction

Figure 15. Double Reduction Open Forward Extrusion in One Pass [8].

Some of the rules for forward extrusion in the system were discussed above to give a general view of how the forward extrusion steps are decided. These and other rules for extrusion are used to establish the forward extrusion sequence to produce the part (See Appendix A). After the forward extrusion sequence is set, upsetting stations are designed as described in the next section.
4.3.8 Finding Upsetting Sequence

The sixth top level subtask in forming sequence design is to find the upsetting sequence (section 4.3.2). Upsetting is one of the principal cold forging operations and was described in section 2.2.1.

Solid part geometry was represented internally as a series of zones (or shape elements) as discussed in section 4.3.5. First the billet diameter is chosen and then the forward extrusion sequence is found. The zones that are to be upset are simply the zones whose diameters are larger than the billet diameter. So, given the zones that are to be upset, this sixth top level subtask finds the sequence of upsettings.

An important parameter in upsetting is the length of the unsupported stock. Length of the unsupported stock is defined in Figure 16. In upsetting one blow can upset about 2.5 diameters of

![Figure 16. Definition of Unsupported Stock Length in Upsetting [9]](image-url)
unsupported stock before losing control of the stock by buckling. Two blows can control about twice this length, and the amount of stock continues to increase with each successive upsetting blow. Again material type is a major factor and maximum possible unsupported stock length is different for different materials. As in the case of extrusion, these limits on the length of the unsupported stock for different materials are stored in the knowledge-base as Prolog facts.

If the length of the unsupported stock is larger than the limit for upsetting a part in one blow, often an additional station with taper dies is used. Figure 17 shows an example of taper upsetting where the length of the unsupported stock is 3.25 times the diameter of the stock. In this case only one additional blow is required. To design the taper for this case, the small diameter of the taper is taken as the billet diameter (B), the large diameter of the taper (C) is 1.25 times the billet diameter (B), and the length of the taper (F) is calculated from the volume constancy (Figure 17). If the taper is designed as shown for the case where unsupported length of stock is 3.25 times the diameter of the stock, the unsupported billet length for the last blow becomes less that 2.25 times the mean taper diameter (Figure 17) and the above rule still holds. Sometimes, in multi-die cold forming, designers try to use not more than two dies for upsetting, in order to retain several other dies for additional forming. So, they use the design method described above (Figure 17) for upsetting stock length equal to 3.25 times the billet diameter. Similar design methods for computing the taper die dimensions for upsetting stock lengths equal to 4 and 4.5 times the billet diameter are given in the literature [8].
Volume finished upset = area B = length A
Length A = diameter B = number of diameters of stock
Diameter B × 1.25 = diameter C
(Area B + area C) / 2 = mean area D
Volume section AB = volume section EB = volume section BCF
Volume section BCF = mean area D = length F

CONE UPSET
3-1/4 diameters

Figure 17. Design of Taper Die for Upsetting Stock Length Equal to 3.25 Times Billet Diameter [8].
So far upsetting up to two blows has been discussed. Sometimes it might be necessary to upset in more than two blows because the amount of volume to be upset might be too much. Another reason that many require more than two blows for upsetting is the number of upset zones. So far upsetting only single zones has been described, but there can be two or three zones next to each other to be upset. If the diameters of these zones are significantly different from each other, each zone might require a separate upsetting blow. Rules for designing taper dies for more than two upsetting blows can be found in the literature [10].

Generate and test method seems to be suitable for the upsetting sequence design subtask. First, a single blow is tried. If it violates any constraints (such as unsupported length is too long), then two upsetting blows are tried. The system finds the shape for the first blow (taper die) and the last blow and computes the "mean diameter" (Figure 17) and checks if it fulfills the constraints. If not, die shapes for upsetting with three blows are generated. This will have two initial taper upsettings and a final upsetting. Again the "mean diameter" for the two tapers are calculated and checked if they satisfy the constraints. If not, die shapes for four blows are generated. This has three taper upsettings and a final upsetting. Again the "mean diameters" for all the three tapers are calculated and checked if they satisfy the rules. If even four blows are not enough to upset the zones, the system displays a warning message and quits the search for an acceptable upsetting sequence since more than four upsetting blows are not normally practical. At that point, the user may ask the system to
go ahead and design an upsetting sequence for 1, 2, 3 or 4 upsetting blows as desired, even though it will not satisfy some rules. Similarly, even when the system can find an upsetting sequence, it will display the number of blows as "recommended" and the user has the flexibility to override this recommended number and enter the number of upsetting blows of his choice. The system will then design the upsetting sequence for that number of blows (See Appendix A). Examples of upsetting sequences are given in Chapter 5.

To have a feel for the upsetting rules in Prolog, let's look at an example. Following is a rule for deciding whether one blow is enough for upsetting. The rule is for the case of only 2 adjacent upsetting zones. Similar rules exists for up to 3 adjacent zones for upsetting in FORMEX. The rule is:

```
rule(upsetting,no_of_blows) :-
    u(1,U),
    material(M,N),
    material(M,N,_,_,_,U1),
    U =< U1,
    shape_dimensions(Z1,_,D1,_,),
    shape_dimensions(Z2,_,D2,_,),
    abs(D1-D2) / D1 < 0.50,
    assert(recommended_upsetting_blows(1)), !.
```

This is one of the rules that decides on the number of upsetting stations required. The body of the rule (see section 3.2.4.4) contains 8 conditions. The first condition

```
u(1,U)
```

is to get U, the unsupported billet length in terms of billet diameters. This value is calculated prior to this rule.
The next 2 conditions

material(M,N) and
material(M,N,_,_,_,U1)

are to get the limit on the unsupported billet length for this material. U1 is the limit for material class M, type N.

The next condition

U <= U1

checks whether the unsupported billet length is less than or equal to that limit. The 2 conditions that follow

shape_dimensions(Z1,_D1,_) and
shape_dimensions(Z2,_D2,)_

are simply to get the diameters (D1 and D2) for the zone to be upset (Z1 and Z2). The next condition

abs(D1-D2) / D1 < 0.50

checks whether the difference of these two diameters is small enough to have only one upsetting blow.

If this condition is also true, finally, the fact

recommended_upsetting_blow(1)

is added to the knowledge-base using the "assert" predicate.

After finding the upsetting sequence the next and the last task is to combine the forward extrusion and upsetting operations in single stations. This last top level subtask is described in the next section.
4.3.9 Combining and Sequencing Extrusion and Upsetting Operations

The seventh and last subtask in designing forming sequences is combining and sequencing the extrusion and upsetting operations (section 4.3.2). Extrusion operations were found in Subtask 5 and upsetting operations were found in Subtask 6. While deciding on these operations, extrusion and upsetting are assumed to be independent of each other. This assumption is fairly realistic for forward extrusion and upsetting but it will not hold for backward extrusion and upsetting.

The main reason for combining forward extrusion and upsetting operations is to reduce the number of stations required. Die designers use several rules and guidelines when they combine these operations in single stations. Sometimes upsetting operations are combined with forward extrusion operations at the last stations of the forming sequence. The main reason for this is that the fingers of the automatic transfer mechanisms can hold the part in balance easier if there are no heads formed on them. If the heads are formed in the initial stages the automatic transfer mechanism may not work. So the upsetting operations to form the heads are combined with the forward extrusion operations at the last stations (see the examples in Chapter 5).

On the other hand, sometimes it is preferred to combine the upsetting operations and the forward extrusion operations at the initial stages of forming. The main reason for this is that otherwise, after several extrusions, the workpiece may get too long before it is upset. These long intermediate workpieces may cause problems in die design and kickoff mechanisms. If, instead, the upsetting is combined with the
forward extrusion operations at the initial stages, due to upsetting, the length of the workpiece is shorter through the rest of the stations and this problem is eliminated. Therefore, according to the given situation, upsetting operations can be combined with the forward extrusion operations either in initial stages or in the last stages of the forming sequence.

If there are enough stations on the machine, there is no need to combine upsetting and forward extrusion operations, instead these operations can be sequenced one after the other. Then the question is whether upsetting should follow forward extrusion or vice versa. There are several points to be considered before deciding. Figure 18 shows an example where upsetting is performed in the first two stages and extrusion in the third. The reason for this sequence is the simplicity of die design. This sequence minimizes the number of dies with stepped inserts. On the other hand Figure 19 shows an example where open extrusion has to be performed before the two upsetting operations. The reason for this sequence is that the diameter right under the head is to be extruded. If the head is upset first, there would not be enough entrance guide to extrude the diameter right under it (Figure 19). These rules for combining and sequencing forward extrusion and upsetting operations are implemented as production rules in the system.

4.3.10 Load Estimations for Cold Forging

One of the constraints in forming sequence design is the maximum load the dies can withstand. The die designer must be sure that the
Simplified die design carries upset load

Number of dies with stepped inserts minimized

Upsetting head first would not allow entrance guide for forward extrusion

Figure 18. Upsetting Before Open Forward Extrusion [8].

Figure 19. Open Forward Extrusion Before Upsetting [8].
load required for a given forming operation does not exceed this limit. Higher loads reduce die life drastically. There is also a maximum force and energy that the press can deliver. Another constraint about the forming load is the requirement of even distribution of load on all the stations in multi-stage cold formers. All of these constraints on load make it an important parameter to estimate during forming sequence design.

Current version of FORMEX does not include the load calculations, but work is underway to estimate the load required at each station. After FORMEX decides for a forming operation, that decision will be double-checked by calculating the required load for that operation. If the load requirement is acceptable, FORMEX will continue to design the forming sequence as before; but if the load requirement is too high, FORMEX will change its decision to lower the load required. For example, FORMEX may decide to have an open forward extrusion at a station. The load required for this extrusion is estimated next. If the load is under the maximum load limit for the given conditions (dies, lubricant, machine, etc.), the forming sequence design continues for the next station. If the load is higher than the limit, FORMEX will change the decision and try to extrude another diameter with a smaller reduction ratio to reduce the load required.

Several empirical methods for calculating load for cold forming processes are compared and summerized by Altan, et al. [1] Generally, these empirical methods rely on direct measurements of load or pressure during actual experiments or trials. These trials and experiments are conducted over a wide range of process parameters. The
experimental data are then analyzed and summarized in the form of empirical equations taking into effect the most important process parameters. The empirical equations, if they have been arrived at properly, are one of the most reliable methods of obtaining forging loads over the applicable ranges. However, they are very expensive to obtain because they require extensive experimentation and measurements with actual tool set ups.

As an example, let's assume that we want to estimate the load required for an open forward extrusion operation. Following is an empirical formula for calculating force in open forward extrusion with reasonable accuracy [79]:

\[ F = A_0 \cdot \bar{\sigma}_A \left[ \bar{\varepsilon}_A \left( 1.01 + \frac{2 \cdot f}{\sin 2\alpha} \right) + 0.77 \cdot \tan \alpha \right] \] (1)

where

- \( F \) = Force required for open die extrusion,
- \( A_0 \) = Initial area of billet,
- \( \bar{\sigma}_A \) = Average flow stress corresponding to deformation strain,
- \( \bar{\varepsilon}_A \), where \( \bar{\varepsilon}_A = \ln \frac{A_0}{A_1} \)
- \( A_1 \) = Final area of billet,
- \( f \) = Friction factor,
- \( 2\alpha \) = Included cone angle of the die.

For materials at room temperature, generally, the strain rate and temperature effects can be neglected and the flow stress, \( \sigma \), can be defined as an equation:

\[ \sigma = K \cdot \varepsilon^n \]
where \( K \) and \( n \) are material constants and can be found from flow curves.

The average flow stress, \( \sigma_A \), for the deformation strain, \( \varepsilon_A \), can be computed as

\[
\bar{\sigma}_A = \frac{1}{\varepsilon_A} \int_{0}^{\varepsilon_A} K \cdot \varepsilon^n \, d\varepsilon
\]

\[
\bar{\sigma}_A = \frac{K \cdot \varepsilon_A^n}{1+n} \tag{2}
\]

Let's assume that we want to estimate load for the following forward extrusion example:

- **Material**: AISI 1045
- **Surface treatment**: phosphated + soap lubricant
- **Initial diameter**: \( d_0 = 11.0 \text{ mm} \)
- **Final diameter**: \( d_1 = 10.0 \text{ mm} \)
- **Friction factor**: \( f = 0.08 \) (assumed)
- **Cone angle of the die**: \( 2\alpha = 24 \text{ degree} \)

\[
A_0 = \frac{\pi}{4} \cdot d_0^2 = \frac{\pi}{4} \cdot 11^2 = 95.0 \text{ mm}^2
\]

\[
A_1 = \frac{\pi}{4} \cdot d_1^2 = \frac{\pi}{4} \cdot 10^2 = 78.5
\]

\[
\bar{\varepsilon}_A = \ln \frac{A_0}{A_1} = \ln \frac{95.0}{78.5} = \varepsilon_A = 0.19
\]
From the flow curves [79] K and n values are obtained and using equation (2) average flow stress, $\bar{\sigma}_A$, is calculated as:

$$\bar{\sigma}_A = \frac{921 \cdot (0.19)^{0.116}}{1 + 0.116}$$

$$= 680.6 \text{ N/mm}^2$$

Using equation (1):

$$F = 95.0 \cdot 680.6 \left[0.19 \left(1.01 + \frac{2 \cdot 0.08}{\sin 24}\right) + 0.77 \cdot \tan 12\right]$$

$$F = 27823 \text{ N}$$

This example shows how to use a simple parametric formula to calculate load for forward extrusion. Similar simple parametric formulas exist to compute load for other cold forging operations. [1] Currently, these parametric formulas are added to FORMEX for estimating load.
CHAPTER 5
IMPLEMENTATION AND RESULTS

A knowledge-based system for forming sequence design which implements the approach described in Chapter 4 is developed with the Prolog language running on a VAX-11 with the VMS operating system. The system is called FORMEX for FORMing EXPert. This chapter described FORMEX and illustrates its use.

5.1 INPUTS

Inputs to FORMEX can be classified in three groups: (1) part geometry, (2) material type, and (3) machine type. Figure 20 shows an example part drawing. In the current version of FORMEX, the input file is created by the user. As mentioned in Chapter 4, it is being considered to interface FORMEX with a CAD system and eliminate the step of preparing an input file using the editor. The input for this part is shown in Figure 21. The cross section of the part is represented with the cartesian coordinates in the input file as described in section 4.3.4. These coordinate pairs are given under the title "PART GEOMETRY" in Figure 21. Material type follows the part geometry coordinate pairs in the input file. In Figure 21, material type is given as carbon steel 1008. Material type is followed by the machine type and machine specifications. Machine specifications include the number of stations,
maximum wire diameter, maximum cutoff length, slide stroke length, maximum punch ejector stroke and maximum die ejector stroke (Figure 21).

After the input file is read, FORMEX consults another input file for the material specifications. A portion of this file is shown in Figure 22. Each line in this file is a Prolog "fact". All the facts have "material" as their predicate (Figure 22). Each fact has six arguments which are (Figure 22):
TITLE
PART # 38

PART DESCRIPTION
This part has upsetting in the middle.

PART GEOMETRY
18
0.800 11.500
0.800 8.800
1.000 8.800
1.000 8.200
1.200 8.200
1.200 7.400
1.000 7.400
1.000 6.400
0.700 6.400
0.700 4.000
0.600 4.000
0.600 3.000
0.500 3.000
0.500 1.800
0.450 1.800
0.450 1.000
0.400 1.000
0.400 0.000

MATERIAL DESCRIPTION
This part is to be made of coil wire.

MATERIAL SPECIFICATIONS
carbon_steel 1045

MACHINE DESCRIPTION
This is a 5-die coldformer.

MACHINE SPECIFICATIONS
machine name : M500
number of stages : 5
wire diameter (max) : 2.0
cutoff length (max) : 15.0
slide stroke length : 10.0
punch ejector stroke : 5.0
die ejector stroke : 10.0
speed (pieces/min) : 90

Figure 21. Input Data File for the Example Part in Figure 20.
FORMEX MATERIAL DATA FILE

This File Contains Information on Cold Formability of
1. Copper and Copper Alloys,
2. Aluminum,
3. Carbon Steels,
4. Alloy Steels,
5. Stainless and Heat-Resistant Steels.

Material ( copper, '102', best, 25, 60, 2.25 ) .
material( copper, '110', best, 25, 60, 2.25 ) .
material( copper, '114', best, 25, 60, 2.25 ) .
material( copper, '122', best, 25, 60, 2.25 ) .
material( copper, '220', best, 25, 60, 2.25 ) .
material( copper, '230', best, 25, 60, 2.25 ) .
material( copper, '240', best, 25, 60, 2.25 ) .
material( copper, '260', best, 25, 60, 2.25 ) .
material( copper, '425', best, 25, 60, 2.25 ) .
material( copper, '443', best, 25, 60, 2.25 ) .
material( copper, '502', best, 25, 60, 2.25 ) .
material( copper, '510', best, 25, 60, 2.25 ) .
material( copper, '521', best, 25, 60, 2.25 ) .
material( copper, '524', best, 25, 60, 2.25 ) .
material( copper, '651', best, 25, 60, 2.25 ) .
material( copper, '655', best, 25, 60, 2.25 ) .
material( copper, '687', best, 25, 60, 2.25 ) .
material( copper, '752', best, 25, 60, 2.25 ) .
material( copper, '762', best, 25, 60, 2.25 ) .

Material ( copper, '150', good, 25, 60, 2.25 ) .
material( copper, '162', good, 25, 60, 2.25 ) .
material( copper, '172', good, 25, 60, 2.25 ) .
material( copper, '182', good, 25, 60, 2.25 ) .
material( copper, '187', good, 25, 60, 2.25 ) .
material( copper, '274', good, 25, 60, 2.25 ) .
material( copper, '314', good, 25, 60, 2.25 ) .
material( copper, '330', good, 25, 60, 2.25 ) .
material( copper, '544', good, 25, 60, 2.25 ) .
material( copper, '697', good, 25, 60, 2.25 ) .
material( copper, '706', good, 25, 60, 2.25 ) .
material( copper, '710', good, 25, 60, 2.25 ) .
material( copper, '715', good, 25, 60, 2.25 ) .
material( copper, '770', good, 25, 60, 2.25 ) .

Figure 22. Portion of the FORMEX Material File.
1. material class,
2. material name,
3. cold formability,
4. maximum reduction in area for open extrusion,
5. maximum reduction in area for trapped extrusion, and
6. maximum unsupported billet length in diameters.

There are five material classes in this data file:
1. copper and copper alloys,
2. aluminum and aluminum alloys,
3. carbon steels,
4. alloy steels, and
5. stainless steels.

The total number of metals in all these five classes is 136.

Cold formability of these 136 metals are classified as poor, fair, good and best. This classification is used by the die designers in forming sequence design. [8] FORMEX recommends to the designer the metals which have better cold formability in that group of metals. For example, if the material specification in the input (Figure 21) is carbon steel 1045, FORMEX will display a warning saying that cold formability of 1045 is poor and display 3 alloy steels whose cold formability is "fair", 5 alloy steels whose cold formability is "good" and 5 alloy steels whose cold formability is "best". Then FORMEX will ask if the user wants to change the material. If the answer is "yes", it asks for the new material. If the answer is "no", it continues with the material which was in the input data file. Given a poor material it may be impossible to find a forming sequence and a warning message is displayed if this is the case (see section 5.4).
The next argument after the cold formability (Figure 22) in the material "fact" is the maximum reduction in area for open forward extrusion. This upper limit is used in the rules for open forward extrusion (see sections 2.2.2 and 4.3.7). Similarly the next argument gives the maximum reduction area for trapped die extrusion for that material. This value is used in the rules for trapped forward extrusion (see sections 2.2.2 and 4.3.7). The last value in the argument list of the material "fact" (Figure 22) is the limit on the unsupported billet length for upsetting in terms of billet diameter. This value is used for calculating the number of upsetting dies as discussed in section 4.3.8.

5.2 OUTPUTS

The output of FORMEX is the plot of the forming sequence and an output file which describes the operations to be performed at each stage. Figure 23 shows the forming sequence plotted by FORMEX for the part in Figure 20. The corresponding output file is shown in Figure 24. The information on the output file is also displayed to the user during the design session.

In the forming sequence plot (Figure 23), the operations to be performed are displayed on top of the stations. For example, in Figure 23, forward extrusion is performed in the first and the second station and upsetting is combined with forward extrusion in Stations 3, 4 and 5. Since the first drawing is the cross section of the billet, "billet" is written on top of it instead of the name of a forming
Figure 23. Forming Sequence Plot for the Example Part in Figure 20.
billet diameter = 0.1600000000E+01
billet length = 0.1037187500E+02

Station 1
-------
trapped forward extrusion
zone 6
diameter = 0.1200000000E+01

Station 2
-------
open forward extrusion,
zzone 7
diameter = 0.1000000000E+01

Station 3
-------
open forward extrusion
zone 9
diameter = 0.8000000000E+00
upsetting
smaller diameter = 0.1600000000E+01
larger diameter = 0.1600000000E+01
mean diameter = 0.1600000000E+01
unsupported length = 0.4300000000E+01
unsupported length / mean diameter = 0.2687500000E+01

Station 4
-------
open forward extrusion
zone 8
diameter = 0.9000000000E+00
upsetting
smaller diameter = 0.1600000000E+01
larger diameter = 0.2000000000E+01
mean diameter = 0.1800000000E+01
unsupported length = 0.383606557E+01
unsupported length / mean diameter = 0.1879781421E+01

Station 5
-------
open forward extrusion
zone 5
diameter = 0.1400000000E+01
upsetting
smaller diameter = 0.1600000000E+01
larger diameter = 0.2340000000E+01
mean diameter = 0.1970000000E+01
unsupported length = 0.2784780468E+01
unsupported length / mean diameter = 0.1413594146E+01

Figure 24. Output File for the Example Part in Figure 20.
operation. This plot shows the billet and the outputs of the five stations forming this part. The shape on the lower right-hand corner is the final part shape.

The corresponding output file shown in Figure 24 summarizes the results. The first portion of the output file is the same as the input file (Figure 21) and is not included in Figure 24 to avoid repetition. The results section of the output file begins with the billet dimensions: billet diameter and length. Then the operations in each station are recorded. In the first station Zone 6 is trapped forward extruded (Figure 24) to a diameter of 1.200 inches from 1.600 inches. The zone numbering is described in Section 4.3.5. Zone 7 is forward extruded to 1.000 inches in Station 2. In Station 3, forward extrusion is combined with tapered upsetting. Zone 9 is forward extruded to 0.800 inches and the Zones 2, 3, and 4 are cone upset. In Figure 24 in Station 3, the shape defined under upsetting is the shape entering that station. There the billet is taper upset, so the smaller, larger and the mean diameters are all the same and are equal to the billet diameter which is 1.600 inches. The unsupported billet length is 4.300 inches. The ratio of unsupported billet length to the billet diameter is 2.688 which is higher than the limit for carbon steel. So tapered upsetting is done in Station 3. The dimensions of the taper can be read from the input shape to the next station (Station 4).

In Station 4, open forward extrusion is combined with taper upsetting again (Figure 24). Zone 8 is open forward extruded to a diameter of 0.900 inch. Zones 2, 3 and 4 are tapered upset. The entrance shape to this taper is the cone from Station 3 with smaller diameter
equal to 1.600 inches (same as the billet diameter), larger diameter equal to 2.000 inches and the mean diameter equal to 1.800 inches. The unsupported billet length is 3.338 inches and the ratio of the unsupported billet length to the mean cone diameter is 1.880 inches.

The fifth and the last station also combines open forward extrusion with upsetting (Figure 24). Zone 5 is open forward extruded to the diameter of 1.400 inches. Zones 2, 3 and 4 are upset to their final dimensions. The entrance cone to this station has a smaller diameter equal to 1.600 inches, larger diameter equal to 2.340 inches and the mean diameter equal to 1.970 inches. The length of the unsupported billet for this station is equal to 2.785 inches. The ratio of the unsupported workpiece length to the mean cone diameter is 1.413 which is well within the limits of maximum unsupported length to diameter ratio for carbon steel.

FORMEX can also plot single stations on the terminal to see the details better. Figure 25 shows such a plot for Station 3 in the forming sequence shown in Figure 23. In this station the top portion of the workpiece is used for additional support on the punch side and is not upset.

5.3 MENU OPTIONS

In order to explain FORMEX in more detail its menu options and their function is described with an example next.
Figure 25. Enlarged Plot of Station 3 from Figure 23.
For this example, the messages from the computer are written in italics and the commands the user enters are underlined.

Prolog interpreter is linked with the object modules of Fortran subroutines that are used for plotting. The executable code is called PROFOR (PROlog and FORtran). First the user runs PROFOR:

```prolog
$PROFOR
Prolog-1

Copyright 1984 Expert Systems (UK) Ltd
Serial Number: 000033
Licensed to: Battelle Columbus Laboratories
    Columbus, Ohio
    U.S.A

?-['FORMEX.PRO'].
```

After the Prolog prompt ?-, user enters the FORMEX file name that is to be consulted. If there are any syntax errors in FORMEX clauses, Prolog displays them at this point. It, then, continues by displaying the FORMEX menu:

**FORMING SEQUENCE DESIGN OPTIONS**

1. start new design
2. check input geometry
3. design forming sequence
4. redesign forming sequence
5. plot forming sequence
6. plot single blow
7. prepare output file
8. Prolog interpreter
9. exit from Prolog
Option: 1

Enter input file name: part38.dat

The forming sequence design starts by choosing menu item 1. When this item is picked, FORMEX asks for a input file name. The user enters the name of the input file (part38.dat) which is shown in Figure 21. Input file structures was explained in Section 5.1.

After reading the input file, the user may want to use the menu item 2 to check the input geometry:

Option: 2

The menu Item 2 causes the input part to be displayed on the terminal as shown in Figure 20. If there is an error in the input file, it will be noticed at this point without going into the forming sequence design.

If the input part geometry is correct, then the user can go ahead and use menu Item 3 to design the forming sequence. Most of the major decisions of the design task are taken by this menu item. First FORMEX checks the cold formability of the given metal and displays the following message:

Option: 3

Material carbon steel 1045 is poor for cold forming.

Fair carbon steels for cold forming are:

1035, 1038, 1040

Good carbon steels for cold forming are:

1018, 1020, 1022, 1024, 1030

Best carbon steels for cold forming are:

1008, 1010, 1013, 1016, 1017

Do you want to change this material (y/n): y
Material type: **carbon steel**

Material no: **1008**

This message displays only the metals in that group, which are better for cold forming than the one given in the input file. In this case the material in the input file is carbon steel 1045 which is only "fair" for cold forming. So FORMEX displays the carbon steels which are better for cold forming than 1045 in three groups, fair, good and best. As described in Section 5.1, cold forming materials are classified in four groups as poor, fair, good and best [8]. If the user wants to change the stock material, he answers "y" to the question above and enters the material type next. If not, he enters "n". If the material entered is not in the knowledge-base, a message is displayed and user is asked to enter another material.

FORMEX, then, uses the rules for the billet diameter and displays its choice as:

*Recommended billet diameter: 1.600*

Do you want to change this diameter (y/n) : n

The billet diameter recommended here is found by the method discussed in Section 4.3.6. If the user wants to use another billet size, he answers "y" to the above question and enters his choice in the next line.

FORMEX, then, finds the forming extrusion sequence and the upsetting sequence and displays them as follows:

**EXTRUSION OPERATIONS**

*Station 1: trapped forward extrusion, Zone 6, diameter = 1.200*

*Station 2: open forward extrusion, Zone 7, diameter = 1.000*
Station 3: open forward extrusion, Zone 9, diameter = 0.800
Station 4: open forward extrusion, Zone 8, diameter = 0.900
Station 5: open forward extrusion, Zone 5, diameter = 1.400

UPSETTING OPERATIONS

Recommended number of upsetting blows: 2

Upsetting Blow 1:

smaller diameter = 1.600
larger diameter = 1.600
mean diameter = 1.600
unsupported length = 3.962
unsupported length/mean diameter = 2.477

Upsetting Blow 2:

smaller diameter = 1.600
larger diameter = 2.080
mean diameter = 1.840
unsupported length = 2.979
unsupported length/mean diameter = 1.619

Do you want to change the number of upsetting blows (y/n): n

First, the sequence of extrusion operations are displayed with the station number, type of extrusion, zone number and the diameter of the zone after extrusion. In the above display, there are five stations with extrusion operations.

Upsetting operations are displayed after the extrusion operations. Initially FORMEX displays the "Recommended number of upsetting blows", which is two for the above example. This number is found using
the "generate and test" method as described in section 4.3.8 and in Appendix A. If the user wishes to change the number of upsetting blows, he answers "y" to the above question, and he enters the number of upsetting blows he wishes to have. The limit for the number of upsetting blows is four since it is not realistic to have more than four upsetting blows as mentioned in section 4.3.8. If the user changes the number of upsetting blows above, then the dimensions of the new upsetting blows are computed and displayed, same as above, starting with upsetting Blow 1.

Menu item 4 is used to redesign another forming sequence for the given part. As mentioned in Chapter 4, it is important that at some points of the design process the user has the ability to override the decisions of the system. For example, the user may first want to design a forming sequence with the recommended billet diameter by FORMEX. Then, he may want to find a forming sequence for the same part but assuming a different billet diameter. In such a case, he can use menu item 4 for the second design. When menu item 4 is picked, first, the Prolog working memory is cleared from the "facts" that are "asserted" during the previous design section. Designer need not have to enter the input file name again when using this option. After he picks this item, FORMEX will display the recommended billet diameter as described above. He answers the question "Do you want to change the billet diameter", "y" and enters the diameter he wishes to use. Similarly menu item 4 can be used to get several alternative forming sequences by changing the number of upsetting blows in each time. Examples of this are given in section 5.5.
Menu item 5 (see above) is used to plot the forming sequence designed. When this item is selected, a Prolog predicate invokes a set of Fortran subroutines for plotting the forming sequence on the terminal. The user terminal must be Tektronix 4010 series compatible to be able to use this menu item. An example plot using this menu item is shown in Figure 23.

Menu item 6 is for plotting single stations in a larger scale. Since the figures in the forming sequence plot, such as in Figure 23, are rather small, it may be desirable to see them with a larger scale. When menu item 6 is picked, FORMEX asks for the station number. Figure 25 shows the plot drawn if the user enters Station Number 3 for the forming sequence in Figure 23. Similar to menu item 5, the user terminal must be Tektronix 4010 series compatible to be able to use menu item 6.

Menu item 7 is for creating output files. If the user wants to keep the results after a forming sequence design session, he can use menu item 7. An example of FORMEX output file is shown in Figure 24.

Menu item 8 is for stopping the design process and accessing the Prolog interpreter directly. The Prolog prompt ?- is displayed on the terminal and the user is at the Prolog interpreter level. This is most useful for debugging purposes. The current facts and the rules can be checked using the Prolog predicates. The user may then reenter FORMEX menu level by typing "prologue".

Menu item 9 is for ending a FORMEX session. It halts the Prolog Interpreter and the operating system prompt is seen at the user terminal.
A second forming sequence for another part can be established without aborting FORMEX by using menu item 1. If menu item 1 is selected, all the facts created in that design session as well as the input data are deleted from the working memory and FORMEX is ready to start a new design session for another part. If it is a redesign for the same part, menu item 4 should be used instead. Menu item 1 is used for only new parts.

5.4 WARNING MESSAGES

Another feature of FORMEX is that it displays "warnings" to the user during the forming sequence design session. Warnings are displayed whenever FORMEX notices a problem but can not solve it by itself. The user may choose one of several options to solve the problem when a warning is displayed. FORMEX suggests a solution in some cases.

Some of the important warning messages are:

1. WARNING! Number of stations required exceeds the number of stations on this machine: As described in section 4.3.9, FORMEX tries to reduce the number of stations required by combining principal cold forming operations in single stations. If the number of stations required is still more than the number of stations available on that machine, FORMEX displays the warning message above. A forming sequence is established which needs more stations than available on the machine. Changing the billet size and trying again may solve this problem.
2. **WARNING! Maximum cutoff diameter is ... for this machine:** As explained in section 4.3.6 user may override the choice of FORMEX for the billet diameter and enter his own choice. If this value exceeds the maximum cutoff diameter of that machine, FORMEX displays the above message and asks for another value.

3. **WARNING! Cutoff length exceeds maximum cutoff length of ...**

   *for this machine:* The cutoff length is calculated from the billet diameter and the part volume. It is compared with the maximum cutoff length for the given machine (section 5.1). When the computed cutoff length exceeds the machine limit, the above warning message is displayed. Increasing billet diameter may solve this problem.

4. **WARNING! Extrusion reduction ratio too high for the zones ...**

   *May want to reduce billet diameter or change material:* This message is displayed if there is one (or more) zones that cannot be forward extruded due to the limit on the reduction in area. FORMEX suggests (1) to reduce billet diameter so the forward extrusion reductions will be less or (2) change the material so the limit on forward extrusion reduction is higher.

5. **WARNING! Needs more than four upsetting blows. May want to change billet diameter:** Maximum number of upsetting blows allowed in FORMEX is four. More than four upsetting blows is not practical and in some cases maximum of two upsetting blows is recommended [9]. If more than four
upsetting blows are required for a head, FORMEX displays the above message. It also suggests that increasing billet diameter may solve this problem.

6. **WARNING! Station ...**

*Die ejector length required for this station = ...*  
*Maximum die ejector length = ...*  
*May want to add spring load on die ejector:* This message warns that the ejector length required on the die side for this station exceeds the limit for that machine (section 5.1). FORMEX recommends to add spring load on the ejector to eliminate this problem.

7. **WARNING! Station ...**

*Punch ejector length required for this station = ...*  
*Maximum punch ejector length = ...*  
*May want to add spring load on the punch ejector:* This message is similar to the warning message 6 above. Instead of the ejector on the die side, this one is concerned with the ejector on the punch side. If the length of the upset exceeds the punch ejector length, this message is displayed on the terminal. Again spring-loaded ejectors are suggested as one solution.

### 5.5 EXAMPLE FORMING SEQUENCES

FORMEX has been tested with several solid parts to find its limitations. Some of these example forming sequences are discussed in this section.
FORMEX recommends the number of upsetting blows for a given part, but this number can be overridden by the user as described in Section 4.3.8. The effects of changing the number of upsetting blows on the upsetting sequence is shown here with some examples. For the part drawing in Figure 20 (the input data is shown in Figure 21), FORMEX finds the forming sequence shown in Figures 23 and 24 for three upsetting blows. The forming sequence can then be redesigned using the menu item "redesign forming sequence" (Section 5.3) with two upsetting blows instead of three. Figure 26 shows the forming sequence plot with only two upsetting blows. In this sequence there is upsetting only on the fourth and the fifth stations but not on the third station, whereas upsetting starts in the third station in the sequence shown in Figure 23.

Another example part is shown in Figure 27 with the corresponding input data file shown in Figure 28. FORMEX recommends two upsetting blows for this part and the final forming sequence is shown in Figure 29. The user may want to try three or four upsetting blows. Forming sequences for the same part with three upsetting blows is given in Figure 30 and four upsetting blows in given in Figure 31. When Figures 29, 30 and 31 are compared, it is seen that the extrusion sequence is the same in all three but the upsetting sequence is different. This is due to the fact that forward extrusion and upsetting were assumed to be independent of each other as described in Chapter 4. This assumption is not true if backward extrusion is also included in the system. Therefore forward extrusion and upsetting rules need to be modified when rules for backward extrusion is added for hollow parts.
Figure 26. Forming Sequence Plot for the Example Part in Figure 20 with 2 Upsetting Blows.
Figure 27. Example Part with 3 Upsetting Zones.
TITLE
PART # 70

PART DESCRIPTION
This is a test part.

PART GEOMETRY
20
0.750 8.100
0.750 7.300
0.800 7.300
0.800 7.100
0.700 7.100
0.700 7.000
0.500 7.000
0.500 5.400
0.412 5.400
0.412 4.600
0.390 4.600
0.390 4.200
0.375 4.200
0.375 3.400
0.360 3.400
0.360 2.400
0.340 2.400
0.340 2.000
0.320 2.000
0.320 0.000

MATERIAL DESCRIPTION
This part is to be made of coil wire.

MATERIAL SPECIFICATIONS
alloy_steel 3115

MACHINE DESCRIPTION
This is a five die boltmaker.

MACHINE SPECIFICATIONS
machine name : Boltmaker B20
number of stages : 5
wire diameter (max) : 1.5
cutoff length (max) : 15.0
slide stroke length : 10.0
punch ejector stroke : 5.0
die ejector stroke : 6.0
speed (pieces/min) : 90

Figure 28. Input Data File for the Example Part in Figure 27.
Figure 29. Forming Sequence Plot for the Example Part in Figure 27 with 2 Upsetting Blows.
Figure 30. Forming Sequence Plot for the Example Part in Figure 27 with 3 Upsetting Blows.
Figure 31. Forming Sequence Plot for the Example Part in Figure 27 with 4 Upsetting Blows.
Another important point about the part given in Figure 27 is that it has three zones to be upset (see Section 4.3.5 for zones and zone numbering). Figure 31 shows how these three zones are being upset in four blows. The first upsetting blow (Station 2) develops the cone. The second upsetting blow (Station 3) widens the cone and forms Zone 3. The third upsetting blow (Station 4) widens the cone even more and forms Zone 2. The fourth and last upsetting blow (Station 5) forms Zone 1 and completes upsetting. Notice that forward extrusion is combined with upsetting in all these three stations.

The forming sequence for two similar parts may be quite different. To show this, two similar parts (Figure 32 and Figure 35) are chosen. The input files for these parts are given in Figures 33 and 36 respectively. Forming sequences established by FORMEX for these parts are given in Figures 34 and 37. When the forming sequences in Figures 34 and 37 are compared, it is seen that the upsetting sequence in both plots are the same since the zones to be upset are the same in both parts. But when the extrusion sequence for these parts are compared in Figures 34 and 37, it is seen that they are entirely different. This example shows the sensitivity of FORMEX. Even though there is a small difference between the two input part shapes, FORMEX can establish completely different forming sequences for these parts. Systems that are developed around classification schemes, usually, give similar results for similar input part shapes. This example shows an advantage of FORMEX that it can establish quite different forming sequences, if required, for very similar parts.
Figure 32. Example Part
TITLE
  PART # 90

PART DESCRIPTION
  This is a test part.

PART GEOMETRY
  18
  0.900  8.600
  0.900  8.400
  1.100  8.400
  1.100  7.200
  0.900  7.200
  0.900  7.000
  0.700  7.000
  0.700  5.400
  0.650  5.400
  0.650  4.400
  0.620  4.400
  0.620  3.200
  0.550  3.200
  0.550  2.800
  0.530  2.800
  0.530  1.800
  0.510  1.800
  0.510  0.000

MATERIAL DESCRIPTION
  This part is to be made of coil wire.

MATERIAL SPECIFICATIONS
  alloy_steel 3115

MACHINE DESCRIPTION
  This is a five die boltmaker.

MACHINE SPECIFICATIONS
  machine name : Boltmaker B20
  number of stages : 5
  wire diameter (max) : 1.5
  cutoff length (max) : 15.0
  slide stroke length : 10.0
  punch ejector stroke : 5.0
  die ejector stroke : 6.0
  speed (pieces/min) : 90

Figure 33. Input Data File for the Example Part in Figure 32.
Figure 34. Forming Sequence Plot for the Example Part in Figure 32.
Figure 35. Example Part
TITLE
PART # 91

PART DESCRIPTION
This is a test part.

PART GEOMETRY
18
0.900 8.600
0.900 8.400
1.100 8.400
1.100 7.200
0.900 7.200
0.900 7.000
0.700 7.000
0.700 5.400
0.650 5.400
0.650 4.400
0.620 4.400
0.620 3.200
0.535 3.200
0.535 2.800
0.530 2.800
0.530 1.800
0.510 1.800
0.510 0.000

MATERIAL DESCRIPTION
This part is to be made of coil wire.

MATERIAL SPECIFICATIONS
alloy_steel 3115

MACHINE DESCRIPTION
This is a five die boltmaker.

MACHINE SPECIFICATIONS
machine name : Boltmaker B20
number of stages : 5
wire diameter (max) : 1.5
cutoff length (max) : 15.0
slide stroke length : 10.0
punch ejector stroke : 5.0
die ejector stroke : 6.0
speed (pieces/min) : 90

Figure 36. Input Data File for the Example Part in Figure 35.
Figure 37. Forming Sequence Plot for the Example Part in Figure 35.
The results of FORMEX are acceptable for the given class of solid parts. The die designers would also choose similar forming sequences for this part class. The part class that FORMEX can accept includes solid parts that requires:

1. only forward extrusion or
2. only upsetting or
3. upsetting on one end and extrusion on the other end or
4. upsetting in the middle and extrusion on only one end.

Current version of FORMEX does not work for solid parts that needs forward extrusion on both ends. In some cases a billet may be extruded on both ends. Cold forming such parts may require rotating the part between stations or extrusion on punch side as well as the die side. Rules and guidelines about these operations are not included in FORMEX yet.
CHAPTER 6

CONCLUSION AND RECOMMENDATIONS

This chapter is divided into 3 sections. In the first section, the main contributions of this study are summarized. The second section discusses the application of KBSs to forming sequence design for cold forging and the results of the study. The recommendations for similar studies and possible new research related to this work are discussed in the last section.

6.1 THE CONTRIBUTION

Forming sequence design is a complex design task which is performed by highly experienced die designers. The state of the art in forming sequence design is to use computer-aided drafting systems for the final drawings of the forming dies. The major decisions, such as which forming operations to use on each station, are made by experienced die designers prior to the final drafting. The number of expert designers is decreasing due to the retirement of older designers and the lack of younger ones. Computer systems which can capture the knowledge and the experience of the expert designers, to help less experienced designers in forming sequence design, are highly desired.

Previous studies for developing such systems focus on very detailed classification schemes. In these systems, axisymmetric parts
with similar geometric properties for cold forming are gathered in a group. A different forming sequence corresponds to every part group. Given a part shape, the forming sequence for that part is chosen by finding out the part group it belongs to. Material type is another factor used in these classification schemes besides geometric properties. This approach has some problems. First of all, even a very detailed classification system is not enough to cover all the axisymmetric parts that are cold formed. If a given part shape is not included in the classification scheme, the system can not do anything for that part. Sometimes very similar parts require different forming sequences and the detail of classification may not be enough. Classification systems are rigid and adding new knowledge may need the reconstruction of the whole classification system.

This study was a new approach for developing CAD systems for cold forming. Instead of storing the cold forming knowledge in detailed classification schemes, the knowledge was represented as a knowledge-base in a production system. The forming sequence design task was formalized and a knowledge-based system was developed for a class of axisymmetric parts. The system can find satisfactory forming sequences for solid round parts.

A main contribution of this study was the partial formalization of the forming sequence design process and the development of a knowledge-based system for solid round parts. The KBS can be expanded later to include other part shapes. There is not a handbook to assist die designers in establishing forming sequences for cold forging. Die designers learn mostly through their years of experience and the
knowledge they accumulate is wasted when they retire. The KBS developed, in this study, captures some of the knowledge used for establishing forming sequences for solid round parts. The system is developed in such a way that the knowledge for establishing forming sequences for other part shapes can be added to the system in separate modules. This system is not only a tool to assist less experienced die designers in forming sequence design but also a means of accumulating the knowledge of the more experienced die designers. The design rules and guidelines, that a die designer develops during his career, will not be lost when he retires anymore.

6.2 THE FORMEX SYSTEM

This section summaries the key features of this study. As described in Chapter 4, forming sequence design was studied carefully before attempting to develop FORMEX. Important properties of the task were analyzed first and FORMEX was structured for these specific needs. There is not a general recipe for developing KBS's, but there are several techniques (knowledge representation schemes, problem solving methods, etc.) each suitable for a different type of problem. So the detailed study of the forming sequence design was essential for choosing the suitable methods for developing the system.

Decomposition of the forming sequence design task into 7 partially independent subtasks reduced the difficulty of the problem to a manageable size. This decomposition is valid for solid parts but it needs to be modified for hollow parts. The Subtasks 5 and 6 (see
sections 4.3.7 and 4.3.8) assume that forward extrusion sequence and upsetting sequence can be established independently and then can be combined to reduce the number of stations. When backward extrusion rules and guidelines are studied, it is seen that they are very much dependent on the forward extrusion and upsetting operations if they are to be performed together. Therefore, a specific rule file for hollow parts (Figure 9) must be structured differently than that of the one for solid parts.

The structure of FORMEX (Figure 9) makes it easier to make modifications such as the one described above for including hollow parts. The structure of FORMEX takes advantage of the fact that axisymmetric parts can be classified into groups like, solid parts with decreasing diameters on one end, solid parts with decreasing diameters on both ends, parts with a cavity on one end, etc. as described in section 4.3.3. This structure makes it easier to add new specific forming rules for a new class of axisymmetric parts without disturbing the rest of the system. This is important since it is almost impossible to create a general forming design system that can handle all the axisymmetric parts. Each company is working with different part shapes most of which have some tricks or exceptions. So, the practical use of such a system in industry is very dependent on how easy it can be modified to handle the parts that are cold formed in a given company.

Forming sequence design involves intensive geometry manipulation. A KBS for forming sequence design needs powerful internal geometry representation schemes. The 2-D and a 3-D geometry representation
scheme developed in this study (section 4.3.5) was adequate for solid parts. But it needs to be modified for hollow parts, because the general shape of the part becomes more important than its small elements (like zones) in backward extrusion.

Developing the system as a forward chaining production system is believed to be the right choice. Implementing the system using the Prolog language was a pleasant experience except for some technical difficulties inherent in the specific Prolog interpreter used. It does not have formatted I/O so these routines had to be written. The Prolog interpreter used also has very limited Fortran interface capability so the plotting routines which are in Fortran are interfaced with some difficulty. Besides these technical problems it was concluded that the language is suitable and adequate for developing such systems.

Currently it is being considered to interface FORMEX with a geometry modelling system (such as ANVIL 4000). This will have two advantages: (1) graphic I/O for FORMEX will be enhanced and (2) it will be easier to integrate FORMEX with other systems (such as finite element analysis systems).

6.3 RECOMMENDATIONS

This section contains recommendations on the application of KBS methodologies to similar tasks in engineering design and manufacturing. First, the recommendations to those who are planning to continue on this study on forming sequence design are stated. In the second and last portion of this section, general recommendations on
application of the knowledge engineering tools to engineering design and manufacturing tasks are discussed.

This work can be expanded in several ways. We are now considering two areas of expansion: (a) modifying the system so that parts with cavities and hollow parts are also included and (b) interfacing FORMEX with a geometry modelling system.

Parts with cavities and hollow parts form an important class among cold formed parts. Rules and guidelines for cold forming hollow parts are similar to those for solid parts. Three major operations which should be added to the operations already considered are forward cup extrusion, backward cup extrusion (section 2.2) and piercing. Two factors that are important when considering hollow parts, are: (a) the internal geometry representation needs to be modified for these parts, and (b) the upsetting and extrusion operations cannot be assumed independent anymore (4.7.7-4.3.9).

Parts with cavities and hollow parts may require some modifications in the internal geometry representation scheme described in section 4.3.5. In the current version of FORMEX, solid parts are divided into simple shape elements such as cylindrical elements and conical elements (section 4.3.5). New shape elements such as hollow cylindrical elements and hollow conical elements will be needed to represent the new parts.

For solid parts upsetting and extrusion operations were assumed to be independent in developing FORMEX. Even though this is a good assumption for solid parts, it is not correct for hollow parts.
Forward cup extrusion and backward cup extrusion are often combined with other extrusion processes and upsetting processes as described in section 2.2.2. When the rules and guidelines for forward cup extrusion and backward cup extrusion are added to FORMEX, it is important to keep in mind that these operations cannot be decided by themselves only and they need to be considered simultaneously with other extrusion operations and upsetting.

The second area where we are considering to modify FORMEX is to interface it with a geometry modelling system such as ANVIL-4000. In the current version of FORMEX, the user enters the input part geometry in a file as explained in section 5.1. Instead of asking the user to prepare an input file to enter part geometry, FORMEX can be interfaced with a commercial geometry modelling system and the user is asked to enter part geometry using the different capabilities of the geometry modelling system. Similarly, the results of FORMEX can be displayed using the geometry modelling system instead of the graphics routines of FORMEX. This new configuration will let the user have the interactive graphics capabilities of the commercial geometry modelling system and the automatic forming sequence design by FORMEX simultaneously. Iterative design steps can be performed without delay if the same system is also interfaced with an FEM program for analyzing stress distribution and load for cold forging.

FORMEX can be interfaced with ANVIL-4000 using the Graphics Applications Programming Language (GRAPL) of ANVIL-4000. GRAPL is a parametric and associative language that can generate most of the ANVIL-4000 entities. It is an algorithmic language with assignment
statements, iteration statements, conditional statements, subscripted variables statement labels and display manipulation (See ANVIL-4000 GRAPL Reference Manual for details). When this interface is completed, the user will enter part geometry with the several options available in ANVIL-4000. He will then get the GRAPL menu and pick "AUTO GRAPL" option to generate a GRAPL program for this part. Next, a file will be created using the "WRITE PROGRAM FILE" option in the GRAPL menu. FORMEX will read this file as the input part geometry file. Then FORMEX will be used to establish a forming sequence for the input part. FORMEX will then represent the forming sequence drawing in a GRAPL program and write it in a file. ANVIL-4000 will read this file using the "READ PROGRAM FROM FILE" option in the GRAPL menu. The next step is to use the "RUN GRAPL PROGRAM" option of the GRAPL menu of ANVIL-4000 to display the forming sequence that was designed by FORMEX. At this point of design, the designer may use ANVIL-4000 to change part geometry and again go through the steps described above to find a different forming sequence.

In the rest of this section general recommendations will be discussed for similar studies in developing KBSs in engineering design and manufacturing. During this study, the general principles stated under "Maxims for Constructing Experts Systems" by B. G. Buchanan et al. [30] were observed to be valid.

One of the most important issues in developing KBSs is that the task should be suitable for this approach. The task should not involve a lot of commonsense knowledge. The scope of the task should be as narrow as possible. Forming sequence design for cold forging was a
suitable task and limiting the initial system to only solid parts narrowed down the scope of the work.

There are not set rules for constructing KBSs. There are several techniques offered in AI to handle different tasks. After carefully studying the domain, appropriate techniques should be chosen and mixed together for the specific task. Different problems require different approaches. Therefore, it is important to study the problem in detail and identify the important aspect of it.

Another important issue in developing KBSs for engineering design and manufacturing is the practical usefulness of the system. In constructing practical systems, the whole system should not necessarily be built with an AI approach but other techniques should be combined with the new AI techniques. For example, in developing CAD systems for product and process design one should start with a commercial geometry modelling system. First the geometry modelling system should be used to develop an interactive design system for that domain. This system will contain applications software specific to that problem. For example, for the forming sequence design task, it should be able to divide parts into smaller shape elements such as cylinders and cones and calculate the volumes of these elements. After completing this customized interactive system, certain steps of the design task, where the user may need some assistance from the systems, should be found out. AI techniques are then applied to supply this assistance to the user. Building this assistance can be a gradual effort. First, little assistance may be available at only a few stages. Later, the available assistance is increased towards the ultimate goal of an automatic design system. By
increasing the level of assistance, less and less experienced designers will be able to perform the task with the help of the system.

An important task in constructing KBSs is engineering design and manufacturing is the internal geometry representation. For engineering design tasks, adequate internal representation of part geometry is essential for a useful KBS (See section 4.3.5). Representation of complex 3-D shapes may be complicated and enough time needs to be spent to have complete representation schemes.

Even though some of the expectations from the field of AI seem unrealistic, there are many areas in engineering design and manufacturing where similar KBSs can be useful. Process planning for sheet metal forming, product and mold design for injection molding and casting die and mold design are some of these areas where KBSs can be used to assist engineers to perform their tasks better.
REFERENCES


APPENDIX A

SAMPLE FORMEX RULES
APPENDIX A

SAMPLE FORMEX RULES

The objective of this appendix is to give the reader some insight about FORMEX and show how some rules and guidelines about forming sequence design are represented as Prolog clauses. The Prolog clauses will not be explained in detail and only their function will be explained. Readers having problems in understanding Prolog terms should refer to "Programming in Prolog" by Clocksin and Mellish [46].

A.1 INTERNAL GEOMETRY REPRESENTATION

One of the subtasks in FORMEX is to convert the input geometry coordinates to an internal geometry representation as described in section 4.3.5. This section of the appendix should be read after rereading section 4.3.5.

Figure A-1 shows some of the FORMEX clauses for constructing the internal geometry representation. The input_to_internal procedure shown in Figure A-1 generates the facts for the internal geometry representation from the list of coordinate pairs for the input part cross section (See section 4.3.5). It uses the built-in predicate "assert" to add facts about the 2-D and the 3-D representations described in section 4.3.5.

The facts for the 2-D representation that are added to the working memory by the input_to_internal procedure (Figure A-1) are zone,
input_to_internal :- initialize_1,
  represent_input.

initialize_1 :- part(List),
  assertz(corners_left(List)),
  assertz(nvertices(0)),
  assertz(nedges(0)),
  assertz(nzones(0)),
  assertz(zones(LJ)).

represent_input :- get_2_corners(CX1,Y13,CX2,Y23),
  make_vertices(E1,V11,Y13,E2,V21,Y23,E3,V31,Y33,E4,V41,Y43),
  make_edges(E1,V1,E2,V2,V3,E3,E4,V4,V1),
  make_zones(2,E1,E2,E3,E4,Type),
  add_zone(Z),
  make_shapes(Z,E11,E21,E31,E41,Type),
  represent_input.

get_2_corners(CX1,Y13,CX2,Y23) :- retract(corners_left(CX1,Y13,CX2,Y23)).

get_2_corners(CX1,Y13,CX2,Y23) :- retract(corners_left(CX1,Y13,T1)),
  assertz(corners_left(T1)),
  retract(corners_left(CX2,Y23,T2)),
  assertz(corners_left(T2)),
  retract(corners_left(CX3,Y33,T3)),
  check_dent(Y2,Y3,T1,T2).

check_dent(Y2,Y3,T1,T2) :- Y2=Y3,
  assertz(corners_left(T2)).

check_dent(Y2,Y3,T1,T2) :- assertz(corners_left(T1)).

make_vertices(E1,V11,Y13,E2,V21,Y23,E3,V31,Y33,E4,V41,Y43) :-
  write_vertex(V1,E1,Y1),
  write_vertex(V2,E2,Y2),
  X3 is 0,
  Y3 is Y2,
  write_vertex(V3,X3,Y3),
  X4 is 0,
  Y4 is Y1,
  write_vertex(V4,X4,Y4).

Figure A-1. Sample FORMEX Rules for Constructing the Internal Geometry Representation
write_vertex(Vold,X,Y) :- vertex(Vold,X,Y).

write_vertex(N1,X,Y) :- retract(vertices(N)),
                  N1 is N+1,
                  assertz(vertex(N1,X,Y)),
                  assertz(vertices(N1)).

make_edges([E1,V1,V2],[E2,V2,V3],[E3,V3,V4],[E4,V4,V1]) :- write_edge(E1,V1,V2),
                         write_edge(E2,V2,V3),
                         write_edge(E3,V3,V4),
                         write_edge(E4,V4,V1).

write_edge(Eold,Vbegin,Vend) :- edge(Eold,Vbegin,Vend).

write_edge(N1,Vbegin,Vend) :- retract(nedges(N)),
                         N1 is N+1,
                         assertz(edge(N1,Vbegin,Vend)),
                         assertz(nedges(N1)).

make_zones(N1,E1,E2,E3,E4,Type) :- vertical_edge(E1),
                        retract(nzones(N)),
                        N1 is N+1,
                        assertz(zone(N1,E1,E2,E3,E4,straight)),
                        assertz(nzones(N1)).

add_zone(2) :- retract(zones(Lold)),
              append(Lold,[Z],Lnew),
              assertz(zones(Lnew)).

vertical_edge(E) :- edge(E,V1,V2),
                vertex(V1,X1,Y1),
                vertex(V2,X2,Y2),
                X1 = X2.

make_shapes(Z,[V1,X1,Y1],[V2,X2,Y2]) :- nzones(N),
              L is Y1 - Y2,
              D1 is 2 * X1,
              D2 is 2 * X2,
              assertz(shape_dimensions(N,L,D1,D2)),
              X3 is -(X2),
              X4 is -(X1),
              assertz(shape_corners(N,n,[X1,Y1],[X2,Y2],[X3,Y3],[X4,Y4])).

Figure A-1. Sample FORMEX Rules for Constructing the Internal Geometry Representation (Continued)
edge, vertex, nzone (for number of zones), nedge (for number of edges) and nvertex (for number of vertices). For the 3-D representation, the input_to_internal procedure adds facts such as shape_dimensions and shape_corners. In 3-D representation the dimensions of the primitive shape elements, such as cylindrical elements and conical elements, are given by the shape_dimensions facts. The orientation of the shape elements are given by the shape_corners facts.

The facts that are added in the working memory by the input_to_internal procedure (Figure A-1) are about the input part shape which is the final workpiece shape in the forming sequence. Similar facts about the zones and shape elements are added to the working memory during the forming sequence design. When FORMEX decides for an operation at a station, new zones and shape elements after the operation are found and added to the working memory. These facts about the zones and the shape elements are also used for plotting the forming sequence.

A.2 FORWARD EXTRUSION RULES

Another important subtask in FORMEX is establishing the forward extrusion sequence. As explained in sections 4.3.7 and 4.3.8, FORMEX first finds a sequence for forward extrusion operations, then it finds the upsetting sequence and finally it combines the extrusion sequence and the upsetting sequence.

Figure A-2 shows some of the rules for establishing the forward extrusion sequence. When a workpiece has several portions to be forward extruded, die designer has to decide the order of the
/* - SINGLE REDUCTION TRAPPED FORWARD EXTRUSION RULE */

rule(extrusion,30,single_trapped) :- nblow(0),
pick_smallest_zone,
area_before_extrusion(AreaO),
area_after_extrusion(Areal), !,
material(M,N),
(AreaO - Areal) * 100 / AreaO < R,
current_zone(Zl),
assert(single_trapped_extrusion(Zl)).

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
pick_smallest_zone :- zones_to_be_extruded([H|TJ]),
retract(current_zone(_)),
assertz(current_zone(H)),
smallest_diameter_zone(T,H).

smallest_diameter_zone([J,Z] :- 
    retract(current_zone(_)),
    assertz(current_zone(Z)).

smallest_diameter_zone([H|T] | Z) :- 
    shape_dimensions(Z,_,Dold,_),
    shape_dimensions(H,_,Dnew,_),
    Dnew > = Dold,
    smallest_diameter_zone(T,Z).

smallest_diameter_zone([H|T] | Z) :- 
    shape_dimensions(Z,_,Dold,_),
    shape_dimensions(H,_,Dnew,_),
    Dnew < Dold,
    smallest_diameter_zone(T,H).

smallest_diameter_zone([H|T] | Z) :- smallest_diameter_zone(T,Z).

area_before_extrusion(A) :- current_zone(Z),
nblow(N),
shape_corners(Z,N,[X1,Y1], [X2,Y2], [X3,Y3], [X4,Y4]),
D is 2 * X1,
circle_area(D,A).

area_after_extrusion(A) :- current_zone(Z),
shape_dimensions(Z,L,D1,D2),
circle_area(D1,A).

circle_area(Diameter,Area) :- Area is 3.1416 * (Diameter/2) ^ 2.

Figure A-2. Sample FORMEX Rules for Forward Extrusion
extrusions. He has to consider two main issues: (1) which portion will be extruded at a given station and (2) which of the several forward extrusion operations, such as open forward extrusion, trapped forward extrusion, double reduction open forward extrusion (see section 4.3.7), will be performed to extrude the specified portion. He would prefer to extrude the smallest diameter first because extrusion over the same diameter is limited to 3 to 4 times. On the other hand, reduction in area during extrusion is limited (section 4.3.7).

FORMEX has similar constraints for forward extrusion. Figure A-2 shows the rule number 30 for extrusion. This is a rule about single reduction trapped extrusion.

There are open forward extrusion rules before this rule. If none of the open extrusion rules are satisfied, the trapped forward extrusion rule will be tried. This rule basically finds the smallest diameter zone among the zones to be extruded, and checks whether the reduction ratio for trapped extruding that diameter is allowable. If it is allowable, a fact, that records the given zone is trapped die extruded, is added to the working memory.

"pick_smallest_zone" clause (Figure A-2) is used to find the smallest diameter zone among the zones that are to be extruded. In "zones_to_be_extruded(L)." , L is a list of the numbers of the zones that have diameters larger than the billet diameter. Whenever a zone is extruded, its number is deleted from the list L.

"area_before_extrusion" clause (Figure A-2) finds the cross sectional area of a given shape element (or zone). Similarly, "area_after_extrusion" clause (Figure A-2) calculates the cross sectional area of the given shape element after extrusion. The
reduction in area is then calculated in the trapped extrusion rule for the given shape element. The reduction in area is compared with the maximum reduction in area for trapped extrusion for the given material. If it is acceptable, the trapped forward extrusion rule succeeds and the corresponding fact is added to the working memory. If the reduction in area is too high, the trapped extrusion rule fails.

A.3 UPSETTING RULES

In establishing the upsetting sequence, a major decision is the number of upsetting blows. FORMEX considers up to 4 upsetting blows (see section 4.3.8). Generate and test approach is used to find the number of upsetting blows. FORMEX first finds the upsetting dimensions (such as unsupported workpiece length) for one blow. It then checks if these dimensions violate any constraints. If they don't, it assigns only one upsetting blow for the given part. If the dimensions of single upsetting blow violate any constraints, FORMEX tries two upsetting blows instead of one. Again, the dimensions of two upsetting blows are compared with the contraints. If these dimensions are acceptable, two upsetting blows are assigned to form the upset. If not, three upsetting blows are tried. This generate and test approach may continue up to four blows. If a given upsetting cannot be formed in four blows, FORMEX displays a warning message and asks the designer to change the material or the billet diameter.

Figure A-3 shows some of the rules for finding the number of upsetting blows. These rules are for upsetting two or three zones next
RULES FOR FINDING THE NUMBER OF UPSETTING BLOWS --- 2 OR 3 ZONES

rule(upsetting,no_of_blows) :- u(1,U),
material(M,N),
material(M,N,_,_,U1),
U <= U1,
shape_dimensions(Z1,_,D1,_) ,
shape_dimensions(Z2,_,D2,_,)
(abs(D1-D2) / D1 < 0.50, assert(recommended_upsetting_blows(1)) , !.

rule(upsetting,no_of_blows) :- u(2,U),
material(M,N),
material(M,N,_,_,U1),
U <= U1,
shape_dimensions(Z1,_,D1,_) ,
shape_dimensions(Z2,_,D2,_,)
(abs(D1-D2) / D1 < 0.50, assert(recommended_upsetting_blows(2)) , !.

rule(upsetting,no_of_blows) :- u(3,U),
material(M,N),
material(M,N,_,_,U1),
U <= U1,
assert(recommended_upsetting_blows(3)) , !.

rule(upsetting,no_of_blows) :- u(4,U),
material(M,N),
material(M,N,_,_,U1),
U <= U1,
assert(recommended_upsetting_blows(4)) , !.

rule(upsetting,no_of_blows) :-
assert(recommended_upsetting_blows(more_than_four)) , !.

Figure A-3. Sample FORMEX Rules for Upsetting
to each other. The five rules in Figure A-3 are tried one by one to find the number of upsetting blows. The dimensions of intermediate upsets for different numbers of blows are computed by other rules. The fact $u(N,U)$ stores the length of unsupported workpiece ($U$) for ($N$) upsetting blows (section 4.3.8). $U$ is compared with the allowable unsupported workpiece length (Figure A-3). Another dimension that is checked in these rules is the difference in the diameters of the zones to be upset. If this difference is high, even the unsupported length is acceptable, separate upsetting blows are required for each zone.

When the first four rules in Figure A-3 fail, the fifth rule is always satisfied. This rule records that the number of required blows for upsetting is more than four. In this case, FORMEX will display a warning message as discussed above. More than four upsetting blows usually are not practical in cold forming.

After the number of upsetting blows is established, FORMEX displays the result as the "recommended number of upsetting blows". At this point, the designer may override the decision of FORMEX and may enter his choice of upsetting blows (sections 4.3.8, 5.3, 5.5). The final dimensions of the upsetting preforms are designed after this.