

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

**Bell & Howell Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA**

UMI[®]
800-521-0600

UNIVERSITY OF ALBERTA

**HYBRID REASONING METHODS FOR
INTELLIGENT OPERATION SUPPORT SYSTEMS**

by

QIJUN XIA ©

A thesis submitted to the Faculty of Graduate Studies and Research in
partial fulfillment of the requirements for
the degree of Doctor of Philosophy

in

Process Control

Department of Chemical and Materials Engineering

Edmonton, Alberta

Fall 1999



National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-46948-4

Canada

UNIVERSITY OF ALBERTA

LIBRARY RELEASE FORM

Name of Author: Qijun Xia
Title of Thesis: Hybrid Reasoning Methods for Intelligent
Operation Support Systems
Degree: Doctor of Philosophy
Year this Degree Granted: 1999

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly, or scientific research purpose only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as hereinbefore provided, neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.

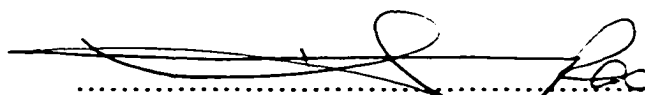


#412, 125 Spruce Street
Fort McMurray, Alberta
Canada T9K 1E2

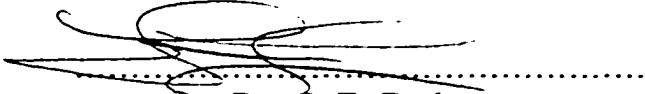
Dated May 25, 1999

UNIVERSITY OF ALBERTA
FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommended to the FACULTY OF GRADUATE STUDIES AND RESEARCH for acceptance, a thesis entitled HYBRID REASONING METHODS FOR INTELLIGENT OPERATION SUPPORT SYSTEMS submitted by QIJUN XIA in partial fulfillment of the requirements for the degree of DOCTOR OF PHILOSOPHY in PROCESS CONTROL.



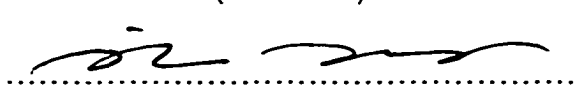
Dr. M. Rao
(Thesis Supervisor)



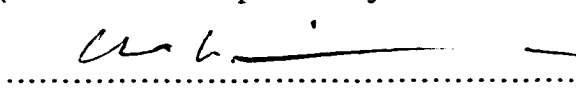
Dr. J. F. Forbes
(Member of Supervisory Committee)



Dr. P. A. J. Mees
(Member)



Dr. M. Q. Meng
(Member of Supervisory Committee)



Dr. C. W. de Silva
(External Examiner)

Dated May 21, 1999

**To my parents
who have provided me with the foundation**

**To my wife
who has shared my ups and downs**

**To my daughter
who brings joy to our world.**

ABSTRACT

Process operations are time critical situations. Faced with vast amount of process data, human operators may not be able to contribute a timely and effective solution. The industry requires new technologies to reduce the cognitive load placed on operators in nonroutine operations. This thesis aims at developing artificial intelligence solutions to the operation support problems.

A multi-dimensional problem solving model is developed following the analysis of the recognition behaviour of human operators and the characteristics of the process operations. A multilayer modularity architecture is proposed to achieve the decomposition and coordination of system functions, production processes and the reasoning methods.

The thesis focuses on the integration of various knowledge representations and reasoning methods. A hybrid reasoning method, which employs the case-based reasoning (CBR) as the principal reasoning paradigm and the other methods as the supplemental paradigms, is developed. A dynamic case-based reasoning (DCBR) method is developed to extend the CBR to dynamic problems. The DCBR utilizes new indexing mechanisms to incorporate system dynamics, and to improve problem solving coverage and flexibility. It allows easy integration of various reasoning paradigms into one environment. The model-based reasoning method is integrated with the DCBR for case

adaptations and novel problem solving. A principal component analysis (PCA) method and a Kalman filter-based algorithm are developed and integrated with the DCBR for problem feature interpretation and abnormal condition identification. By compensating the limitations of the individual reasoning methods, the best system performance can be achieved. To enhance the intelligent operation support system (IOSS) with the capability of handling multimedia information, an intelligent hypermedia on-line manual is developed as the external knowledge base and multimedia operator interface of the IOSS.

An actuator and sensor design algorithm is developed for operation support systems based on fault distances and objective trees that represent the instrumentation requirement of the IOSS. This algorithm selects an optimal set of actuators and sensors that ensures good system performance and the least hardware cost.

The methods developed in this thesis improve the problem coverage, problem solving efficiency, knowledge representation and operator acceptability of the IOSS. A prototype IOSS has been developed and implemented on a bleached chemi-thermo-mechanical pulp plant with satisfactory results.

ACKNOWLEDGEMENT

I wish to express my sincerely gratitude to my supervisor, Dr. M. Rao, for his enthusiastic supervision, help and encouragement during the course of my PhD program. I am also thankful to my supervisory committee members, Dr. J. F. Forbes and Dr. M. Meng, for their useful guidance. Thanks are also due to Dr. C. de Silva of the University of British Columbia and Dr. P. A. J. Mees for their careful reviews of the thesis.

I would like to extend my thanks to my colleagues in the Intelligence Engineering Laboratory for their assistance and stimulating discussions.

Financial and technical supports from the Natural Science and Engineering Research Council of Canada (NSERC), the Canada-Alberta Partnership Program in Forestry, Slave Lake Pulp Corporation, MoDo Chemetics, and Perde Enterprise are gratefully acknowledged. I would also like to express my gratitude to the financial support from the University of Alberta in the form of the J Gordin Kaplan Graduate Student Award and the Andrew Memorial Graduate Prize, and from the Pulp and Paper Research Institute of Canada (Paprican) in the form of Du Pond/Paprican Scholarship.

TABLE OF CONTENTS

1	INTRODUCTION	1
1.1	Introduction to Operation Support Systems	2
1.2	Introduction to Reasoning Methods	5
1.3	Contributions of this Thesis	8
1.3.1	Contributions to operation support system theory	8
1.3.2	Contributions to intelligent systems	9
1.3.3	Contributions to industrial applications	10
1.4	Organization of the Thesis	10
2	KNOWLEDGE ARCHITECTURE AND SYSTEM DESIGN	12
2.1	Introduction	13
2.2	Problem Formulation	13
2.2.1	Functional requirements	15
2.2.2	Desired features	16
2.3	Recognition Behavior of Human Expert	18
2.3.1	Reasoning with episodic experience	18
2.3.2	Navigation among various methods	20
2.3.3	Learning from operations	21
2.4	Multidimensional Problem Solving Model	23
2.4.1	Functionality dimension	24
2.4.2	Process dimension	27
2.4.3	Methodology dimension	28
2.5	Multilayer Modularity Design	29
2.6	Conclusions	33

3	ACTUATOR AND SENSOR DESIGN FOR OPERATION SUPPORT SYSTEMS	34
3.1	Introduction	35
3.2	Problem Formulation	36
3.3	Overview of the Bleach Plant	38
3.4	Sensor Design for Fault Diagnosis	40
3.5	Actuator and Sensor Design for Corrective Operations	50
3.6	Case Study in the Bleach Plant	55
3.7	Conclusions	62
4	DYNAMIC CASE-BASED REASONING METHOD	63
4.1	Introduction	64
4.2	Introduction to the CBR Approach	65
4.3	Indexing Feature Interpretation	68
4.3.1	Static index features	72
4.3.2	Dynamic index features	73
4.3.3	Composite index features	76
4.4	Abnormal Symptom-Based Indexing Path	77
4.4.1	Time-tagged indexes	77
4.4.2	Indexing mechanism	79
4.4.3	Case retrieval	84
4.4.4	Remembrance factor for case memory separation	86
4.5	Problem Description-Based Indexing Path	87
4.6	Operational Problem Solving	91
4.6.1	Problem solving with perfectly matching cases	92
4.6.2	Problem solving with a partially matching case	93
4.6.3	Problem solving by case mirroring	96
4.7	Conclusions	98

5	INTEGRATION OF MULTI-VARIATE STATISTICAL CONTROL WITH CASE-BASED REASONING	99
5.1	Introduction	100
5.2	Problem Formulation	101
5.2.1	Abnormal operating conditions	101
5.2.2	Introduction to PCA method	103
5.2.3	Nonlinear principal analysis methods	104
5.3	PCA for Case Index Feature Interpretation	107
5.3.1	Squared prediction errors (SPE)	108
5.3.2	Principal scores	109
5.3.3	Accumulated principal scores	110
5.3.4	Principal loading vectors	111
5.4	Case Similarity Evaluation Using PCA	111
5.4.1	Case similarity evaluation using SPE and scores	112
5.4.2	Case similarity evaluation using accumulated scores	115
5.4.3	Case similarity evaluation using principal loading	116
5.5	Case-Based Reasoning with PCA	117
5.6	Conclusions	119
6	KALMAN FILTER-BASED FAULT DETECTION AND FAULT-TOLERANT CONTROL	120
6.1	Introduction	121
6.2	Process Specification	122
6.3	Fault Detection and Reorganization	126
6.3.1	Scheme 1--single-failure case	129
6.3.2	Scheme 2--multiple-failure case	132
6.3.3	Scheme 3--reduced-order technique	134
6.4	Fault-Tolerant Control of the Headbox	137
6.5	Conclusions	144

7	INTEGRATION OF INTELLIGENT OPERATION SUPPORT SYSTEM WITH MULTIMEDIA TECHNOLOGY	145
7.1	Introduction	146
7.2	Problem Formulation	148
7.3	IHOM System Structure	149
7.3.1	Integration model	149
7.3.2	Software environment	152
7.4	IHOM Application in a BCTMP Plant	155
7.4.1	Manual structure	155
7.4.2	Searching mechanism	156
7.4.3	Computations and real time data access	159
7.4.4	Integration with IOSS	160
7.5	Conclusions	162
8	IOSS IMPLEMENTATION ON A BCTMP PROCESS	163
8.1	Introduction	164
8.2	Description of the BCTMP Process	165
8.3	Introduction to MOPS	169
8.4	IOSS Implementation	170
8.4.1	Introduction to Meta-COOP	171
8.4.2	Hardware layout	171
8.4.3	Interface with MOPS	173
8.4.4	Implementation results	174
8.5	Conclusions	182
9	CONCLUSIONS AND RECOMMENDATIONS	183
9.1	Conclusions	183
9.2	Recommendations	186
	BIBLIOGRAPHY	187

LIST OF TABLES

Table 8.1 Causes of P2 low brightness

179

LIST OF FIGURES

Fig. 2.1	Information and knowledge integration in IOSS	14
Fig. 2.2	Three-dimensional problem solving model.	24
Fig. 2.3	Possible suboptimal operations in abnormal situations	25
Fig. 2.4	Detail reasoning procedure	26
Fig. 2.5	Function dimension: functional decomposition	27
Fig. 2.6	Multilayer architecture of IOSS	31
Fig. 3.1	Overview of the bleach plant	39
Fig. 3.2	Flow chart of P-1 bleach tower	40
Fig. 3.3	Causal relation between F and Y	45
Fig. 3.4	Possible plant states in abnormal situations	51
Fig. 3.5	Control structure of the bleach plant (part)	53
Fig. 3.6	Operation tree for actuator and sensor placement	54
Fig. 3.7	Recursive sensor and actuator design	57
Fig. 3.8	Graph for fault-symptom relations	58
Fig. 3.9	Operation tree for actuator and sensor placement	61
Fig. 3.10	Flow chart of P-1 bleach tower	61
Fig. 4.1	Principles of case-based reasoning systems	66
Fig. 4.2	Interpretation of static features	73
Fig. 4.3	Evolution of a process variable	74
Fig. 4.3	Evolution of a process variable	74
Fig. 4.5	Symptoms of a fault changing with time	78

Fig. 4.6	Quality variables affected by operating variables	89
Fig. 4.7	Classification of consequences	90
Fig. 4.8	Structure of a dynamic case-based reasoning system	91
Fig. 5.1	Autoassociative neural network for NLPCA	105
Fig. 5.2	Integration of PCA with CBR	107
Fig. 5.3	SPE versus score plot for performance monitoring	109
Fig. 5.4	Representation space for various operating conditions	112
Fig. 5.5	Case memory in hierarchical structure	118
Fig. 6.1	Schematic diagram of the pressurized headbox	122
Fig. 6.2	Block diagram of pressurized headbox	124
Fig. 6.3	Principle diagram of the fault-tolerant control	128
Fig. 6.4	Response of fault-tolerant control in total head sensor failure	142
Fig. 6.5	Response of conventional control in total head sensor failure	142
Fig. 6.6	Response of fault-tolerant control in pulp stock level sensor failure	142
Fig. 6.5	Response of conventional control in pulp stock level failure.	143
Fig. 6.6	Response of fault-tolerant control in air pressure sensor failure.	143
Fig. 6.6	Response of conventional control in air pressure sensor failure	143
Fig. 7.1	Integration of multimedia system with intelligent systems	150
Fig. 7.2	Client – server architecture	153
Fig. 7.3	The structure of IHOM for a BCTMP plant	156
Fig. 7.4	The table of the contents of the IHOM	157
Fig. 7.5	Topic of the principle of screening	157
Fig. 7.6	The process flowchart of the BCTMP plant	158
Fig. 7.7	Small window shows the referenced picture	159
Fig. 7.8	Primary screens reject rate calculation	160
Fig. 7.9	IHOM integration with IOSS	161

Fig. 7.10	Operation support for P2 tower level problem	161
Fig. 8.1	Overview of the BCTMP process	165
Fig. 8.2	MOPS data flow	169
Fig. 8.3	Hardware layout and data flow	172
Fig. 8.4	Principle diagram of IOSS	175
Fig. 8.5	Illustration of the Matrix Simulator	176
Fig. 8.6	Matrix Simulator user display	176
Fig. 8.7	Displays for the Summery Pages	180
Fig. 8.8	Operator interface for problem solution	181

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
ASSO	Automatic Start-up and Shutdown Operation
BCTMP	Bleached Chemical Thermo-Mechanical Pulping
CBR	Case-Based Reasoning
DCBA	Dynamic Case-Based Reasoning
DCS	Distributed Control Systems
IHOM	Intelligent Hypermedia On-Line Manual
IOSS	Intelligent Operation Support System
MBR	Model-Based Reasoning
MIS	Management Information System
MOPS	Mill Wide Optimization System
MSPC	Multivariate Statistical Process Control
NLPCA	Nonlinear Principal Component Analysis
PCA	Principal Component Analysis
PLC	Programmable Logic Controller
RBR	Rule-Based Reasoning
SPC	Statistical Process Control
SPE	Squared Prediction Error
SQC	Statistical Quality Control

Chapter 1

INTRODUCTION

This thesis focuses on the development and implementation of new methods for intelligent operation support systems in process industry. It investigates the operation support problems from the operator's recognition behavior and process operation characterization. The thesis emphasizes on the integration of various knowledge representations and reasoning methods to provide improved reasoning efficiency and problem solving flexibility. This chapter provides a brief introduction to the operation support systems. It compares the performances of various reasoning methods, and analyzes the limitations of the existing methods. The main contributions of the thesis are summarized.

1.1 Introduction to Operation Support Systems

Computer systems and information technology have been extensively applied in industrial production processes. This trend is motivated by ever increasing requirements for better quality, higher production profits, safer operation and stringent environment regulation. Distributed control systems (DCS) and management information systems (MIS) have been implemented in most production companies. The main function of a DCS is to handle normal disturbances and maintain key process parameters in prespecified local optimal levels (Murdock and Hayes-Roth, 1991). A MIS stores and manipulates historical process data and management information. It presents these data to plant personnel in an easy-to-understand way for the purpose of production management, scheduling and operations (Tien and Croke, 1990). However, DCS and MIS are designed mainly for normal and routine operations. They have little functionality for abnormal operating conditions. In such situations as the process and equipment failures, human operators have to rely on their own experience to find out what is the problem and how to handle the problem. Faced with vast amount of raw process data, they may find it difficult to contribute a timely and effective solution. The industry requires new technologies to reduce the cognitive load placed on operators in nonroutine and abnormal operations by providing them with decision making support.

Process operations are knowledge-intensive tasks. Numerical computation based control technologies are not suitable to deal with the problems that require personal expertise and heuristics. Because of their capability of capturing and utilizing broad sources of knowledge (Dvorak and Kuipers, 1991), intelligent systems are the alternative to the conventional control technologies for solving such kind of problems.

There have been considerable research in the operation support related fields, such as fault diagnosis (Grantham and Ungar, 1990; Lapointe et al., 1989; Rich and Venkatasubramanian, 1987), real-time advisory control systems (Clancy et al., 1992), and expert systems for complex operations (de Silva and Wickramarachchi, 1998; Rao and Corbin, 1992; Rao et al., 1994a; Sachs, 1986; Saelid et al., 1989). However, unlike

computer integrated manufacturing systems (CIMS) (Clemons et al., 1992), the concept and scope of the process operation support systems have not been clearly defined. Many important problems remain unsolved. This thesis was intended to address these problems, and develop new methods for intelligent operation support systems (IOSS) design and application.

Artificial intelligence was originated from the effort to mimic human being's problem solving. A good understanding of human operator's cognitive behaviour may affect every aspect of operation support system design, especially in knowledge representation, knowledge acquisition, and reasoning mechanism. An intelligent operation support system that incorporates human operator's cognitive behaviour often has better user acceptance and higher problem solving efficiency. However, the research on this problem was limited (Leitch and Stefanini, 1989; Rasmussen, 1993). Most intelligent systems did not consider this issue. Research is required, for the purpose of system design, to investigate human operator's problem solving and find out the following results:

- the reasoning paradigms a human operator applies;
- the knowledge he/she possesses;
- the procedures he/she uses in problem solving;
- the relationship he/she has with other human operators and machines;
- the way he/she learns from problem solving.

Process operation support is a very complicated task as a result of the complex industrial processes and process operations. The systematic design of operation support system requires a problem solving model to formulate the philosophy and methodology of reasoning and computation processes for the purpose of efficient problem solving. The model guides all stages of system development. Research has been done in a few areas, such as the operator function model for space mission and reference model for intelligent control (Jones et al., 1995; Huang, 1996). They did not consider some important issues that are special in the process operation support, such as the multiple reasoning paradigms. The problem solving model for the IOSS should incorporate the

characterizations of the process, process operations, and the understanding about the human operator's recognition behavior.

One of the most important research topics in the intelligent operation support system is the reasoning mechanism. Process operations are time critical and require diverse process knowledge and experience. The IOSS requires a reasoning method that is highly efficient, is able to utilize all types of knowledge available in process operations, and is consistent with operator's problem solving. The next section will review the reasoning methods.

The prerequisite of designing a successful intelligent operation support systems is the adequate number of sensors and actuators. Sensors provide process information for problem identification and operation state evaluation. Actuators, together with sensors, are required for process control in normal and abnormal situations. Theoretically, larger number of sensors and actuators implies robustness in problem identification and flexibility in system reorganization. However, the cost consideration limits the number of sensors and actuators installed in production processes. Extensive cost and performance analysis must be performed to identify the best set of sensors and actuators. Past research on the optimal placement of sensors and actuators was mainly on controllability, control dynamics identification and large space structure control application (de Silva, 1989; DeLorenzo, 1990). There were also considerable research on the fault diagnosability by using fault tree and entropy-based methods (Ishida et al., 1987; Seong et al., 1994). Further research is required on the sensor and actuator design that directly meet the needs of problem identification, system state evaluation and system reorganization.

An important problem in the IOSS design is operator interface. Multimedia system technology has gained increasing attention in process industry. The IOSS needs to have ability to represent and reason with multimedia information. A solution to this problem is to integrate intelligent operation support systems with multimedia systems (Maybury, 1992; Shu et al., 1995). Commercial multimedia systems do not have the integration capability. A new integration model is required to develop an intelligent hypermedia manual that provides:

- the hypermedia manual serving as an extended multimedia operator interface and an external hypermedia knowledge base.
- a hypermedia on-line manual that has reasoning ability and can be run independently, if necessary, to replace traditional paper manual.

This thesis was intended to address these problems.

1.2 Introduction to Reasoning Methods

There have been numerous methods available for fault diagnosis. These methods can be classified into three categories: rule-based reasoning (RBR) (Calandranis et al., 1990), model-based reasoning (MBR) (Davis, 1984), and case-based reasoning (CBR) (Kolodner et al., 1985; Bareis, 1989) techniques.

RBR systems are among the earliest to assist operators in fault diagnosis. In a RBR system, the basic unit of knowledge is a production rule. A rule consists of a premise (condition) and an action (result). The problem solving knowledge is represented by a set of production rules. A RBR system searches through the knowledge base and activates the rules whose premises are satisfied. To capture uncertain and fuzzy knowledge, fuzzy logic (de Silva, 1995; Dubois and Parade 1980; Zadeh, 1965; 1978) has been extensively applied in RBR. There have been considerable number of successful industrial applications of RBR systems (Calandranis et al., 1990; Becraft et al., 1991).

RBR systems have the advantage of high reasoning efficiency, and easy knowledge representation and acquisition. Plant engineers and operators often express their operation knowledge using IF-THEN rules (production rules). However, because of the crispy and unstructured knowledge base, RBR are usually not suitable to solve large problem. For a typical engineering application, the number of rules could be up to hundreds. It is very difficult to make the knowledge base complete and consistent. Since the rule sets are strongly process specific, RBR systems have poor adaptability and portability. A RBR system can only handle “known” problems.

MBR is an important approach that uses process models for problem solving. A process fault, can be defined as a discrepancy of the process behavior from its normal behavior described by process models (Rasmussen, 1993). The fault diagnostic problem is to explain the discrepancy. If a hypothesis can properly explain the discrepancy, it is considered as a solution to the fault diagnostic problem.

Process models generally consist of the following knowledge:

- Structure knowledge describes the structural construction of the process by physical devices. It is used to construct process behavior from the behavior of individual devices.**
- Behavior knowledge describes the expected behavior of individual process devices in various conditions.**
- Functional knowledge describes the designated functional purpose of devices. Even if a device meets its expected behavior, it is considered ill-functioning if it does not fulfill its designed functional purpose.**

The types of models used in MBR vary widely, including quantitative models, semiquantitative models, signed directed graphs, causal models and fuzzy relationship (Dvorak and Kuipers, 1991). Therefore, a large variety of reasoning approaches belong to this category, such as fault tree (Lapp and Powers, 1977; Kramer and Palowitch, 1987), signed directed graph (Iri et al., 1979; Kokawa et al., 1983), qualitative simulation (Kuipers, 1986), and even observer based methods (Iserman, 1984; Xia and Rao, 1992).

The advantages of the MBR method come from the fact that it employs structured and deep process knowledge. The problem solving results are usually more precise and reliable. Since the knowledge applied is less process dependent, MBR systems have good generality and portability. Provided sufficient knowledge, a MBR system is able to solve novel problems that have never occurred before. However, it is usually difficult to obtain the process models that describe structural, functional and behavioral knowledge. Problem solving in MBR systems may require complicated computation and inference reasoning. Excessive time may be required to find a solution.

CBR is the result of the attempt on modeling the human recognition process, including memory, learning and problem solving, with computer models. The research on CBR can date back to 1975 (Schank, 1975;1981). The theory about the episodic information plays an important role in the invention of CBR (Slade, 1991). A few survey papers gave good overview of the research on CBR (Slade, 1991; Bareiss, 1989).

CBR is a reasoning approach that lies between RBR and MBR in the sense of the level of knowledge employed. A CBR system includes a case memory to simulate the episodic human memory. The basic unit of knowledge is a memory episode, or a case. Each case is the solution to a specific problem. These cases are stored in a structure with rich index. Faced with a new problem, the system retrieves similar cases from the case memory, and adapts solutions contained in the cases to suit the new problem.

CBR has high problem solving efficiency. It solves a problem by directly applying the knowledge of the previously solved problems rather than from scratch. Much less computational and reasoning effort is required compared to MBR. Since the episodic knowledge is very similar to the plant engineers and operators' experience, a CBR system has the advantage of easy knowledge acquisition and good operator acceptability. However, the CBR system is usually not able to solve novel problems. The existing CBR techniques are applicable only to static problems.

There are a few approaches that do not fit into the above categories, such as neural networks (Hoskins et al., 1991; Venkatasubramanian and Chan, 1989) and multivariate statistical process control (MSPC) approaches (Martin and Morris, 1996). These approaches are based on the assumptions that the system behaviors, either normal or abnormal, are all captured in the data. By characterizing these data, decision can be made on the status of the production process.

It is seen from the comparison of various reasoning methods that none of them is superior to others in a general sense. Each individual reasoning method has its own advantages and limitations. To compensate for the limitations of the individual reasoning methods and to achieve the best system performance, this thesis was intended to develop a method that is able to integrate various reasoning approaches.

1.3 Contributions of This Thesis

The main contribution of this thesis is a group of new methods for the design of intelligent operation support systems. These methods provide efficient problem solving, easy knowledge acquisition, and good operator acceptability. The results obtained in this research are a step advance to the success of intelligent operation support systems in process industry.

1.3.1 Contributions to operation support system theory

The thesis investigates the operation support problems from the roots of operator's recognition behavior and process operation characterization. The following new results provide a methodology for systematic design of high performance operation support systems:

- The analyses of human recognition behavior in problem solving reveal many psychological issues that are significant in process operations. The new problem solving model, which incorporates not only the human recognition behaviors but also the system requirements, is applied as the basis of operation support system design.
- The multilayer modularity architecture achieves the decomposition and coordination of function, process and methodology, as well as the separation of general and specific knowledge. The full integration of the intelligent operation support systems with the existing plant facilities helps the company to protect the previous investment and makes the intelligent operation support system more useful and powerful.
- The new hybrid reasoning method, the dynamic case-based reasoning (DCBR), uses case-based reasoning as the principal reasoning paradigm and other approaches as the supplemental reasoning paradigms. It is superior in terms of the problem coverage, problem solving efficiency, knowledge representation, and user acceptability.

- The method of actuator and sensor design is based on the fault distance and objective tree that represent the instrumentation requirement of operation support systems. The design method produces the scheme of actuator and sensor locations that ensures good system performance and least hardware investment.
- The intelligent hypermedia on-line manual provides the operation support system with capability of handling multimedia information. It greatly enhances the operator interface and can be used as an external knowledge base. As a result, the IOSS will have better user acceptance and higher efficiency.

1.3.2 Contributions to intelligent systems

This research focuses on the integration of various knowledge representations and reasoning methods to provide improved reasoning efficiency and problem solving flexibility. The dynamic case based reasoning (DCBR) method is developed to achieve this purpose:

- The dynamic case-based reasoning (DCBR) method extends the traditional case based reasoning (CBR) methods with the capability of solving dynamic problems. The DCBR utilizes a new indexing mechanism to represent time-tagged features and multiple indexing paths. The time-tagged features provide a feasible way to incorporate system dynamics in CBR, and thus lead to more accurate and timely problem solving. The multiple indexing paths significantly improve problem solving coverage and adaptation capability of CBR. A new learning mechanism is proposed to constantly refresh memory by “forgetting” old cases and “remembering” new cases.
- The DCBR has a structure that can integrate various reasoning paradigms (CBR, MBR, RBR) into one environment. By complementing each other in problem solving, the disadvantages of the individual reasoning methods are avoided. As a result, better system performance can be achieved.
- The integration of the DCBR with the principal component analysis (PCA) method enables the DCBR to employ process variable correlation and context-related

information extracted by PCA for case retrieval and evaluation. It avoids the low fault identifiability of individual PCA approaches, and improves the problem solving accuracy and efficiency of the DCBR.

- The DCBR is integrated with a Kalman filter-based fault diagnosis and fault-tolerant control algorithm for fault diagnosis and control system reorganization. It is illustrated that a numerical model-based method can be easily fit in the DCBR environment as a subsystem or as a data processing package.
- The integration model of intelligent system technology with multimedia technology enhances the intelligent system with the capability of multimedia information representation.

1.3.3 Contributions to industrial applications

The new methods developed for operation support systems are feasible in industrial application. A prototype intelligent operation support system has been developed for a bleached chemical thermo-mechanical pulping (BCTMP) plant. The system has been implemented in the plant computer systems and tested with real-time plant operations. It is integrated with the existing plant automation systems, including a DCS (Fisher Provox 2000) and a management information system (MOPS). The results were very encouraging. With the help of the operation support system, operators were able to identify the undesirable conditions earlier and reduce off-spec product and production cost.

1.4 Organization of the Thesis

The thesis is organized as follows. Chapter 2 is devoted to the knowledge architecture and system design for the intelligent operation support systems. A multidimensional problem solving model is developed as the guide for the intelligent operation support system design.

The design approach for sensor and actuator placement in process operation support systems is developed in Chapter 3. An example in the bleach plant of a pulp mill is illustrated.

In Chapter 4, a dynamic case-based reasoning (DCBR) method, which extends the case-based reasoning approaches to dynamic problems, is developed. DCBR has the ability to integrate with other computational and reasoning methods, including MBR and RBR. Chapter 5 extends the DCBR to the multivariate statistic process control method by integrating with the principal component analysis approach. A model-based fault detection and fault-tolerant control method is developed in Chapter 6, which is also directly integrated with the DCBR.

A new integration model of intelligent system with multimedia technology is proposed in Chapter 7. An intelligent hypermedia on-line manual (IHOM) is developed as the external knowledge base and multimedia operator interface of the intelligent operation support system. Chapter 8 presents the implementation of the intelligent operation support system in a bleached chemical thermo-mechanical pulping plant. The result is very encouraging. Finally Chapter 9 presents overall conclusions and recommendations for future research.

Chapter 2

KNOWLEDGE ARCHITECTURE AND SYSTEM DESIGN†

This chapter presents an intelligent operation support system (IOSS) structure using rich knowledge representation and hybrid reasoning strategy. The functional requirements and desired features for IOSS are defined. The human operator's recognition behavior is analyzed. It is shown that a hybrid reasoning environment that combines case-based reasoning (CBR), model-based reasoning (MBR) and rule-based reasoning (RBR) is consistent with operator's problem solving. A multidimensional problem solving model is proposed to incorporate these requirements and human recognition behavior. The IOSS is designed by using the problem solving model as the guide.

† Some contents of the chapter have been published: Rao and Xia, 1994, "Integrated distributed intelligent system for on-line monitoring and control of pulp process", *Canadian Artificial Intelligence*, 33, 5-10.

The extended and modified version has been submitted for publication: Xia and Rao, 1999, "Knowledge architecture and system design for intelligent operation support systems", *Expert Systems with Applications*.

2.1 Introduction

Distributed control systems (DCS) and management information systems (MIS) have been extensively applied in modern process industry. The main function of DCS is to handle normal disturbances and maintain key process parameters in prespecified local optimal levels. MIS stores and manipulates historical process data and management information, and presents them in a more understandable format for the purpose of production management, scheduling and operations. Despite their great success, DCS and MIS have little function for abnormal and nonroutine operations. In such situations as process faults, human operators have to rely on their own experience to contribute a solution. Faced with complex processes and vast amount of data, they are under a tremendous working pressure (Hoskins and Himmelblau, 1988; Petti et al., 1990), which leads to “cognitive error” (Long, 1984). The process industry demands new computer integrated technologies to reduce operator’s working burden by providing operation support.

Process operations are knowledge-intensive tasks. Artificial intelligence (AI) has been considered promising to deal with problems that require personal expertise and heuristics (Dvorak and Kuipers, 1991). Considerable research has been done in various fields related to operation support, such as automatic trouble shooting (Grantham and Ungar, 1990), real-time advisory control systems (Clancy, 1992), and expert system for complex operations (de Silva and Wickramarachchi, 1998; Rao et al., 1994a; Sachs et al., 1986; Saelid et al., 1989). However, many problems remain unsolved. Taking a bleached chemical thermo-mechanical pulp plant as an example, this chapter systematically investigates the problems involved in process operation support including production requirements, human recognition behaviors and problem solving models. A feasible system architecture and a reasoning strategy are proposed.

2.2 Problem Formulation

Unlike computer integrated manufacturing systems (Clemons et al., 1992), the concept

and scope of IOSS have not been clearly defined. In verbose fashion, any system that assists operators in operation is an operation support system. Based on the investigation on industrial requirements, the definition of IOSS is formalized as follows:

IOSS is a mill-wide real-time intelligent system that integrates production information of all areas and from all resources. It manipulates the information and various types of knowledge to provide decision support to operators. IOSS assists operations in two different ways: (1) generating and directly implementing new operating conditions or corrective actions on control action devices, e.g., DCS; or (2) making feasible operation recommendations to operators who make final decision and take actions. The objectives of IOSS are to help operators to maximize production, to improve product quality, to reduce production costs, and to minimize environmental impacts through more consistent and timely operations.

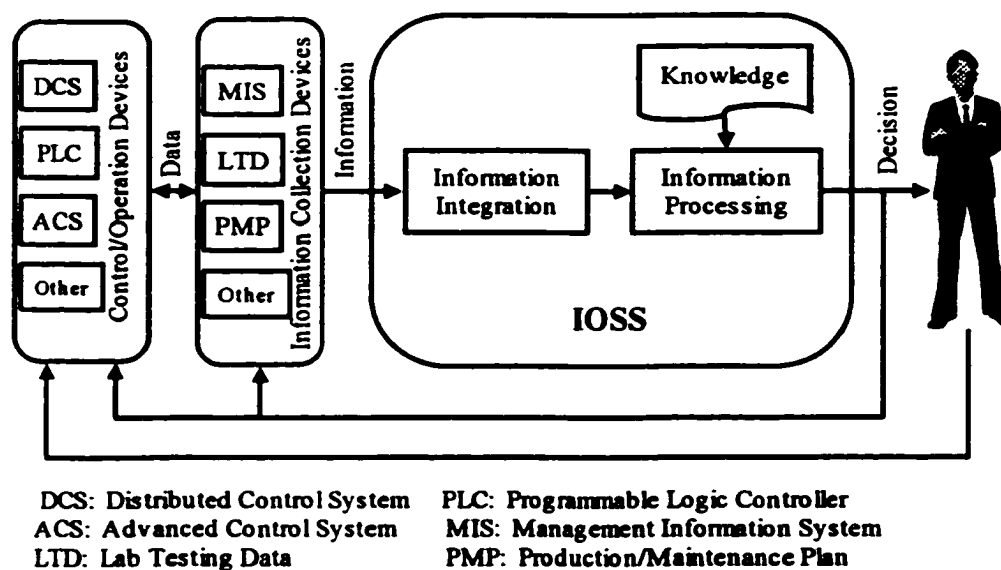


Fig. 2.1. Information and knowledge integration in IOSS.

The principle diagram of IOSS is shown in Fig. 2.1. IOSS emphasizes in information and knowledge integration. It considers a mill facility as either an information collection

device or a control/operation action device. From the former, it receives and integrates information. To the latter, or indirectly through operators, it issues control and operation actions. IOSS is a bridge between production process and operators through information and knowledge integration.

· The next section will discuss the factors that determine the system design and related technology development: the functional and featural requirements.

2.2.1 Functional requirements

An IOSS is intended to provide operation supports for the situations in which human operators feel difficult to contribute a timely and effective solution. Human operators perform well for routine operations, but not as good for nonroutine operations. An IOSS is designed to provide decision support for the following nonroutine operations:

- (1) *Product/grade transition*: Modern operation companies tend to produce product of multiple grades in small ordering volume to cope with ever increasing competition in the global market. In the investigated BCTMP mill, grade change occurs very often, sometimes, once a week. A new target pulp grade requires a new set of optimal operating conditions, such as temperature, chemical charging rate and refining energy. The IOSS can make plant more responsive to multiple grades, rates and products production by automatically generating new operation conditions.
- (2) *Production optimization/disturbance compensation*: In a production process, DCS and advanced control systems maintain process variables at constant levels. Most of small process disturbances can be compensated by DCS control loops. However, DCS is not efficient enough for some significant disturbances. In face of significant process disturbances and drifts, production optimization is required to obtain a set of new optimal conditions. For example, when the chip quality/mix changes, the settings for refiner energy and bleaching reactant must be properly adjusted to maintain a desired pulp quality and production profit. The production

optimization may include complex calculations and reasoning using mass and heat balance axioms, which can be better done by using computers.

(3) *Fault correction*: Process faults are defined as undesired situations or incidents that could lead to off-specification products. There are two types of process faults: device failure and operation fault:

- **Device failure**: A device is failed if it can not function properly. An obvious indication is the discrepancy between the presenting behavior and the designated or nominal behavior.
- **Operation fault**: It is referred to the improper operation actions taken by operators, such as improper setting of control mode and accidental opening of a bypass valve.

Fault diagnosis and correction are very difficult problems in process operation. Operators need to identify the fault, evaluate the severity and recoverability, and decide the redeem actions. Operation support is of great reward in preventing or reducing the loss caused by process faults.

Automatic start-up and shutdown operation (ASSO) is an important research topic in computer integrated system applications in process industrial (Rao et al., 1994a). It is an important component of IOSS. However, ASSO itself is a large research topic. The technology applied in ASSO is quite distinct. In most pulp companies, the automatic ASSO has been achieved, though not perfect, using sequential control by DCS. For this reason, ASSO is not considered as a component in IOSS.

2.2.2 Desired features

The weakness of past developed intelligent systems for decision support and fault diagnosis was addressed by many researchers (Rich and Venkatasubramanian, 1987; Lapointe et al., 1989). Certain system features were paid special attention in every specific application. Boy (1993) emphasized the coexistence of human operators with artificial intelligent agents on a cooperative base in space system design. He claimed that human operators tend to lose vigilance if machine automation level is too high. The

integration problem of classic techniques and AI methods was addressed by Murdock (1991) in the monitoring system design for semiconductor production. Lapointe and his coworkers (1989) proposed to divide knowledge base into the knowledge about plant and the knowledge necessary for problem solving. Open architecture and distributed knowledge organization were also considered critical in developing, operating and maintaining an intelligent system (Huang, 1996; Rao et al., 1993).

After carefully investigating the requirements of pulp production, the following features are considered critical to the IOSS:

- *Information/knowledge integrated environment:* Integration with existing plant facilities is an essential requirement to all high-tech applications. DCS, MIS, PLC and other systems such as SPC and SQC have been performing irreplaceable roles in modern pulp production. A standalone intelligent system cannot function efficiently. The IOSS needs to be integrated and embedded into plant production and automation network and to take full advantage of the existing facilities.
- *Operator's involvement in the decision making:* Real-world applications of automation system bring a dual problem: without enough automation, an excessive workload appears; Conversely, too much automation may “push” operators “far” away from the production process and tending to lose their control of the production. Keeping operators in decision making (Rao and Qiu, 1993) is a critical issue in designing the IOSS.
- *Timely decision:* Process operation is a time critical situation. As a real time application, the IOSS must provide timely and knowledgeable decision whenever a complex situation occurs. Reasoning efficiency is important to the system.
- *Process generality:* Process generality is necessary to make IOSS portable and enable easy accommodation of process configuration changes. It is also important to make users easy to maintain and upgrade the system.
- *Full use of operation knowledge:* Knowledge acquisition has long been identified as the bottleneck in developing intelligent systems. Knowledge in process

operation exists in various levels and types. To overcome the difficulty, the IOSS should be able to make use of knowledge of all levels and types.

- *Learning capability:* Because of the complexity of process operations, it is impossible to make the knowledge base complete. The IOSS is required to have capability to learn from problem solving in order to expand the knowledge base in the operation.
- *Operator acceptability:* Operators are the end user of the IOSS. Good acceptance by operators is crucial for the system to succeed (Rao and Xia, 1998). Besides functionality, easy to understand reasoning process and operator interface are important to encourage operators to use and maintain the system.

2.3 Recognition Behavior of Human Expert

In order to develop an efficient operation support system, we need to analyze the recognition behavior of human operator's problem solving in process operations. The characteristics of human operator's problem solving that may significantly affect the design of operation support system include: reasoning with episodic experience, navigation among various reasoning paradigms, and learning from operations.

2.3.1 Reasoning with episodic experience

Modern process operation involves knowledge from various areas. It is impossible for an individual plant operator to have all knowledge. Research has discovered that operators do not usually apply deep domain knowledge for problem solving. Instead, the principal knowledge they apply is experience, the explanations and solutions to the problems they encountered and solved in the past. A senior operator has encountered a lot of problems in his domain. He generalizes these problems and stores the knowledge in his memory for the future use. Just as Schank's episodic memory theory (Schank, 1981), this knowledge is stored as discrete episodes. Each episode is the explanation and solution to a single solved problem, for example, the situation of excessive bleaching chemical charge with

low product brightness. The memory of a human operator has a very complex storing and indexing structure. These episodes are related to each other according to sophisticated conceptual features and criteria. This complex structure enables the human operator to relate a new problem to one or more memory episodes.

A human operator monitors production from central control room by checking the states of important process variables through the consoles of distributed computer systems (DCS) and management information systems (MIS). He has a clear idea about what variables to monitor. The monitoring variables are either quality variables, such as brightness and freeness, or important operating variables, such as temperature and specific energy. He barely notices the actual values of these variables. What he concerns most is whether the variables are in normal range or not. Since industrial production is a dynamic process, an experienced operator pay attention also to trends, the evolution of the monitoring variables. Trend information is extremely important in dynamic processes since it provides additional information from the same set of data. With the trend information, the operator can foresee a problem before relevant variables actually go out of normal ranges. For example, if a brightness is half way to high and still increasing rapidly, the operator will be able to anticipate that the brightness will go high in certain amount of time. Proper actions need to be taken to prevent off-specification brightness.

If all monitoring variables are in normal ranges, the operator assumes that the production is in normal condition and no nonroutine operation needs to be taken. If one or more monitoring variables go out of normal ranges, a potential abnormal problem may have occurred. He has to explain the problem and contribute a solution to it, if necessary. The basic reasoning steps he takes are reminding, solving and learning. By giving a quick survey of the relevant process variables, he gets a rough picture of the production status. This picture reminds him of similar problems he encountered in the past. The explanation and solution to the reminded problems or the adapted version will serve as the solution to the new problem. He will remember what he learned from the problem solving.

An important characteristic of operator reasoning is the similarity reasoning. Industrial production processes are extremely complex. A lot of factors are involved in the

definition of a problem. A production process will not repeat a past situation exactly. The operator is not looking for an exact match between a new problem and a past solved problem. Instead, he is looking for the analogy or similarity. He has a perceptiveness to collect all the important facts and ignore the unimportant ones.

The reminding, solving and leaning mechanisms of a human operator are very sophisticated. For a single problem to be solved, he may remind himself a number of different past solved problems from different perspectives. These complex mechanisms enable the human operator to obtain a solution for a new problem from more than one past problems that are similar in one perspective or another.

2.3.2 Navigation among various methods

An important phenomenon in human operator's problem solving is continuing switching among various reasoning paradigms using different levels of his knowledge. Rasmussen (1993) emphasized the same phenomenon in diagnostic reasoning. The specific reasoning paradigm that an operator applies depends on his experience, his domain knowledge, problem solving objective and time limitation. For the problem, a different operator may reason in a different way. A senior operator, having experienced plenty of problems in his operation history, often reminds himself of similar past problems. An engineer, having more domain knowledge, tends to come up a solution from domain knowledge, including the structural and behavioral description of the production process. A novice operator, however, having neither much experience nor domain knowledge, may have to figure out a solution using very simple rules. For example, if pulp brightness is low, a novice operator might increase peroxide charge because peroxide is the dominating influential factor of pulp brightness in his knowledge. A solution so obtained may be wrong. A senior operator may find immediately from the low brightness and high peroxide residual that increasing temperature is the best solution. An engineer, knowing that the brightness is affected by incoming brightness, temperature, retention time, and peroxide/caustic/silicate/DTPA charge, will arrive the same conclusion from the careful analysis of domain knowledge. It is usually time consuming to solve operation problems using domain knowledge. That is why a senior operator may be able to solve a problem

more efficiently than a less experienced engineer. However, even an experienced operator may be confronted with a problem that he is not completely familiar. In this situation, he has to navigate among all types of knowledge, including domain knowledge, past solved problem and operation rules, to solve the problem.

A main purpose of operation support systems is to avoid production loss in abnormal situations. The first response of an operator to an emerging abnormality is to stabilize critical functions so as to prevent the accidental situation. This response is time critical. He is required to respond as quickly as possible even if the problem is not totally understood. In such situations, only simple operation rules count. After the critical functions have been stabilized, he may apply more time consuming methods, including domain knowledge, to find a solution of more precision.

2.3.3 Learning from operations

Active learning is one of the most important advantages of human operators over computer-based systems. The performance of most computer-based systems does not get improved in the course of running. In fact, it will be deteriorated as the plant and its environment drift from original settings. On the other hand, a human operator can quickly learn from process operations. He may not be knowledgeable when he is first assigned to a position. However, in a short period of time, he will quickly gain enough experience to handle most of operation problems. A basic sign of artificial intelligence is the learning ability. However, efficient learning in intelligent system design is still not totally solved. The understanding of the learning process of human operators can be applied as a guide to the operation support system design.

The updating of the memory content and structure is called learning. Operators learn mainly in three ways:

- *Learning from documented knowledge*: This learning process is mainly fulfilled in the pre-operation training stage for novice operators. Its purpose is to enable operators to handle the typical operation problems that either happen frequently or have significant impact on production. The knowledge about these problems is

usually in the form of well-formulated operation procedures. Operators are required to follow the procedures strictly whenever a problem occurs. But this learning process is not limited to pre-operation training stage. Most pulp companies maintain records of all incidents occurred in production. A diligent operator can get valuable knowledge from reviewing these records and talking to the personnel who made the records.

- *Learning from successful problem solving:* An operator gains most valuable knowledge from his own problem solving in operations. Confronted with a problem, he has to resort to all type of knowledge he has and be cooperated with other mill expert, if necessary. Trial-and-error method could be applied in order to get an effective solution. After the problem is satisfactorily solved, the knowledge about this problem will become an episode in the memory of all participants'. Next time the same problem occurs, every one of them will be able to contribute a timely solution. Extracting episodic knowledge from domain knowledge is a main part of this learning process. Domain knowledge plays a primary role in solving new problems. In the problem solving using domain knowledge, the operators gain a clear description of the problem, extract sufficient features and obtained a solution. This knowledge will be applied to create a new episode in his memory.
- *Learning from problem solving failure:* Failure-driven learning is another important characteristic of human operators. When confronted with a problem, an operator will come up a solution in the way he is most familiar with. This solution could either succeed or fail. If the solution is proven to be a failure, his knowledge needs to be updated. From the explanation to the failure, i.e. the discrepancy between the actual result and his expectation, he revises their knowledge.
- *Learning from forgetting:* Efficient forgetting is the basis of efficient memorizing. The active memory space of our brain is limited. Human being has ability to forget past events eventually. For example, we have no difficulty to recall something happened yesterday. However, it is almost impossible for us to recall something happened twenty years ago unless it had affected our life significantly. The same

rule applies to human operators in operations. They have a good memory of the events occurred recently, but not of those happened long time ago.

Another phenomenon that may affect system design is the interconnection between episode knowledge and domain knowledge. An operator has a certain amount of domain knowledge. It is discovered that the domain knowledge is not independent of episodic knowledge. They are highly interconnected. A problem specific episode is related to a certain part of domain knowledge the human operator possesses. This relation enables the operator to adapt the solution from a specific episode for solving new problems.

2.4 Multidimensional Problem Solving Model

The analysis of human operator's recognition in process operation directly affects the intelligent system design. This analysis needs to answer what activities operators carry out to solve problems and how they perform. Jones et al. (1995) proposed an operator function model for space mission operations. The model is a heterarchic-hierarchic network of nodes. The nodes, from top level to bottom level, are respectively the major operation functions, subfunctions, tasks and actions. This operator function model addresses the sequential operator operation activities and their hierarchical decomposition. Rasmussen (1993) discussed many important issues in diagnostic reasoning in various domains such as process control, maintenance and medicine. He presented the nature of diagnosis from various perspectives. A related research was done by Huang (1996). He proposed a multidimensional reference model for intelligent control. The limitation of the reference model is that it does not consider the fact that human operators are applying different reasoning mechanism facing with different situations.

For the purpose of system design, the human operator recognition analysis needs to develop a problem solving model that can be used as a working environment of IOSS. It is revealed that the problem solving model is better to be represented in a three-dimensional coordinate as in Fig. 2.2: functionality dimension, process dimension, and methodology dimension.

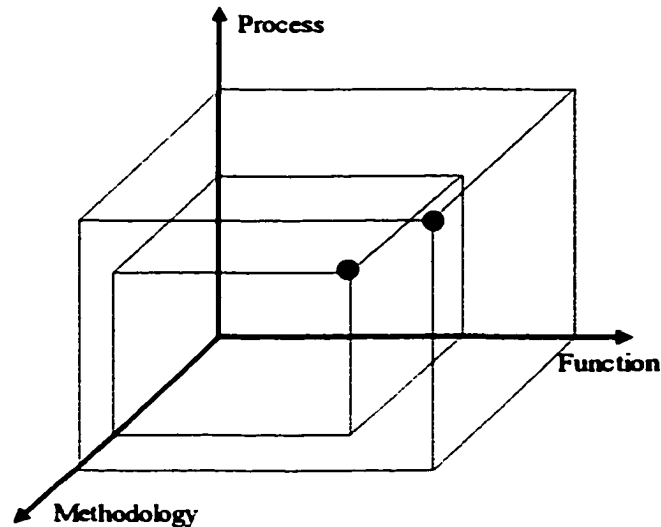


Fig. 2.2. Three-dimensional problem solving model.

2.4.1 Functionality dimension

Take the pulp production as an example. The goal of pulp production is to run the production process in optimal states. The optimality of production is measured by the following criteria:

- best use of wood fibers,
- optimized product quality,
- optimized production rate,
- minimized use of chemicals and energy,
- minimized environmental impact,
- safe operation (equipment, human).

In normal situations, the target state is optimal operation. However, in some abnormal situations, optimal operation may not be achievable. There is no choice but to accept degraded (suboptimal) operations. To an extreme, when the profitable production can not be maintained, the production process has to be shut down with the least production loss. Fig. 2.3 presents the potential suboptimal operations with deteriorated optimality top down.

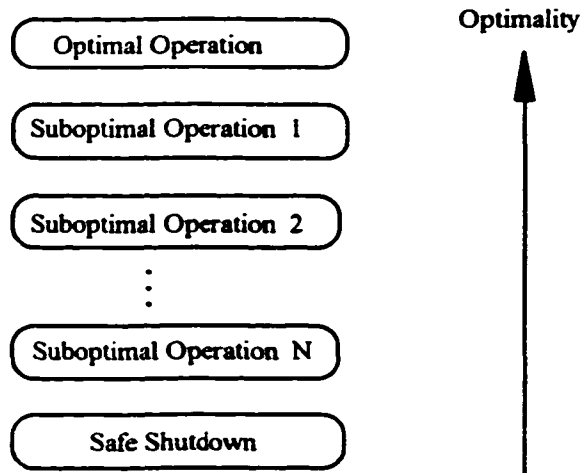


Fig. 2.3. Possible suboptimal operations in abnormal situations.

The objective of operation support is to find the achievable operation of best optimality with the available resources and the corresponding corrective actions. The problem solving goes through process monitoring, problem identification, and corrective action planning:

- **Process monitoring**
 - **Data interpretation:** to extract features of sensory data.
 - **Problem detection:** to detect the presence of problem.
- **Problem identification**
 - **Original cause isolation:** to locate the problem.
 - **Consequence analysis:** to predict the consequences.
- **Problem correction**
 - **Problem evaluation:** to find achievable operation.
 - **Corrective scheme planning:** to plan operation scheme.
 - **Corrective scheme verification:** to verify the scheme.

A more detail reasoning procedure is depicted in Fig. 2.4.

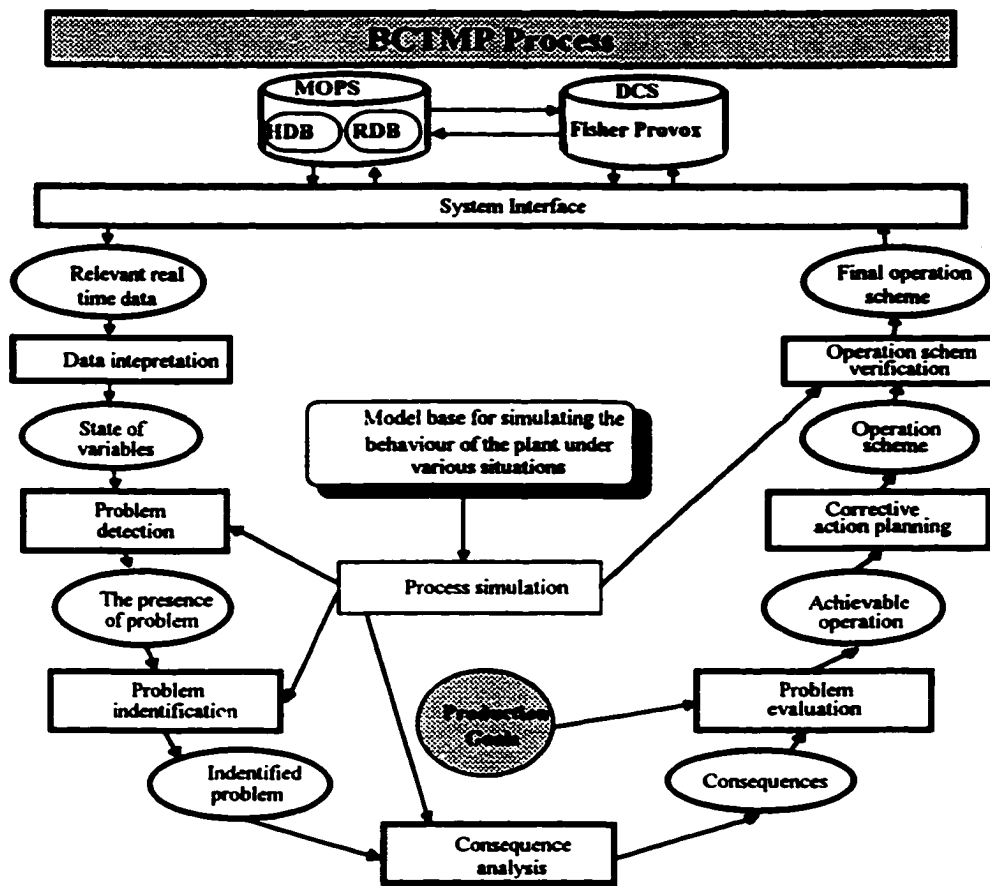


Fig. 2.4. Detail reasoning procedure.

The functionality dimension represents the continuous decomposition of reasoning functions into simple reasoning tasks in a hierarchical form, as similar to Jones's operator function model (Jones et al., 1995). An example is presented in Fig. 2.5. These functions, both in the same level or different level, are coordinated to solve an operation support problem.

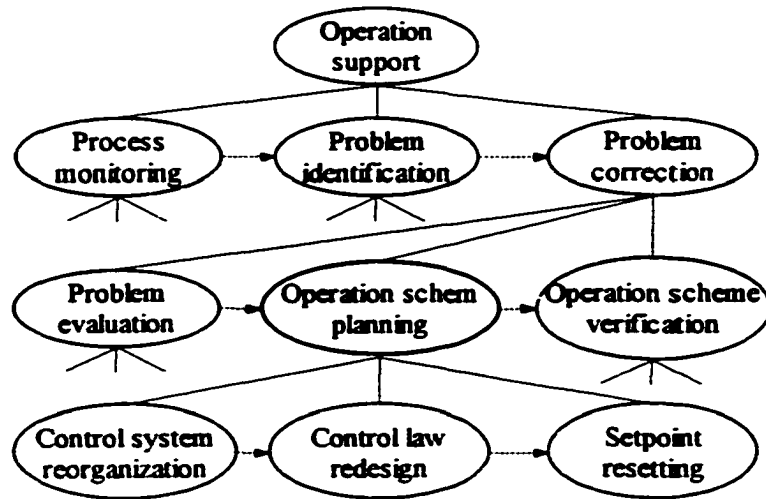


Fig. 2.5. Functionality dimension: functional decomposition.

2.4.2 Process dimension

The process dimension represents the decomposition of entire production process into a number of nodes at various levels. Pulp production is a long and complex process that consists of impregnation, refiner, bleach plant and drying. Human operators do not consider the production process as a whole in problem solving. Instead, they instinctually decompose the process. For example, when an abnormal situation occurs, a human operator first examines whether it is a bleaching problem or a refining problem. He then narrows down the fault to an individual component. Accordingly in the IOSS design, the production process should be properly decomposed into a number of subprocesses.

We decompose the entire pulp production process into six levels:

Level 1:	Mill level	Pulp production
Level 2:	Plant level	Refiner, bleach plant, etc.
Level 3:	Stage level	Interstage bleaching and washing, etc.
Level 4:	Key unit level	P1 bleaching tower, etc.
Level 5:	Control loop level	Peroxide charging, etc.
Level 6:	Component level	Pumps, valves, etc.

The process decomposition also makes open system structure possible. IOSS can be developed independently for individual nodes (usually at plant or stage level). These local systems are then integrated into a mill-wide system.

2.4.3 Methodology dimension

There are two aspects to be discussed in methodology dimension: knowledge types and reasoning paradigms.

From the process dependability, information necessary for operation support problem solving can be divided into two groups:

- **Generic knowledge:** This portion of knowledge describes the process independent problem solving procedures.
- **Process specific knowledge:** This portion of knowledge characterizes a specific production process.

From the difference in knowledge level, information included in problem solving can be classified into various types with different confidence:

- **Human operator's individual experience:** This is usually simple experience in production rule form.
- **Episodic human operator's experience:** This is the knowledge about the well defined past solved problems.
- **Domain knowledge:** The structural, behavioral and functional descriptions of pulp process belong to this category.
- **Process operation manual:** Technical documents and process operation manuals are the important sources of knowledge.

Different reasoning paradigms have to be applied for different types of knowledge. The following facts about human recognition behavior may affect the IOSS design: reasoning with episodic experience, switching among various reasoning paradigms, and active learning.

An operator may encounter a lot of problems in his operation. He generalizes these problems and stores the knowledge for future use. This knowledge is stored as discrete episodes. Each episode is an explanation and solution to a single solved problem. These episodes are related to each other according to some complex conceptual features and criteria. When encountered a new problem, he will remind himself the past solved problems and create a solution from similar episodes rather than from scratch. In artificial intelligence (AI), there is a reasoning paradigm, namely case-based reasoning (CBR), to simulate this recognition behavior. CBR is the principal reasoning paradigm of human operators.

CBR is not the only reasoning paradigm that human operators apply. For a new problem, an operator with sufficient domain knowledge tends to apply the structural, behavioral and functional descriptions. This reasoning is corresponding to model-based reasoning (MBR) paradigm. Rule-based reasoning (RBR) is another important paradigm that is often applied to solve simple problems or to respond quickly to emergency situations. The selection of reasoning paradigm depends on operator's personal experience, the domain knowledge and other background knowledge he possesses, the problem solving objective, and the time limitation. Human operators navigate constantly among CBR, MBR and RBR in order to solve a complex problem efficiently.

As has been emphasized in the previous section, active learning capability is one of the most important advantages of human being.

The above multidimensional problem solving model can be directly applied to system design and served as a working environment of IOSS.

2.5 Multilayer Modularity Design

The performance of an intelligent system is usually evaluated by two parameters: scope and power. The scope is the range of applications that can be developed using the intelligent system. The power is represented by how easy it is to develop a specific application using the intelligent system. These two parameters are closely related. As the scope increases, the power will usually decrease. In developing intelligent systems, scope

and power must be carefully specified (Leitch and Stefanini, 1989). The high level AI languages, such as LISP, Prolog and C++, are high in scope but low in power. On the other extreme, highly process specific expert systems are of least scope (Lapointe et al., 1989). Among these two extremes, there are expert system shells and expert system designkit. Provided with knowledge representations and inference mechanisms, expert system shells, such as Meta-COOP (Rao et al., 1993), KEE, ART and Nexpert (Laurent et al., 1986) have more power than high level AI languages. To further improve the power of intelligent system, though in the expense of scope, designkits are developed that have more domain specific facilities (Stephanopoulos et al., 1987; Leitch and Stefanini, 1989).

The IOSS is configured to achieve the functional requirements and desired features outlined in Section 2.2.2. According to the design objectives, IOSS is an intelligent system aiming at solving a range of automation problems for a specific type of process industry. Therefore, it lies between general process automation design tools and process specific AI systems. As a result of applying the multidimensional problem solving model in system design, and also as an effort to achieve the desired system features, the IOSS for the BCTMP process is designed in a multilayer modularity architecture as illustrated by Fig. 2.6.

The following facts about the IOSS are obvious from Fig. 2.6:

- IOSS is programmed in C++ under ALPHA machine in open VMS System 1.0. Above the high level language C++, there are three layers: object-oriented intelligent system tool, operation support function modules, and IOSS application shell.
- The first layer, Meta-COOP, is an object-oriented intelligent system tool coded in C++ (Rao et al., 1993). Meta-COOP adopts the object-oriented programming technique and frame-based knowledge representation to implement the organization, management, maintenance and applications of complex knowledge base systems. In Meta-COOP, the organization structure of the knowledge bases can be divided into several components. It provides such distinct characteristics as the integration of various knowledge representations and inference methods.

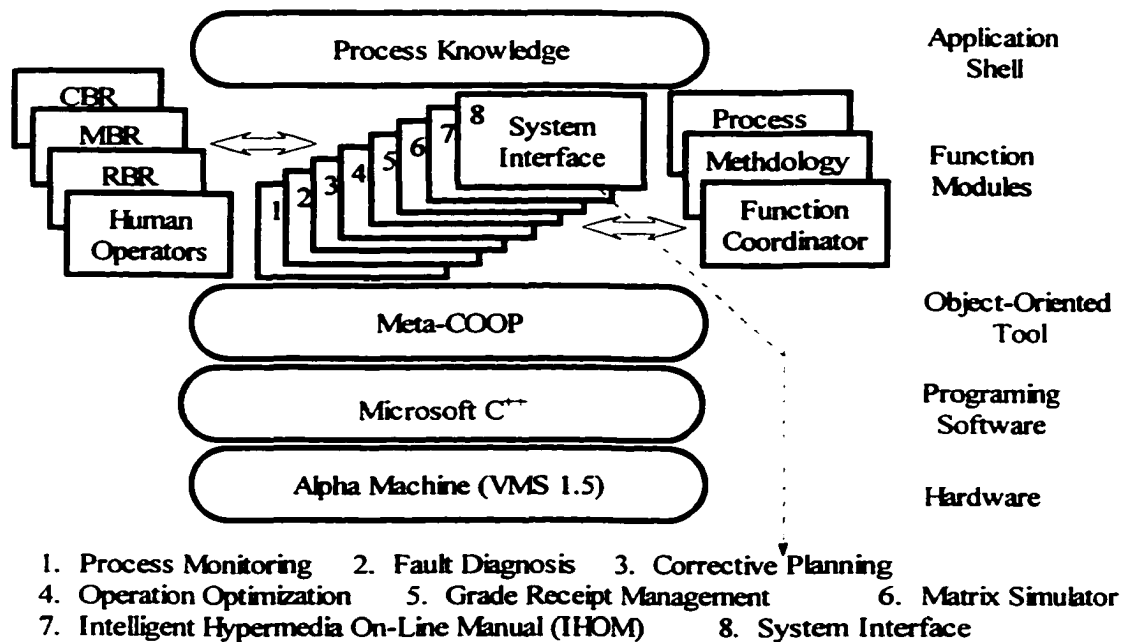


Fig. 2.6. Multilayer architecture of IOSS.

- The second layer includes the function modules based on the modeling of the generic problem solving tasks. With the distinct characteristics of Meta-COOP, the programming of these function modules is much easier than directly from high level AI language. These function modules, based on their roles in problem solving, can be classified into three groups:
 - Operation support function modules: These modules, depicted in the middle of the layer, are the generic operation support functionality. Examples of these modules are process monitoring, fault diagnosis, matrix simulator (Hassan et al., 1995) and IHOM.
 - Reasoning paradigm modules: These modules, depicted on the left side of the layer, fulfill various reasoning paradigms, including CBR, MBR, and RBR. It should be noticed that IOSS treats human operators as an integrated external

reasoning agent of the system. When none of the reasoning paradigms can be applied, IOSS will communicate to human operators for help.

- Coordination modules: The modules on the right side of the layer include process coordinator, methodology coordinator and function coordinator. The process coordinator integrates the local IOSS systems to a mill-wide system. Methodology coordinator fulfills the coordination among CBR, MBR, and RBR. The function coordinator combines and queries the problem solving function modules to solve complex engineering problems.
- The third layer is an application shell for customizing a specific IOSS application. Process dependent knowledge is implemented in this layer. Unique process dependent knowledge makes a unique IOSS system.

This multilayer modularity architecture fully reflects the above multidimensional problem solving model. The operation support function modules implement the function dimension. The reasoning paradigm modules implement methodology dimension. The process coordinator implements process dimension. The three coordinators fulfill the coordination among nodes in these three dimensions.

Knowledge of different levels and different process dependability is separated. The generic knowledge is implemented in function modules, whereas the process dependent knowledge is in application shell. Knowledge of different levels is respectively applied by CBR, MBR and RBR.

It is obvious that the desired system features, such as process generality, rich knowledge representation and user acceptability, are achieved with this architecture.

The motivation and philosophy of the above approach are somewhat similar to the cooperative expert systems, cooperative problem solving and distributed intelligent system technology (Durfee et al., 1989; Rao, 1991). In this research, we do not simply break up complex operation support problems into simple tasks. Instead, the system is designed based on desired features and a generic problem solving model that is proposed according to human operator's recognition behavior. Human expert is treated as a crucial

component of the IOSS. These considerations make the IOSS remarkably different from conventional distributed intelligent systems.

2.6 Conclusions

The operation support system technology is in great demand by process industry. This chapter presented the architecture and design of an intelligent operation support system (IOSS) for process industry. IOSS is designed based on the analysis of production requirement and human operator's recognition behavior. It employs rich knowledge representation and hybrid reasoning strategy that combines case-based reasoning (CBR), model-based reasoning (MBR) and rule-based reasoning (RBR). It is shown that the proposed design has many advantages such as debottlenecking knowledge acquisition, high reasoning efficiency, and better user acceptability.

Chapter 3

ACTUATOR AND SENSOR DESIGN FOR OPERATION SUPPORT SYSTEMS[†]

This chapter proposes a design approach for sensor and actuator placement in process operation support systems. The objective of the design is to find a set of sensors and actuators that satisfies the requirements of operation support systems, and meanwhile minimizes the instrumentation costs. The requirements of actuators and sensors in operation support systems are systematically formulated. The concept of fault distance is defined to measure the fault diagnosability and system diagnosability. The operation tree that represents the control structure in abnormal situations is introduced. The design is performed based on the fault distance and operation tree. A design example in the bleach plant of a pulp mill is illustrated.

[†] This chapter has been accepted for publication: Xia and Rao, 1999, "Actuator and sensor design for operation support systems", *Computers in Industry*.

3.1 Introduction

Sensor and actuator design is a fundamental problem towards control, fault diagnosis and operation support systems (de Silva, 1989; Xia et al., 1994a). In control systems, the problem is to choose sensors and actuators to optimize a controllability-observability related performance criterion subject to an hardware cost constraint, such as that in multivariable control systems (Ghosh and Knapp, 1989), optimal control (Geromel, 1989), distributed parameter systems (Kubrusly and Malebranche, 1985), and large space structure control (DeLorenzo, 1990). In fault diagnostic systems, the problem is to design sensor placement such that all faults are diagnosable. The design methods vary with the fault diagnostic approaches. In state observer/parameter estimator-based approach (Iserman, 1984), the basic requirement is the observability/identifiability such that the observer/estimator can be constructed. In the design approach proposed by Park and Himmelblau (1987), the linear state and measurement equations for every possible instrumentation scheme have to be available. The selection of proper scheme is based on observability and identifiability analysis of these equations. Such a method may be difficult to be applied to large industrial processes where the complete state and measurement equations are not available. In knowledge-based diagnostic systems, the faults are diagnosed using their abnormal symptoms (Xia et al., 1996). A system is diagnosable if every fault corresponds to a distinct syndrome. Ishida and his coworkers (Ishida et al., 1986; 1987) proposed a graph-theoretic model for diagnosability analysis. The model is described by a set of potential faulty units, a set of measurements, and an incident matrix that expresses the relation between the faults and the abnormal syndromes on the measurement. It was shown that this method could be applied to error correcting codes in computer systems. Another attractive method is entropy-based. Entropy is applied as a measure of the difficulty in system diagnosis relating to system complexity and signal incompleteness (Rouse and Rouse, 1979; Seong et al., 1994).

So far, there are few reports on sensor and actuator design for operation support systems in process industry (Xia et al., 1994a). The functions of an operation support

system include fault diagnosis, plant state evaluation, and corrective operations. Fault diagnosability is an important but not the only requirement of operation support systems. Sensors and actuators have to be designed to satisfy the needs of the various operation support functions.

In this chapter, the requirements of sensors and actuators in operation support systems are systematically formulated. Fault distances are defined to represent fault diagnosability and system diagnosability. The sensor design for fault diagnosis is based on the fault distances. An operation tree, which relates achievable plant states to the available control components, is introduced for the design of actuators and sensors for corrective operations. The design criterion is to find at least one successful path from the available control components to a prescribed plant state, that is, there exist corrective operation schemes to bring the plant from abnormal states to normal or safe states. The case study is conducted on the bleach plant of a bleached chemi-thermo-mechanical pulp (BCTMP) mill.

3.2 Problem Formulation

Intelligent operation support systems emerged from the complexity of modern industrial plants and the availability of inexpensive computer hardware (Hoskins et al., 1991; Modarres and Cadman, 1986; Petti et al., 1990; Venkatasubram and Rich, 1988; Rao and Corbin, 1992; Xia and Rao, 1997a). Modern industrial plants often collect vast amount of process data in distributed control systems and management information systems. Operators are faced with a problem commonly referred to as “data overload”. Because of the large volume of data, it is very difficult for operators to extract useful information for decision making. Process operations are time critical situations. Time stress due to data overload and decision uncertainty increases the risk of operator errors. It is discovered that even with thorough training, human operators may not always have in-time and efficient solutions to abnormal situations (Xia et al., 1993). Comprehensive operation support for the operator in abnormal situations to reduce operator errors are strongly suggested (Long, 1984). The solution to this problem is to design intelligent operation

support systems that reduce the cognitive load placed on operators by providing guidance for knowledge-based decision making. The system anticipates the operator's knowledge, performs necessary calculation, and prompts the operator with derived and formatted information and decision instruction (Nimmo, 1995).

· Process operations in abnormal situations include problem identification and corrective operations. An intelligent operation support system detects and diagnoses the abnormalities, predicts the consequences and plans for the corrective actions. One of the main functions of intelligent operation support systems is fault diagnosis that helps operator to make quick assessments of plant state and identify the root causes. Another function is decision making on corrective actions. It aids operator to decide corrective operations to bring the plant to normal states or prevent the plant from severe incidents.

Sensors and actuators are the instrumentation basis of intelligent operation support systems. In order to fulfill the above functions, an intelligent operation support system must have adequate plant information and sufficient flexibility for taking corrective actions that can only be provided by on-plant actuators and sensors. The placement of actuators and sensors must satisfy the following conditions:

- Sensors should provide, as a minimum, sufficient information for fault diagnosis. During the recovery stage, the sensors should also provide adequate information as whether the process has been in the specified target plant state. That is to say, the measurements should always be adequate to assess the critical process states.
- Sufficient sensor information is also required for the process control in normal and abnormal situations. It should be adequate to achieve the operation objectives in both normal and abnormal situations.
- Actuators should be sufficient for process control in the normal situation. In abnormal situations, they should provide sufficient flexibility for system reconfiguration and corrective actions to bring the plant to normal or safe states.

However, the instrumentation cost limits the number of sensors and actuators available in intelligent operation support systems. Extensive performance and cost analyses must be

performed to select a relatively small number of sensors and actuators that meet the requirements of the operation support functions. The purpose of this chapter is to introduce a method for optimal placement of actuators and sensors for operation support systems. The criterion of the design is to satisfy the requirements, and at the same time, to minimize the instrumentation cost.

3.3 Overview of the Bleach Plant

Bleach plant is one of the major processes in pulp and paper production. Its operation affects almost all pulp quality parameters. The process diagram of the bleach plant in a bleached chemi-thermo-mechanical pulp (BCTMP) mill is briefly presented in Fig. 3.1 which is a screen copy of MOPS display, a commercial management information system developed by MoDo Chemetics. The bleach plant consists of four washing stages and two bleaching stages: first stage washing, interstage bleaching and washing, third stage washing and bleaching, and fourth stage washing. Bleach towers are the major components of the bleach plant. The chemical reactions with peroxide that oxidize lignin to a less colored state are taking place in the bleach towers. There are two bleach towers in the bleach plant: P-1 and P-2 towers. The detail design of actuators and sensors will be illustrated on P-1 tower.

P-1 bleach tower has a bleaching zone and a dilution zone. The pulp of 38% consistency is discharged from the twin wire press (TWP) of the washing stage and dropped into the pulp conveyor. The bleach chemicals, including peroxide, caustic, silicate, DTPA and water, are added in the static mixer and mixed with pulp in the pulp conveyor. The pulp conveyor discharges the pulp mixed with the bleach chemicals into the Impco double-shafter steam mixer. Steam is added through four steam ports there. The heated pulp is then charged into P-1 bleach tower.

The pulp stock from P-1 bleach tower is pumped to the headbox of the twin wire press in the next washing stage. Its consistency is controlled by the second stage weak pressate added to the dilution zone (major dilution) and to the suction of the pulp stock pump (minor dilution). The detail flow chart of P-1 bleach tower is shown in Fig. 3.2.

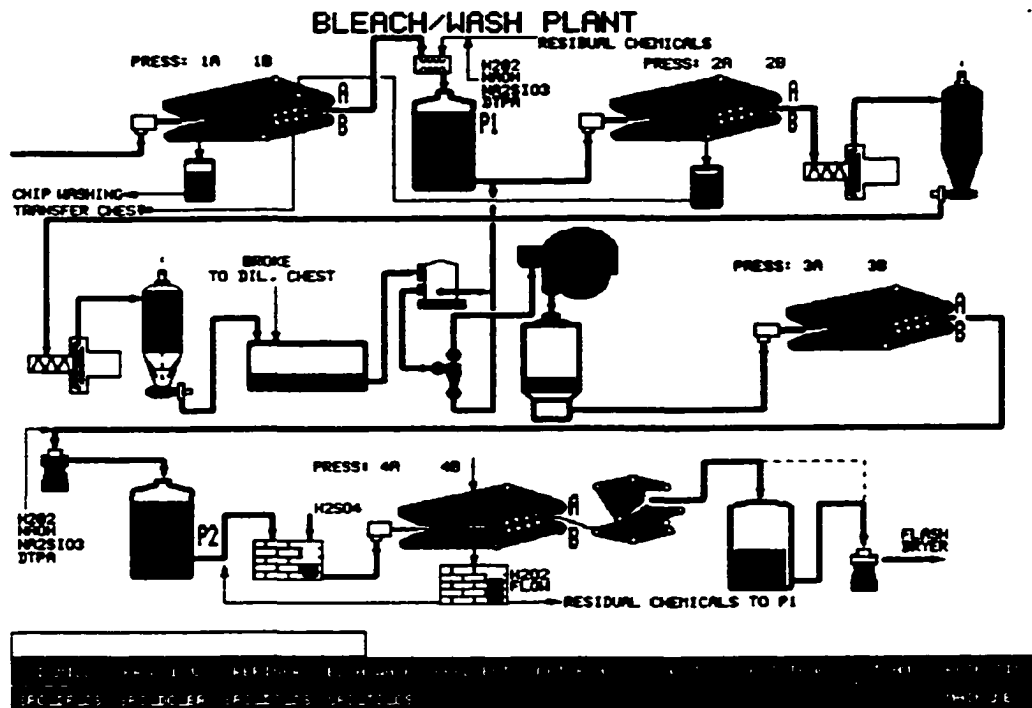


Fig. 3.1. Overview of the bleach plant.

The control loops in P-1 tower in normal situations include:

- bleaching zone level control (LIC-006);
- pulp stock consistency control (NIC-008);
- pulp temperature control (TIC-005);
- bleach chemical (water, caustic, silicate, DTPA, and peroxide) flow controls (FFIC-001, FFIC-002, FFIC-003, FFIC-004, FFIC-005).

The potential process faults include:

- pulp conveyer blocked up;
- steam mixer blocked up;
- loss of second stage pressate;
- loss of steam supply;

- loss of bleach chemicals supply;
- temperature sensor failure;
- bleaching zone level sensor failure;
- consistency sensor failure.

The placement of sensors and actuators is designed for the control functions in normal situations and the corrective operations in abnormal situations. The design principle is: the sensors and actuators to meet the minimum requirement for normal operations are selected first; the additional sensors are selected for the purpose of fault diagnosis; finally, the actuators and sensors to meet the requirements of corrective operations are selected.

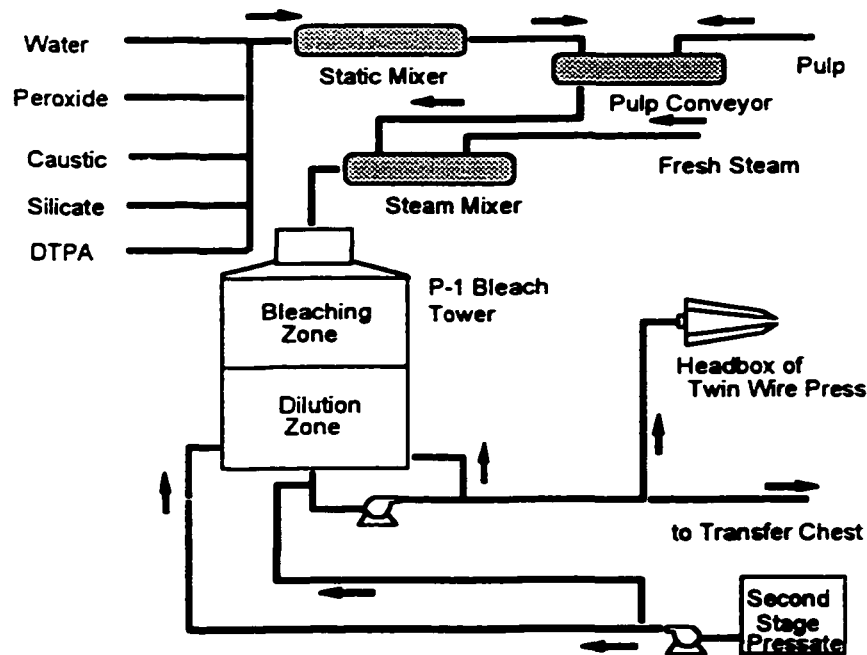


Fig. 3.2. Flow chart of P-1 bleach tower.

3.4 Sensor Design for Fault Diagnosis

The objective of the sensor design for fault diagnosis is to satisfy the requirement of fault

diagnosability, that is, all the process faults must be able to be identified from the symptoms with sensory information. In accordance with the reality of process industry, our discussions will focus on the single faults. It is assumed that no other independent faults occur than the said fault in the course of fault diagnosis. This single fault assumption only rules out the possibility of *independent faults*, but not the secondary faults that are caused by the primary faults.

Ishida et al. (1987) proposed a failure diagnosis model (FDM) that is defined by a triple (X, Y, Λ) , where $X = \{x_i\} \subset \mathcal{R}^n$ is a set of units that may fail, $Y = \{y_j\} \subset \mathcal{R}^m$ is a set of measurements, and $\Lambda = \{\lambda_{ij}\} \subset \mathcal{R}^{n \times m}$ is an incident matrix that describes the binary relations between X and Y . If the failed unit, x_i , leads to the abnormal state of the measurement y_j , $\lambda_{ij} = 1$; otherwise $\lambda_{ij} = 0$. The concept of the fault distance of a pair of distinct fallible units x_i and x_j is defined in terms of the distinct syndrome a failed unit produces. The fault diagnosability is analyzed based on the fault distance.

The above method was successfully applied to error-correcting codes in computer systems. The application to the design of instrumentation systems was also illustrated. However, it is difficult to apply Ishida's fault distance to the operation support system design. The requirements of sensory information for process fault diagnosis vary with the diagnostic strategies. In most intelligent operation support systems, the fault diagnosis is performed based on the behavior (symptoms) of the monitored process variables (Xia and Rao, 1997b). These symptoms are either static or dynamic, as will be explained in detail later. To distinguish a fault, at least one element of the fault syndrome needs to be distinct from that of the other faults. The fault distance needs to be defined to represent the number of syndrome elements that are distinct.

Based on these considerations, a triple description of fault diagnosis model similar to Ishida's, (F, Y, Λ) , is applied, where $F = \{f_i\} \subset \mathcal{R}^n$ is a set of process faults, Y is a set of measurements, and Λ represents the relationship between F and Y . A new fault distance is defined as follows.

Definition 1. The fault distance, $d(f_i, f_j)$, of a pair of process faults f_i and f_j is defined as

$$\begin{aligned} d(f_i, f_j) &= (|S(f_i)| - |S(f_i) \cap S(f_j)|) + (|S(f_j)| - |S(f_i) \cap S(f_j)|) \\ &= |S(f_i)| + |S(f_j)| - 2|S(f_i) \cap S(f_j)| \end{aligned} \quad (3.1)$$

where $S(f_i)$ and $S(f_j)$ are the symptoms produced by the faults f_i and f_j , respectively; $|S|$ is the cardinality of the set S .

It is obvious that the defined fault distance, $d(f_i, f_j)$, represents the number of distinct elements in the symptoms produced by the faults f_i and f_j . The following lemma proves that $d(f_i, f_j)$ satisfies three axioms for distances.

Lemma 1. The fault distance $d(f_i, f_j)$ in Eqn. (3.1) satisfies the following formulas

$$d(f_i, f_i) = 0 \quad (3.2)$$

$$d(f_i, f_j) = d(f_j, f_i) \quad (3.3)$$

$$d(f_i, f_j) + d(f_j, f_k) \geq d(f_i, f_k) \quad (3.4)$$

Proof: Substituting $S(f_i) = S(f_i) \cap S(f_i)$ into Eqn. (3.1), Eqn. (3.2) is obtained directly. Eqn. (3.3) is obvious since f_i and f_j are commutable in Eqn. (3.1). To prove (3.4), we first rewrite the right side of the Eqn. (3.4) as

$$\begin{aligned} d(f_i, f_j) + d(f_j, f_k) &= |S(f_i)| + |S(f_j)| - 2|S(f_i) \cap S(f_j)| \\ &\quad + |S(f_j)| + |S(f_k)| - 2|S(f_j) \cap S(f_k)| \end{aligned}$$

Rearranging the above equation gives

$$d(f_i, f_j) + d(f_j, f_k) = |S(f_i)| + |S(f_k)| - 2|S(f_i) \cap S(f_k)| \\ + 2|S(f_j)| + 2|S(f_i) \cap S(f_k)| - 2|S(f_i) \cap S(f_j)| - 2|S(f_j) \cap S(f_k)|$$

or

$$d(f_i, f_j) + d(f_j, f_k) = d(f_i, f_k) + I \quad (3.5)$$

where

$$I = 2|S(f_j)| + 2|S(f_i) \cap S(f_k)| - 2|S(f_i) \cap S(f_j)| - 2|S(f_j) \cap S(f_k)| \quad (3.6)$$

If $I \geq 0$, Eqn. (3.4) can be obtained directly from Eqn. (3.5). To prove $I \geq 0$, we use the following fact

$$2|S(f_i) \cap S(f_j)| = |S(f_i)| + |S(f_j)| - |S(f_i) \cup S(f_j)| \quad (3.7)$$

Substituting (3.7) into (3.6) and rearranging, we obtain

$$I = |S(f_i) \cup S(f_j)| + |S(f_j) \cup S(f_k)| - |S(f_i) \cup S(f_k)| \quad (3.8)$$

It is well known from the set theory that I given in Eqn. (3.8) satisfies $I \geq 0$. Thus Eqn. (3.4) is proven. Q.E.D.

Lemma 1 validates the fault distance given in Definition 1. A system is diagnosable if every fault can be identified from its corresponding symptoms. This requires every distinct fault correspond to a distinct syndrome, i.e. $S(f_i) \neq S(f_j)$ for all $j \neq i$. This condition can be described in terms of the fault distance.

Theorem 1. A system is diagnosable if and only if it has the fault distances satisfying the following conditions

$$d(f_i, f_j) \geq 1 \quad \forall i, j = 1, 2, \dots, n; \quad i \neq j \quad (3.9)$$

The proof of Theorem 1 is straightforward. Eqn. (3.9) implies that every pair of distinct faults produces at least one distinct symptom. Therefore, all faults are identifiable. A greater fault distance implies more elements that are distinct. As a result, the faults can be identified with more accuracy. To represent the degree of fault diagnosability, we define the concept of l -diagnosability in terms of fault distances.

Definition 2. A process fault f_i is l -diagnosable if and only if

$$d(f_i, f_j) \geq l \quad \forall j = 1, 2, \dots, n; \quad j \neq i \quad (3.10)$$

where $l \geq 1$ is an integer. A system is l -diagnosability if all faults are l -diagnosability.

The l -diagnosability implies that there are at least l distinct elements in the syndromes of every pair of distinct faults. The objective of sensor placement for fault diagnosis is to satisfy the fault diagnosability condition (3.10). From the definition of fault distance and fault diagnosability, it is obvious that fault diagnosability depends not only on the placement (number and location) of sensors, but also on the type and number of abnormal symptoms defined for each measurement. With the same sensor placement scheme, the fault distance and the fault diagnosability may be different if the abnormal symptoms defined for the measurements are different. Therefore, the minimum number of sensors required to realize the specified fault diagnosability depends also on the symptoms that are applied in fault diagnostic strategy.

The diagnostic strategy developed by Xia and Rao (1997b) applies two types of the symptoms for fault diagnosis: the static symptoms and the dynamic symptoms (See Chapter 4). The static symptoms, which are the most commonly used, are symbolic state attributes of process variables formulated by instant measurement values. They are

obtained by comparing the measurements of the monitored process variables to their standard limits. A finely defined process variable may have five state attributes: {"high (+1)", "extreme-high (+10)", "normal (0)", "low (-1)", "extreme-low (-10)"}. A less accurately defined variable may only have three state attributes {+1, 0, -1}. In the case that only static symptoms are applied, the incident matrix $\Lambda = \{\lambda_{ij}\} \subset \mathfrak{R}^{n \times m}$, which shows the fault relation between F and Y , has its elements belonging to the set $\{+10, +1, 0, -1, -10\}$. If fault f_i has no effect on measurement y_j , $\lambda_{ij} = 0$. Otherwise, if fault f_i causes measurement y_j to become high, $\lambda_{ij} = +1$; if f_i causes y_j to be extremely high, $\lambda_{ij} = +10$; if f_i causes y_j to be low, $\lambda_{ij} = -1$; if f_i causes y_j to be extremely low, $\lambda_{ij} = -10$. Fig. 3.3 shows a simple graph that represents the relation between F and Y . The graph consists of two components: nodes and edges. The nodes represent process faults and state attributes (symptoms) of measurements. The edges represent the causality between process faults and process measurements. For example, an edge $\lambda_{ij} = +1$ indicates that f_i will cause y_j to become high.

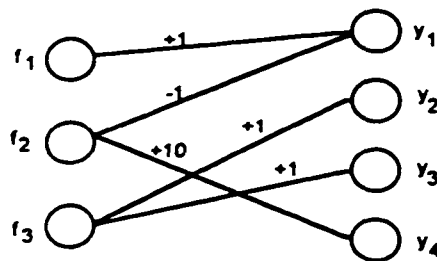


Fig. 3.3. Causal relation between F and Y .

Production processes in process industry are dynamic in nature. When a fault occurs, the production process will undergo a transient process until it reaches a new steady state. Static symptoms are not sufficient to describe the behavior of dynamic process variables. Additional information, which represents the dynamic behavior of the process variables, can be extracted from the current and historical sensory data. It may play an important role in fault diagnosis. Xia and Rao (1997b) defined two types of dynamic symptoms:

instant dynamic symptoms and interval dynamic symptoms (see Chapter 4). The instant dynamic features describe the instant changing speed of a process variable, which include three attributes:

- increasing
- decreasing
- unchanging.

The interval dynamic symptoms describe the evolution of a process variable in the period of interest. The interval dynamic symptoms include

- eventual increase with a final change M
- eventual decrease with a final change M
- abrupt increase with a final change M
- abrupt decrease with a final change M
- increase and then return with a peak change M
- decrease and then return with a peak change M.

The dynamic symptoms are the additional diagnostic information extracted from the same set of process data. Using the dynamic symptoms together with the static symptoms, the composite fault distance defined in Eqn. (3.1) can be rewritten as

$$d(f_i, f_j) = d_s(f_i, f_j) + d_d(f_i, f_j) \quad (3.11)$$

where $d_s(f_i, f_j)$ is the static fault distance of f_i and f_j that is represented by the static syndromes, $S_s(f_i)$ and $S_s(f_j)$; $d_d(f_i, f_j)$ is the dynamic fault distance of f_i and f_j that is represented by the dynamic syndromes, $S_d(f_i)$ and $S_d(f_j)$. As in Definition 1, the static and dynamic fault distances are defined as:

$$d_s(f_i, f_j) = |S_s(f_i)| + |S_s(f_j)| - 2|S_s(f_i) \cap S_s(f_j)|$$

$$d_d(f_i, f_j) = |S_d(f_i)| + |S_d(f_j)| - 2|S_d(f_i) \cap S_d(f_j)|$$

Obviously, the composite fault distance, $d(f_i, f_j)$, that employs both static and dynamic symptoms is greater than the static fault distance that employs static symptoms alone. For a pair of faults that has the same static symptoms, the dynamic symptoms can be different because of their distinct fault evolution processes. Therefore, using dynamic symptoms can lower the requirement for sensors and improve the reliability and accuracy of fault diagnosis. When no dynamic symptoms are applied for fault diagnosis, the composite fault distance is reduced to the static fault distance.

For the convenience of sensor design, the following fault distances are defined for the overall system.

Definition 3. Assume that l -diagnosability is the desired fault diagnosability. The fault distances of the overall system are defined as:

$$d_1 = \sum_{i=1}^{n-1} \sum_{j=i+1}^n \min\{l, d(f_i, f_j)\} \quad (3.12)$$

$$d_2 = \sum_{i=1}^{n-1} \sum_{j=i+1}^n d(f_i, f_j)$$

The following results are generated from the above definitions:

- (1) $d_1 \leq \frac{1}{2}n(n-1)l$ and $d_2 \geq d_1$. If $d_1 = \frac{1}{2}n(n-1)l$, the system is l -diagnosability.
- (2) When $d_1 = \frac{1}{2}n(n-1)l$, the greater d_2 , the better is the fault diagnosability of the systems.
- (3) d_1 and d_2 are the functions of the measurement set, Y . For two sets of measurements Y^1 and Y^2 , if $Y^1 \subset Y^2$, then $d_1(Y^1) \leq d_1(Y^2)$ and $d_2(Y^1) \leq d_2(Y^2)$.

The following algorithms give the detail procedures in which the design of sensors for fault diagnosis is carried out.

Algorithm 1. The sensors for fault diagnosis can be designed in the following procedures using the sensitivity of the fault distances d_1 and d_2 to each individual sensor:

Step 1. Identify the control functions in normal situations; identify all the possible selections of sensors (measurement of process variables), M .

Step 2. Design the sensors required to implement the control functions in normal situations, $Y_n \subset M$.

Step 3. Identify all the potential process faults F ; define the abnormal symptoms for each measured process variable; define the desired fault diagnosability.

Step 4. Develop the fault graph to represent the relation between F and M ; let $Y = Y_n$.

Step 5. Calculate the fault distances of the system, $d_1(Y_n)$; if $d_1(Y_n) < \frac{1}{2}n(n-1)l$, go to Step 6; otherwise stop. $Y = Y_n$ is the set of sensors for fault diagnosis.

Step 6. Partition M (excluding Y_n) into several groups M_i ($i = 1, 2, \dots, p$) based on the price, reliability, and easiness in installation and maintenance. The sensors in the preceding groups have the priority to be selected.

Step 7. Try every new sensor, y_{new} , from the first group of M_i ($i = 1, 2, \dots, p$) that is not empty; let $Y' = (Y, y_{new})$; compute $d_1(Y')$ and $d_2(Y')$; select the sensor that makes d_1 increase the most. If there are several sensors that increase d_1 by the same amount, select the one that makes d_2 increase the most.

Step 8. If $d_1(Y') < \frac{1}{2}n(n-1)l$, let $Y = Y'$ and go to Step 7; otherwise stop, and $Y = Y'$ is the set of sensors for fault diagnosis.

The above algorithm is easy to implement, however, the selected sensors may not be the minimum set. The following algorithm is proposed to solve the problem.

Algorithm 2. The minimum set of sensors for fault diagnosis can be obtained by using the following procedures (Steps 1-5 are the same as in Algorithm 1):

Step 6'. let $k = 1$.

Step 7'. try all possible k-dimensional subsets $Y_{new} \subset M - Y_n$; let $Y = (Y_n, Y_{new})$; compute $d_1(Y)$ and $d_2(Y)$; select the subset Y_{new} that corresponds to the greatest d_1 . If there are several subsets that correspond to equally greatest d_1 , select the one that corresponds to the greatest d_2 .

Step 8'. If $d_1(Y) < \frac{1}{2}n(n-1)l$, let $k = k + 1$ and go to Step 7'; otherwise stop. Y is the minimum set of sensors for fault diagnosis.

The above two design algorithms do not explicitly incorporate the costs of sensors. In a practical design, the cost of various sensors may be dramatically different. Cost limitations will significantly affect the design results.

The cost of a sensor, C_i , consists mainly of the purchase, installation and maintenance expenses

$$C_i = C_{ip} + C_{ii} + C_{im} \quad (3.13)$$

where C_{ip} , C_{ii} and C_{im} are the purchase, installation and maintenance expenses of the sensor y_i , respectively. For a set of sensors, Y , the total cost is

$$C_Y = \sum_{i \in Y} C_i \quad (3.14)$$

Algorithm 3. The set of sensors for fault diagnosis with a minimum cost can be obtained by using the following procedures (Steps 1-5 are the same as in Algorithm 1):

Step 6". Select all the possible subsets $Y_i \subset M - Y_n$; calculate its cost C_{Y_i} using Eqn. (3.14); arrange the subsets in the order of the cost: $C_{Y_1} \leq C_{Y_2} \leq \dots \leq C_{Y_k} \leq C_{Y_{k+1}} \leq \dots \leq C_N$; let $k = 1$.

Step 7". Let $Y = (Y_n, Y_k)$; compute $d_1(Y)$.

Step 8". If $d_1(Y') < \frac{1}{2}n(n-1)l$, let $k = k + 1$ and go to step 7; otherwise stop. Y is the set of sensors for fault diagnosis with minimum cost.

The selection of a proper algorithm depends on the characteristics of the engineering problems. If all the sensors are assumed of the same cost, Algorithms 2 and 3 will generate the same result.

3.5 Actuator and Sensor Design for Corrective Operations

Process operations intend to run the plant in target production states. In normal situations, the target state is optimal production. In the event of some process faults, optimal states are restorable after reconfiguring the production and control systems. However, in the event of some substantial process faults, optimal plant states are no longer achievable. There is no alternative but to accept a degraded or suboptimal production, in which the key process variables are maintained in acceptable ranges, and thus the plant produces product with the acceptable (may not be desirable) quality and profits. To an extreme, when the profitable production can not be maintained, the production process has to be shut down with the least production loss. Fig. 3.4, as indicated in Chapter 2, presents the possible suboptimal states with deteriorated optimality top-down.

The purpose of operation support is to assist operators to find the best achievable plant state with the available resources and the corresponding corrective actions (Rasmussen and Goodstein, 1987). The decision making process can be viewed as to find the mapping among three finite sets: a set of process faults, a set of target plant states, and a set of corrective actions (including the reorganization of available sensors and actuators) to restore the target states.

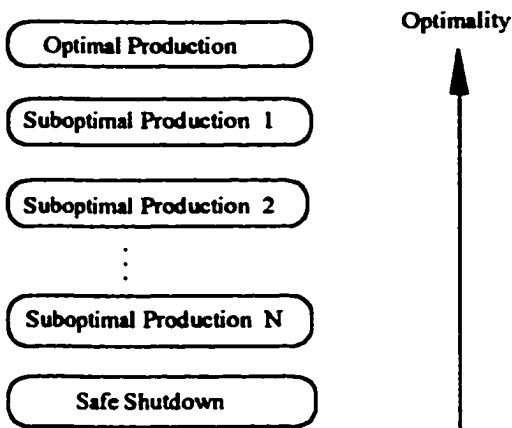


Fig. 3.4. Possible plant states in abnormal situations.

Note that a plant state is supported by a number of key process variables or control functions, while a process variable is supported by a number of process and control components, including sensors and actuators. Therefore, for the purpose of sensor and actuator design, a production system can be abstracted with the following four abstract symbols:

- (1) Process faults $F = \{f_i\}$: the potential anomalies of the production system;
- (2) Target plant states $T = \{t_i\}$: each fault is attached with a target plant state that is to be restored if the fault occurs, that is $f_i \rightarrow t_i$;
- (3) Key process variables (control functions) $V = \{v_i\}$: a plant state t_i is supported by a group of process variables, that is, $V_i \rightarrow t_i$;
- (4) Control components $U = \{u_i\}$: the control components include sensors and actuators (valve, pump, etc.), as well as human actions in some situations. A control function is supported by a set of control components, that is, $U_i \rightarrow v_i$.

These four symbols are inherently related to each other by the above definition. In order to solve the problem of sensor and actuator placement, a representation is required to organize these abstract symbols and incorporate their relations. To illustrate this representation, the physical system shown in Fig. 3.2 is investigated. If "pulp is fed to the

succeeding washing stage normally" is considered a required plant state, Fig. 3.2 can be interpreted to rules about the way in which control components have to be organized so that the plant state can be achieved, as illustrated by Fig. 3.5. The nodes in the first layer represent control components. Those in the second and third layers represent process control functions. The node in the fourth layer is the plant state. The process control functions are decomposed into two layers for the purpose of convenience. In some situations, more than two layers of control functions are required, depending on the complexity of the production process and control system. A control function is not necessarily a true control loop. However, it is always a critical process variable. The links from control components to a control function define what control components have to be operated upon to fulfill the control function. The links from the control functions in the second layer to a control function in the third layers define what control functions in the lower layer must be working properly to fulfill the control function in the higher layer. The links from the control functions to the plant state indicate that in order to achieve the plant state, those control functions need to perform normally.

Modarres and Cadman (1986) proposed a feasible process goal tree for alarm system analysis. Following the discussion above, it is found that a structure similar to Modarres and Cadman's goal tree can be applied to design control components (sensor and actuator). In this chapter, a similar structure, namely operation tree, is constructed to represent the knowledge about the relations among the four abstract symbols in hierarchical way: target plant states, control functions, control components, and faults. Each node in the tree is one of the abstract symbols. The top layer is the production objectives, which may include improving product quality, maximizing production profit, reducing loss in emergency situation and improving production safety. The second layer is the three types of plant states: optimal production, suboptimal (degraded) production and safe shutdown. Below the plant states, there are one or more layers for the control functions that are required in order to achieve the plant states. For example, in Fig. 3.5, the control functions are arranged in two layers. The lowest level represents the control components and human actions required to fulfill the control functions. The process faults

(faulty components) are also included in this layer. The structure of the operation tree is shown in Fig. 3.6.

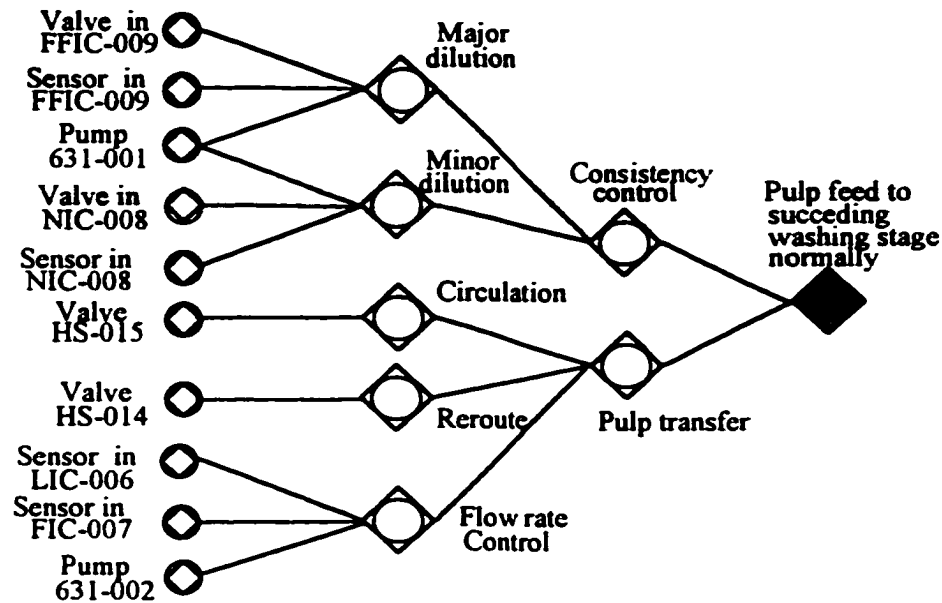


Fig. 3.5. Control structure of the bleach plant (part).

To develop the operation tree, the potential faults and the corresponding target plant states are identified first. The operation tree is developed top-down from the target plant states to the first level control functions, then detailed downward to lower level control functions, and finally to the control components and human actions. To achieve an optimal placement of sensors and actuators, the entire candidate sensors and actuators must be appeared in the operation tree. There are two causal logics used in the operation tree, "OR" and "AND". "OR" implies that there is more than one alternative to achieve the goal represented by the upper layer's node. "AND" means all the conditions represented by the nodes in the layer must be satisfied in order to achieve the goal represented by the node in the upper layer. In the normal conditions, there must be at least one success path from the control components to the full production objectives. When a process fault presents, there must be at least one success path from the remaining

control components to the target plant state attached to the fault. The objective of sensor and actuator design is to select the best set of control components (sensors and actuators) from those in the lowest layer of the operation tree. Here, the "best" means the least number or minimum cost of actuators and sensors that satisfy the requirements for the success paths under normal and abnormal situations. The cost of the control components can be computed using Eqns. (3.13) and (3.14).

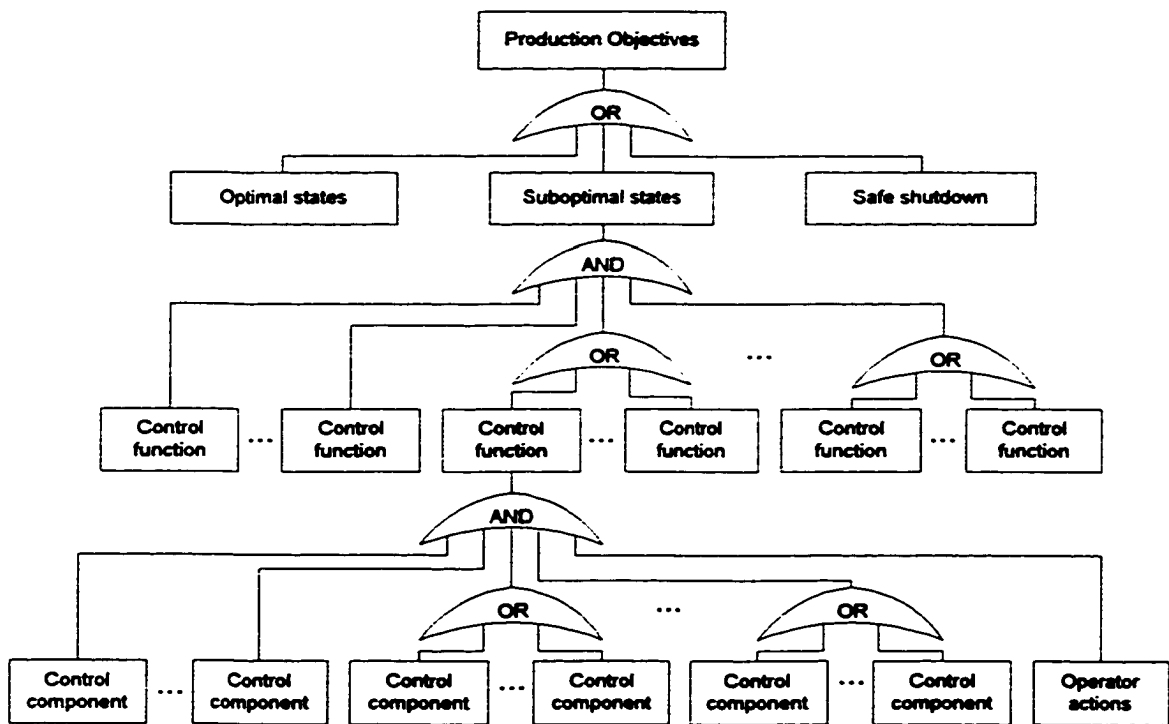


Fig. 3.6. The operation tree for actuator and sensor placement.

The following algorithm gives the procedures to design the sensor and actuator placement. It is assumed that the actuators and sensors for process control in the normal situation and for fault diagnosis have been selected.

Algorithm 4. Denoting the set of all candidate control components by Z , and those used for normal control and fault diagnosis by Z_0 ($Z_0 \subset Z$). The control components for corrective operations can be selected by using the following procedures:

- (1) For every potential process fault, $f_i \in F$, specify a target restoring plant state $t_i \in T$.
- (2) Develop the operation tree that includes all the candidate control components and incorporates the potential process faults.
- (3) Find all the control component placement schemes $Z_1, Z_2, Z_3, \dots, Z_N$ that have at least one success path from each process fault to the prespecified target restoring plant state. Note that $Z_i = (Z_0, Z'_i)$, and for any scheme $Z_j \supset Z_i$, Z_j will not be considered since it has a higher cost than Z_i does.
- (4) Compute the cost of each scheme by using $C_{z_i} = \sum_{k \in Z_i} C_k$, the one with the lowest cost is the optimal scheme.

It is a possibility that the developed operation tree does not have any success path for a process fault even if all the control components are selected. One way to solve the problem is to change the target restoring plant state. The higher the optimality of the restoring plant state, the higher the requirement for the control components. The selection of the proper plant state to be restored in the event of a fault is a trade-off between the production optimality and the cost of the control components. Another way to solve the problem is to increase the selection of control components, which implies more nodes in the lowest layer of the operation tree. With proper specifications of candidate control components and the restoring plant states, an optimal placement scheme is always obtainable.

3.6 Case Study in the Bleach Plant

In the previous sections, the design of sensors for fault diagnosis is completely

independent of that for corrective operations. However, the sensors designed for fault diagnosis also function in corrective operations. Similarly, the sensors designed for corrective actions can be applied for fault diagnosis. In order to obtain the most economical design, the interactions need to be considered. Moreover, additional sensors may be required for plant state evaluation.

The objective of the sensor design for plant state evaluation is to provide adequate information to determine whether the plant is in the prespecified restoring states (optimal or suboptimal). The information needed for plant state evaluation is denoted by $I(T)$. As has been pointed out, a plant state is supported by a number of control functions. Therefore, the plant state can be evaluated by using the measurement of the process variables related to these control functions. In most cases, the sensors for fault diagnosis are adequate for plant state evaluation.

Algorithm 5 presents the design of sensors and actuators for operation support systems.

Algorithm 5. The combined design of sensors and actuators for operation support is carried out in the following steps:

- Step 1. Design (for a new control system) or identify (for an existing control system) the sets of sensors and actuators for process control in normal situations, denoted by Y_n and A_n , respectively;
- Step 2. Design the additional sensors that will be definitely used (without any alternative) in corrective operations, denoted by Y_{c1} ;
- Step 3. Design the additional sensors for fault diagnosis, denoted by Y_f ;
- Step 4. Design the additional sensors, if necessary, for plant state evaluation, denoted by Y_e ;
- Step 5. Design the additional actuators and sensors for corrective operations, denoted by Y_{c2} and A_c ;

Step 6. The sensor and actuator placement schemes are $Y = (Y_n, Y_{c1}, Y_f, Y_s, Y_{c2})$ and $A = (A_n, A_c)$, respectively.

To obtain the most economical design, the recursive procedures as shown in Fig. 3.7 can be applied.

Using an object-oriented intelligent system building tool, Meta-COOP (Rao et al., 1993), the fault relation graph and the operation tree can be readily implemented. The fault distance, fault diagnosability, success paths of the operation tree, and the cost of each sensor and actuator placement scheme are automatically calculated. The most economical design is selected.

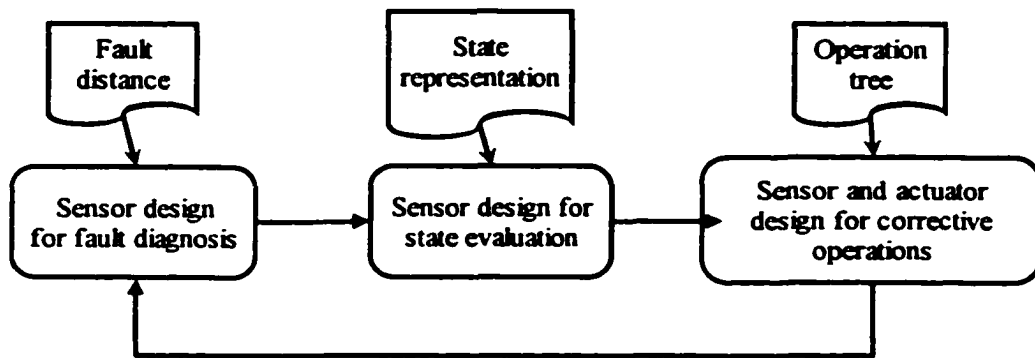


Fig. 3.7. Recursive sensor and actuator design.

Consider the design for P-1 bleach tower as an example. The actuators and sensors for the control functions in the normal situation are identified (refer to Section 3.3):

- Bleaching zone level control: bleach zone level sensor, and the control valve and pump in the pulp line to the succeeding twin wire press.
- Pulp stock consistency control: consistency sensor, and the pump and control valves in the pressate line.
- Pulp temperature control: steam control valve and temperature sensor.

- **Water, peroxide, caustic, silicate and DTPA controls: each control loop requires a control valve and a flowrate sensor.**

The attached target restoring plant states for the potential process faults are:

- **Pulp conveyor blocked up → safe shutdown**
- **Steam mixer blocked up → safe shutdown**
- **Lost second stage pressate → optimal production**
- **Lost steam supply → safe shutdown**
- **Lost bleach chemicals supply → optimal production**
- **Temperature sensor failed → suboptimal production**
- **Bleaching zone level sensor failed → suboptimal production**
- **Consistency sensor failed → safe shutdown.**

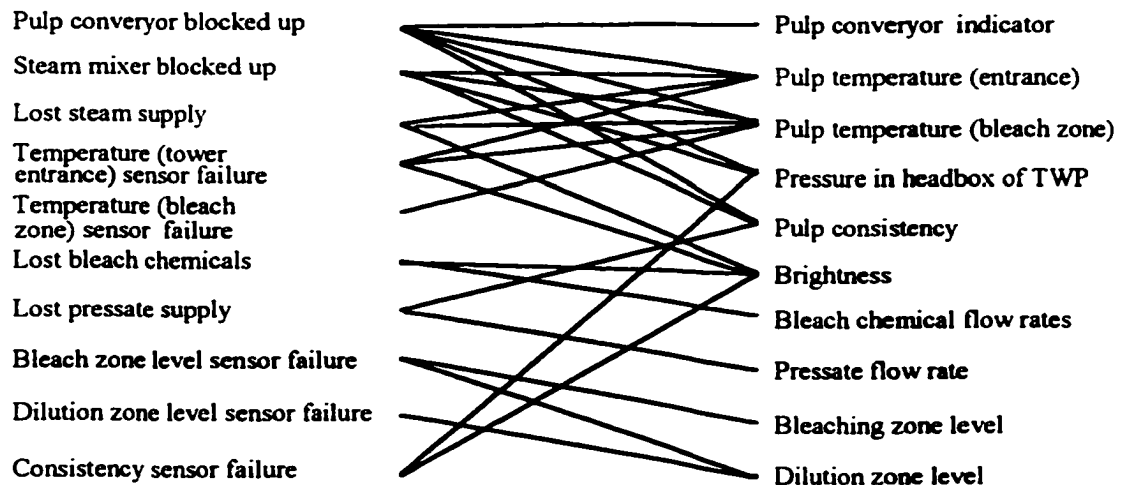


Fig. 3.8. Graph for fault-symptom relations.

The process mechanism analysis indicates that a temperature sensor is required in the bleaching zone to achieve suboptimal production when the pulp temperature sensor fails. Similarly, a level sensor is required in the dilution zone to achieve suboptimal production

when the bleaching zone level sensor fails. These two sensors are also useful for operators to monitor the operation condition of the bleach tower. Pulp brightness is the most important quality variable in the bleach plant. A brightness sensor is installed, though may not be always accurate.

The graph for the relation between the potential process faults and the candidate measurements is depicted in Fig. 3.8. The additional sensors for fault diagnosis are the pulp conveyer amperage indicator and the pressure sensor in the headbox of the twin wire press.

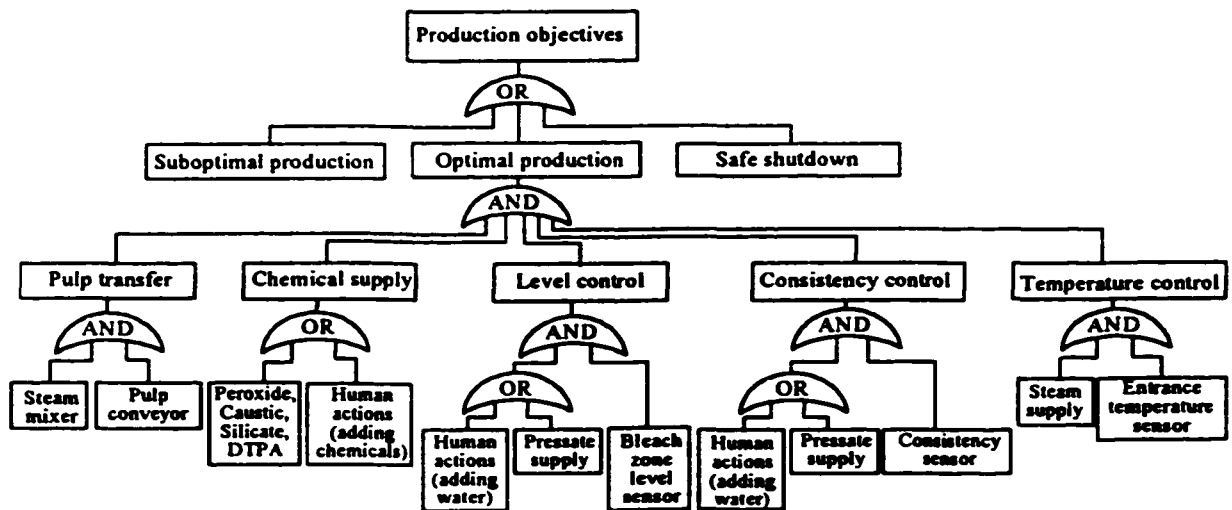
The fault incidents, which are not indicated in Fig. 3.8, are given by the following incident matrix:

$$\Lambda = \begin{bmatrix} +10 & * & 0 & -1 & -10 & * & 0 & 0 & 0 & 0 \\ 0 & * & * & -1 & -10 & * & 0 & 0 & 0 & 0 \\ 0 & -10 & -10 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & \pm 1 & \pm 1 & 0 & 0 & \pm 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \pm 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & -10 & 0 & 0 & 0 \\ 0 & 0 & 0 & +1 & +10 & * & 0 & -10 & 0 & 0 \\ 0 & 0 & 0 & * & 0 & 0 & 0 & * & \pm 1 & \pm 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \pm 1 \\ 0 & 0 & 0 & \pm 1 & \pm 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

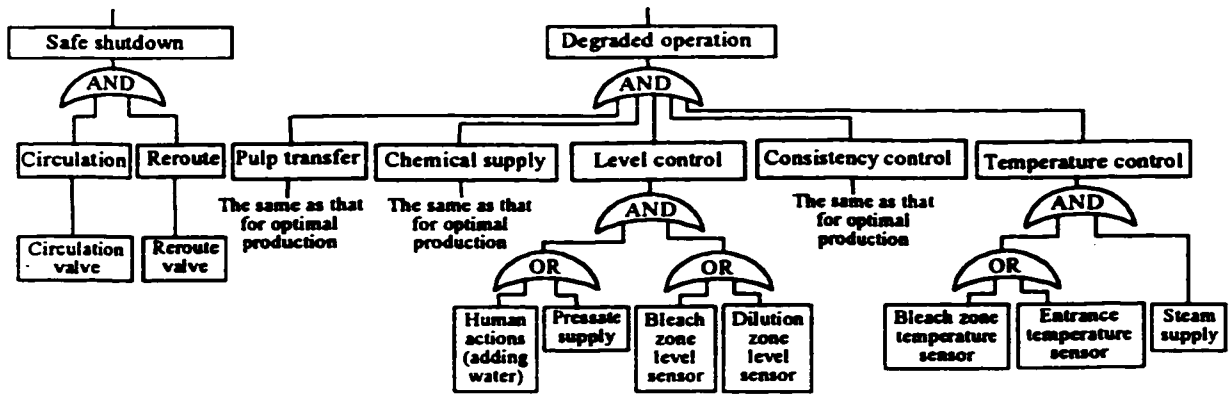
where "*" represents the uncertain incidents. Note that there are five chemical flows and five corresponding process faults. For purpose of convenience, only one chemical flow is shown in Fig. 3.8 and in the incident matrix. It is assumed only the static symptoms are used for fault diagnosis. The desired system diagnosability is 1-diagnosability.

The operation tree is depicted in Fig. 3.9. For simplicity, only the process faults (faulty components) and the additional actuators and sensors for corrective operations are included. Fig. 3.9a shows the paths for optimal production, while Fig. 3.9b shows that for suboptimal production and safe shutdown. From the operation tree, it is obvious that by adding the circulation valve and reroute valve, all the restoring plant state can be achieved. The corrective operation scheme is as follows:

- When the pulp conveyor is blocked up, or the steam mixer is blocked up, or the consistency sensor fails or steam supply is lost, open the circulation valve and reroute valve to allow pulp to be circulated back to the tower and reroute to transfer chest and then shut down the plant.
- When the second stage pressate supply is lost, operators will add fresh water into the pressate tank and maintain the plant in the optimal production state.
- When bleach chemicals supply is lost, operators will add chemicals in chemicals tanks and maintain the plant in the optimal production state.
- When the temperature sensor in the entrance of the bleach tower fails, use the temperature sensor in the bleaching zone as feedback signal of the control loop and run the plant in a degraded production state.
- When the bleaching level sensor fails, use the dilution zone level sensor as feedback signal of the control loop and run the plant in a degraded production state.



(a)



(b)

Fig. 3.9. The operation tree for actuator and sensor placement.

The hardware implementation of the control system for the P-1 bleach tower of the bleach plant is depicted in Fig. 3.10.

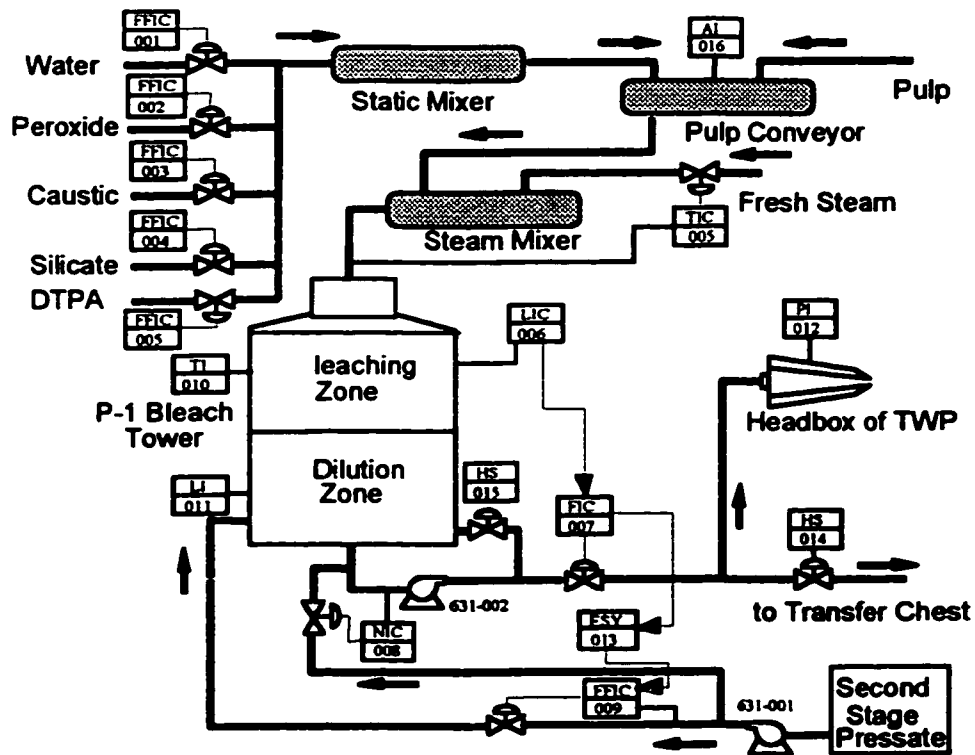


Fig. 3.10. Flow chart of P-1 bleach tower.

3.7 Conclusions

A method for the design of sensor and actuator placement for operation support systems is proposed in this chapter. The objective of the design is to meet the requirements of the operation support and meanwhile minimize the instrumentation cost. The application of the method to the bleach plant of a pulp mill is illustrated. This method can be applied to the design a new industrial plants and the innovation of industrial plants.

Chapter 4

DYNAMIC CASE-BASED REASONING METHOD[†]

This chapter is intended to develop an efficient reasoning method for operation support systems. It is pointed out that case-based reasoning (CBR), which is based on the concept that human memory is episodic in nature, is consistent with operator's problem solving. Despite their successful application to the solution of many problems, case-based reasoning methods are mostly static. Process operation support systems require a CBR method that can represent system dynamics and fault-propagation phenomena, and can be integrated with other proven reasoning methods, such as model-based reasoning (MBR) and rule based-reasoning (RBR). To solve this problem, a new approach, namely dynamic case-based reasoning (DCBR), is developed. DCBR introduces a number of new mechanisms including time-tagged indexes, dynamic and composite features, and multiple indexing paths. As a result, it provides flexible case adaptation, timely and accurate problem solving, and an ability to incorporate other computational and reasoning methods.

[†] This chapter has been accepted for publication: Xia and Rao, 1999, "Dynamic case-based reasoning for operation support systems", *Engineering Applications of Artificial Intelligence*.

4.1 Introduction

The most common reasoning approaches in problem solving in artificial intelligence (AI) include case-based reasoning (CBR) (Bareiss, 1989; Kolodner et al., 1985), model-based reasoning (MBR) (Davis, 1984) and rule-based reasoning (RBR) (Calandranis et al., 1990). Each of them has shown advantages over others in one situation and disadvantages in another. Rule-based reasoning applies production rules for problem solving. It is efficient at reasoning, and provides for relatively easy knowledge acquisition. Because of its unstructured knowledge organization, however, rule-based reasoning is rarely applied to solve complex problems. Model-based reasoning, on the other hand, uses structured deep domain knowledge for problem solving. The domain knowledge describes the structural, functional and behavioral information about the investigated process. Model-based reasoning has good generality, and can be applied to the solution of novel problems. However, its reasoning efficiency may be low, and the knowledge is usually difficult to obtain. Case-based reasoning is a reasoning approach that lies between RBR and MBR in the sense of the level of knowledge applied. The knowledge used in this approach is the experience about problems solved in the past. Case-based reasoning creates a solution to a new problem by relating it to the previously solved problems, and by adapting the earlier solutions to suit the new problem. It is especially useful in the areas where the domain theory is weak (Portinale et al., 1993). The suitability of the above approaches varies with the situation, depending on the type and size of the problem to be solved, as well as the availability of knowledge.

Case-based reasoning was founded on the belief that human memory is episodic in nature (Schank, 1975). This episodic memory, which comprises human knowledge, is accumulated from past experience. Each memory episode is contributed by a single past situation or event. Faced with a new problem, a human being often relates the problem to one or more memory episodes, and creates a solution from these episodes, not from scratch. These highly interconnected memory episodes are dynamically updated during problem solving. CBR is a computer program to simulate this human recognition process.

It has been successfully applied to solve many problems, such as natural language processing (Stanfill, 1988), cooking (Hammond, 1989), diagnostic problem solving (Hammond and Hurwitz, 1988; Kolodner and Kolodner, 1987)), law (Branting, 1989; Selfridge and Cuthill, 1989), mediation (Kolodner et al., 1985), medicine (Bareiss, 1989; Turner, 1988), and resource allocation and scheduling (Koton, 1988).

Investigations into pulp and paper production discovered that human operators rely heavily on the memory of previously solved problems in process operations (Xia et al., 1996). When a new problem occurs, the operator will refer to his memories of previously solved problems to obtain a solution. This fact implies that CBR is consistent with the recognition behavior of human operators. It would therefore be advantageous to apply CBR as a principal reasoning paradigm in operation support systems with MBR and RBR as the complement. However, the existing CBR methods are static in nature. They are most suitable for solving static problems, but not for dynamic ones. It is difficult to integrate CBR with other reasoning or calculation methods. These facts limit the application of CBR in operation support systems where the problems to be solved are dynamic.

To overcome this difficulty, this chapter proposes a dynamic case-based reasoning (DCBR) method, an extension of existing CBR to dynamic problems. DCBR applies dynamic and composite features, time-tagged indexes and multiple indexing paths. The new indexing and retrieval mechanisms enable the incorporation of system dynamics and fault propagation phenomena, and the augmentation of other qualitative and quantitative algorithms. As a result, the performance of operation support systems can be improved.

4.2 Introduction to the CBR Approach

The design of a CBR system is a result of the attempt to model the human recognition process with computer models, including memory, learning and problem solving, etc. The research on CBR can be dated back to 1975 when the theory of episodic human memory was proposed (Schank, 1975). This theory played an important role in the invention of CBR. A few survey papers give good overview of the research to date on CBR (Bareiss,

1989; Slade, 1991).

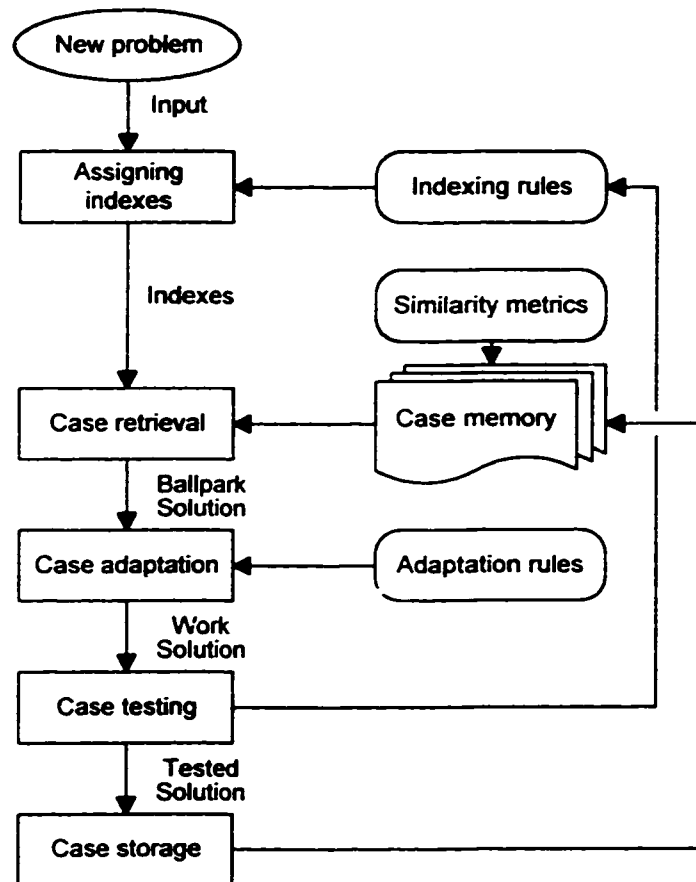


Fig. 4.1. Principles of case-based reasoning systems.

A CBR system includes a case memory to simulate the episodic human memory. The basic element of knowledge is a case, which is the solution to a specific understood problem. These cases are stored in a structure with a rich index. To employ these cases efficiently in problem solving, a CBR system typically consists of the following reasoning modules, as depicted in Fig. 4.1:

- **Assign indexes:** The indexes are the crucial features to characterize an event and determine how cases are stored in the case memory. The purpose of indexing is to allow a cased-based reasoner to retrieve one or more cases that are similar to a new problem.

- **Retrieval:** The indexes of a new problem are used to retrieve similar cases from the case memory. Efficient case retrieval is especially crucial in time-critical situations when the case memory is large.
- **Adaptation:** Since the retrieved cases may not be exactly the same as the new problem, the solutions from these cases need to be modified and adapted to the new problem.
- **Test:** The proposed solution will be tried out to see if it really solves the problem. In the process industry, there are two ways of trying out of a solution: on-plant installations and simulation. For safety reasons, simulation is usually preferred if simulation models are available.
- **Storage:** If the new problem is considered to be conceptually different from the existing cases, a new case needs to be created. The new case needs to be properly indexed with the proven explanation and solution as its content.

Besides the case memory, a CBR system also needs knowledge in order to fulfil the above reasoning tasks. *Indexing rules* identify the features of the input to provide appropriate indexes to the case memory. *Similarity metrics* provide a measure to determine how similar the retrieved cases are to a new problem. *Adaptation rules* decide how to modify the ballpark solution for the new problem.

Because of its successful applications in various fields (Bradshaw, 1987; Hammond, 1989; Kolodner, 1987; Simpson, 1985), CBR is beginning to attract attention from the process industry. The motivation for applying CBR in operation support systems is to make the underlying reasoning mechanism consistent with the natural problem solving of the human operators. As a result, the operation support systems become more powerful in problem solving, facilitate the process of knowledge acquisition, and are better accepted by plant operators. However, the following problems must be solved in order to apply CBR successfully:

- **Selection of appropriate indexing features:** Operational problems, such as the diagnosis of process faults, must be properly characterized by indexing features.

- **Incorporation of system dynamics:** Process operations are dynamic in nature. A feasible CBR must be able to incorporate system dynamics.
- **Incorporation of propagation phenomena:** Because of the system dynamics and signal and material transportation, a single operational problem may propagate different symptoms at different times. The CBR system must have an efficient mechanism to handle the propagation phenomena.

The solution to these problems is dynamic case-based reasoning, which is described in detail below.

4.3 Indexing Feature Interpretation

Typical non-routine operational problems include fault diagnosis and handling, product/grade transition, and process optimization. For the purposes of simplicity, the following discussion will focus on fault diagnosis and handling.

The process being investigated is a bleached chemi-thermo-mechanical pulp (BCTMP) mill, which consists primarily of impregnation, refining, bleaching, and drying stages. This chapter focuses on the bleach plant.

The bleach plant can be divided up into four stages: first stage washing, interstage bleaching and washing, third stage washing and bleaching, and fourth stage washing. The product quality is maintained by ensuring that operating variables fluctuate within permissible ranges. The most important quality variable in a bleach plant is pulp brightness. However, the operating conditions in the bleach plant also affect other pulp quality variables, such as freeness, bulk, breaking length, tear index, burst index, opacity, cost and yield.

One of the main operating objectives is to maintain the pulp brightness at the required level, while minimizing the chemical consumption. Pulp brightness is affected by a dozen operating conditions, including caustic charge, peroxide charge, silicate charge, DTPA, retention time, pulp consistencies, reaction temperatures, wood species and chip quality. There are numerous causes that can lead to low brightness. In order to identify the causes

of low brightness and/or excess chemical consumption, operators have to monitor closely various process variables, such as brightness, residual alkali (pH), temperature, pulp consistency, and so on. The complexity of the pulp production process makes the fault diagnosis and decision making extremely difficult. Plant operations require a knowledge-based operation support system to achieve timely and accurate decision making, and to reduce the operator's working load. Through early problem identification, the operator is able to take corrective actions to minimize off-spec product and avoid excess chemical consumption.

In the operation support system, each case provides information concerning the characterization, explanation and solution of a previously solved problem. For the purposes of case adaptation, a case should also provide information about any relevant causal models. A case with these considerations comprises of the following main components:

- **Case description (*D*):** This provides a short description of the case explains the nature and location of the event, such as “high bleaching temperature in interstage washing and bleaching”.
- **Indexing mechanism (*S*):** This determines the organization of the case memory and the retrieval of similar cases.
- **Propagation model (*T*):** This component gives information about the propagation of the main problem.
- **Validation requirement (*V*):** These suggest the further checks and tests to be performed by operators to validate the case.
- **Solution to the problem (*H*):** This includes an initial cause, a final consequence, achievable plant states, and corrective actions.
- **Relation to causal models (*CM*):** This gives the information to be used by model-based reasoning if the solution from the retrieved cases needs to be adapted.

The case can be represented by

$$C = \langle D, S, V, H, T, CM \rangle . \quad (4.1)$$

Indexes are used to organize the case memory and characterize new problems. The indexing mechanism is one of the most important elements of a CBR systems, and largely determines the problem solving efficiency. The following properties are considered important in an indexing mechanism:

- **Efficiency in case selection and retrieval:** The efficiency can be improved by reducing the case searching space.
- **Distinguishability of various cases:** Maximum distinguishability results in the best diagnosis precision and the most effective solutions.
- **The promptness of decision making:** The sooner a problem is identified, the sooner the corrective actions can be taken to prevent the accidental situation.
- **Generality:** A case is to be retrieved even if it does not exactly match a new problem. In the different reasoning activities of operation support problem solving, the matching criteria used are different. The indexing mechanism needs to provide enough generality to enable multiple matching criteria.

Much of the research on CBR has addressed the indexing problem. Kolodner organized the case memory in a hierarchy of generalized norms (Kolodner, 1983). The search for a matching case is done in a top-down fashion. Extending Kolodner's results, Simpson introduced a set of orthogonal indexing structures in addition to the hierarchical norms so as to improve the efficiency of case retrieval (Simpson, 1985). These indexing mechanisms were proved to be suitable for the specific problems they investigated.

Operation support systems have special requirements in terms of their indexing mechanisms. An operation support system is required to monitor the production process by checking the states of important process variables, namely the monitored variables. The monitored variables are either quality variables, such as brightness and freeness, or important operating variables, such as temperature and specific energy. Rather than the absolute values of measurement, the symbolic attributes that indicate whether the process variables are in normal range play an important role in decision making. If all monitored

variables are in normal ranges, the production process is usually assumed normal, and no nonroutine operational actions need to be taken. If one or more monitored variables go outside normal ranges, a potentially abnormal problem may have occurred. The system has to explain the problem and contribute a solution to it. It is usually advantageous for the operation support system to apply trend information on the evolution of the monitored variables over time, to obtain additional information from the same set of data and to foresee a problem before it actually happens. For complex systems, deep domain knowledge, such as qualitative and quantitative equations describing the material and energy balances, may also be required for accurate and prompt decision making. The system obtains context-dependent information about the production process by interpreting the computational results using the deep domain knowledge.

The operation support problem solving fits neatly into CBR paradigm. The monitored variables serve as *index variables*. The symbolic attributes or symptoms used to identify problems serve as *index features*. The index features representing the static behaviors of the monitored variables are called *static features*, and those representing the dynamic behaviors are *dynamic features*. The information obtained from the deep domain knowledge, such as multivariate statistical process control methods and Kalman filter-based methods, does not have a one-to-one relationship with the physical monitored variables. Artificial index variables, namely *composite variables*, need to be defined. The features attached to the composite variables are called *composite features*.

Feature interpretation is a crucial step in CBR. It maps the measurement space to the feature space. In principle, feature interpretation is similar to the well-established pattern recognition and qualitative interpretation. The task of feature interpretation is to transform the patterns of quantitative measurements to the patterns of qualitative features. Assume that \aleph is the n-dimensional measurement space that covers the possible sensor output ranges of all index variables, and Φ is the m-dimensional feature space that characterizes the process operation problems, feature interpretation can be represented by:

$$\xi: \aleph \rightarrow \Phi. \quad (4.2)$$

That is, from a pattern $X \in \mathcal{N}$ to a pattern $I \in \Phi$:

$$\begin{aligned} X &= (x_1, x_2, \dots, x_n) \\ I &= (\varphi_1, \varphi_2, \dots, \varphi_m). \end{aligned} \tag{4.3}$$

In this chapter, Φ is composed of a static feature space Φ^s , a dynamic feature space Φ^d and a composite feature space Φ^c :

$$\Phi = \Phi^s \oplus \Phi^d \oplus \Phi^c. \tag{4.4}$$

4.3.1 Static index features

Static index features refer to those formulated by the instantaneous values of process variables. They are obtained by comparing the measurements of process variables to their standard limits. In the pulp and paper production, grade recipes often use standard limits such as:

UH - ultimate high	UL - ultimate low
HH - very high	LL - very low
MH - moderate high	ML - moderate low
NH - normal high	NL - normal low
DS - desired setting	

The “ultimate” high and low represent the physical constraints of the sensor or the utmost variation of the process variable. The “normal” high and low represent the range within which the process variable is defined as normal. The interpretation of static features is used to generate the symbolic state attributes from the physical values of process variables, as illustrated in Fig. 4.2. Therefore, the static features are the symbolic state attributes in the set $\{-4, -3, -2, -1, 0, +1, +2, +3, +4\}$. Considering the noise corruption found in sensory data, simple low pass filter techniques and sophisticated statistical methods can be applied to reduce the influence of noise in feature interpretation.

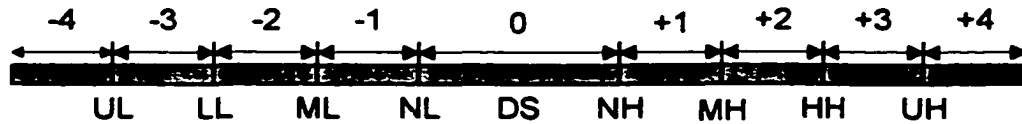


Fig. 4.2. Interpretation of static features.

It should be pointed out that not all process variables are defined in such detail. The quality variables, such as brightness and freeness, usually have detailed definitions. However, the operating variables, such as temperature and consistency, often have only two standard limits: high and low. Accordingly, the static features they have are: normal (0), high (+1) and low (-1).

4.3.2 Dynamic index features

An efficient means of incorporating process dynamics is to use dynamic features of the process variables. Dynamic features refer to those representing the dynamic behavior of the process variables in a time period of interest. Static features are adequate to describe static systems and dynamic systems in the steady state, but not for dynamic systems in transient state. Pulp and paper production is a dynamic process. When a process fault occurs, the production process will undergo a transient process until it reaches a new steady state. The dynamic behavior of the process variables is very important in production state assessment, fault detection and diagnosis (Stephanopoulos, 1991).

Two types of dynamic features are introduced in this chapter: *instant dynamic features* and *interval dynamic features*. “Instant” dynamic features describe the changing speeds of index variables in terms of the slope of the tangent line at a time instant, such as α at time t_1 in Fig. 4.3. There are three different instant dynamic features:

- (I, α) - increasing with a speed α
- (D, α) - decreasing with a speed α
- (U) – unchanging.

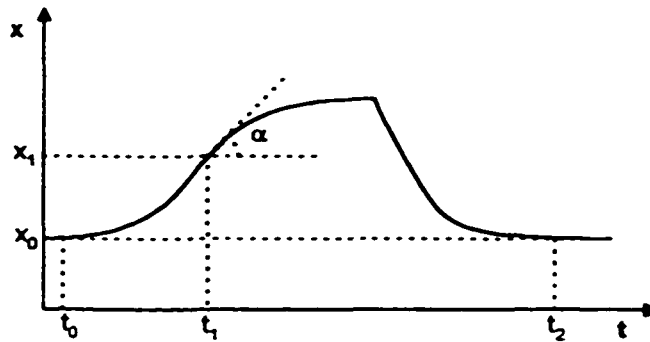


Fig. 4.3. Evolution of a process variable.

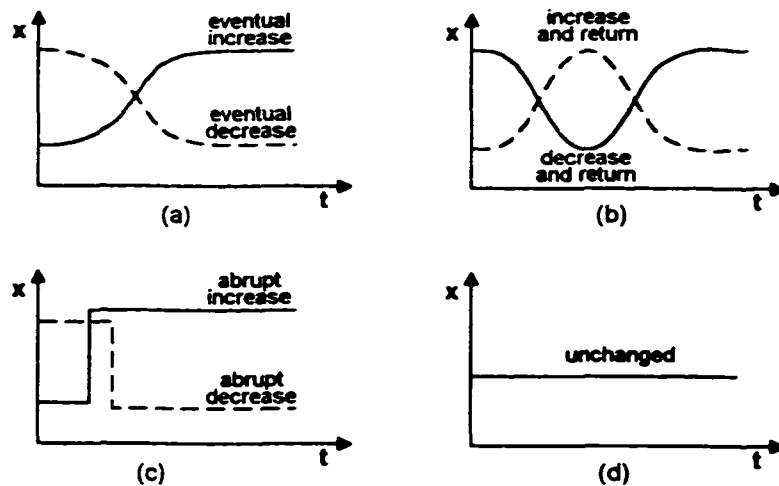


Fig. 4.4. Patterns of interval dynamic features.

The interval dynamic features present the evolution of a process variable over a period of time, say $[t_0, t_2]$ in Fig. 4.3. Fig. 4.4a-d show seven different interval dynamic features:

- (EI, M) - eventual increase with a magnitude M
- (ED, M) - eventual decrease with a magnitude M
- (AI, M) - abrupt increase with a magnitude M
- (AD, M) - abrupt decrease with a magnitude M

- (IR, M) - increase and then return with a peak magnitude M
- (DR, M) - decrease and then return with a peak magnitude M
- (CS)- constant.

It can be seen that a dynamic feature is represented by a doublet, such as (EI, M). The first element describes the qualitative changing pattern, and the second describes its quantitative magnitude. In most situations, the qualitative element is sufficient to characterize an operational problem. Therefore, in the following discussion, the quantitative elements M and α are ignored.

The interpretation of instant dynamic features is used to calculate the time derivative of a process variable in the neighborhood of a specified time instant. Many methods can be applied to carry out this calculation. However, it is much more difficult to interpret the interval dynamic features. Fortunately, a few methods proposed for the pattern recognition of sensory data can be applied for the purpose of dynamic interval feature interpretation (Cheung and Stephanopoulos, 1990; Janusz and Venkatasubramanian, 1991; Konstantinov and Yoshida, 1992; Rengaswamy and Venkatasubramanian, 1995).

One of the benefits of introducing dynamic features in dynamic case based reasoning (DCBR) is the improved case distinguishability. Pulp and paper production has many potential process faults that lead to distinct static and/or dynamic features. The static features alone may not be sufficient to distinguish these faults. For example, because of material cycling and control loops, a process variable x may change as depicted in Fig. 4.3 when a fault F occurs at t_0 . From the static feature of x at time t_2 , no abnormality is observable. However, with the interval dynamic feature in the period of $[t_0, t_2]$, the system can tell that the process variable has undergone an abnormal cycle. The pattern of this cycle is the important information for accurately identifying the fault.

Another benefit is the improved promptness of decision making; that is, the earlier identification of process faults. Using static features only, no diagnostic result can be obtained until the index variables exceed their standard limits. With dynamic features, however, an intelligent system can identify a fault and propose corrective actions well before the index variables go beyond their limits. In Fig. 4.3, for example, the variable x

has not reached the high limit x_1 at time t_1 . However, with the information of the changing speed at t_1 , the system will be able to predict that the variable will reach the high limit in a short time. With this foresight, corrective actions can be taken to prevent x from going up any further.

4.3.3 Composite index features

Complex index features are introduced to enable DCBR to take advantage of other well-established qualitative/quantitative fault diagnosis and decision making algorithms as well as available deep domain knowledge. It is an efficient means of enhancing the problem solving capability of DCBR systems.

Unlike static and dynamic index features, a composite index feature is not attached to any individual process variables. It is an indication of the combined effect of a number of process variables. In principle, all kinds of reasoning and computation algorithms can be applied to generate composite index features by using relational and context-dependent knowledge about the investigated process. DCBR treats these algorithms as data processing modules. The results from these modules are considered as the features of a number of non-physical (artificially defined) index variables, namely composite index variables. These non-physical variables are applied in case indexes in the same way as the physical index variables. Since the composite index features are more knowledge intensive, they can improve the reasoning efficiency of DCBR.

Consider a Kalman filter-based fault diagnostic algorithm as an example (Chapter 6). A bank of Kalman filters KF_i is constructed with respect to potential fault modes H_i ($i = 0, 1, \dots, M$). The probability that fault H_i is true, $p_i(k)$, is calculated from the corresponding estimation errors, and plays an important role in fault diagnosis and control system reorganization subsequent to a fault. DCBR can simply define $M+1$ non-physical index variables p_i ($i = 0, 1, 2, \dots, M$). The computational results of $p_i(k)$ from the Kalman filter-based algorithms can then be easily applied in case-based reasoning.

The following sections will discuss the indexing mechanism of DCBR. Two indexing paths, the abnormal symptom-based index path and the problem description-based index path, will be introduced.

4.4 Abnormal Symptom-Based Indexing Path

In the previous sections, it has been pointed out that a number of process variables (which may include non-physical variables, namely composite index variables) will present abnormal symptoms when a process fault occurs. The design of the abnormal symptom-based indexing path is based on this observation.

4.4.1 Time-tagged indexes

Because of different dynamic behaviors (time delay and time constant), some variables respond to a fault very quickly, whereas the others respond slowly. The quickly responding variables present abnormal symptoms first, and then the slowly responding variables. For example, a failure of a steam control valve in the interstage bleaching and washing stage leads to a change in the bleaching temperature almost immediately. However, the interstage brightness will not change significantly until one hour later, and the final brightness three hours later. This fact indicates that the symptoms of a process fault change with time. Fig. 4.6 shows that after a fault F_i happens at time t_0 , the symptoms will change from $S_i(1)$ at t_1 , to $S_i(j)$ at t_j , and finally $S_i(n_i)$ at t_{n_i} .

This phenomenon makes the selection of index variables difficult. If only quickly responding variables are selected as index variables, the fault can be identified in the early stages. However, the reliability and distinguishability of case retrieval may not be satisfactory. To achieve the highest reliability and distinguishability, all the affected process variables have to be used for case indexing. Denoting the indexes of a case (a previously solved problem) by $S = [X, \Phi]$, with X representing index variables and Φ the corresponding index features, the case is perfectly confirmed if and only if

$$X = Y \quad \Phi = \Psi \quad (4.5)$$

where Y are process variables that present abnormal symptoms, and Ψ are the corresponding symptoms. The case cannot be confirmed until the slowest responding index variable shows abnormal symptoms, which may take hours. Time-tagged indexes are proposed to solve this problem.

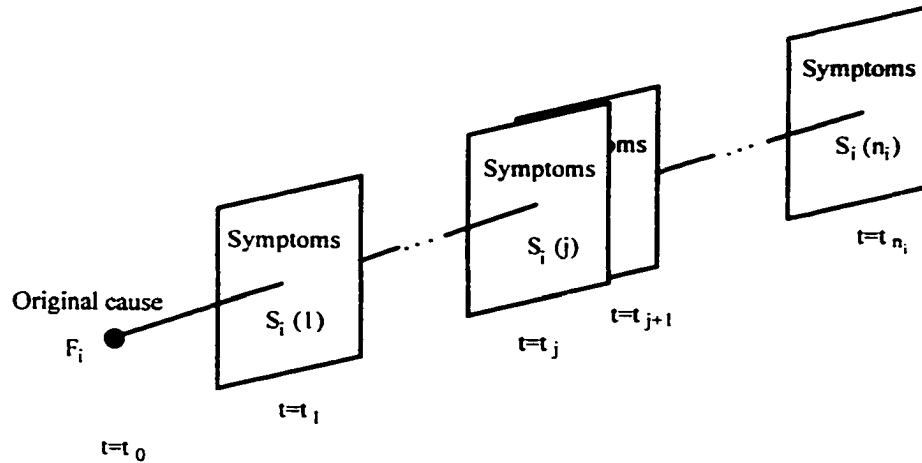


Fig. 4.5. Symptoms of a fault changing with time.

Time-tagged indexes classify the index variables and features according to their responding times. A chain of propagation cases, C_{ij} , is defined under a principal case, C_i . The subscript i represents the i th fault, F_i , and j represents the j th propagation phase. From Fig. 4.5, it is obvious that

$$F_i \xrightarrow{t_1=t_1-t_0} C_{i1} \xrightarrow{t_2=t_2-t_1} C_{i2} \dots \xrightarrow{t_j=t_j-t_{j-1}} C_{ij} \dots \xrightarrow{t_n=t_n-t_{n-1}} C_{in} \dots \quad (4.6)$$

The indexes of the propagation case C_{ij} are

$$S_i(j) = [X_i, \Phi_i(j)] \quad (4.7)$$

where X_i is composed of the index variables for case C_i , and $\Phi_i(j)$ are the corresponding features of X_i in the j th propagation phase. Denoting $X_i(j) \subseteq X_i$ as the index variables that have abnormal symptoms in $\Phi_i(j)$, then $X_i(1)$ consists of only the quickest responding variables, whereas $X_i(n_i)$ consists of both quickly and slowly responding variables; thus

$$X_i(1) \subseteq X_i(2) \subseteq \dots \subseteq X_i(n_i). \quad (4.8)$$

The time-tagged indexes and the chained propagation case structure make it possible to achieve early and accurate fault identification. As soon as the first cycle of abnormal symptoms in $X_i(1)$ presents, the system will assume the occurrence of F_i , though with low certainty. As more variables present abnormal symptoms, the system continues to confirm and verify the previous diagnostic results until the desired certainty level is achieved. Another advantage of this structure is the convenience in predicting the consequences of process faults. The abnormal symptoms imply the effects of the fault on the production.

4.4.2 Indexing mechanism

The indexes are made up of four fields:

- indexed case: the case that is characterized by the indexes;
- index variables: the variables whose symptoms are employed to index the cases;
- index features: the static, dynamic and composite symptoms of the index variables;
- similarity evaluator: the mechanism used to calculate the case similarity (to a new problem), a nonnegative number within $[0, 1]$.

A similarity of zero leads to a rejected case. At the other extreme, a similarity of one leads to a validated case. The effect of an index feature on case similarity evaluation is either positive or negative. On the positive side, the matching of an index feature increases the similarity or even validates the case; on the negative side, the matching of a feature decreases the similarity or even discards the case. For example, low brightness and

high peroxide charge are the positive features of a bleaching temperature control problem. However, low peroxide residual is a negative feature.

From the discussions in Section 4.4.1, it is obvious that $S_i(j) = [X_i, \Phi_i(j)]$ will be matched if the new problem is exactly described by the propagation case C_{ij} . The matching of $S_i(j)$ ($j < n_i$) is a necessary condition to confirm the case, but not a sufficient condition. In the early propagation phase, e.g. $j = 1$, many important symptoms have not yet emerged. Even if $S_i(1)$ are exactly matched, the case may not be confirmed; that is, the case similarity may still be less than one. To suit the chained propagation case structure, DCBR applies a two-step case similarity evaluation method:

Step 1: Evaluate the *phase matching degree*, κ_{ij} , by comparing the index features of the propagation case C_{ij} with the symptoms of the new problem;

Step 2: Calculate the *case similarity*, λ_{ij} , by using κ_{ij} and the *phase completeness*, η_{ij} :

$$\lambda_{ij} = \eta_{ij} \cdot \kappa_{ij}. \quad (4.9)$$

In the above formulation, κ_{ij} represents how close the symptoms of the new problem are to the index features of C_{ij} . η_{ij} is defined as the case similarity when all index features of C_{ij} are exactly matched. It is a measure of the completeness of the propagation process. Obviously, $\eta_{i n_i} = 1$.

Two types of case similarity evaluators are proposed: that based on individual index variables, and that based on group index variables.

4.4.2.1 Similarity evaluator based on individual index variables

In the similarity evaluator based on individual index variables, each index feature is attached with an *index strength* that represents the importance of the feature in the calculation of the phase matching degree. A positive index feature represents a positive

contribution to the phase matching degree. The contribution of an individual index feature does not rely on other index features. For example, if the strength of high bleach temperature is 0.4, the matching of this feature will increase the phase matching degree by 0.4 no matter whether other index features are matched or not.

For the simplicity of similarity evaluation, the index features are given two different natures: *equivalency feature* and *difference feature*. An equivalency feature contributes the credit specified by its strength to a case if it is matched. Conversely, a difference feature contributes the credit if it is not matched. Before going further into the calculation of case similarity, the following definition is introduced:

Definition 1: The weighted equivalence function is defined as

$$EQW(\Psi, \Phi, \Omega) = \sum_{k=1}^l eqw(\varphi_k, \phi_k, \omega_k) = \sum_{k=1}^l \omega_k \cdot eq(\varphi_k, \phi_k) \quad (4.10)$$

where the weighted difference function is defined as

$$DFW(\Psi, \Phi, \Omega) = \sum_{k=1}^l dfw(\varphi_k, \phi_k, \omega_k) = \sum_{k=1}^l \omega_k \cdot df(\varphi_k, \phi_k) \quad (4.11)$$

where $\Psi = \{\varphi_k\} \in \mathfrak{R}^l$, $\Phi = \{\phi_k\} \in \mathfrak{R}^l$, $\Omega = \{\omega_k\} \in \mathfrak{R}^l$, and

$$eq(\varphi, \phi) = \begin{cases} 0 & \forall \varphi \neq \phi \\ 1 & \forall \varphi = \phi \end{cases} \quad (4.12)$$

$$df(\varphi, \phi) = \neg eq(\varphi, \phi)$$

where \neg represents the negation operator. The weighted combination function is defined as

$$COM(\Psi_e, \Phi_e, \Omega_e; \Psi_d, \Phi_d, \Omega_d) = EQW(\Psi_e, \Phi_e, \Omega_e) + DFW(\Psi_d, \Phi_d, \Omega_d). \quad (4.13)$$

Using the above definitions, the calculation of case similarity can be easily carried out.

Macchion and Vo classified the features into five types according to their contributions to case validation: sufficiency, necessity, exclusion, remembrance and inhibition features (Macchion and Vo, 1993). In DCBR the features are classified into three types:

- Sufficiency features: the matching of a sufficiency feature leads to $\kappa_{ij} = 1$;
- Necessity features: the matching of a necessity feature leads to a case being retained for consideration. In other words, the violation of a necessity feature leads to $\kappa_{ij} = 0$;
- Grading features: the matching of a grading features increases or decreases κ_{ij} .

The different types of index features can be represented by the values of their index strengths. A sufficiency feature is assigned a strength +10, a necessity feature is assigned a strength -10 of difference nature, whereas a grading feature is assigned a strength within (-1, +1). Considering that a case may include both equivalence and difference features, the indexes of case C_i at the j th propagation phase can be represented by:

$$S_i(j) = (\hat{S}_i(j), \eta_{ij}) \quad (4.14)$$

$$\hat{S}_i(j) = (X_{ie}(j), \Phi_{ie}(j), \Omega_{ie}(j); X_{id}(j), \Phi_{id}(j), \Omega_{id}(j))$$

where $X_{ie} \in \mathfrak{R}^1$, $\Phi_{ie} \in \mathfrak{R}^1$ and $\Omega_{ie} \in \mathfrak{R}^1$ are the equivalence index variables, index features and their strengths, respectively; $X_{id} \in \mathfrak{R}^2$, $\Phi_{id} \in \mathfrak{R}^2$ and $\Omega_{id} \in \mathfrak{R}^2$ are the difference index variables, index features and their strengths, respectively. Assume that X_{ie} and X_{id} present the symptoms Ψ_{ie} and Ψ_{id} under a new problem, respectively, the similarity of the case C_i at the j th propagation phase is then calculated as:

$$\kappa_{ij} = \text{Min}\{1, \text{Max}\{0, \text{COM}(S_i(j))\}\} \quad (4.15)$$

$$\lambda_{ij} = \kappa_{ij} \cdot \eta_{ij} \quad (4.16)$$

where $\bar{S}_i(j)$ is $\hat{S}_i(j)$ shown in Eqn. (4.14) with X_{ie} and X_{id} substituted by Ψ_{ie} and Ψ_{id} , respectively. The function $COM(\cdot)$ is defined in Eqn. (4.13).

4.4.2.2 Similarity evaluator based on group variables

In the similarity evaluator based on group variables, a phase matching degree is granted to a case only if a group of index variables simultaneously matches their predefined features. For example, low brightness in the interstage bleaching and washing stage has no means for similarity evaluation. Only if low brightness and low bleaching temperature occur simultaneously, is the case of bleaching temperature control failure granted a phase matching degree of 1.0. For the convenience of similarity calculation, the following functions are defined:

Definition 2. The conjoint equivalence function is defined as

$$CEQ(\Psi, \Phi, \rho) = \rho \cdot [eq(\varphi_1, \phi_1) \cap eq(\varphi_2, \phi_2) \cap \dots \cap eq(\varphi_l, \phi_l)] = \rho \cdot \bigcap_{k=1}^l eq(\varphi_k, \phi_k) \quad (4.17)$$

where $\Psi = \{\varphi_k\} \in \mathcal{R}^l$, $\Phi = \{\phi_k\} \in \mathcal{R}^l$ and ρ is a scalar number within [0, 1]. The conjoint difference function is defined as

$$CDF(\Psi, \Phi, \rho) = \rho \cdot [df(\varphi_1, \phi_1) \cap df(\varphi_2, \phi_2) \cap \dots \cap df(\varphi_l, \phi_l)] = \rho \cdot \bigcap_{k=1}^l df(\varphi_k, \phi_k). \quad (4.18)$$

The combined conjoint function is defined as

$$CCOM(\Psi_e, \Phi_e; \Psi_d, \Phi_d; \rho) = \rho \cdot \left(\bigcap_{k=1}^l eq(\varphi_{ek}, \phi_{ek}) \right) \cap \left(\bigcap_{k=1}^m df(\varphi_{dk}, \phi_{dk}) \right) \quad (4.19)$$

where $\Psi_e = \{\varphi_{ek}\} \in \mathcal{R}^l$, $\Phi_e = \{\phi_{ek}\} \in \mathcal{R}^l$, $\Psi_d = \{\varphi_{dk}\} \in \mathcal{R}^m$, and $\Phi_d = \{\phi_{dk}\} \in \mathcal{R}^m$.

The similarity evaluation can be easily obtained by applying the above definitions. The indexes of the case C_{ij} are represented by several groups of index features, $S_i^h(j)$, the attached matching degrees, $\rho_i^h(j)$, and the phase completeness, η_{ij} :

$$S_i(j) = (S_i^1(j), S_i^2(j), \dots, S_i^{g_{ij}}(j); \eta_{ij}) \quad (4.20)$$

$$S_i^h(j) = (X_{ie}^h(j), \Phi_{ie}^h(j); X_{id}^h(j), \Phi_{id}^h(j); \rho_i^h(j)) \quad h = 1, 2, \dots, g_{ij}$$

where g_{ij} is the number of the index variable groups for C_{ij} ; X_{ie}^h and Φ_{ie}^h are the equivalence index variables and corresponding features of group h , respectively; X_{id}^h and Φ_{id}^h are the difference index variables and corresponding features, respectively; and ρ_i^h is the matching degree granted to the propagation phase C_{ij} if both the equivalence and difference features in group h are satisfied. The phase matching degree κ_{ij} and the case similarity λ_{ij} can be calculated by the following equations:

$$\kappa_{ij} = \max\{0, \min\{1, \max\{CCOM(\bar{S}_i^h(j)) \mid h = 1, 2, \dots, g_{ij}\}\}\} \quad (4.21)$$

$$\lambda_{ij} = \kappa_{ij} \cdot \eta_{ij} \quad (4.22)$$

where $\bar{S}_i^h(j)$ is $S_i^h(j)$ given in Eqn. (4.20) with X_{ie}^h and X_{id}^h replaced by their symptoms under the new problem being investigated.

4.4.3 Case retrieval

The purpose of case retrieval is to select and retrieve from the case memory the cases that serve best in problem solving. The retrieval of cases is done in three steps: collecting plausible cases, searching for a perfectly matching case, and selecting the best matching case.

The first step is to find a set of plausible cases by pruning the case memory using necessity features. DCBR defines a pruning index $S_i(0)$ that contains the necessity features of case C_i . The matching of $S_i(0)$ determines whether C_i is a plausible case that is worthy for further evaluation. The size of the plausible case set depends on the selection of $S_i(0)$. With multiple pruning index features in $S_i(0)$, the case memory can be organized hierarchically, and the number of the plausible cases can be reduced.

DCBR searches in the plausible case set for a perfectly matching case, the case whose sufficiency feature is matched or whose similarity is 1.0. If a perfectly matching case is found, DCBR will retrieve the case for problem solving. Otherwise, the similarities of all the plausible cases will be evaluated to find the best matching case.

The best matching case is the one with the greatest case similarity to the problem being investigated. The plausible cases are ranked in terms of their similarities. The best matching propagation phase p of case C_i is determined using Eqn. (4.15) or (4.21) if:

$$\kappa_{ip} = \max\{\kappa_{i1}, \kappa_{i2}, \dots, \kappa_{in_i}\}. \quad (4.23)$$

The similarity of case C_i to the new problem is

$$\lambda_i = \kappa_{ip} \cdot \eta_{ip}. \quad (4.24)$$

Case C_q is the best matching case if and only if

$$\lambda_q = \lambda_{\max} = \max\{\lambda_1, \lambda_2, \dots, \lambda_L\} \quad (4.25)$$

where L is the size of the plausible case set.

A threshold, $\bar{\lambda}$, is defined. Only if $\lambda_q = \lambda_{\max} \geq \bar{\lambda}$, will C_q be used for problem solving. Otherwise, temporal reasoning is applied for further case evaluation. Comparing the case index features at phases p and $p+1$, $\Phi_{qe}(p)$ and $\Phi_{qe}(p+1)$, and denoting the features that are different in the two phases by $\tilde{\Phi}_{qe}(p)$ and $\tilde{\Phi}_{qe}(p+1)$, the symptoms of the investigated problem must have the change in the time interval t'_p :

$$\tilde{\Phi}_{qe}(p) \xrightarrow{t'_p} \tilde{\Phi}_{qe}(p+1). \quad (4.26)$$

If the symptoms of the new problem do not have such changes, case C_q is dropped from consideration. The second best matching case will be considered. Using this

mechanism, temporal reasoning can be easily incorporated in further case similarity evaluation.

4.4.4 Remembrance factor for case memory separation

Human operators have an efficient memory refreshing and forgetting capability. They remember recent operational problems well and remind themselves of these problems with priorities, but not those that are far in the past. DCBR simulates this human behavior by introducing a new parameter, namely the remembrance factor, to improve the reasoning efficiency.

The remembrance factor, π_i , represents the recentness and frequency of occurrence of case C_i . Considering a moving window from month l to month M (present time), π_i is defined as

$$\pi_i = \sum_{k=1}^M \frac{N_{ik}}{\alpha^{M-k}} \quad (4.27)$$

where N_{ik} is the number of occurrences of the case in the k th month; $\alpha \geq 1$ is the forgetting rate. It is obvious that every occurrence contributes to the remembrance factor. However, the past occurrences are being forgotten at a rate of $1/\alpha$. The forgetting rate can be appropriately selected according to the operational requirements.

For the convenience of updating, Eqn. (4.27) needs to be transformed into a recursive form:

$$\begin{aligned} \pi_{i,k,l+1} &= \pi_{i,k,l} + 1 \\ \pi_{i,k+1,0} &= \frac{1}{\alpha} \pi_{i,k,N_s} \\ \pi_{i,0,0} &= 1 \end{aligned} \quad (4.28)$$

where $\pi_{i,k,l}$ is the remembrance factor of case C_i in the k th month after the l th occurrence.

The cases with very small remembrance factors have not occurred for a long time and may never occur again because of process innovation. In a DCBR system with a large case memory, these cases can be considered inactive, and can be separated from the other cases. The case memory is thus separated into two parts: the active case memory and the inactive case memory. Only if the active case memory is not amenable for problem solving, is the inactive case memory searched. By doing so, the case searching space can be significantly reduced.

The remembrance factor can also serve as an additional case selection criterion. If two or more cases have an equal similarity, the one with the larger remembrance factor will be selected as the best matching case.

4.5 Problem Description-Based Indexing Path

One of the significant capabilities of human operators is the application of the experience obtained in one situation to another. In the BCTMP pulp production process shown in Fig. 4.2, there are a number of units with very similar structures, such as the three refining processes, the two bleaching processes, and the four washing processes. The resemblance in structure enables the solution for one unit to be easily adapted to another. For example, the knowledge about the steam control valve problem in the primary refiner can be easily applied to solve a similar problem in the secondary refiner. An abnormal symptom-based indexing path cannot recall cases that are structurally or functionally similar. A different indexing path is required.

DCBR applies a secondary indexing path, namely the problem description-based indexing path, to retrieve structurally and functionally similar cases for problem solving. The indexing path employs the indexes of six fields in the form

$$Index(IC, PL, PS, AV, CS, CV). \quad (4.29)$$

The physical meanings of the fields are:

- *Indexed case (IC)* is the case characterized by the indexes. Since the cases are numbered, for instance CASE #32, IC is actually an integer.
- *Problem location (PL)* indicates the process unit in which the problem originates. *PL* consists of two components: the type of the process unit (UNIT), and the stage (STAGE), as represented by the following doublet

$$PL (UNIT, STAGE).$$

According to the decomposition of the BCTMP process, UNIT belongs to the set {*Impregnator, Refiner, Bleaching, Washing, Dryer*}. The value of STAGE depends on the type of UNIT. For example, the BCTMP process has three refiners: primary refiner, secondary refiner and tertiary refiner. Therefore, for UNIT *Refiner*, STAGE belongs to the set {*Primary, Secondary, Tertiary*}. The primary refiner is represented by *PL (Refiner, Primary)*.

- *Problem source (PS)* gives information about the original faulty device or operation that causes the problem. The common devices include {*Valve, Pump, Sensor, Controller, etc.*}. *PS* needs also to specify the function of the device in production and the trouble it has. A triple is sufficient to give this description:

$$PS (DEVICE, FUNCTION, FAULT).$$

For example, *PS (Valve, Temperature, Stuck)* represents that the valve for temperature control is stuck.

- *Affected operating variable (AV)* is the operating variable that is directly affected by the process fault. Operating *variables* have a direct influence on pulp quality and production profits. For example, the operating variables that affect pulp brightness in the bleach plant include bleaching temperature, production rate

(retention time), pulp consistency, peroxide charge, caustic charge, silicate charge and DTPA, etc., as shown in Fig. 4.6. A process fault will first affect one of these operating variables before it affects pulp quality. *AV* can be represented by

$$AV(VARIABLE, STATE)$$

where *VARIABLE* belongs to {*Temperature, Production_rate, Consistency, Peroxide, etc.*}. *STATE* belongs to {*Low, Normal, High*}. A low bleaching temperature can be represented by *AV(Temperature, Low)*.

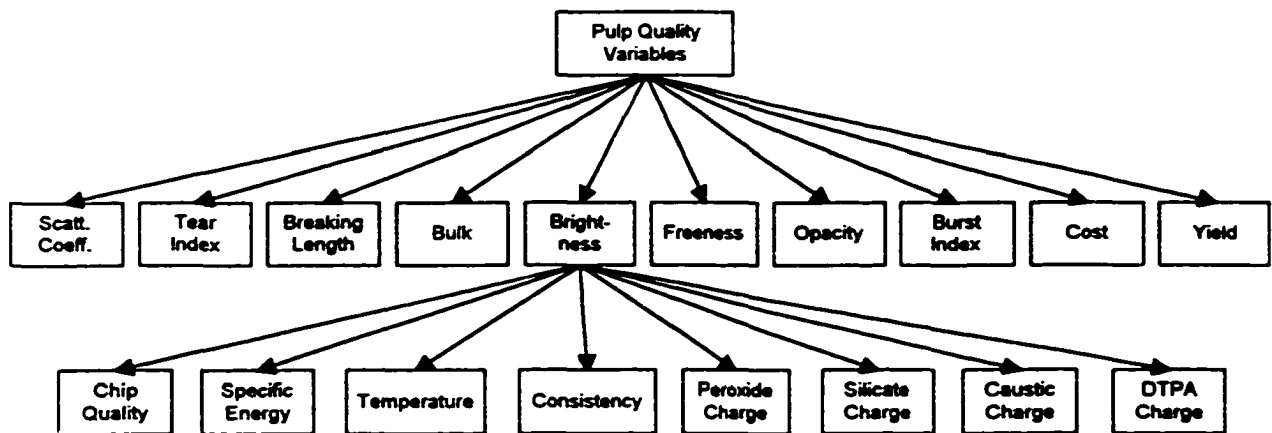


Fig. 4.6. Quality variables affected by operating variables.

- *Consequence (CS)* indicates the main effect of *the* fault on production if no corrective action is taken. The general classification of the consequences is shown in Fig. 4.7. *CS* can be described by

$$CS(EFFECT, STAGE).$$

The off-range third stage brightness is represented by *CS(Brightness, Thirdstage)*.

- *Corrective operating variable (CV)* is the operating variable that is adjusted to restore the production. If *AV* is still manipulatable, the best choice of *CV* is *AV* since it implies exact restoration of the production. If *AV* is no longer

manipulatable, other operating variables can be adjusted to compensate for the effect of *AV*. *CV* is represented by a triple

$$CV (VARIABLE, CHANGE)$$

where *VARIABLE* belongs to the same set as for *AV*, and *CHANGE* belongs to $\{Increase, Decrease\}$.

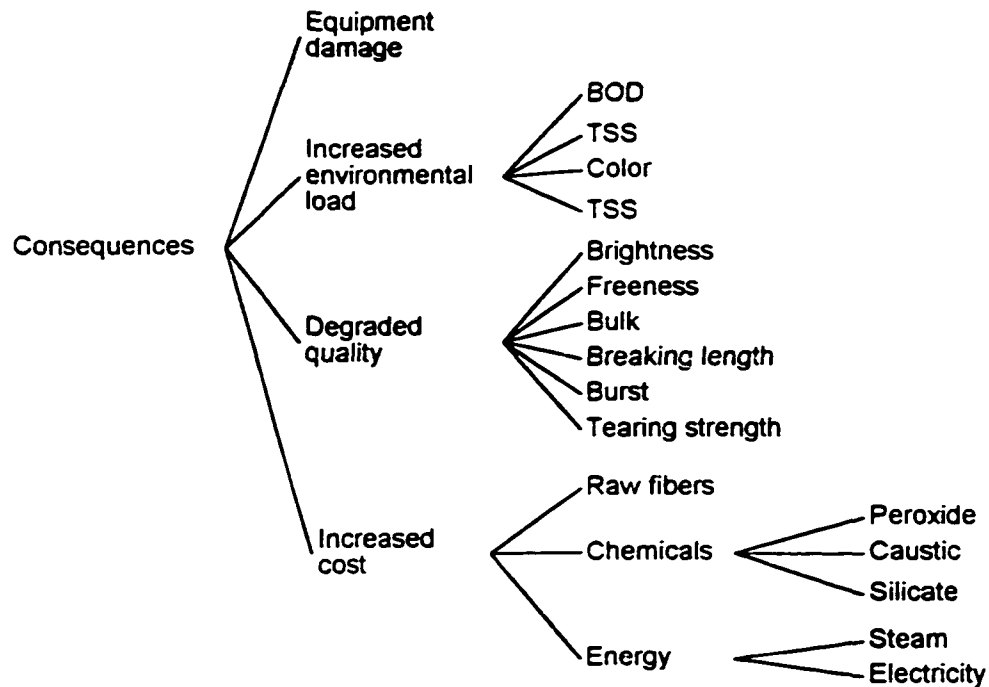


Fig. 4.7. Classification of consequences.

According to the requirements of problem solving, the system selects one or more fields to be the matching criteria for recalling the similar cases.

The problem description-based indexing path characterizes a case by a number of structural and functional definitions. The retrieval of similar cases depends totally on the problem solving goal. This indexing path can be applied to solve operational problems that are not known exactly. In the next section, the retrieval of cases using the problem description-based indexing path will be illustrated using an example.

4.6 Operational Problem Solving

Based on the discussions in the previous sections, the structure of DCBR can be sketched as in Fig. 4.8. The following observations can be made:

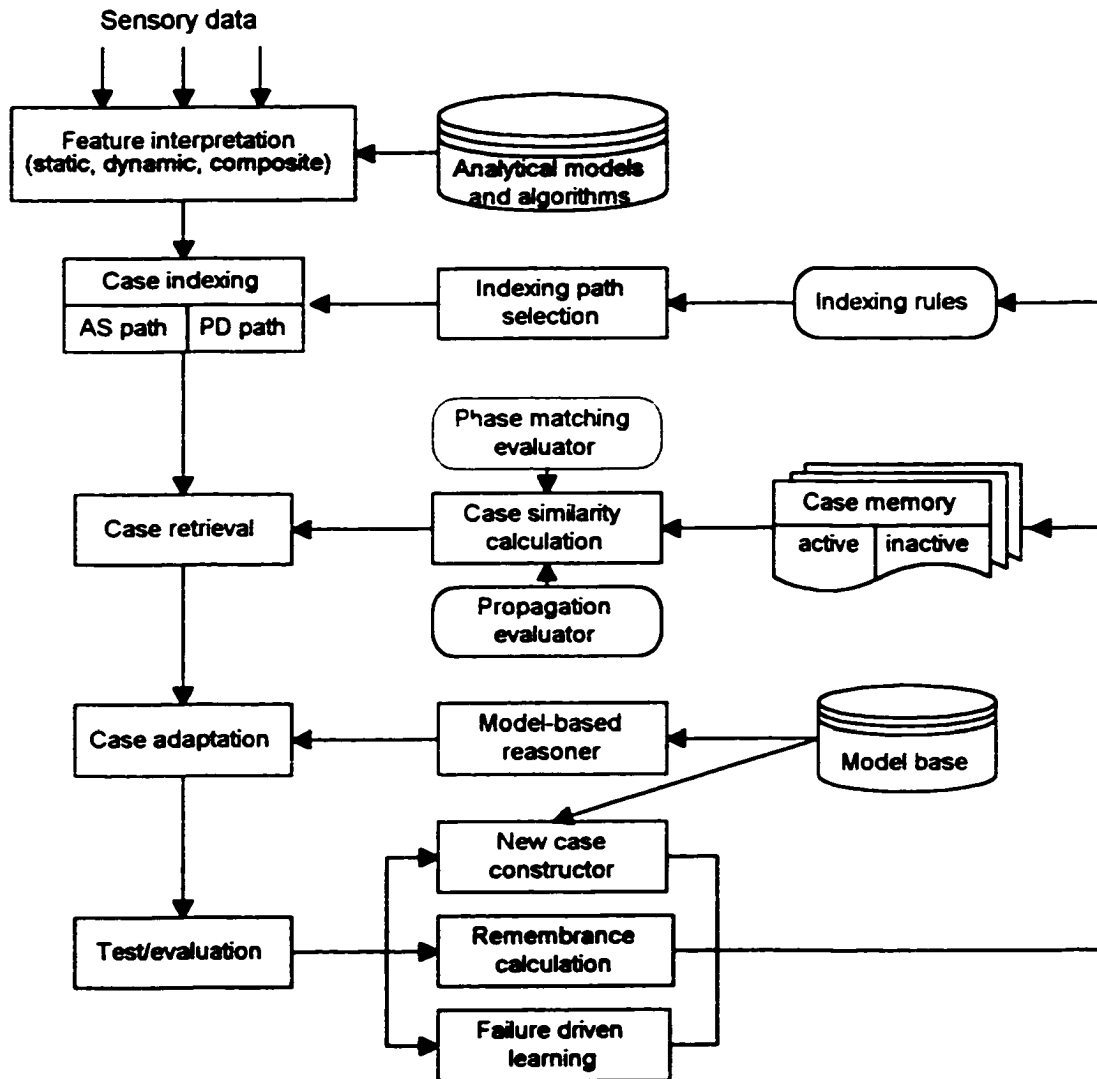


Fig. 4.8. Structure of a dynamic case-based reasoning system.

- Not only the static features, but also the dynamic and composite features are interpreted from sensory data. Analytical models and algorithms can be used for dynamic and composite feature interpretation.
- Instead of one indexing path, DCBR applies two indexing paths: the abnormal symptom (AS)-based path and the problem description (PD)-based path. According to the problem analysis and decomposition, DCBR automatically selects the appropriate indexing path for case retrieval and problem solving.
- There are two new modules to support case similarity evaluation. A phase matching evaluator calculates the phase matching degree of a case to the new problem, while a propagation evaluator employs temporal reasoning for further case similarity evaluation as described by Eqn. (4.26) using time tagged indexes.
- The remembrance calculation module updates the remembrance factors of existing cases. The case memory is divided into two parts: those with large remembrance factors are stored in the active memory, and the rest in the inactive memory. Only if the search in the active memory does not solve the problem, is the inactive memory searched. In this way, the search space is reduced.

When an operational problem occurs, the problem solving procedures depend on how well the case memory covers the problem.

4.6.1 Problem solving with perfectly matching cases

A perfectly matching case is defined in Section 4.4.3 as a case whose sufficiency feature is matched, or whose similarity is 1.0. If a perfectly matching case is retrieved by the abnormal symptom-based indexing path, the problem solving is simple. DCBR simply applies the explanation and solution from the perfectly matching case (that is, component *H* in the case representation (4.1)) directly to the problem being investigated. No MBR activity is required for problem solving in this situation.

The operational actions required to handle an operational problem are usually composed of action procedures and action magnitudes. To improve the generality of a

case, mathematical formulations instead of fixed values are embedded for the calculation of action magnitude.

Take the correction scheme that restores pulp freeness by adjusting the specific energy as an example. The case solution indicates that the primary refiner requires a higher specific energy setting in order to obtain the desired pulp freeness. However, the next question is how much specific energy is required to increase. This depends on the current pulp freeness and the desired pulp freeness. The pulp freeness csf in a refiner follows the following equation

$$csf = \exp(k_1 - k_2 \cdot KWH) - k_4 \cdot (Caustic - k_5) + csf_0 \quad (4.30)$$

where $k_1 \sim k_5$ are the constants depending on the production rate, pulp grade, chip quality, plate age and plate gap, etc; KWH is the specific energy; $Caustic$ is the chemical charge; and csf_0 is the pulp freeness before entering the refiner. The required change in KWH can thus be calculated by

$$\Delta KWH = -\frac{1}{k_2} \ln(1 + \Delta csf \cdot \exp(-k_1 + k_2 \cdot KWH)). \quad (4.31)$$

This mathematical formulation is embedded in the case in the place of the action magnitude. One case is able to cover all the situations where the specific energy needs to be adjusted.

4.6.2 Problem solving with a partially matching case

A plausible case with a best phase matching degree $\kappa_{op} < 1$ is called a partially matching case. The solution from a partially matching case cannot be directly applied to solve the problem. If the investigated problem does not have an exact matching case, some adaptation is required. Research on CBR led to a few case adaptation approaches, such as the feasible adaptation strategy that integrates CBR with MBR proposed by Portinale and coworkers (1993).

Partially matching is the result of an inconsistency between the symptoms of the new problem and the indexes of the case. The description of the inconsistency is used to retrieve appropriate adaptation rules. The following notations are first introduced:

$$\begin{aligned} D_1 &= \Phi_q^+(p) - \Psi & D_2 &= \Psi \cap \Phi_q^-(p) \\ D_3 &= \Psi - \Phi_q^+(p) & D_4 &= D_3 - D_2 \end{aligned} \quad (4.32)$$

where Ψ are the symptoms of the new problem, and $\Phi_q^+(p)$ and $\Phi_q^-(p)$ are the positive and negative index features of the propagation case C_{qp} , respectively. It is obvious that D_1 represents the positive features that do not present in the symptoms of the investigated problem, D_2 represents the symptoms that are the negative features, D_3 represents the symptoms that are not the positive features, and D_4 represents the symptoms that are neither the positive nor the negative features.

Strictly, a perfectly matching case must satisfy $D_i = \emptyset$ for $i = 1, 2, 3, 4$. However, only D_1 and D_2 will affect the evaluation of the phase matching degree. Note that $D_3 = D_2 \cup D_4$.

The possible causes of inconsistencies include:

- The best matching case is not the solution to the investigated problem. In this situation, no adaptation can be performed to remove the inconsistency. The second best matching case needs to be examined for problem solving. If the problem is not covered by the case memory, other reasoning methods (MBR and RBR) will have to be applied.
- The best matching case is the solution to the investigated problem. However, the case is not accurately indexed. For example, in case indexing, the index features are usually selected from the symptoms that will *definitely* present. Using a symptom that *may* or *may not* present as the index feature will be likely to result in a non-empty D_1 , or in inconsistency. Inaccurate time tags for a symptom may also result in inconsistency. In this situation, MBR should be invoked to remove the

inconsistency. The explanation and solution from the case can be applied directly to solve the problem.

- The best matching case is the solution to the investigated problem. However, there is a secondary problem that has occurred simultaneously. The symptoms of the secondary problem are superimposed on that of the primary problem described by the best matching case. In this situation, MBR and/or CBR can be used to remove the inconsistencies. The explanation and solution from the best matching case can be applied directly. However, additional actions must be taken to handle the secondary problem.

As in Eqn. (4.1), a case is represented by $C = \langle D, S, V, H, T, CM \rangle$. H and CM are used as the starting point of the model-based inference engine. Inconsistency removal is performed by the following procedures:

- (1) For the inconsistent static symptoms included in D_1 , check the instant dynamic symptoms of the index variables. If an index variable has a tendency to present the symptom in D_1 , the inconsistency is removed. For example, even if the variable is only halfway to the high limit, a positive rate of change indicates that it is going to reach the high limit soon. The remaining inconsistencies are denoted by \bar{D}_1 .
- (2) Apply $D_2 \cup D_3 \cup \neg\bar{D}_1$ as the symptoms of a secondary problem to search for a case, namely secondary case, in the case memory. If a case is successfully retrieved and verified, the secondary problem that occurred simultaneously is found, and the inconsistencies are removed. The solution from both the primary case and the secondary case are combined to solve the problem being investigated. Otherwise, go to the next step. Here $\neg\bar{D}_1$ represents the negation of the symptoms in \bar{D}_1 .
- (3) In this situation, MBR has to be invoked. Reason forward with MBR from the original cause, H , of the primary case to the set of consequences: if within a certain time frame, the consequences do not include some of the symptoms in \bar{D}_1 ,

these inconsistencies are removed. The remaining inconsistencies in \bar{D}_1 are denoted by \hat{D}_1 ; if within a certain time frame, the consequences include some of the symptoms in D_2 and D_4 , these inconsistencies are removed. The remaining inconsistencies are denoted by \hat{D}_2 and \hat{D}_4 .

- (4) Modify the indexes of the case on the basis of the MBR results. If $\hat{D}_1 = \hat{D}_2 = \hat{D}_4 = \emptyset$, all the inconsistencies are removed. The solution from the case can be directly applied to solve the investigated problem. Otherwise, go to the next step.
- (5) Apply $\hat{D}_2 \cup \hat{D}_4 \cup \neg\hat{D}_1$ as the problem symptoms to search for a secondary case in the case memory as in Step (2). If a case is successfully retrieved and verified, the secondary problem that occurred simultaneously is found, and the inconsistencies are removed. The solutions from the primary case and the secondary case are combined to solve the problem. Otherwise, reason backward with MBR from the symptoms in $\neg\hat{D}_1$, \hat{D}_2 and \hat{D}_4 . If a cause is reached to explain these symptoms, the inconsistencies are removed. The solution from the primary case can be applied. However, additional corrective actions have to be taken to deal with the problem found with MBR.

If the above inconsistency removal is not successful, the second best matching case needs to be examined using the same procedures. It is obvious from the above procedure that inconsistency removal is an interactive reasoning process between the human operator, CBR and MBR. For the problems not covered by CBR and the MBR knowledge base, human operators have to take actions.

4.6.3 Problem solving by case mirroring

The problem description-based indexing path is employed when the cases retrieved by the abnormal symptom-based indexing path do not solve the problem. Unlike the abnormal symptom-based indexing path, this indexing path retrieves the cases that are functionally or structurally similar to the new problem. Such kinds of similar cases are called

mirroring cases, mirroring in the structure or function. The process of retrieving the cases is called “case mirroring”. For example, a case concerning the failure of the pressate pump in the interstage washing and bleaching can be mirrored in structure to third stage washing and bleaching for solving the same problem.

Eqn. (4.29) represents the mechanism of the problem description-based indexing path. To find a mirroring case, one or more fields of $Index(IC, PL, PS, AV, CS, CV)$ must be selected as the matching criterion. The following example illustrates the problem solving procedures using this indexing path.

Assume that a new problem presents the symptoms: P1 peroxide residual is high and brightness is low. The indexes of the new problem are thus

$$AV_N = (Peroxide_residual, High)$$

$$CS_N = (Brightness, Low)$$

$$PL_N = (Bleaching, Interstage).$$

The mirroring cases are those represented by

$$CASE_p = Index(*, PL_p, *, AV_p, CS_p, *)$$

with the matching criteria

$$AV_p = AV_N, \quad PL_p = (Bleaching, *), \quad CS_p = CS_N$$

where “*” represents any reasonable value. A case $CASE_p$ satisfying the above criteria is found to have the following fields:

$$PL_p = (Bleaching, Thirdstage)$$

$$CV_p = (Temperature, Increase, Thirdstage).$$

Therefore, increasing the temperature in the interstage bleaching and washing is the solution to the new problem. To obtain a solution as how to increase the temperature in interstage bleaching and washing (P1 bleach tower), one or more other cases matching the following criteria need to be retrieved:

$$CASE_Q = Index (*, PL_Q, *, *, *, CV_Q)$$

with

$$CV_Q = (Temperature, Increase, Interstage)$$

$$PL_Q = (Bleaching, Interstage).$$

It can be seen from this example that problem solving with case mirroring is more flexible. A number of cases can be combined to solve a single problem.

4.7 Conclusions

This chapter has investigated operation support problems in a BCTMP process. It was pointed out that a case-based reasoning approach is consistent with the natural problem solving of human operators. Considering that the industrial production processes are dynamic in nature, a dynamic case-based reasoning (DCBR) method was proposed. DCBR applies a new mechanism to reason with transient behaviors and fault propagation phenomena. DCBR has the advantages of fast, accurate and flexible problem solving.

The structure of DCBR makes it convenient for the integration with other computational and reasoning methods. MBR and RBR are actively applied in case adaptation and new case creation. Through the composite features, it is easy to incorporate other well-established algorithms in DCBR problem solving.

Chapter 5

INTEGRATION OF MULTI-VARIATE STATISTICAL CONTROL WITH CASE-BASED REASONING

The dynamic case-based reasoning (DCBR) method proposed in Chapter 4 has the advantage of combining various knowledge types and reasoning methods. Integrated with other reasoning methods, the DCBR is efficient in solving the operation support problems. This chapter investigates the application of principal component analysis (PCA) approaches in the DCBR system. The DCBR applies PCA for dimension reduction of representation space. The context-depended information extracted by PCA from data set is used as the composite features in case indexing of the DCBR system. Three case similarity evaluation mechanisms are proposed: that by using the squared prediction error (SPE) and principal scores, that by using the trajectory of the accumulated scores, and that by using the first principal loading direction. The DCBR system applies the combination of the three mechanisms for case retrieval and best matching case selection, which compensates the low fault distinguishability of individual PCA approaches. Since the information extracted by PCA contains correlation among process variables, the problem solving accuracy and efficiency of the DCBR are improved.

5.1 Introduction

Process operation support systems have been becoming an integral and important part of the production systems in modern process industry. Various methods have been proposed for the operation support related tasks, especially for the identification of abnormal conditions (Gertler, 1988; Patton et al., 1989; Frank, 1990; Hoskins et al., 1991; Iserman, 1984; Venkatasubramanian and Chan, 1989). The dynamic case-based reasoning method proposed in Chapter 4 has the advantage of integrating various knowledge types and reasoning methods together. It combines the static, dynamic and composite features for case indexing. The composite features are the problem solving results from other methods or systems. To improve the efficiency of the operation support systems, the other effective fault identification methods are to be integrated with the dynamic case-based reasoning. Multivariate statistical process control (MSPC) is one of these methods.

MSPC has been gaining increasing interest in process performance monitoring and fault detection (Martin and Morris, 1996). The bases of MSPC are the projection of process variables on to a reduced set of latent variables. There are two projection techniques: principal component analysis (PCA) (Wold et al., 1987) and projection to latent structure (PLS) (Höskuldsson, 1988) approaches. The advantage of MSPC is its dimension reduction. Instead of monitoring a large number of process variables, the system needs only to monitor a few projected latent variables. The most common method in MSPC for the performance monitoring is by using the plots of the squared prediction errors (SPE) and the scores in the latent variable spaces. It is assumed that a process malfunction will cause the SPE and scores moving to a distinctive region on the plot. This method is very effective to detect the deviation from the normal operation. It is able to classify a small number of process abnormal conditions (Kaspar and Ray, 1992; Kresta et al., 1991; Martin et al., 1996). However, when the number of the abnormal conditions is relatively large, the above method may suffer from low fault distinguishability. Besides, the classification of the abnormal conditions relies on visual inspection of the scores plot. Zhang and his coworkers (1995a) proposed a PCA based fault signature extraction

method to assist fault classification. A fault would cause the process measurement to be polarized in certain direction. The proposed method classified the fault by comparing the current data direction with a library of fault direction.

MSPC approaches apply the correlation among process variables and extract the context-depended information from data. Even though the fault distinguishability of each individual approach is relatively low, the utilization of the context-depended information in the dynamic case-based reasoning environment will improve the efficiency of problem solving. This chapter investigates the extraction of context-depended features from MSPC to be used in CBR systems.

5.2 Problem Formulation

5.2.1 Abnormal operating conditions

Consider a set of measured process variables representing the system states, $x(k)$. The process is governed by a set of the equality and inequality relations in the normal operations (Dunia et al., 1995):

$$\begin{aligned} G(x(k), x(k-1), \dots, x(k-n_d)) &= 0 \\ H(x(k), x(k-1), \dots, x(k-n_d)) &< 0 \end{aligned} \tag{5.1}$$

where $x \in \mathfrak{R}^n$ is the n dimensional measured vector, n_d is the number of the past terms used in the relations. The selection of n_d depends on the process characteristics. In the normal situations, the process follows the trajectory specified by the equality relation G and limited by the inequality H . The process parameters are maintained at the desired targets or ranges by control systems. The possible variations of operating variables, which is the response of the control system to normal perturbation, are limited to certain range, which is referred steady state behaviour. This normal steady state behavior is corresponding to one or more regions in the measurement space represented by the pattern vector, X :

$$X = (x(k)^T \quad x(k-1)^T \quad \dots \quad x(k-n_d)^T)^T \quad (5.2)$$

When an abnormal condition occurs, the equality and inequality relations will be violated. The system moves to a new steady state governed by new relations, resulting in the operating data moving to a new region in the measurement space. The classification of the abnormal conditions can be accomplished by identifying the regions in the measurement space. Since the distribution of the measurement data in industrial processes are usually poor or unknown, clustering-based pattern recognition methods are appropriate to determine the similarity of the operating data in the measurement space (Whiteley and Davis, 1994). The fault transition or migration can be simply viewed as the transition of the operating conditions from one cluster to another.

The difficulty of applying the above method is the high dimension of the measurement space. The dimension is even higher if the past data is used to incorporate process dynamics, as in Eqn. (5.1). Using the initial measurement variables, the calculation requirement is high and the efficiency of clustering is low. The dimension reduction by multivariate statistical process control can be applied to solve the problem.

In an industrial process, most of the measurements are correlated. The initial space of measurement, X , can be mapped to a lower dimensional representation space, Θ :

$$\begin{aligned} \xi: \quad X &\rightarrow \Theta \\ \eta: \quad \Theta &\rightarrow X \end{aligned} \quad (5.3)$$

or

$$\begin{aligned} \theta &= \xi(x) \\ x &= \eta(\theta) + e \end{aligned} \quad (5.4)$$

where e is the residual that represents the difference between the initial measurement and the de-mapped measurement. Instead of the initial measurement space, X , the fault classification or clustering can be based on the much lower dimensional representation

space, Θ . That is, the cluster range for the normal or abnormal operations can be confined by the following equations:

$$\begin{aligned} F^i(\theta(k)) &= 0 \\ G^i(\theta(k)) &< 0 \end{aligned} \quad (5.5)$$

where the superscript “ i ” represents the presence of the i th fault, k represents the time.

5.2.2 Introduction to PCA method

The primary multivariate statistical process control methods (MSPC) are principal component analysis (PCA) (Wold et al., 1987) and projection to latent structures (Geladi and Kowalski, 1986). Both methods have been extensively applied in process performance monitoring (MacGregor, 1994; MacGregor et al., 1994; Martin and Morris, 1996). This chapter will be focused on PCA. Good survey and tutorial of PLS approach can be found in the papers by MacGregor et al (1994), Geladi and Kowalski (1986), Hoskuldsson (1988) and Martin and Morris (1996).

Principal component analysis is an approach to map high dimensional data to lower dimensions without or with minimum loss of information. It has been successfully applied in data summarization, outlier detection, fingerprinting for fault identification, and early prediction of potential malfunctions.

Assume that there are n measured variables and N samples, the system can be described by the observation matrix $X \in R^{N \times n}$:

$$X = [\mathbf{x}_1 \quad \mathbf{x}_2 \quad \cdots \quad \mathbf{x}_n] \quad (5.6)$$

where $\mathbf{x}_i \in R^N$ ($i = 1, \dots, n$) is either a process variable or a quality variable. The PCA method decomposes the observation matrix, X , as follows:

$$X = TP^T + E = \sum_{i=1}^l t_i p_i^T + E \quad (5.7)$$

where t_i and p_i is the i th score vector and loading vector, respectively; $t_i p_i^T$ is the i th principal component; E is the residual matrix; $l < n$ represents the number of the principal components. The loading vector t_i are defined as:

$$\begin{aligned}
 t_1 &= X p_1^T & E_1 &= X - t_1 p_1^T \\
 t_2 &= E_1 p_1^T & E_2 &= E_1 - t_2 p_2^T \\
 t_i &= E_{i-1} p_{i-1}^T & E_i &= E_{i-1} - t_i p_i^T
 \end{aligned} \tag{5.8}$$

$$|p_i| = 1$$

where the score vectors p_i ($i = 1, 2, \dots, l$) are the eigenvectors of the covariance matrix of X . The first loading, p_1 , describes the direction of greatest variability. The first score, t_1 , a linear combination of the columns of X , describes the greatest amount of variability in X . It represents the projection of each data set onto p_1 . The second score, t_2 , represents the second greatest variability. The elements in a loading vector represent the contribution of each variable to the corresponding component.

Because of the process variable correlations, the first three principal components are usually sufficient to explain the major variances in the data. The rest represents only the noise or unimportant variances. Discarding these minor components will not cause major information loss. As a matter of fact, it may even reduce the effect of process noise in modeling. Cross-validation method can be employed to determine the number of principal components required to contain most of the information in the data (Wold, 1978).

5.2.3 Nonlinear principal analysis methods

Most of the engineering problems in process industry are nonlinear in nature. Previous research found that linear PCA is not always adequate to solve nonlinear problems (Dong and McAvoy, 1996; Palus and Dvrak, 1992). The minor components may contain important information. For example, in the application of PCA for chemical process

monitoring, the minor components may provide stronger evidences of the occurrence of a process upset than the principal components do (Martin, et al, 1996). Using only the principal components in nonlinear processes may yield erroneous results. On the other hand, if all the minor components are to be included, the advantage of PCA techniques will no longer exist. Nonlinear principal component analysis techniques (NLPCA) have been proposed to solve the problem.

The concept of NLPCA is similar to linear PCA. Instead of using the linear correlations among the process variables, NLPCA uses nonlinear correlations.

Kramer (1992) proposed a NLPCA method by applying the autoassociative neural network and the mapping and de-mapping concepts. The neural network has five layers: input, mapping, bottleneck, de-mapping and output. The mapping and de-mapping layers have nonlinear nodes, whereas other three layers have linear nodes (Fig. 5.1). The input, mapping and bottleneck layers compresses the process measurement, X , into a lower dimensional space, Θ , i.e. nonlinear principal components. The de-mapping and output layer decompresses the principal component space to initial measurement space, \hat{X} . The number of nodes in the mapping and de-mapping layers depends on the complexity of the engineering problem. Provided that adequate number of the bottleneck layer is selected, no major information loss will occur. The constructed measurement, \hat{X} , is roughly the same as the initial measurement, X .

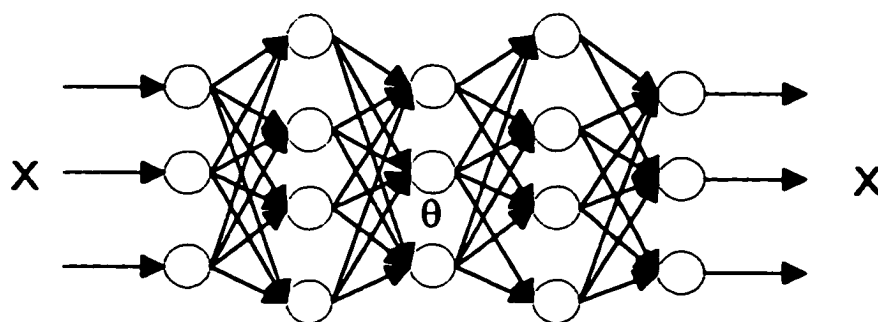


Fig. 5.1. Autoassociative neural network for NLPCA.

The neural network is trained to minimize the following objective function:

$$E = \sum_{j=1}^N \sum_{i=1}^n (x_{i,j} - \hat{x}_{i,j})^2 \quad (5.9)$$

It is difficult to determine the number of nodes in the mapping, bottleneck and de-mapping layers. An extension to this method is the principal curve method proposed by Hastie and Stuetzle (1989). Principal curves are the generalizations of principal components in which all the data points are projected orthogonally down onto the principal curves. For every data point, the projected point on the principal curves is the length along the curve, defined as the principal score.

Mathematically, the NLPCA can be represented by the following equation:

$$X = \sum_{i=1}^l f_i(\theta_i) + E \quad (5.10)$$

or in an iterative way:

$$X = f_1(\theta_1) + E_1 \quad (5.11)$$

$$E_{i-1} = f_i(\theta_i) + E_i \quad (5.12)$$

where θ_i is the i th principal score vector, f_i is the i th principal curve, or principal loading function. E_1 is the residual of the first principal component, which is used for the calculation of second principal component. The procedure continues until most of the variance is captured by the principal components. The number of the principal components to be selected depends on the percent of variance explained by the principal components.

The type of the nonlinear loading functions to be selected depends on the characteristics of the engineering problems. Different engineering problems may require different principal loading functions. Dong and McAvoy (1996) proposed a neural network method for learning the nonlinear principal loading functions. The neural network for NLPCA consists of two three layer neural networks. The first one maps n -

dimensional original data to the l -dimensional principal score, the second one maps l -dimensional scores to the m -dimensional corrected data set.

5.3 PCA for Case Index Feature Interpretation

The PCA method is to be used by the dynamic case-based reasoning method (DCBR) to extract context-depended information from the initial measured data. The context-depended information is used as the composite features defined in Chapter 4 for case indexing.

PCA approach is applicable to the process performance monitoring because of the fact that the measured process variables are highly correlated. The PCA converts an n -dimensional measurement space X into an l -dimensional principal component space Θ ($l < n$). The DCBR then maps the principal component space, Θ , to the process abnormal condition space, F , as shown in Fig. 5.2:

$$\begin{aligned} \xi: X &\rightarrow S \\ \pi: \Theta &\rightarrow F \end{aligned} \tag{5.13}$$

where $F = (F_1, F_2, \dots, F_M)$ is the M -dimensional process abnormal condition space.

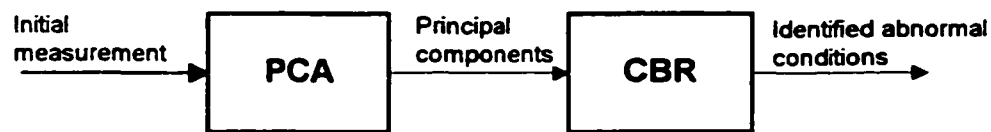


Fig. 5.2. Integration of PCA with CBR.

The purpose of the feature interpretation is to convert the initial n -dimensional measurements to the features that can be directly used for case indexing and case retrieval. For the efficiency of problem solving, it is required that these features are sensitive to

process faults and present distinctive symptoms under various abnormal conditions. The following features extracted by PCA approaches are considered of these properties.

5.3.1 Squared prediction errors (SPE)

The principal component models developed from the nominal data set, as given by Eqns. (5.7) and (5.10), represent the correlations among the process variables in normal process conditions. Given the linear principal model (5.7), the predicted values can be calculated from each new observation:

$$\begin{aligned} t_i(k) &= x(k)p_i^T \\ \hat{x}(k) &= t(k)p^T = \sum_{i=1}^l t_i(k)p_i^T \end{aligned} \quad (5.14)$$

where $x(k)$ and $\hat{x}(k)$ is the process measurement and the prediction at time k , respectively; $t_i(k)$ is the i th principal score calculated using the nominal model; and p_i is the i th nominal principal loading vector. In the case of nonlinear PCA, the predicted value can be calculated by using the nonlinear model (5.10):

$$\hat{x}(k) = \sum_{i=1}^l f_i(\theta_i) \quad (5.15)$$

The squared prediction error (SPE) is defined as the squared difference between the predicted value and the measurement:

$$SPE(k) = \sum_{j=1}^n (x_j(k) - \hat{x}_j(k))^2 \quad (5.16)$$

An abnormal process condition might change the correlations among the process variables. The principal component model developed from the nominal data will no longer represent the process characteristics. As a result, the SPE will increase. A significant increase in SPE usually indicates an abnormal condition if the data for developing the nominal model covers the entire normal operating conditions.

It should be noted that not all abnormal conditions result in an increase in SPE (Martin and Morris, 1996). Some process faults do not change the relationship between the process variables. The process moves to the regions that are normally not in the operation ranges. However, the SPE will remain at the acceptable level.

SPE is very useful for fault detection. For the classification of abnormal conditions, however, SPE has to be used with other features, such as principal scores.

5.3.2 Principal scores

One of the common methods for performance monitoring is by using the plots of the squared prediction errors (SPE) and the scores in the latent variable spaces (Kaspar and Ray, 1992; Kresta, et al., 1991; Martin et al., 1996), as shown in Fig. 5.3. An abnormal process condition may change the direction of the score movement in the latent variable space. When the system reaches its new steady state, the operating data will be located at a region that is not in the normal operation ranges. Assume that every fault causes a distinctive score movement, the abnormal conditions can be classified by the region at which the data is located.

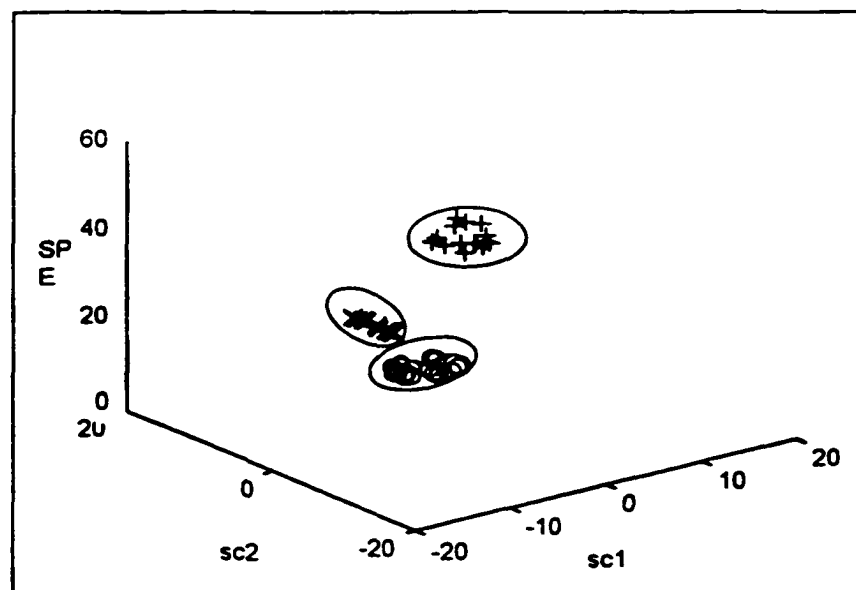


Fig. 5.3. SPE versus score plot for performance monitoring.

Nominal principal component model is required to calculate the principal score vector of a new observation. In linear PCA, the principal score vector is calculated by Eqn. (5.14). In nonlinear PCA, the principal score vector is calculated by feeding the new observation to the input layer of the neural network as shown in Fig. 5.1, or by using the nonlinear loading functions, as given in Eqns. (5.10) – (5.12).

Fig. 5.3 shows the plot of SPE, first score t_1 and second score t_2 . The points represented by 'o' are the nominal data, that represented by 'x' are the abnormal data that does not violate the correlation structure, and that represented by '+' are the abnormal data that violates the nominal correlation structure.

The SPE and scores plots have been proven effective when the number of faults is small. The best principal components used in the plot might not be the major principal scores but the minor ones. However, the score movement may not be distinctive for every pair of abnormal conditions. For this reason, this approach is not suitable to classify the large number of abnormal conditions.

5.3.3 Accumulated principal scores

The accumulated scores, A_i , are defined as follows (Zhang, et al., 1995b):

$$A_i = \sum_{k=1}^k (\alpha_i(k) - \bar{\alpha}_i) \quad (5.17)$$

where α_i is the i th principal score, $\bar{\alpha}_i$ is the mean of the i th nominal score.

Previous research indicated that the score movements for various abnormal situations are not necessarily significantly different. The abnormal conditions cannot be distinguished from the scores plot. However, because of its cumulative nature, the accumulated scores plot can be more effective in distinguishing abnormal conditions.

5.3.4 Principal loading vectors

The features used in the previous sections are calculated based on the nominal principal component model. The models developed for individual abnormal conditions can be also used for fault classification.

Assume that the principal component model developed from the data under the k th abnormal condition is given by:

$$X = T_j P_j^T + E_j = \sum_{i=1}^l t_{ji} p_{ji}^T + E_j \quad j = 1, 2, \dots, M \quad (5.18)$$

where M is the number of abnormal conditions. It was indicated by Zhang and his coworkers (1995a) that the first principal loading, p_{j1} , can be used as the fault direction extracted from the data set. Each fault has distinctive first principal loading direction. The fault can be identified by comparing the current data direction with the fault direction library.

The above features have relatively low fault distinguishability. To improve the problem solving effectiveness when there are large number of potential abnormal conditions, the combination of these features as case indexes is required.

5.4 Case Similarity Evaluation Using PCA

The integration of PCA with case-based reasoning is intended to evaluate the case similarity by utilizing the context-dependent features extracted by PCA approaches. Since the context-dependent features contain much more explicit information than the initial measured data, the problem solving accuracy and efficiency of the dynamic case-based reasoning will be improved. This section discusses how the features identified in section 5.3 are used for case similarity evaluation.

5.4.1 Case similarity evaluation using SPE and scores

As shown in Fig. 5.3, the case similarity evaluation is a pattern recognition problem using data clustering approach. There are two basic approaches of pattern recognition: the synthetic approach (Fu, 1992; Gonzalez and Thomason, 1978) and the geometric approach (Duda and Hart, 1973). Three approaches are available for geometric pattern recognition: linear/nonlinear discriminate functions, Bayesian-based approach and clustering approach. In the abnormal condition identification, the prototypes for learning are usually not complete. Clustering approach is most appropriate since it can identify the situations that are not covered by the learning prototypes and treat these patterns as “unknown”. A certainty factor can be assigned to a pattern that is located outside all the learned prototypes.

The objective of the data clustering is to divide the representation space into a number of regions representing various operation conditions. Using the PCA, the representation space has the dimension of the principal components. The representation space, as shown in Fig. 5.4, consists of four types of regions: normal operating conditions, known abnormal operating conditions, unknown abnormal operating conditions, and transition states.

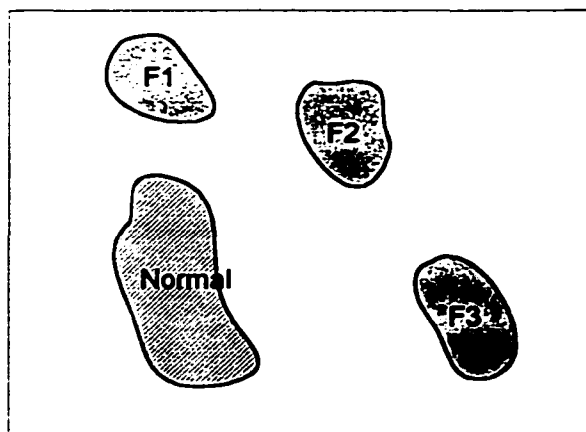


Fig. 5.4. Representation space for various operating conditions.

Normal operating conditions are the steady state process behavior maintained by control systems. The patterns falling to the region of representation space (marked “Normal” in Fig. 5.4) represent the normal process behaviour. There is often large amount of “normal” process data available from the data-to-day operations. However, it is not necessary that the available data covers the entire normal process behaviour.

The abnormal operating conditions are the results of process or equipment faults. For the known abnormalities, process data are available from either the historical database or process simulations. Each region in the representation space marked “F1”, “F2”, and “F3” represents the process behaviour under a specific abnormal condition.

The transition states are the result of fault propagation. When a process evolves from “normal” to “abnormal”, the fault symptoms are propagated from the first affected variable to eventually all the other affected variables. Before the process reaches the new steady state, the patterns fall to the transition state regions.

In process industry, there are always some abnormal conditions that are not known or the data is not available. Since no patterns can be created for these situations, the operation support system has to declare the situation “unknown”.

In the operation support problem investigated here, the data available for clustering has been labeled with “normal” or a specific abnormal condition. The clustering algorithm is intended to identify a bounded decision region for each cluster. The size of the decision region should be large enough to include most of the training patterns but small enough not to include the patterns belong to other clusters.

There have been a number of methods to determine the boundary of a data cluster, such as neural network methods. A common assumption made in process industry is that the data in the cluster is normally distributed. The multivariate normal density function for the j th pattern cluster can be represented by:

$$p(\theta|\omega_j) = \frac{1}{(2\pi)^{k/2} |C_j|^{k/2}} \exp\left[-\frac{1}{2}(\theta - m_j)^T C_j^{-1}(\theta - m_j)\right] \quad (5.19)$$

where $m_j = E_j[\theta]$ is the mean vector of the prototypes in cluster j , $C_j = E_j[(\theta - m_j)(\theta - m_j)^T]$ is the covariance matrix of the prototypes. The mean vector is intended to capture the center of the cluster, and the covariance to capture the shape. The principal axes of the hyperellipsoids (contours of equal probability density) are given by the eigenvectors of C_j with the eigenvalues determine the relative length of these axes. When $l = 1$, the region $|\theta - m_j| < 2\sigma$ covers 95 percent of the prototypes, where σ is the standard deviations.

For the purpose of the simplicity, we apply the hyperellipsoid to represent the boundary of region of a data cluster. The hyperellipsoid is represented by the following distance known as Mahalanobis distance:

$$r_j^2 = (\theta - m_j)^T C_j^{-1} (\theta - m_j) \quad (5.20)$$

where m_j and C_j is the mean and covariance calculated with the samples:

$$m_j = \frac{1}{N_j} \sum_{\theta_i \in S_j} \theta_i \quad (5.21)$$

$$C_j = \frac{1}{N_j} \sum_{\theta_i \in S_j} [(\theta_i - m_j)(\theta_i - m_j)^T] \quad (5.22)$$

where N_j is the number of prototypes in cluster j ; S_j represents the set of data in cluster j . The contour of the cluster can be represented by:

$$r_j^2 = (\theta - m_j)^T C_j^{-1} (\theta - m_j) < \beta_j \quad (5.23)$$

where β_j is a number in the neighborhood of 1 to be selected based on the distribution of the data in the cluster.

The case similarity is evaluated with the following fuzzy membership:

$$\lambda_j = \begin{cases} 1 & \text{if } (\theta - m_j)^T C_j^{-1} (\theta - m_j) \leq \beta_j \\ \frac{1}{(\theta - m_j)^T C_j^{-1} (\theta - m_j)} & \text{if } (\theta - m_j)^T C_j^{-1} (\theta - m_j) > \beta_j \end{cases} \quad (5.24)$$

where λ_j is the similarity of the Case j to a new problem. Case 0 represents the normal operating conditions.

5.4.2 Case similarity evaluation using accumulated scores

Using the accumulated scores, the similarity of a case in the case memory to the new problem is evaluated by comparing the accumulated score trajectories. Each case that uses the accumulated scores as index has a built-in accumulated score trajectory. The score that moves the most under the specific situation is selected as the primary score. Assume that the p th principal score is selected as the primary score, the trajectory is represented by:

$$A_{ji} = S_{ji}(A_{jp}) \quad i = 1, 2, \dots, l; i \neq p \quad (5.25)$$

where the subscript j represents Case j , i represents the i th score.

Assume that the trajectory of the accumulated scores for the new problem is:

$$A(k) = (A_1(k), A_2(k), \dots, A_l(k))$$

where k represents time. The calculated trajectory using the index in the j th case is:

$$A_j(k) = (A_{j1}(k), A_{j2}(k), \dots, A_p(k), \dots, A_{jl}(k))$$

with

$$A_{ji}(k) = S_{ji}(A_p(k))$$

The similarity of Case j to the new problem is calculated as:

$$\lambda_j(k) = \frac{\sum_{r=k-W}^k A(r)A_j(r)^T}{\left(\sum_{r=k-W}^k A(r)A(r)^T\right)^{\frac{1}{2}} \left(\sum_{r=k-W}^k A_j(r)A_j(r)^T\right)^{\frac{1}{2}}} \quad (5.26)$$

where W is the width of the data window for case similarity calculation. It is clear that if the current accumulated score trajectory is exactly aligned with the calculated trajectory of Case j , the case similarity λ_j is 1.

5.4.3 Case similarity evaluation using principal loading

To calculate the case similarity using the first principal loading vector, each case should have a fault direction as its indexes. The fault direction is the first principal loading vector obtained using the data under the specific abnormal condition. The first principal loading of the new problem is calculated from the data within a moving window of size W , i.e., within $(k-W, k)$.

Denoting the first principal loading of the new problem by p_1 , and the fault direction of Case j (j th abnormal condition) by M_j , p_1 and M_j will be closely aligned if the j th abnormal condition has occurred. $p_1^T M_j$ will be close to 1. Similar to Eqn. (5.24), the case similarity is evaluated with the following fuzzy membership:

$$\lambda_j = \begin{cases} 1 & \text{if } p_1^T M_j \geq \beta_j \\ \frac{\beta_j}{p_1^T M_j} & \text{if } 0 < p_1^T M_j < \beta_j \\ 0 & \text{if } p_1^T M_j \leq 0 \end{cases} \quad (5.27)$$

where β_j is a number within $[0.7, 1]$. Usually the larger the variation of the data within the cluster, the smaller the β_j should be selected.

5.5 Case-Based Reasoning with PCA

For the process faults that change the relationship between the process variables, SPE is the best parameter to detect the abnormal condition. If the SPE increases beyond an acceptable range, an abnormal condition is detected. However, as pointed out in the previous sections, an acceptable SPE may not necessarily imply that the process is in normal conditions. A process abnormality may only cause one or more process variable to have large shift, but does not change the relationship between the process variables. In this situation, SPE will remain at an acceptable level. It is better to use the principal score movement, or the case similarity given by Eqn. (5.24), to detect such kind of abnormality.

Using Eqn. (5.24), the certainty that the system is in normal operating conditions is calculated as:

$$\lambda_0 = \begin{cases} 1 & \text{if } (\theta - m_0)^T C_0^{-1} (\theta - m_0) \leq \beta_0 \\ \frac{\beta_0}{(\theta - m_0)^T C_0^{-1} (\theta - m_0)} & \text{if } (\theta - m_0)^T C_0^{-1} (\theta - m_0) > \beta_0 \end{cases}$$

where the subscript 0 represents the cluster for normal operating conditions. An abnormality is detected if:

$$SPE > \alpha_1 \quad \text{or} \quad \lambda_0 \leq \alpha_2 \quad (5.28)$$

where α_1 and α_2 are the detection thresholds. α_2 is usually selected between 0.6 to 0.85.

When an abnormal condition is detected, further classification is required. The dynamic case-based reasoning system uses the context-depended features extracted from the PCA approaches to retrieve the plausible cases, and uses the case similarity evaluators given in Section 5.4 to determine the best matching case. The explanations and solutions contained in the best matching case will be used as the solution to the problem. The details of the problem solving process are given in Chapter 4.

As shown in Section 5.4, A case can use more than one similarity evaluation mechanism to calculate the case similarity. To further improve the efficiency of case

retrieval and evaluation, the entire case memory can be organized in a hierarchical structure by using various indexes, as shown in Fig. 5.5.

Fig. 5.5 shows that the entire case memory is first divided into a number of level-two case memories by the accumulated scores index. Each level-two case memory is divided into a number of level-three case memories by using the principal loading index. The SPE and scores are then used to index the individual cases in the level-three case memories. Accordingly, the problem solving process is to locate a level-two case memory first, and then narrow down to a level-three case memory, from there an individual case will be identified. This problem solving process is consistent with the process decomposition proposed in Chapter 2. The static features and dynamic features, together with the features extracted by PCA, are used for case memory decomposition.

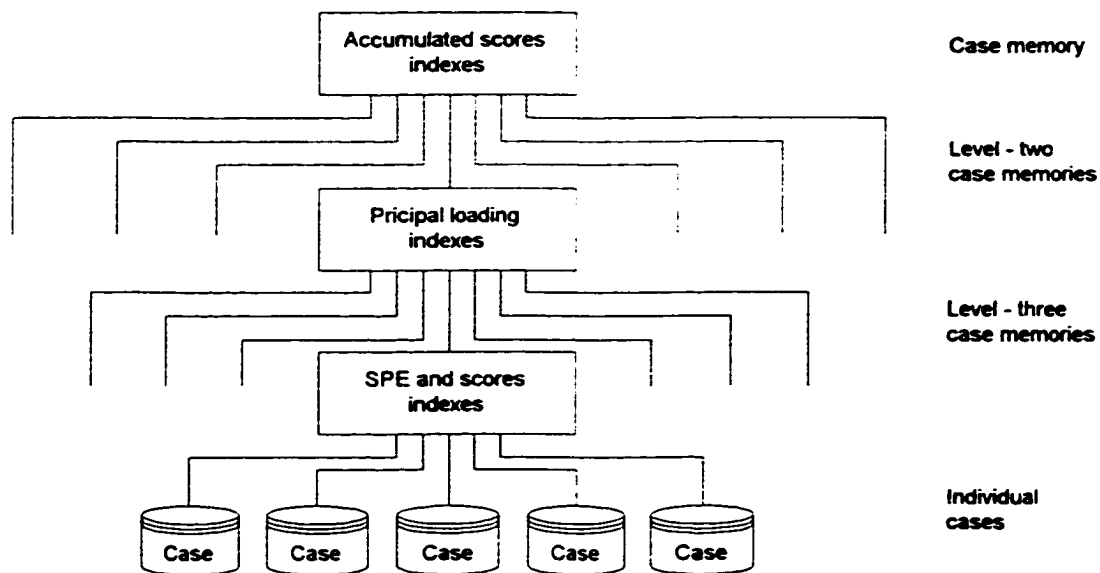


Fig. 5.5. Case memory in hierarchical structure.

5.6 Conclusions

MSPC has been gaining increasing interest in process performance monitoring and fault detection. The advantage of MSPC is its dimension reduction. Instead of monitoring a large number of process variables, the system needs only to monitor a few projected latent variables. However, the individual MSPC approaches have relatively low fault distinguishability. The classification of the abnormal conditions mainly relies on visual inspection of various plots.

This chapter proposed a method of applying the principal component analysis (PCA) approaches in the dynamic case-based reasoning (DCBR) system. The PCA approaches were applied for dimension reduction and the extraction of the correlations between process variables. The DCBR applies the information extracted by PCA as the composite features for case indexing and evaluation, which includes the squared prediction error, principal scores, accumulated scores, and principal loading vectors. Accordingly, three case similarity evaluators were proposed. Through the application of these context-dependent features, together with the features obtained by other methods, DCBR is able to improve the fault distinguishability and problem solving efficiency.

Chapter 6

KALMAN FILTER-BASED FAULT DETECTION AND FAULT-TOLERANT CONTROL[†]

This chapter presents a model-based fault detection and fault-tolerant control technique and its integration with the dynamic case-based reasoning (DCBR) system. A bank of Kalman filters is constructed corresponding to all the possible sensor failure modes. The possibility that a failure mode hypothesis is true is calculated using measurement innovation processes. The sensor failures are detected and located based on the calculated possibilities of the hypotheses. The controller and state estimator are automatically reorganized subsequent to the occurrence of failures to ensure the stability and acceptable performance of the closed-loop system. The dynamic case-based reasoning system applies the calculated possibilities of the hypotheses as the composite features for case evaluation. The reorganization of the controller subsequent to a failure is contained in the cases as the solutions to the corresponding problems. The issues of system hardware redundancy and computational requirement as well as implementation complexity are taken into account in the system design. Simulation results have shown satisfactory performance of the headbox control system after applying the presented technique.

[†] This chapter has been published: Xia and Rao, 1992, "Fault-tolerant control of paper machine headbox", *Journal of Process Control*, 2, 171-178.

6.1 Introduction

Paper machine headbox is one of the critical units of a pulp and paper process. A good control of headbox is extremely important to improve pulp and paper quality and to increase economic profit. Many sophisticated control algorithms have been successfully developed for paper machine headboxes, such as adaptive control (Borisson, 1979; D'Huster et al., 1983), nonlinear control (Gunn and Sinha, 1983), multivariable optimal control with a reference model (Lebeau et al., 1980), and optimal decoupling control (Xia et al., 1993).

Those algorithms, however, are developed under the assumption that all the system components are of high reliability and will not fail. Once a failure occurs, the system performance will degrade and the system may even diverge. However, some components, especially the measurement instruments (sensors) applied in the headboxes of paper machines are poor in reliability. The control systems for headbox are required to deal with the problem of component failures.

Fault-tolerant control system can adapt to significant changes of environments. When all components are in normal condition, the system has desired performance. If some key components (e.g., sensors and actuators) fail, the system continues to maintain acceptable performance (Siljak, 1980). Fault-tolerant control is a challenge to the classical control techniques. The advances of computer science and parallel computing technology have made it possible to realize the fault-tolerant control techniques. There have been extensive simulation and application studies of fault tolerant control in the area of flight control (Deckert et al., 1977), but few have been reported in pulp and paper industry.

This chapter investigates the fault detection and fault-tolerant control of headboxes. A control technique that is tolerant of sensor failures is developed and applied to a practical pressurized headbox. The technique has moderate computational requirement. It is easy to be implemented in most industrial processes.

6.2 Process Specification

The pressurized headbox is a volume chamber. Its main purpose is to distribute the water-fiber suspension onto the fourdrinier wire as evenly and steadily as possible. The lower part of the headbox is occupied by pulp stock and above the pulp stock is air. Inside the headbox is a rectifier that provides even distribution of pulp fiber onto the fourdrinier wire.

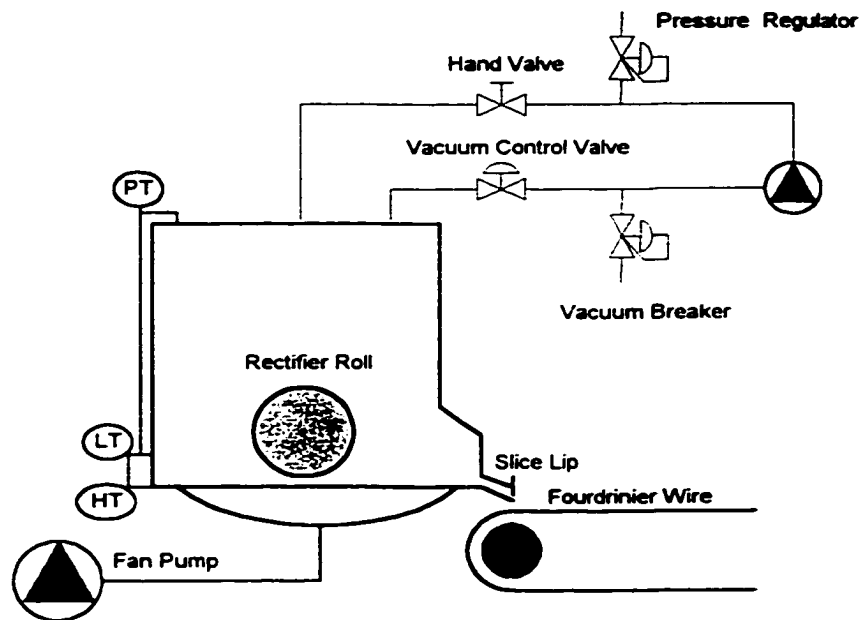


Fig. 6.1. Schematic diagram of the pressurized headbox.

The pulp stock is pumped into the headbox by a fan pump and flows onto the wire through a slice lip as a result of the total head. The flow rate of the pulp stock into the headbox is controlled by the speed of a fan pump. The air is circulating in the air chamber of the headbox with a vacuum pump. The air pressure in the headbox is controlled by a vacuum valve located in the suction side of the vacuum pump. A vacuum breaker, a pressure regulator and a hand valve are also installed in order to protect the vacuum pump, limit the maximal value of pressure in the headbox and adjust the position of the

vacuum control valve under normal operation condition. Fig. 6.1 is the schematic diagram of the pressurized headbox.

The two most important control objectives are:

- (1) maintaining the pulp stock level in the headbox: the optimum operation condition for the rectifier roll is the pulp stock level just above the top of the rectifier. Therefore, the level must be held constant;
- (2) maintaining the rush/drag ratio: the term rush/drag ratio refers to the ratio of the velocity of the jet of slurry onto the wire to the speed of the wire. In order to improve the physical properties of pulp and paper product, the ratio should be maintained at some specific value. For the headbox investigated here, the desired ratio is 1.02.

The velocity of the jet of slurry is proportional to the square root of the total head. To maintain a constant rush/drag ratio, the total head must be adjusted with the variation of wire speed (machine rate). Hence, the direct controlled variables in the headbox are total head and pulp stock level.

From mechanism analyses, it is obvious that the transfer functions relating pulp stock level to fan pump speed and that relating pressure to vacuum valve position are both first-order if without the interactions between the pressure and the level. It is also obvious that the pulp stock level variation has no steady state effect on the air pressure, and neither does the air pressure variation on the total head. These facts can be explained by the mass balance and gas axiom: the increase of the pulp stock level causes air pressure to increase and subsequently causes the outflow rate of air to increase until the air pressure regains its original value. The steady state value of the air pressure can only be affected by the vacuum valve position. The total head is the summation of the air pressure and pulp stock level. When the air pressure increases, the total head will increase simultaneously. However, the increase of the total head will cause the outflow rate of pulp stock to increase. As a result, the level decreases, and the total head returns to its original value. The steady state value of the total head can only be affected by the fan pump speed.

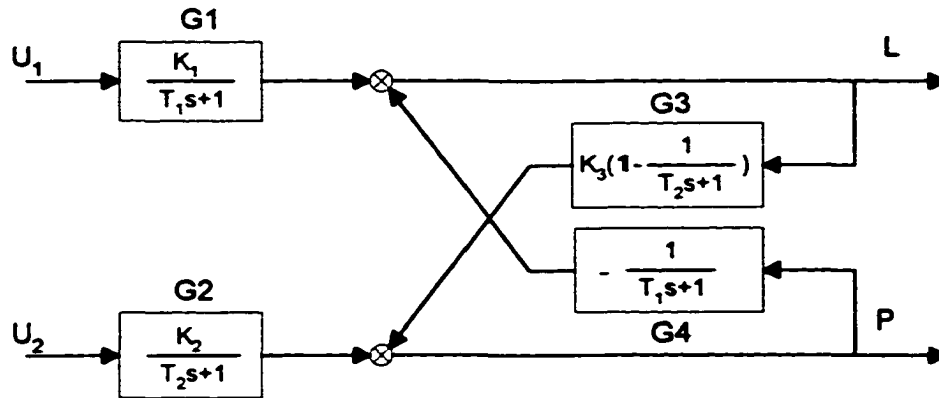


Fig. 6.2. Block diagram of pressurized headbox.

According to the mechanism analyses above, the block diagram of the pressurized headbox can be depicted as Fig. 6.2, with L , P , U_1 and U_2 representing the pulp stock level, air pressure, fan pump speed and vacuum valve opening, respectively (Xia et al., 1993). The transfer functions are:

$$T_1sL(s) + L(s) = K_1U_1(s) - P(s) \quad (6.1)$$

$$T_2sP(s) + P(s) = K_2U_2(s) + K_3T_2sL(s) \quad (6.2)$$

The pressurized headbox investigated in this chapter is of size 235 inches wide, 48 inches deep and 57 inches high. A number of bump tests have been done on the headbox. Using the data obtained from the bump tests, it is calculated that $K_1 = 1.75$, $K_2 = 1.54$, $K_3 = 13.12$, $T_1 = 0.2308$ and $T_2 = 0.1710$. The continuous state equations are thus:

$$\dot{L}(t) = 4.33L(t) - 4.33P(t) + 7.58U_1(t) \quad (6.3)$$

$$\dot{P}(t) = -56.8L(t) - 62.7P(t) + 99.5U_1(t) + 9.01U_2(t) \quad (6.4)$$

Discrete time steady state equations are required for the digital control of the headbox. By integrating the differential Eqns. (6.3) and (6.4) over the sampling interval of $T = 0.1$ min, the discrete equations of the headbox are obtained:

$$L(k+1) = 0.905L(k) - 0.0629P(k) + 0.166U_1(k) - 0.049U_2(k) \quad (6.5)$$

$$P(k+1) = -0.825P(k) + 0.0586L(k) + 1.44U_1(k) + 0.180U_2(k) \quad (6.6)$$

Noting that the total head H is the summation of L and P , we have:

$$H(k+1) = -0.0042H(k) + 0.148L(k) + 1.61U_1(k) + 0.138U_2(k) \quad (6.7)$$

$$L(k+1) = 0.9682L(k) - 0.0629H(k) + 0.166U_1(k) - 0.049U_2(k) \quad (6.8)$$

or rewrite it into state space representation by defining the state vector $x(k) = (H(k) \ L(k))^T$, the output vector $y_r(k) = x(k)$ and the control vector $u(k) = (U_1(k) \ U_2(k))^T$:

$$x(k+1) = \begin{pmatrix} -0.0042 & 0.148 \\ -0.0629 & -0.968 \end{pmatrix} x(k) + \begin{pmatrix} 1.61 & 0.138 \\ 0.166 & -0.049 \end{pmatrix} u(k) + w(k) \quad (6.9)$$

$$y_r(k) = x(k) \quad (6.10)$$

where $w(k)$ is the process noise. The pulp stock level, air pressure and total head in the headbox are measured on-line. Denoting $y(k) = (H(k) \ L(k) \ P(k))^T$, the measurement equation is:

$$y(k) = \begin{pmatrix} 1.0 & 0.0 \\ 0.0 & 1.0 \\ 1.0 & -1.0 \end{pmatrix} x(k) + v(k) \quad (6.11)$$

where $v(k)$ is the measurement noise.

Because of the strong interactions between the control of the two output variables, H and L , a decoupling control law is required:

$$u(k) = F \cdot x(k) + G \cdot v_m(k) \quad (6.12)$$

where $v_m(k)$ represents the setpoints.

The problem in the control of the headbox is that the measurement instruments are not reliable. The sensors for the pulp stock level, air pressure and total head may fail during operation. When any of the sensors fails, the measured data will no longer represent the true operation conditions of the headbox. The control systems without considering the possible failures will lead to a poor performance. To improve the reliability and the dynamic performance of the headbox control system, failure detection and modification are necessary. In the subsequent sections, a new method is proposed.

6.3 Fault Detection and Reorganization

To maintain acceptable performance in the case of fault, the fault-tolerant system will perform two tasks:

- early and accurate process fault detection and identification;
- effective control system reorganization.

A great number of failure detection techniques have been developed in the past two decades, such as failure-sensitive filter, generalized likelihood ratio test and multiple Kalman filter with a standard multiple hypothesis testing (Montgomery and Caglayan, 1974; Willsky, 1976). The design of failure detection systems for industrial processes involves the consideration of the following issues:

- (1) Industrial processes may involve slow change and abrupt change failures. The failure detection schemes must be applicable to both kinds of the failures;
- (2) Industrial processes usually do not possess many back-up components. The requirement for costly hardware redundancy should not be too high;

- (3) Industrial processes usually apply computers with relative small storage and low computing speed. The scheme should have reasonable computational complexity;
- (4) Industrial process models are usually erroneous. The failure detection schemes are required to be robust to model errors;
- (5) The system should be easy to be reorganized in order to retain integrity in the presence of failures.

According to the above requirements, a fault detection technique is developed in this section.

Consider the linear discrete stochastic system with the following system dynamics:

$$x(k+1) = A(k)x(k) + B(k)u(k) + w(k) \quad (6.13)$$

$$y(k) = C(k)x(k) + v(k) \quad (6.14)$$

where $x \in \mathfrak{R}^n$, $y \in \mathfrak{R}^m$ and $u \in \mathfrak{R}^r$; w and v are zero-mean, independent, white Gaussian sequences with the covariances defined by:

$$\begin{aligned} E[w(k)w^T(j)] &= Q(k)\delta_{kj} \\ E[v(k)v^T(j)] &= R(k)\delta_{kj} \end{aligned} \quad (6.15)$$

where δ_{kj} is the Kronecker delta. The idea of the failure identification technique is as follows: assume M ($M \leq m$) out of m sensors are of poor reliability and tend to fail. Without the loss of generality, it is assumed that the unreliable sensors correspond to the 1st, 2nd, ..., M th elements of the measurement vector, y . A number of measurement equations are constructed under various failure hypotheses, based on which Kalman filters are designed. Since the performance of the Kalman filters is the good indication of sensor failures, the dynamic case-based reasoning system applies the performance parameters as the composite features (see Chapter 4 for details) for case retrieval and evaluation. The solution contained in the corresponding case will be used for control system reorganization.

Fig. 6.3 shows the principle of the fault-tolerant control system. The pulp machine can be divided into four sections: pulp preparation, pressurized headbox, fourdrinier wire, and dryers. As pointed out in Chapter 2, to improve the problem solving flexibility and efficiency, the process is decomposed into a few sub-processes (sections). The DCBR system first applies a global case memory to locate the section that is in abnormal condition. It then applies the Kalman filter performance parameters as the composite features to find the best matching case from the local case memory for the section that is in abnormal condition. The solution of the best matching case contains the control system reorganization scheme that modifies the control system to maintain acceptable performance. In this chapter, we assume it has been found that the pressurized headbox is in abnormal condition.

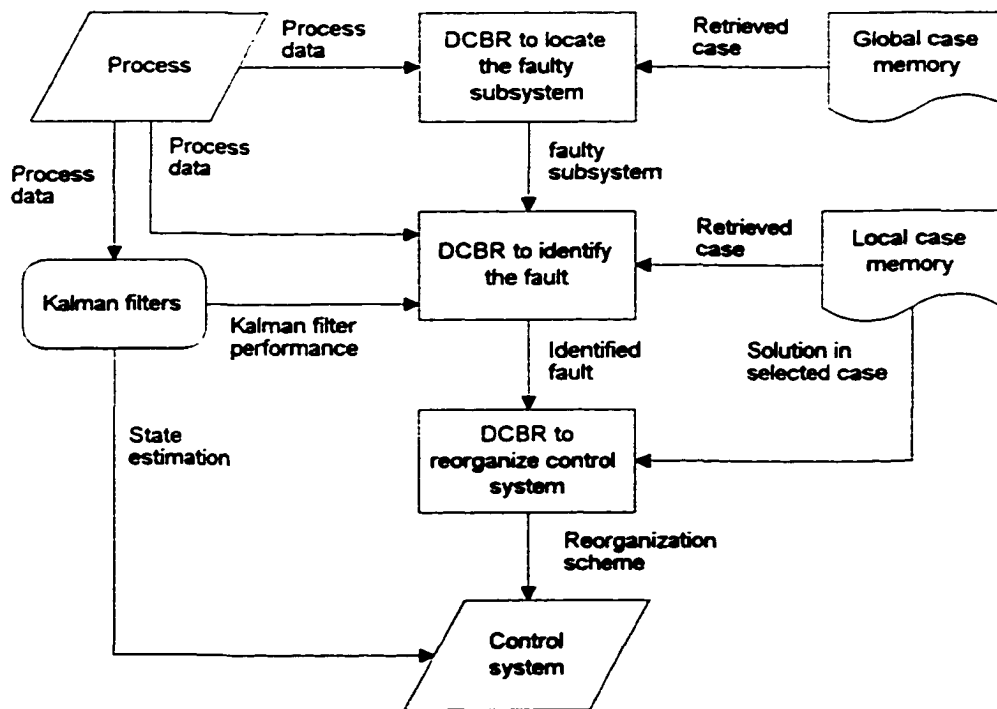


Fig. 6.3. Principle diagram of the fault-tolerant control.

6.3.1 Scheme 1--single-failure case

In this scheme, we focus on single-failure situations, i.e., it is assumed that there will be no more than one sensor fails simultaneously.

For a system with M unreliable sensors, there are $M+1$ hypotheses to consider: the normal condition and the abnormal conditions when the i th ($i = 1, 2, \dots, M$) sensor has failed.

For the normal condition:

$$H_0 : \quad y(k) = C_0(k)x(k) + v(k) + d_0(k) \quad (6.16)$$

where $C_0(k) = C(k)$ and $d_0(k) = 0$.

When the i th sensor fails:

$$H_i : \quad y(k) = C_i(k)x(k) + v(k) + d_i(k) \quad i = 1, 2, \dots, M \quad (6.17)$$

where $d_i(k) = (0 \dots 0 \ m_i(k) \ 0 \dots 0)^T$. $C_i(k)$ is $C(k)$ matrix with the row corresponding to the i th sensor replaced by zero; $m_i(k)$ is the unknown mathematical expectation of the output of the i th sensor. We shall solve the problem as if $m_i(k)$ were known.

There is usually analytical redundancy among dissimilar measurement instruments in industrial control systems. That is to say, not only the system (6.13) and (6.14) is observable, it remains observable when one of the unreliable sensors has failed. That is, $(C_i(k), A(k))$ is observable for all $i = 0, 1, \dots, M$.

The selection of the most probable hypothesis is based on a finite set of measurements, $Y(k) = \{y(1), y(2), \dots, y(k)\}$. $M+1$ Kalman filters, denoted by KF_0, KF_1, \dots, KF_M , respectively, are built based on the measurement equations (6.16) and (6.17):

$$\hat{x}_i(k+1|k) = A(k)\hat{x}_i(k) + B(k)u(k) \quad (6.18)$$

$$\hat{x}_i(k) = \hat{x}_i(k|k-1) + K_i(k)\gamma_i(k) \quad (6.19)$$

$$\gamma_i(k) = y(k) - C_i(k)\hat{x}_i(k|k-1) - d_i(k), \quad i = 0, 1, \dots, M \quad (6.20)$$

where \hat{x}_i ($i = 0, 1, 2, \dots, M$) is the state estimation under the hypothesis H_i . The filter gain $K_i(k)$ is calculated from the following equations:

$$P_i(k+1|k) = A(k)P_i(k)A^T(k) + Q(k) \quad (6.21)$$

$$K_i(k) = P_i(k|k-1)C_i^T(k)[C_i(k)P_i(k|k-1)C_i^T(k) + R(k)]^{-1} \quad (6.22)$$

$$P_i(k) = P_i(k|k-1) - K_i(k)C_i^T(k)P_i(k|k-1) \quad (6.23)$$

According to Bayes' rule, $p_i(k)$, the probability that H_i is true, can be calculated by the measurements using the following relation:

$$p_i(k+1) = \frac{F_i(\gamma_i(k+1))p_i(k)}{\sum_{j=0}^M F_j(\gamma_j(k+1))p_j(k)} \quad (6.24)$$

where $F_i(\gamma_i(k+1))$ is the probability density of $\gamma_i(k+1)$ under the hypothesis H_i . Obviously, under the condition that the hypothesis H_i is true, $\gamma_i(k)$ is the zero-mean, Gaussian innovation process with the covariance as:

$$V_i(k) = E(\gamma_i(k)\gamma_i^T(k)) = C_i(k)P_i(k|k-1)C_i^T(k) + R(k) \quad (6.25)$$

$$F_i(\gamma_i(k+1)) = \frac{\exp[-\frac{1}{2}\gamma_i^T(k+1)V_i^{-1}(k)\gamma_i(k+1)]}{(2\pi)^{\frac{n}{2}}[\det(V_i(k))]^{\frac{1}{2}}} \quad (6.26)$$

where $\det(\cdot)$ denotes the determinant of matrix.

In Eqn. (6.20), the true value of $m_i(k)$ in $d_i(k)$ is not available. It can be estimated from the sample mean of the output of the i th sensor. However, because of the special structure of the matrix $C_i(k)$, the unknown mean $m_i(k)$ does not enter the filter equation.

The probability that H_i is true implies good performance of the filter KF_i . If no failure occurs, the innovation processes of all the $M+1$ Kalman filters are zero-mean Gaussian processes with limited covariances. Thus $p_i(k)$ for all $i=0,1,\dots,M$ have a lower bound, i.e., $p_i(k) \geq \varepsilon$. Here $0 < \varepsilon < 1$ is a detection threshold. When the i th sensor fails, H_i is the only true hypothesis. The innovation processes of all Kalman filters except KF_i will no longer be zero-mean. Their covariances will increase because of the wrong measurement data. The performance of those filters will degrade dramatically. As a consequence, $p_j(k)$ decreases. Thus $p_i(k) \gg p_j(k)$ and $p_j(k) < \varepsilon$ for all $j \neq i$.

Based on these observations, a number of cases can be built for fault identification and control system reorganization:

- **Case # i ($i = 1, 2, \dots, M$) - i th sensor failure**

Case validation conditions:

$$p_i(k) \gg p_j(k) \quad \forall j \neq i \quad (6.27)$$

and

$$\text{Max}\{p_0(k), \dots, p_{i-1}(k), p_{i+1}(k), \dots, p_M(k)\} < \varepsilon \quad (6.28)$$

Control system reorganization:

All the Kalman filters except KF_i are no more effective. Set the final state estimate to be the estimate of the filter KF_i :

$$\hat{x}(k) = \hat{x}_i(k) \quad (6.29)$$

- **Case # 0 – normal condition**

Case validation conditions:

There is no any $i (i \neq 0)$ satisfies the condition (6.27) and (6.28).

Control system reorganization:

Set the final state estimate is a weighted summation of the state estimate of all Kalman filters. Since $p_i(k)$ represents the relative accuracy of the state estimate of each Kalman filters, and noting that:

$$\sum_{i=0}^M p_i(k) = 1 \quad (6.30)$$

$p_i(k)$ can serve as the weighting factor in the summation. The final state estimate is given by:

$$\hat{x}(k) = \sum_{i=0}^M p_i(k) \hat{x}_i(k) \quad (6.31)$$

The detection threshold ε is selected according to the practical situation of the systems.

6.3.2 Scheme 2--multiple-failure case

When there are more than one sensors have failed, none of the Kalman filters in Scheme 1 are designed under true hypothesis. All the Kalman filters are driven by wrong measurements. Therefore, Scheme 1 can not be applied to multiple-failure case. In order to detect multiple failures, we design a bank of filters such that each sensor failure only affects one filter.

Consider $M+1$ hypotheses: the normal condition and when all the unreliable sensors except the i th ($i = 1, 2, \dots, M$) sensor have failed.

In the normal condition, the measurement equation is given by:

$$H_0 : \quad y(k) = \bar{C}_0(k)x(k) + v(k) + d_0(k) \quad (6.32)$$

where $\bar{C}_0(k) = C(k)$ and $d_0(k) = 0$. When all the unreliable sensors except the i th ($i = 1, 2, \dots, M$) one have failed:

$$H_i : \quad y(k) = \bar{C}_i(k)x(k) + v(k) + d_i(k) \quad i = 1, 2, \dots, M \quad (6.33)$$

where $d_i(k) = (m_1(k) \dots m_{i-1}(k) \ 0 \ m_{i+1}(k) \dots m_M(k) \ 0 \dots 0)^T$. $\bar{C}_0(k)$ is $C(k)$ with the rows corresponding to all the unreliable sensors except the i th one replaced by zero. $m_j(k)$ is the unknown mathematical expectation of the output of the j th sensor. If $(\bar{C}_i(k), A(k))$ is observable for all $i = 0, 1, \dots, M$, then just as in Scheme 1, $M+1$ Kalman filters can be designed based on the system dynamics (6.13) and the measurement equations (6.32) and (6.33). Because of the special structure of the matrix $C_i(k)$, the failure of the i th sensor only affect the filters KF_0 and KF_i . Multiple failures can be detected by evaluating the performance of the Kalman filters.

The filter equations and the calculation of $p_i(k)$ are similar to those in Scheme 1. Defining $J = \{j_1, j_2, \dots, j_s\} \subset \{1, 2, \dots, M\}$, the algorithm for fault detection and state estimation is expressed as follows:

- **Case # 1 - j_1 th, j_2 th, \dots , j_s th sensors have failed**

Case validation conditions:

$$p_0(k) < \varepsilon \quad (6.34)$$

and

$$p_i(k) < \varepsilon \quad \forall i \in J \quad (6.35)$$

Control system reorganization:

Set the final state estimate as

$$\hat{x}(k) = \sum_{j \in J} \frac{p_j(k)}{1 - p_0 - \sum_{i \in J} p_i(k)} \hat{x}_j(k) \quad (6.36)$$

- **Case # 0 – normal condition**

Case validation conditions:

There is no any subset $J \subset \{1, 2, \dots, M\}$ satisfies (6.34) and (6.35).

Control system reorganization:

Set the final state estimation as

$$\hat{x}(k) = \sum_{i=0}^M p_i(k) \hat{x}_i(k) \quad (6.37)$$

Obviously, Scheme 2 requires more sensor redundancy than Scheme 1. The best scheme to be selected depends on the sensor redundancy and the potential of multiple failures.

6.3.3 Scheme 3--reduced-order technique

As pointed out in Scheme 1, the output of the i th sensor does not participate in the state estimation of the filter KF_i , because of the special structure of the matrix $C_i(k)$. For the sake of convenience in industrial application, it is preferable to design M reduced-order Kalman filters, denoted by KF_1, KF_2, \dots, KF_M respectively, based on the system equation (6.13) and measurement equation (6.38):

$$y_i(k) = \hat{C}_i(k)x(k) + v_i(k) \quad i = 1, 2, \dots, M \quad (6.38)$$

where $y_i(k)$ is $y(k)$ vector with the element corresponding the i th unreliable sensor deleted, $\hat{C}_i(k)$ and $v_i(k)$ are the corresponding measurement matrix and measurement noise vector. The estimate error sequences γ_i are:

$$\gamma_i(k) = y_i(k) - \hat{C}_i \hat{x}_i(k) \quad i = 1, 2, \dots, M \quad (6.39)$$

It is clear from Eqns. (6.38) and (6.39) that the dimensions of the measurement vectors have been reduced to $m-1$. The estimation of the mathematical expectation of the output of sensors in failure is also avoided. Thus the computational requirement is reduced.

The multiple-filter algorithm is intended to measure how well each Kalman filter works. $p_i(k)$ is a convenient measure of the performance of the Kalman filters relative to each other and to what they are expected to work. Since the performance of the Kalman filters depends on the performance of the sensors, $p_i(k)$ is also a measure of the accuracy of the sensors. In the normal condition, the filter KF_0 should give the best result since it applies the most measurement information. When any of the unreliable sensors fails, the performance of KF_0 will degrade. Only one Kalman filter KF_i ($i \neq 0$) will still work well. Thus we can use the prediction errors of the filters KF_1, KF_2, \dots, KF_M to detect failures and use the estimate of the filter KF_0 as the final state estimate in the normal condition.

A new failure detection scheme is proposed that is a little different from Scheme 1.

Calculate $p_i(k)$ as:

$$p_i(k+1) = \frac{F_i(\gamma_i(k+1))p_i(k)}{\sum_{j=1}^M F_j(\gamma_j(k+1))p_j(k)} \quad i = 1, 2, \dots, M \quad (6.40)$$

where

$$F_i(\gamma_i(k+1)) = \frac{\exp[-\frac{1}{2}\gamma_i^T(k+1)V_i^{-1}(k)\gamma_i(k+1)]}{(2\pi)^{\frac{m-1}{2}} \det(V_i^{-1}(k))^{\frac{1}{2}}} \quad (6.41)$$

$$V_i(k) = E(\gamma_i(k)\gamma_i^T(k)) = \hat{C}_i(k)P_i(k|k-1)\hat{C}_i^T(k) + R_i(k) \quad (6.42)$$

There are differences between Eqns. (6.24)–(6.26) and Eqns. (6.40)–(6.42): In Eqn. (6.42), $\hat{C}_i(k)$ and R_i instead of $C_i(k)$ and R are used. Here R_i is the covariance of the measurement noise v_i ; In Eqn. (6.40) the sum is from $j = 1$ to M instead of from $j = 0$ to M ; In Eqn. (6.41), $(2\pi)^{\frac{n-1}{2}}$ instead of $(2\pi)^{\frac{n}{2}}$ is used.

The following cases are built for fault identification and system reorganization.

- **Case # i ($i = 1, 2, \dots, M$) - i th sensor failure**

Case validation conditions:

$$p_i(k) \gg p_j(k) \quad \forall j \neq i \quad (6.43)$$

$$\text{Max}\{p_0(k), \dots, p_{i-1}(k), p_{i+1}(k), \dots, p_M(k)\} < \varepsilon \quad (6.44)$$

Control system reorganization:

Set the final state estimate as

$$\hat{x}(k) = \hat{x}_i(k) \quad (6.45)$$

- **Case # 0 – normal condition**

Case validation conditions:

There is no any $i \in \{1, 2, \dots, M\}$ that satisfies the conditions (6.43) and (6.44).

Control system reorganization:

Set the final state estimate as

$$\hat{x}(k) = \hat{x}_0(k) \quad (6.46)$$

The $p_i(k)$ expressed in Eqn. (6.40) may not be exactly the probability that the i th unreliable sensor has failed since the M Kalman filters use different measurement data. However, this scheme does reduce the computational complexity of state estimate and

failure detection. It is preferable in practical application. This simplified scheme can also be applied to the multiple-failure case by employing the results obtained in Scheme 2.

To further reduce the real-time computational requirement, the steady state Kalman filters can be applied for time-invariant systems. In steady state Kalman filter, $K_i(k)$ and $\hat{V}_i^{-1}(k)$ are both constant matrices and can be calculated off-line. In this way, the real-time computational requirement can be reduced to a very moderate level.

One of the most significant advantages of this technique is its inherent ability for state estimator reorganization subsequent to the occurrence of sensor failure.

6.4 Fault-Tolerant Control of the Headbox

To realize the fault-tolerant control, the problem of equal importance to that of fault detection is the issue of system reorganization to retain system integrity in the presence of failures. In the previous section, three fault detection and state estimator reorganization schemes have been developed. In this section, we apply Scheme 3 for the design of the fault-tolerant control system for the pressurized headbox.

The system dynamics and measurement equation of the pressurized headbox have been given in (6.9)-(6.11). As mentioned in Section 6.2, all the three sensors may fail during operation. There are, therefore, four hypotheses to be considered:

$$H_0 : \quad y(k) = \begin{pmatrix} 1.0 & 0.0 \\ 0.0 & 1.0 \\ 1.0 & -1.0 \end{pmatrix} x(k) + v(k) \quad (6.47)$$

$$H_1 : \quad y(k) = \begin{pmatrix} 0.0 & 0.0 \\ 0.0 & 1.0 \\ 1.0 & -1.0 \end{pmatrix} x(k) + d_1(k) + v(k) \quad (6.48)$$

$$H_2 : \quad y(k) = \begin{pmatrix} 1.0 & 0.0 \\ 0.0 & 0.0 \\ 1.0 & -1.0 \end{pmatrix} x(k) + d_2(k) + v(k) \quad (6.49)$$

$$H_3 : \quad y(k) = \begin{pmatrix} 1.0 & 0.0 \\ 0.0 & 1.0 \\ 0.0 & 0.0 \end{pmatrix} x(k) + d_3(k) + v(k) \quad (6.50)$$

where H_0 is the hypothesis for normal condition. H_1 , H_2 and H_3 are the hypotheses assuming the total head sensor, stock level sensor and air pressure sensor have failed, respectively. For all the hypotheses, the system is completely observable.

The selection of the variance of system noise $w(k)$ should consider the uncertainty of the knowledge in the process dynamics, the relative scale of the variables and environmental disturbances. The selection of the variance of measurement noise $v(k)$ should consider the accuracy of the sensors. After careful investigation, the variance matrices are selected as $Q = \text{diag}(0.025^2, 0.015^2)$ and $Q = \text{diag}(0.035^2, 0.02^2, 0.05^2)$.

To reduce the real-time computational requirement, four steady state Kalman filters, denoted by $KF_0 - KF_3$, are designed under various failure assumptions. $p_1(k)$, $p_2(k)$ and $p_3(k)$ are calculated by Eqns. (6.40)-(6.42). If the pulp stock level sensor fails, both $p_1(k)$ and $p_3(k)$ will be less than ε . As a result, $p_2(k)$ will be larger than $1 - 2\varepsilon$. Similar results can be obtained easily if the total head sensor or air pressure sensor fails.

The total head H is the summation of the stock level L and the air pressure P . Since the three sensors have the same scale, there is a direct relation between the output of the sensors:

$$y^1(k) = y^2 - y^3(k) \quad (6.51)$$

where y^i ($i = 1, 2, 3$) is the i th component of the measurement vector y . When one sensor fails, the above relation will no longer true. Thus, Eqn. (6.51) can be applied to fault detection.

Selecting $\varepsilon = 0.005$, the fault detection and system reorganization scheme can be outlined as follows:

- **Case # 1 – The normal condition**

Case validation conditions:

$$|y^1(k) - y^2(k) - y^3(k)| \leq 0.15 \text{ or } \text{Max}\{p_1(k), p_2(k), p_3(k)\} \leq 0.99$$

Control system reorganization:

Set the final state estimate as

$$\hat{x}(k) = \hat{x}_0(k) \tag{6.52}$$

- **Case # 2 - The total head sensor failure**

Case validation conditions:

$$|y^1(k) - y^2(k) - y^3(k)| > 0.15 \text{ and } p_1(k) > 0.990$$

Control system reorganization:

Set the final state estimate as

$$\hat{x}(k) = \hat{x}_1(k) \tag{6.53}$$

- **Case # 3 - The stock level sensor failure**

Case validation conditions:

$$\text{If } |y^1(k) - y^2(k) - y^3(k)| > 0.15 \text{ and } p_2(k) > 0.990$$

Control system reorganization:

Set the final state estimate as

$$\hat{x}(k) = \hat{x}_2(k) \quad (6.54)$$

- **Case # 4 - The air pressure sensor failure**

Case validation conditions:

$$\text{If } |y^1(k) - y^2(k) - y^3(k)| > 0.15 \text{ and } p_3(k) > 0.990$$

Control system reorganization:

Set the final state estimate as

$$\hat{x}(k) = \hat{x}_3(k) \quad (6.55)$$

The decoupling control law is realized by applying the final state estimate:

$$u(k) = F \cdot \hat{x}(k) + G \cdot v_m(k) \quad (6.56)$$

where the feedback gain F and the feedforward gain G are synthesized using the state feedback decoupling algorithm proposed by Gilbert (1969):

$$F = \begin{pmatrix} 0.498 & -0.291 \\ 0.402 & 2.45 \end{pmatrix}$$

$$G = \begin{pmatrix} 0.0731 & 0.260 \\ 0.247 & -3.20 \end{pmatrix}$$

For performance comparison, a decoupling controller is also implemented by applying the state estimate of the Kalman filter under normal assumption:

$$u(k) = F \cdot \hat{x}_0(k) + G \cdot v_m(k) \quad (6.57)$$

Note that in this section, the failure detection conditions (6.43) and (6.44) have been reduced to the simpler approximate form, $p_i > 1 - (M - 1)\varepsilon$, for convenience.

From (6.40), it can be seen that if $p_i(k)$ is very small, the $p_i(k+1)$ will grow very slowly at best. To avoid this problem, a lower bound of 0.001 is set on the $p_i(k)$. If the calculated $p_i(k)$ is below the lower bound, set $p_i(k)$ to be the lower bound. Another problem is the divergence of the Kalman filter. When a sensor fails, the Kalman filter based on the incorrect hypothesis may diverge. Even after the sensor has been repaired, it will take a long time before the Kalman filter gives meaningful state estimation again. To avoid this problem, the estimate of the potentially divergent Kalman filter is set to be equal to the final state estimate.

The performance of the headbox fault-tolerant control system has been studied using digital simulation. The initial state is assumed to be zero. The setpoint of total head is increased to 1 at $k = 150$ and then goes back to 0 at $k = 350$. The setpoint of pulp stock level is increased to 1 at $k = 50$ and then goes back to 0 at $k = 250$. It is assumed that failures occur at $k=100$ and the failed sensors have a constant output, 0.2. Figs. 6.4 and 6.5 show the responses of the fault-tolerant control system and the normal decoupling control system, respectively, when the total head sensor has failed; Figs. 6.6 and 6.7 show those when the stock level sensor has failed; Figs. 6.8 and 6.9 show those when the air pressure sensor has failed. The solid curves represent the setpoint and actual response of the pulp stock level. The dotted curves represent that of the total head. From these figures, it is obvious that when any sensor has failed, the dynamic performance of the normal decoupling control system becomes very poor and the system is no longer decoupled. However, the fault tolerant control system maintains satisfactorily dynamic and decoupling performances.

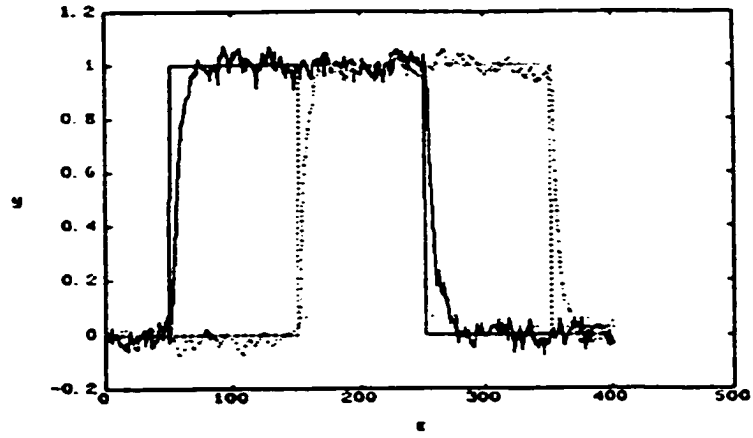


Fig. 6.4. Response of fault-tolerant control in total head sensor failure.

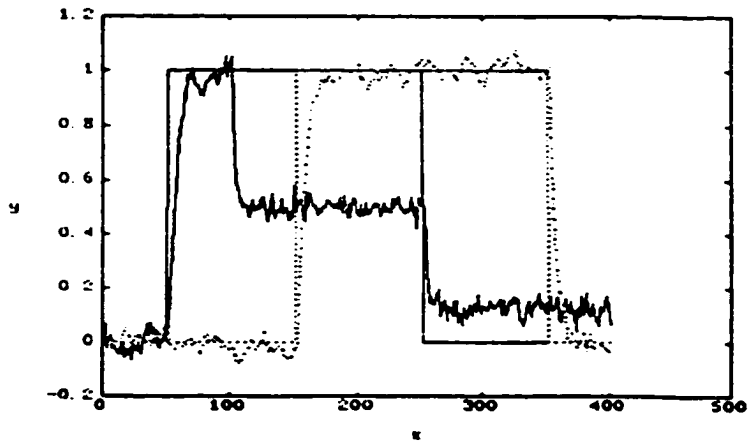


Fig. 6.5. Response of conventional control in total head sensor failure.

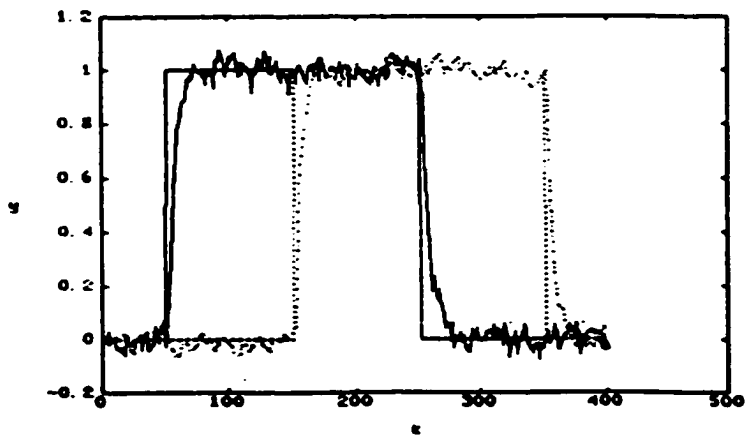


Fig. 6.6. Response of fault-tolerant control in pulp stock level sensor failure.

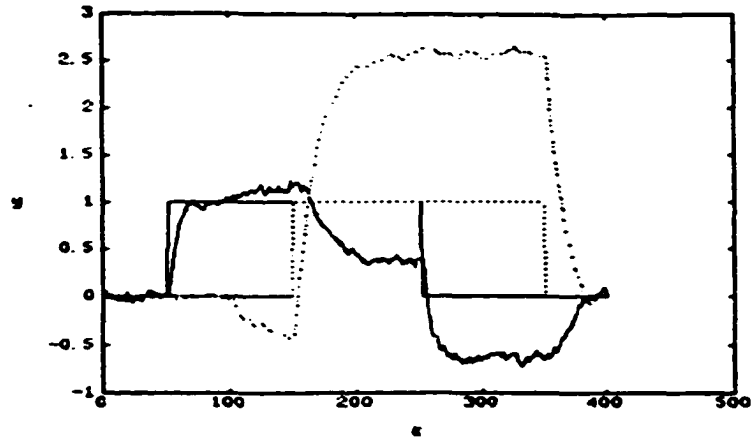


Fig. 6.7. Response of conventional control in pulp stock level failure.

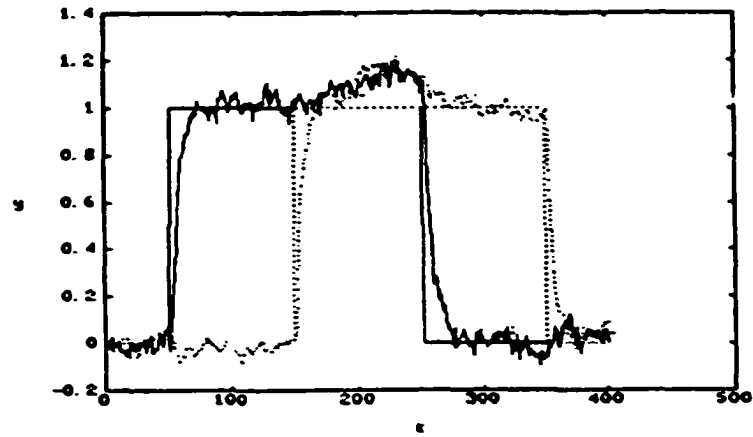


Fig. 6.8. Response of fault-tolerant control in air pressure sensor failure.

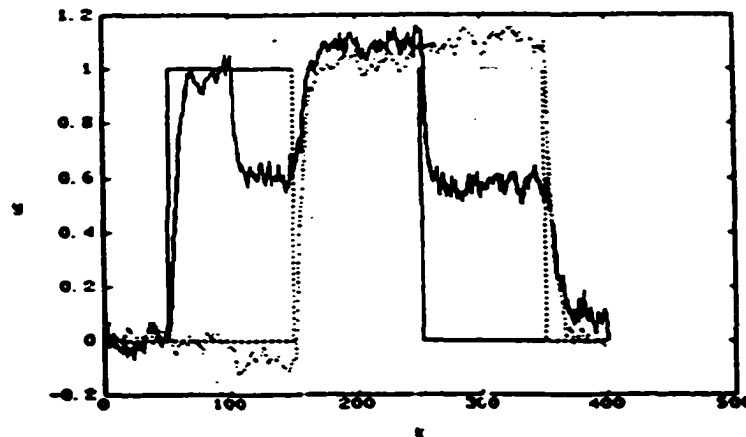


Fig. 6.9. Response of conventional control in air pressure sensor failure.

6.5 Conclusions

In this chapter, a fault-tolerant control technique that tolerates sensor failures was proposed. The technique includes fault detection schemes and state estimator reorganization schemes subsequent to the occurrence of a failure. The fault detection is accomplished by the dynamic case based reasoning (DCBR) system coupled with a bank of Kalman filters using the analytic redundancy among dissimilar measurements. The state estimator is reorganized to give the most accurate state estimate in normal unfailed condition and when some sensor has failed. A decoupling control law is realized using the state estimate.

The technique has reasonable real-time computational requirement and is easy to be implemented. An important feature of the technique is its inherent capability of automatic reorganization of the system structure in the presence of the failure. Simulation results in a real world pressurized headbox showed satisfactory performance.

Chapter 7

INTEGRATION OF INTELLIGENT OPERATION SUPPORT SYSTEM WITH MULTIMEDIA TECHNOLOGY[†]

This chapter presents the design and application of an intelligent hypermedia on-line manual (IHOM). This research is motivated by two facts. First of all, the intelligent operation support system (IOSS) requires a manual to be fully integrated with it and serve as an extension of knowledge base and a multimedia operator interface. Secondly, the plant operations require a manual that has more functionality than the traditional paper manuals. A new integration model of intelligent system technology with multimedia technology is proposed. The designed IHOM has built-in knowledge base to perform numerical calculation and symbolic reasoning. Real time data can be embedded in the manual. By changing the link relationship of the intelligent link in IHOM, the IOSS is able to use IHOM as a multimedia operator interface that delivers operation suggestions to operators. By moving some of the calculation and reasoning tasks to IHOM, the IOSS is able to reduce the complexity of its core system. As a result, the system will have better user acceptance and higher efficiency. The IHOM has been applied to a bleached chemical thermo-mechanical pulp (BCTMP) plant.

[†] This chapter has been submitted for publication: Xia, Liu, Shu and Rao, 1999, "Integration of intelligent operation support system with multimedia technology", *Engineering Application of Artificial Intelligence*.

7.1 Introduction

One of the important problems in the intelligent operation support system (IOSS) design is operator interface. An operation support system monitors the production processes and analyses potential process and operation problems. It provides operators with operation suggestions. Operators take proactive and corrective actions based on the suggestions and their own judgment. It is crucial that the information provided by the operation support system is sufficient and easy-to-understand. However, the operator interface of traditional intelligent systems can usually only handle simple text information, which limits the effectiveness and user acceptability of intelligent systems.

One of the requirements for operator interface is the capability of handling the information in multimedia forms. Multimedia information has been extensively applied in process operations. The operation support system should provide operation suggestions in the original multimedia forms instead of converting it to a simple text form. Otherwise, information loss may occur during the conversion. Another requirement is the capability of providing detailed process and operation information as required. The modern production process is extremely complex. Operators may need the details of the process and operation procedures to verify a process problem or evaluate a corrective plan. Such information is usually available in process or operation manuals. Searching through a traditional paper manual is inefficient, which is not desirable in a time critical situation. The solution to this problem is a multimedia on-line manual that is fully integrated to the intelligent operation support system. The on-line manual, by using the problem solving results of the IOSS, has "intelligence" to help the operators to find the relevant information quickly. It serves as an enhanced multimedia operator interface and an external knowledge base. This research is intended to solve this problem.

Ragusa (1994) defined multimedia as the computer-facilitated integration of multiple information formats. He classified the multimedia to two groups according to their time characteristics: static media and dynamic media. Static media includes text, data, graphics and still images. Dynamic media includes animation, full-motion video, speech and

nonspeech audio. The development of multimedia computers (Bailey, 1990) and multimedia devices such as compact disc read only memory (CD-ROM), recordable CD-ROM, and erasable rewritable optical discs (EROD), makes the integrated displays of multimedia information with computer possible (Maule, 1991; Staub and Wetherbe, 1989). Today the integrated use of multimedia information has been considered as a new and efficient way to improve the training and operations.

The powerfulness of multimedia technology comes from the fast and effective integration and accessing of information in various media forms (Short, 1992). Hypermedia systems, a technology to combine various media forms in a unified information delivery system, have gained more and more attention in process industry.

Hypermedia is the management system for multimedia information (Paske, 1990). It integrates information pieces (data objects) in different media forms as nodes in a vast network. These nodes can be organized hierarchically or non-hierarchically according to the nature of the problems. An information piece can be referred from several sources. Through nonlinear links and associative links, user can interactively navigate through an information base and find a specific content (node) quickly. Because of its capability of integrating multiple information form and high efficiency in searching, hypermedia technology has been widely used in almost every aspect of our life. However, the most successful applications in process industry are limited to off-line activities, such as new operator training and presentations. Very rare are real time on-line applications. There has been considerable number of researches and commercial systems available in organizing and displaying information in multimedia formats (Raskin, 1990; Yagaer, 1991; Paske, 1990). However, it is difficult to integrate them with the intelligent operation support system.

This chapter will outline the structure of the intelligent hypermedia on-line manual (IHOM) and the applications to a BCTMP plant.

7.2 Problem Formulation

The development of intelligent hypermedia on-line manual (IHOM) is motivated by two facts. First of all, the IOSS requires a manual to be fully integrated with it and serve as an extension of knowledge base and operator interface. Secondly, the plant operations require a manual that has more functionality than the traditional paper manuals (Liu et al., 1995). Paper manuals can no longer satisfy the requirement of modern production because of its low efficiency in searching, high cost to update, and failure to incorporate multimedia information.

For the above purposes, IHOM has to be an electronic document that has the following features:

- **Capable of including multimedia information:** The system should be able to manage and deliver all the information forms used in process operations, such as animation, video, sounds, speech and pictures. For example, the burning processes in pulp production are usually video monitored.
- **Efficient in searching:** The IHOM is intended to help the operators to find the relevant information quickly by using built-in features, such as effective nonlinear links in the design environment and the incorporated cognitive process in arranging the information pieces.
- **Capable of numerical calculation:** The IHOM often contains formulas to calculate various process variables, such as production cost and bleach chemicals addition rate. The numerical calculation capability will also allows the IOSS to move some of its calculation tasks to the IHOM.
- **Capable of symbolic reasoning:** An inference engine is required in the IHOM for symbolic manipulation. Similar to the numerical calculation capability, this feature will allow the IOSS to move some reasoning activities to IHOM to reduce the complexity of IOSS.
- **Intelligent links:** Unlike the associative links that connect a source to a single deterministic reference, the intelligent links allow a source to be connected to

multiple references controlled by the linking conditions. With this feature, a single source “Suggested Corrective Actions” can be connected directly to the solution to an operation problem that is identified by the IOSS.

- **Integration with the IOSS:** Since IHOM serves as a multimedia operator interface and an external knowledge base of the IOSS, the integration of these two systems is essential.
- **On-line real time:** As an on-line manual, the IHOM is required to have access to real time production data that is commonly from a DCS or from data processing systems or packages, such as the management information systems.
- **Visual development tool:** For the easy use of nonprogrammers, the development tool is a visual environment, that is, what you see is what you get.

The limitation of the traditional hypermedia systems has been addressed by previous researches (Halasz, 1988; Ragusa, 1994). Most of the above features are difficult to be implemented with available hypermedia systems. These limitations have limited the application of the hypermedia technology in process industry.

The key to solve the problem is to embed knowledge into hypermedia systems, that is, to integrate the intelligent systems with the hypermedia systems. The next section will outline the structure of the IHOM.

7.3 IHOM System Structure

7.3.1 Integration model

The key issue is how to integrate intelligent system with multimedia information system to meet the requirements indicated in Section 7.2. Several integration models were proposed (Coyne, 1990; Maybury, 1992; Ragusa and Coyne, 1992; Sipior and Garrity, 1992). Ragusa (1994) classified the integration models into five categories in terms of software architectures: (1) standalone, (2) translational, (3) loose coupling, (4) tight coupling, and (5) full integration; and three categories in terms of integration orientations: (1) expert

system (ES) supported by multimedia system (MM), (2) MM supported by ES, and (3) complementary systems. Each model has advantages and disadvantages in terms of maintenance effort, development speed, flexibility and potential for problem solving.

In the IOSS, we need an intelligent hypermedia system that replaces the traditional paper manual and serves as an extended operator interface and external knowledge base of IOSS. These requirements conclude that the integration between the intelligent system and the multimedia system is quite complicated. On one hand, multimedia system is supported by intelligent system with full integration. On the other hand, intelligent system is supported by multimedia system with translational integration. None of the above integration models can meet the requirements.

In accordance with the above considerations, a new integration model is proposed for the IOSS as illustrated in Fig. 7.1.

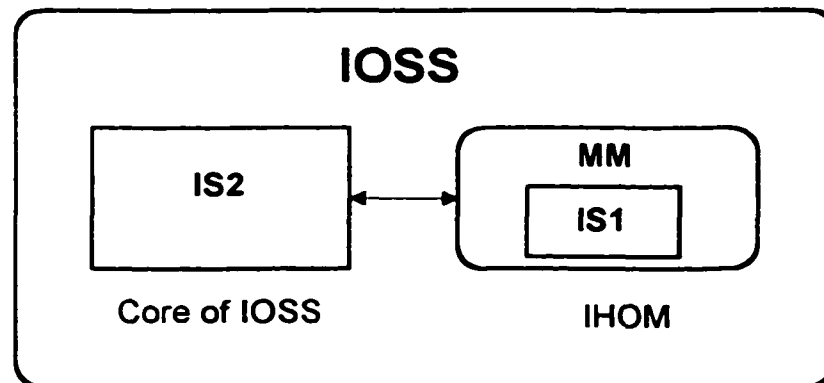


Fig. 7.1. Integration of multimedia system with intelligent systems.

In this integration model, there are two intelligent systems and one multimedia system. The intelligent system IS_1 is fully integrated with the multimedia system (MM) and supports MM to become IHOM. The intelligent system IS_2 is the core of the intelligent operation support system (IOSS). The combination of intelligent system IS_2 and IHOM constructs IOSS with enhance operator interface and external knowledge base. The

integration of IS_2 and IHOM is translational. IS_2 is supported by IHOM and affects IHOM only through data transmission. Both IS_2 and IHOM can run independently.

IHOM is an MM embedded with an intelligent system. It is a graph defined by a set of nodes (NODE) and the link relationship (LINK) between them as well as embedded rule sets (RULE) and procedures (PROC):

$$IHOM = (NODE, LINK, RULE, PROC)$$

The NODE consists of files (FILE), topics (TOPIC) and link sources (LNSR) (keywords and regions) in the topics

$$N = (FILE, TOPIC \cup LNSR)$$

IHOM is a topic-based application tool. Each topic is an information fragment, which consists of a tag name, static media contents, dynamic real-time data and dynamic variables. The dynamic real-time data is obtained from IS_2 and MIS, and updated in a fixed time period. The dynamic variables, implemented with the embedded rule sets and procedures provided by IS_1 , fulfill numerical calculations.

The link relationship (LINK) connects information fragments. With the reasoning capability of IS_1 , IHOM can not only implement hyperlinks, the prespecified nonlinear links, but also intelligent links. An intelligent link connects a single link source to different targets, controlled by the defined linking conditions. Based on the different conditions, one or none of the link will be activated and the source is connected to the activated reference.

The numerical calculation and symbolic reasoning capabilities is implemented with the knowledge base in IHOM. There are two ways to build a knowledge base. One is by rule sets (RULE), the other is by procedures (PROC). Any button, marked text or dragged rectangular area in a picture can be linked to a rule set or a procedure and act as a trigger. The rule set or procedure will be executed when its trigger is clicked. A procedure can also be defined as an auto-procedure which is triggered cyclically in a given time interval, or as a configure procedure which is triggered when entering the topic every time. Since

the knowledge base is divided into many pieces in various topics, the efficiency of problem solving is high.

The integration of the core of IOSS, i.e., IS1, with IHOM is through data communications. IOSS sends the problem solving results as data to IHOM, which affects the reasoning procedures and intelligent links in IHOM. IOSS also receives calculated data from IHOM to be used in its problem solving.

7.3.2 Software environment

The IHOM development tool is developed using an object-oriented programming technique, Meta-COOP (Rao, et al., 1993; Shu et al., 1995). Meta-COOP is coded in C++, and runs under UNIXTM, VMSTM and DOSTM operating systems. Meta-COOP adopts the object-oriented programming technique and frame-based knowledge representation to implement the organization, management, maintenance and applications of a complex knowledge base system. It provides such distinct characteristics as the integration of various knowledge representations and inference methods.

Meta-COOP distributes its meta-knowledge into many knowledge units. Each knowledge unit is a set of rules, operation commands or numerical models to deal with an operation problem and attached to a *frame* or an *object* that is dedicated to the problem. *Frame-based* or *object-based* knowledge representation constructs the knowledge base hierarchically to represent the internal relationship between the knowledge units.

To solve a complicated engineering problem, the problem domain is decomposed into many basic sub-problems. A sub-problem is solved independently in a slot or a frame by a set of production rules, operation commands, analysis methods, external procedures written in any languages, and internal subroutines written in C++. Meta-COOP accomplishes communication and conflicting coordination between sub-problem solving by applying the internal relationship between the knowledge units.

IHOM is a visual development environment with hypermedia functions and embedded knowledge bases. User can easily create multimedia topics, create link relationship, select screen transitions, define topic variables and build knowledge bases.

IHOM has the advantage of both hypermedia systems and intelligent systems. It allows hypermedia nodes to associate with a knowledge base of an expert system. It provides a reasoning mechanism to browse the media.

Client/server computing is a paradigm in which applications, data, and even processing power can be distributed between a centralized server computer and all other computers that access it. To implement the client-server support, IHOM uses the Open Database Connectivity (ODBC) interface. The ODBC allows applications to access data from database management systems (DBMS). The interface permits maximum interoperability - a single application can access diverse database management systems. User can develop, compile, and ship an application without targeting a specific DBMS. He can then add the modules called database drivers that link the application to the selected database management systems.

With the ODBC interface capability, IHOM supports SQL statements for DBMS operations. Applications can access to different DBMS products. The client-server architecture is shown in Fig. 7.2.

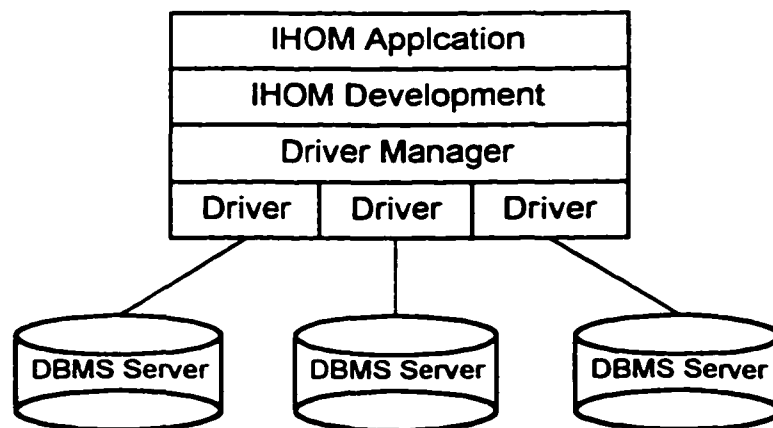


Fig. 7.2. Client – server architecture.

To integrate hypermedia static database and real time database, IHOM has a data processing system built in to process both static data, real time data and user input data.

Because of the special characteristics of Meta-COOP, IHOM is an object-oriented development tool that makes innovative use of object-oriented techniques to user-interface objects. It treats each topic as an object. A topic can combine formatted text, embedded graphics, video, sound, and animation together.

Two types of variables can be defined in IHOM, topic variables and global variables. Topic variables are valid only within the topic where they are defined. Global variables are shared by all topics.

The most important functions in IHOM are procedures and rule sets. A procedure or a rule set is a knowledge unit built in a "method" in IHOM. There are three types of procedures: configure procedure, auto-procedure and normal procedure. A configure procedure is used to describe the action when a topic is activated. An auto-procedure is used to describe actions after any change happens in the topic, or the topic screen is refreshed. Users can also create normal procedures for a link icon and it will take action when the link icon is activated. The function of a rule set is similar to a normal procedure. However the rule set can represent heuristic knowledge. A rule set can be activated by clicking its link icon. It can also be activated from a procedure through a function call.

Intelligent link is one of the most important features of IHOM. Hyperlink in a hypermedia system is static and fixed linkage set up when the system is developed, whereas intelligent link is a dynamic linking mechanism. The link destination is decided at the run time. The intelligent linkage is realized by using a rule set associated with the link source. The rule set is activated when the link source is clicked. The rule whose condition part is matched is executed and the destination in the action part of the rule is displayed. Intelligent link provides powerful browsing mechanism. Navigation in a hypermedia system is much like traveling in a huge concept tree. Because of the large number of levels and branches in the tree, it is quite difficult to find a destination node. The user has to know where he is in the network and how to enter other topics. They may get lost, commonly called disorientation problem. The intelligent links and other knowledge built in the nodes of IHOM will greatly reduce the number of relevant branches and speed up the searching process.

7.4 IHOM Application in a BCTMP Plant

The IHOM application for a bleached chemical thermo-mechanical pulp (BCTMP) plant is composed of three portions:

- **Process Details:** the description of process and equipment in the pulp production.
- **Operation Support:** operation suggestions in various abnormal situations.
- **Company Overview:** general introduction to the company.

This manual has been installed in the plant computer system and interconnected with the intelligent operation support system (IOSS).

7.4.1 Manual structure

The structure of the on-line manual is shown in Fig. 7.3. The Process Details are composed of 13 chapters. Each chapter consists of a number of sections, and each section is divided into a few topics. The user can go to any topic through the table of contents or process flowchart. The Operation Support consists of a number of topics that describe the causes and corrective actions of the identified abnormal process situations. Each topic is corresponding to a case in the case memory of the IOSS, and is activated by the intelligent operation support systems. The Company Overview consists mainly of pictures and video clips. These three portions are connected to each other. For example, the Operation Support topic for the interstage bleaching temperature control problem is directly connected to the process and equipment details of the interstage bleaching and washing in the Process Details.

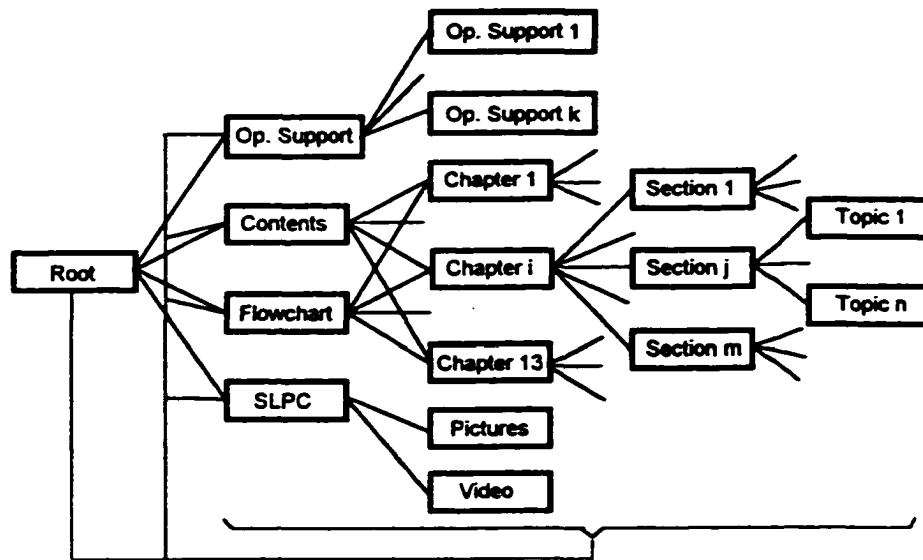


Fig. 7.3. The structure of IHOM for the BCTMP plant.

7.4.2 Searching mechanism

To improve the efficiency and convenience of searching for relevant information, a number of searching methods are built in the IHOM for topic searching:

- Table of contents
- Topic index
- Process flow chart
- Link sources embedded in topics
- Intelligent link triggered by IOSS.

The table of contents is organized into a tree structure, as shown in Fig. 7.4. It gives the user a clear picture of the organization of the manual and allows them to find the content they want. There are 13 tables of contents, each for a chapter. For example, by clicking Principles of Screening, the screen will be switched to the topic of the description of the principles of screening immediately (Fig. 7.5).

INTE MOR	
File Index Option Help	
Ranger SLPC On-line Manual	
Introduction and Overview	Introduction and Overview
Refining Equipment	Principles of Screening
Chip Washing	Components
Screening	- Latency Chest
Primary Refining	- Latency Chest Aniliner
Inter-stage Washing and Bleaching	- Latency Chest Pumps
2nd Stage Washing	- Primary & Secondary Screens
Secondary and Tertiary Refining	- Primary Screens Rejects Collection Tank (RSPCT)
Screening	- RSPCT Aniliner & RSPCT Pumps
3rd Stage Washing and Bleaching	- Rejects De-watering Screens Heat Tank
4th Stage Washing and Bleaching	- Rejects De-watering Sidehill Screens
Refining and Drying	- Rejects Collection Tank (RCT)
Heat Recovery	- RCT Aniliner & RCT Pumps
	- Cloudy White Water Distribution Pumps
	- Screens Feed Dilution Pumps
	Detailed Process Flow
	- Latency Chest
	- Latency Chest Level
	- Latency Chest Pumps
	- Pressure Tank
	- Primary Screen Control
	- Primary Screens Rejects Collection Tank
	- Secondary Screen Control
	- Rejects Sidehill Screens & Collection Tank

Fig. 7.4. The table of the contents of the IHOM.

INTE MOR

File Index Option Help

Ranger SLPC On-line Manual

Screening: Principles of Screening

Fibres discharged from the final stage of refining are stiff, twisted and curled due to the refining action. This is known as latency which must be removed to efficiently screen the pulp. The Latency Chest provides latency removal treatment by diluting the pulp to 3 or 4% consistency (from 35%) using hot cloudy white water at 70 to 90<198> C and agitating for about 20 minutes. The hot white water causes the individual fibres to relax and straighten out before being fed to the screens. (Fig 9-2)

The objective in the screening process is to eliminate unwanted fibres such as shives, fibre bundles and other large debris. In a pressure screen system there are three basic stock flows: feed coming into the screen, the accepts from the screen and rejects from the screen (Fig 9-3).

The incoming stock enters near the top of the unit and under pressure is forced down through the narrow zone between the rotating drum and the inside of the screen. Acceptable fibres pass through the screen plate and are discharged to the accepts stream. The rejects (shives, fibre bundles etc.) are retained on the inside of the screen and discharged at the bottom.

In order to keep a mat from forming on the screen basket, the rotating drum has a number of bumps on its surface that pass very close to the surface of the screen. As the drum rotates, it creates a vacuum and turbulence around the surface of the screen and helps to break up any mat formation produced by fibres bridging across the screen holes.

Dilution is also provided in order to prevent high concentration of rejects.

Fig. 7.5. Topic of the principle of screening.

Another primary searching method is by using the topic index. Each topic in the IHOM has a unique name. By using the command Index on the menu bar, all topic names will be popped up alphabetically. The users can find the topic of interest by typing in the topic name or by scrolling through the topic list.

The process flowchart searching method is mainly used by the users who are familiar with the process. Fig. 7.6 shows the flowchart of the BCTMP process. The regions on the flowchart are linked to relevant topics. By clicking any equipment in the flowchart, the manual will be switched to the process details of the equipment, from there the user can access the identified abnormal situations in the equipment.

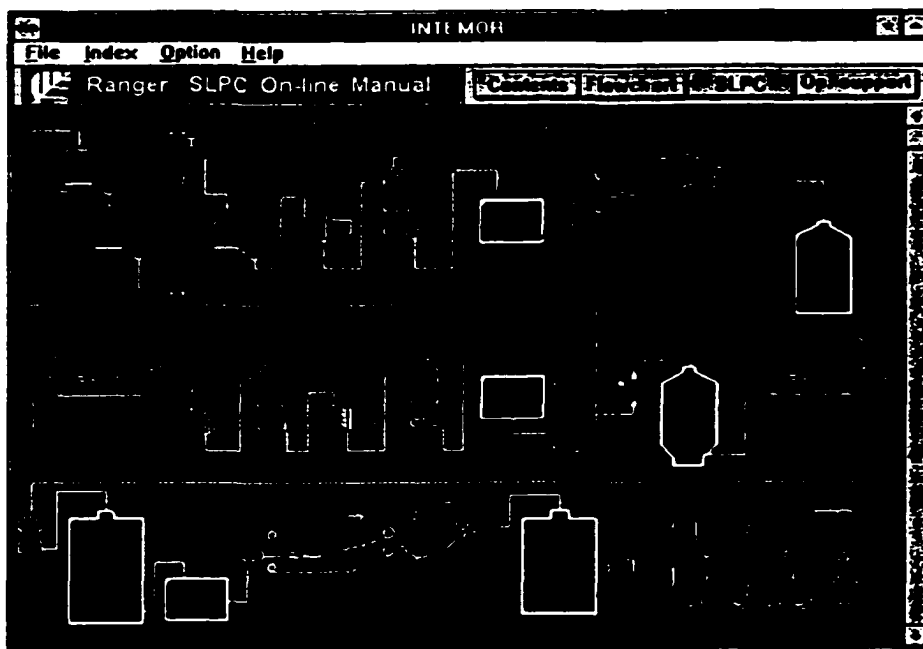


Fig. 7.6. The process flowchart of the BCTMP plant.

A topic can also be accessed by clicking a link source embedded in any other topics. For example, by clicking the link source [Fig. 9-3](#) in "Screening: Principles of screening" (See Fig. 7.5), the diagram of Screen will be shown as in Fig. 7.7.

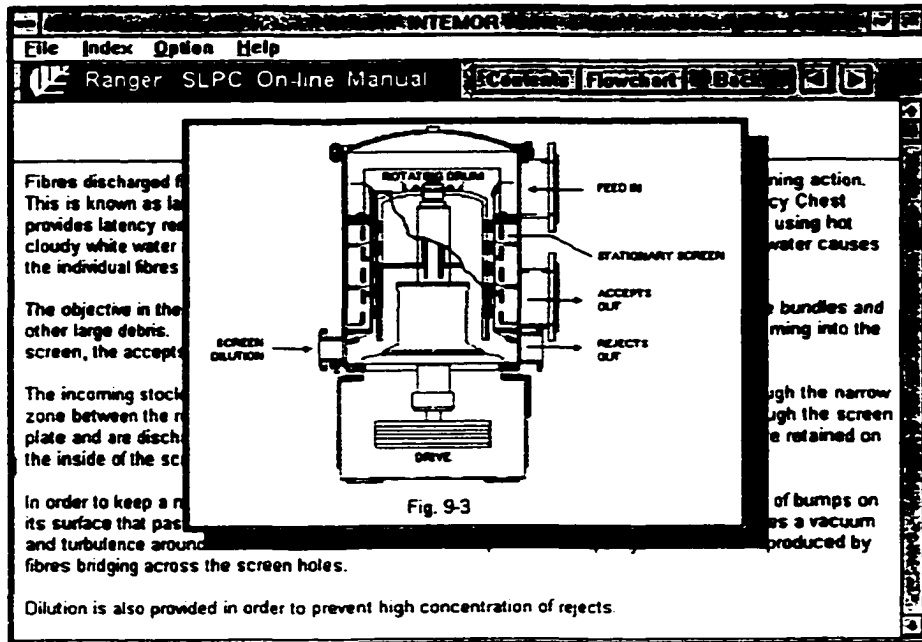


Fig. 7.7. Small window shows the referenced picture.

The linkage applied in the above searching methods can be changed at run time by the IOSS through modifying the values of the topic variables and global variables in the IHOM.

7.4.3 Computations and real time data access

Fig. 7.8 shows an example in which the Primary Screen Reject Rate is calculated automatically from five input parameters: Refiner production, 1A Screen Reject Flow, 1A Screen Reject Consistency, 1B Screen Reject Flow and 1B Screen Reject Consistency. The calculation is implemented by auto-procedures. The IHOM obtains real time data through on-line data access to DCS, MIS and IOSS and performs the numerical calculations automatically.

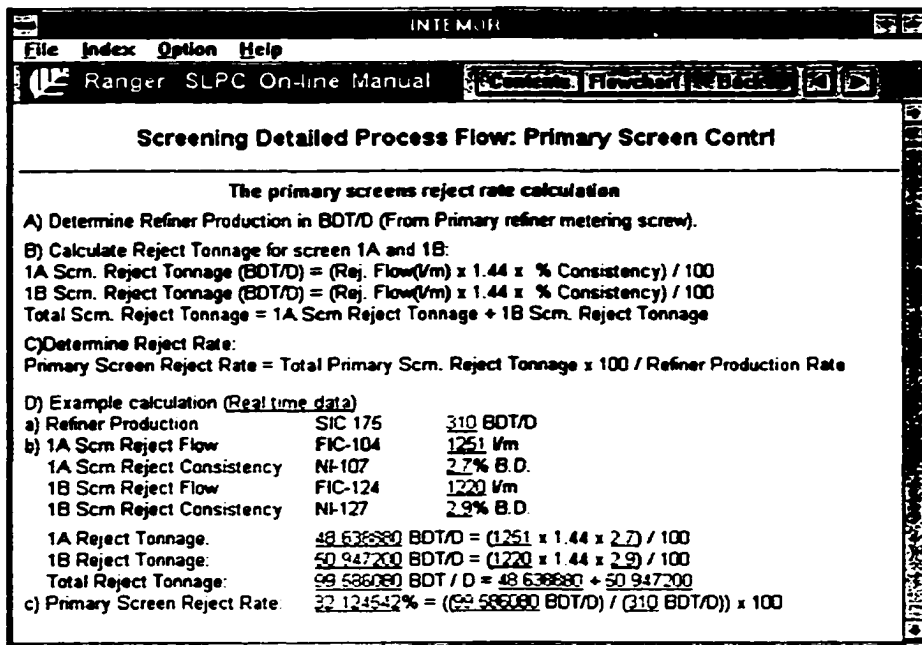


Fig. 7.8. The primary screens reject rate calculation.

7.4.4 Integration with IOSS

IHOM is required to be integrated with the intelligent operation support system (IOSS) and serves as a multimedia operator interface and external knowledge base. With IHOM's intelligent link function, IOSS is able to control the link relationship in IHOM and bring up desired multimedia information fragments to operators. With IHOM's numerical calculation and symbolic reasoning functions, IOSS is able to distribute simple information processing tasks to IHOM and thus reduce the complexity of its core system.

The integration of the IOSS with IHOM is shown in Fig. 7.9. The core system of the IOSS residing in the VMS monitors the real time data from the management information system, MOPS. If any abnormal operation problem is identified, the IOSS sends the problem solving results to IHOM through MOPS EDE/2. The data from the IOSS changes the link relationship in IHOM and automatically brings up the information fragments that are relevant to the identified problem. Fig. 7.10 shows an abnormal condition that is identified in the IOSS but displayed in the IHOM.

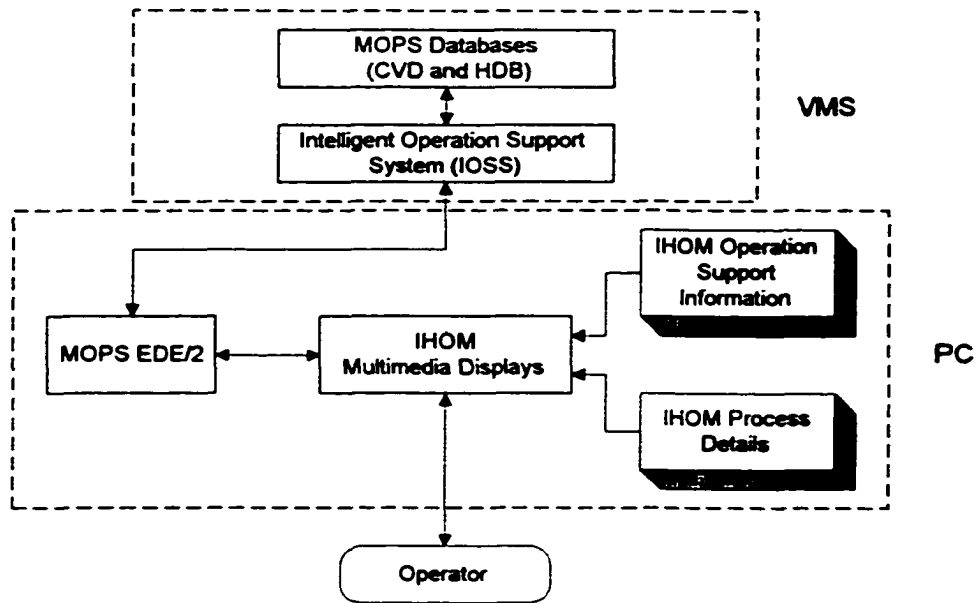


Fig. 7.9. IHOM integration with IOSS.

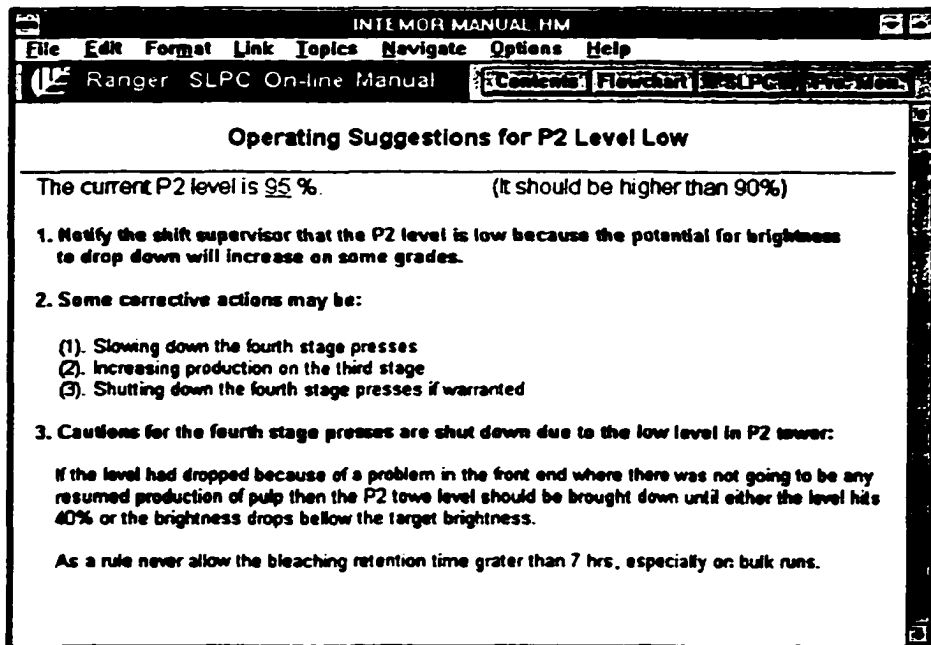


Fig. 7.10. The operation support for P2 tower level problem.

7.5 Conclusions

The intelligent hypermedia on-line manual (IHOM) presented in this chapter is not a simple replacement of traditional paper manuals. It provides functions that are required by the modern industrial production. Its intelligent link feature enables the user to find the relevant information quickly, which is especially important in the time-critical situations. The IHOM has built-in knowledge base and embedded real time data. It has the capability of numerical calculation and symbolic reasoning. IHOM can be used not only as a multimedia manual, but also as a process monitoring system.

Most of all, IHOM can be integrated with the intelligent operation support system (IOSS). It serves as an enhanced multimedia interface and external knowledge base of the IOSS. The integration of the IOSS with the IHOM significantly improves the efficiency and user acceptability of the IOSS.

Chapter 8

IOSS IMPLEMENTATION ON A BCTMP PROCESS[†]

Most modern pulp companies have successfully installed distributed computer systems (DCS) and management information system (MIS). However, process operations still rely on individual operator's experience. The operators may find it difficult to contribute a quick solution when faced with nonroutine situations. This chapter outlines the implementation results of the intelligent operation support system (IOSS) in a bleached chemical thermo-mechanical pulp (BCTMP) plant . The IOSS applied dynamic case-based reasoning methods, matrix simulator and intelligent hypermedia on-line manual. The system fulfilled such functions as abnormal situation identification, corrective action planning, process simulation, and on-line process manual. It emphasized on the integration of the IOSS with DCS and MIS. Implementation results showed that reduced off-spec product and the chemical consumption can be achieved by timely corrective actions. The methods proposed in the development of the system are very promising in process industry.

[†] The sections of this chapter have been published: Xia, Rao, Henriksson and Farzadeh, 1997, "Case-based reasoning for intelligent fault diagnosis and decision making in pulp processes", *Pulp & Paper Canada*, 98(2) , 26-30.

8.1 Introduction

Pulp and paper operation is a knowledge intensive task. The numerical computation based control technologies are not suitable to deal with such kind of problems that require considerable symbolic reasoning. Intelligent system technology is an alternative that can capture and utilize more broadly based sources of knowledge (Dvorak and Kuipers, 1991). The monitoring, analysis and control of process operations can benefit from the improved knowledge representation schemes and advanced reasoning control strategy (Stephanopoulos, 1990).

Research has been done on the knowledge-based systems for pulp and paper processes, such as intelligent operation support system for batch digester (Rao and Corbin, 1992), diagnostic expert system for solving pitch problems (Kowalske, 1991), decision support system for pulp blending strategy (Dane and Harvey, 1992), and expert system for the on-line monitoring of wastewater treatment process (Lapointe et al., 1989). The results from the research have made significant contributions to the success of artificial intelligence technology in process industry.

This chapter outlines the implementation of an intelligent operation support system (IOSS) on a bleached chemical thermo-mechanical pulp (BCTMP) plant in Slave Lake Pulp Corporation (SLPC). SLPC is utilizing the best available technology to produce high quality pulp products. A distributed control system, Fisher Provox 2000, and a mill wide information system, MOPS, have been successfully installed on the plant. The benefit from integrating MOPS with DCS has been significant (Henriksson et al., 1992). To further improve the pulp production, SLPC requires an intelligent system to be integrated with MOPS to provide operation support functions (Xia et al., 1993).

The IOSS applies the new methods proposed in the previous chapters, including multilayer system structure, dynamic case based reasoning, and intelligent hypermedia on line manual, to improve the problem solving capability and user acceptability. Special emphasis is on the integration of the intelligent system with the existing plant automation systems.

8.2 Description of BCTMP Process

The BCTMP process produces bleached pulp from aspen or mixture of aspen and softwood by using a combination of thermal energy, chemical pre-treatment, mechanical refining energy and hydrogen peroxide bleaching. The production process comprises of Chip Washing and Conditioning, Refining, Screening and Cleaning, Bleaching, Washing and Dewatering, as well as Flash Drying and Finishing, as shown in Fig. 8.1. This chapter is focused on the bleaching plant. The purpose of the bleaching operation is to achieve desired pulp brightness.

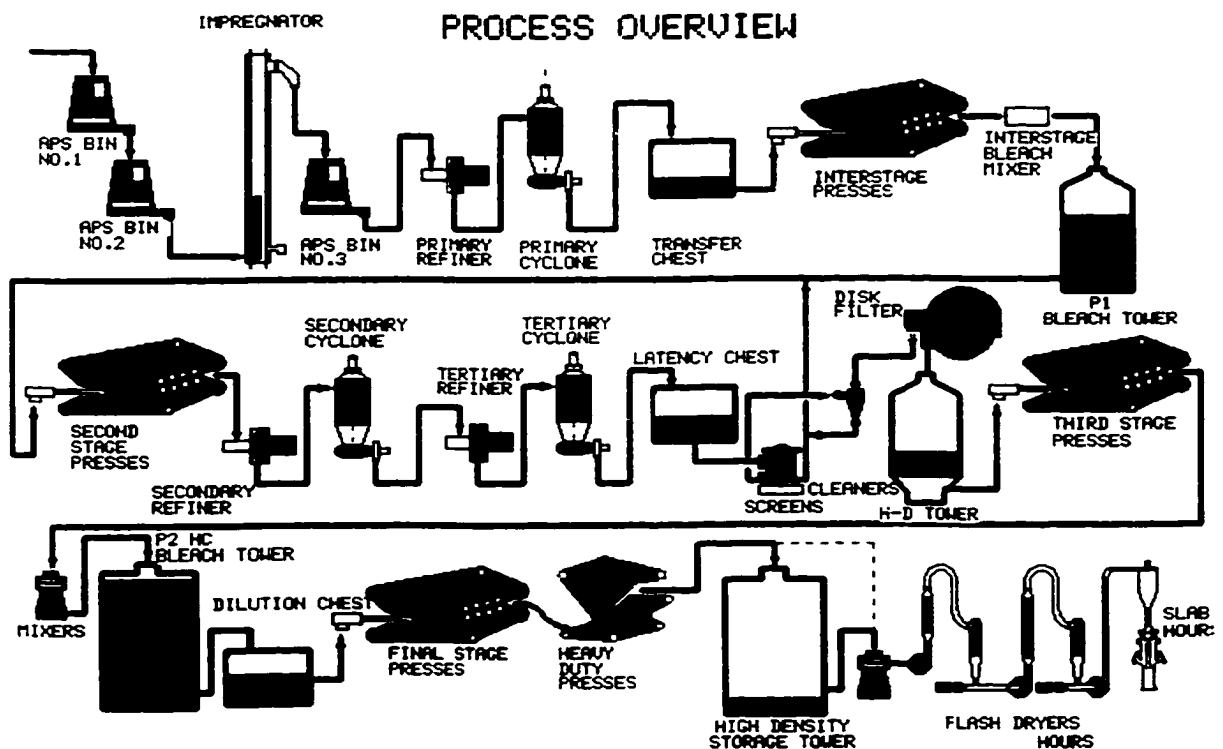


Fig. 8.1. Overview of the BCTMP process.

The bleach plant can be divided up into four components: first stage washing, interstage bleaching and washing, third stage washing and bleaching, and fourth stage of washing. The product quality is maintained by assuring that operating variables fluctuate

within permissible ranges. If operating conditions go beyond these limits, the product quality may be in jeopardy.

The color, or brightness, of pulp is dependent on its light absorbing/reflecting properties. If a material reflects all (visible) wavelengths of light, it will be seen as white; if it absorbs all (visible) wavelengths, it will be seen as black. The pulp consists of cellulose, hemi-cellulose, lignin and some “impurities”. The cellulose, hemi-cellulose and lignin are essentially non-absorbers of light. This leaves the various chemical compounds bonded to the lignin as the causes of low brightness. The objective of bleaching is to convert these compounds to a form which will absorb less light. Hydrogen peroxide has strong oxidizing capability to break chemical bonds and thereby convert compounds to non-absorbing forms. It is used to realize large gains in brightness.

Interstage Bleaching and Washing consists of a Transfer Chest, Wash Presses, pulp Conveyors and a Bleaching Tower. The Transfer Chest is a holding tank where the consistency of the pulp is reduced from approximately 35% to 5%. It receives pulp stock from the Primary Refiner. Two agitators within the transfer chest mix the hot pulp and dilution water together to make uniform slurry. This 5% pulp stock is pumped to two Andritz “Twin Wire Presses” where it is fed into a nip formed by the two traveling wires of the Twin Wire Press. The pulp is sprayed with hot water and then dewatered on its route through the press until it is discharged at a consistency of approximately 38%.

The pressate from the Interstage Wash Presses is divided into two streams, one goes to the Interstage Pressate Tank for future use, and the other goes into the top of the Transfer Chest for dilution. The purpose of the Interstage Wash Presses is to receive low consistency pulp stock and, spread this pulp out evenly over the surface area of the bottom wire so that the pulp sheet can be washed and dewatered on its pass through the press. The pulp is washed for three reasons:

1. Improve the bleachability of the pulp and increase bleaching efficiency.
2. Remove Dissolved Solids and DCM's in the pulp.

3. It is more economical to wash the pulp between Refining stages because of the higher freeness.

A 38% consistency pulp sheet is discharged from the Interstage Twin Wire Press into a Shredder Conveyor. The purpose of the Shredder Conveyor is to receive the pulp sheet and tear it up into coffee-bean size particles. It then conveys the pulp to the Discharge Conveyor. This Discharge Conveyor carries the pulp to the Interstage Pulp Conveyor.

Both Twin Wire Press Discharge Conveyors feed into the Interstage Pulp Conveyor where peroxide and other bleach chemicals are added via a Static Mixer. The Static Mixer is used to mix the bleach chemicals thoroughly with the peroxide before the mixture touches the pulp. The Interstage Pulp Conveyor carries the pulp and bleach chemicals to the Double-Shafted Bleach Mixer where steam is added to the pulp. Steam is added to the pulp and bleach chemical mixture to maintain the ideal bleaching temperature of 55 to 60 °C. The double Shafted Bleach Mixer mixes the pulp, the bleach chemical mixture and the steam together before discharging into P-1 Bleach Tower.

P-1 Bleach Tower is used to receive 15% pulp and acts as a pulp storage place for the 2 to 2.5 hours while the bleach reaction is taking place. The bleach chemical consists of Water (H_2O), Caustic ($NaOH$), Silicate (Na_2SiO_3), Peroxide (H_2O_2) and DTPA (Chelating agent). Water carries the bleach chemical to the desired location. Caustic in the bleach chemical provides the necessary alkaline environment for the peroxide to react. Silicate is a buffer used in the bleach chemical to “protect” the bleach reaction. Silicate encourages the peroxide to bleach instead of decomposing. Peroxide is the active bleaching ingredient that is added to the bleach chemical just prior to being added to the pulp. DTPA is a chelating agent included in the bleach chemical because of its ability to “tie-up” metal ions. Metal ions in the pulp or bleach chemical are undesirable because they cause peroxide to decompose.

The bleach reaction is dependent on several variables in combination:

1. pH (measure of hydrogen ion concentration): as the pH increases, more peroxide is converted to its active bleaching form, OOH^- , which drives the bleach reaction harder. However, this form is very unstable. Excessive pH will result in the loss

of peroxide and cause pulp to yellow. Careful control of the pH/peroxide combination is required to ensure good bleach response and maintain a peroxide residual at the bottom of the tower.

2. **Time/temperature:** the combination of time and temperature determines how complete the bleach reaction will be. The rate of the bleach reaction doubles for every 8-10 °C of increased temperature. Obviously, decreasing the temperature from 58 °C to 50°C would require about double the retention time to get the same degree of bleaching. High temperature also accelerates peroxide decomposition reactions. A limit of about 60 °C is practical to avoid the loss of peroxide.
3. **Pulp consistency:** the brightness gain, for a given amount of peroxide, increases as pulp consistency increases. The maximum final brightness obtainable also increases with increased consistency. Increasing the bleaching consistency means that the peroxide (in OOH^- form) is in a more concentrated, and more intimate contact with the fibers. High consistency bleaching requires very good mixing.
4. **Impurities:** metal ions such as iron, copper and manganese cause peroxide to decompose into water and oxygen. This is a direct cost to the production as the lost peroxide must be replaced. Metal ions are removed by complexing them with DTPA. The DTPA forms a very stable, water soluble complex with metal ions. This complex can then be pressed out of the pulp with the white water. The amount of these complexes carried forward to the bleach plant is dependent on the amount of white water (consistency of pulp) carried forward. This is important because under the high pH conditions of bleaching some of the complexes will break down releasing metal ions which will decompose peroxide. It is very important to press up to high consistency and remove these metal complexes prior to bleaching.
5. **Wood:** Species and chip age also influence the brightness response. Fresh cut aspen chips are preferred.

The operating conditions in the bleach plant affect also the other pulp quality variables as well, such as freeness, bulk, breaking length, tear index, burst index, opacity and yield.

For such a complex process, an operation support system will help operators to make timely decision and consistent operations.

8.3 Introduction to MOPS

MOPS, the Mill wide information and OPTimization System, is an integrated software package developed exclusively for pulp and paper industry by MoDo-Chemetics. By providing real-time operations monitoring and mill-wide information management, MOPS helps mill personnel improve quality, reduce cost and increase environmental awareness (MoDo Chemetics, 1990).

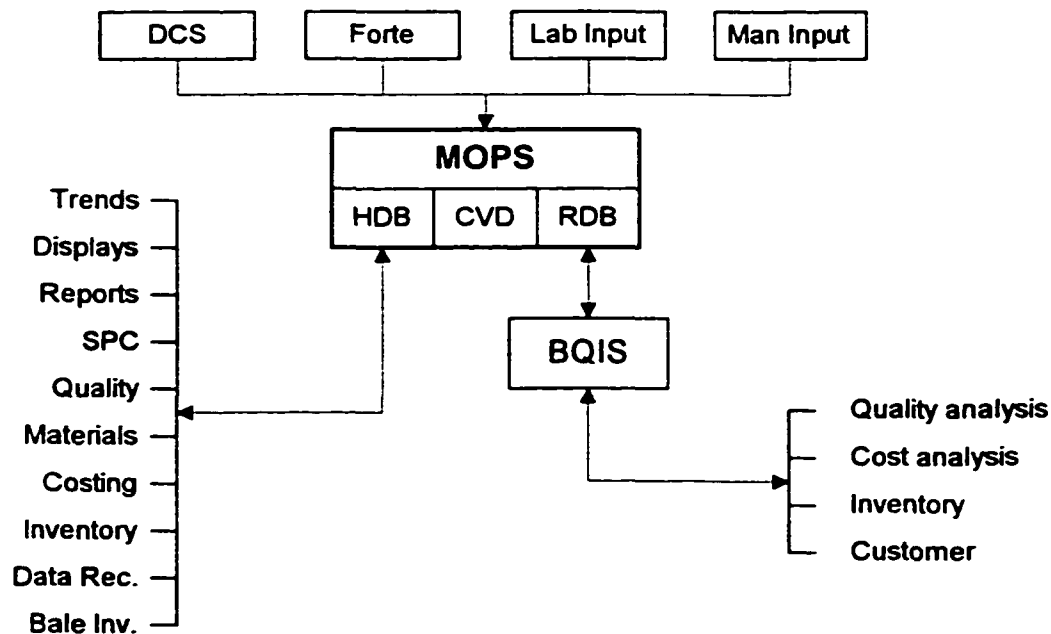


Fig. 8.2 MOPS data flow.

Fig. 8.2 shows the principle diagram of MOPS data flow. The integrated database technology is the core of MOPS. MOPS collects thousands of points about process, production, quality and business planning from various sources, such as DCS, other plant systems, lab test data, manually entered data and calculated data. The data is then saved

in several databases. Data sources may be synchronously sampled, event-driven or synchronously requested on demand.

The following are two internal databases in MOPS that are utilized in IOSS:

- **Current-Value Database (CVD):** The CVD can store all types of plant data including numbers, text, status information, mathematical expressions and records. It collects all of the current data from every MOPS communications point in the plant.
- **Historical Database (HDB):** This database maintains a historical record of plant operations. The HDB receives plant information from CVD and then compresses the data before they are stored to reduce the storage requirement. All compressed data can be instantly recreated for analysis or display. The HDB is optimized for speed, to provide ultra-high system performance and immediate operator response.

The primary functions of MOPS include display handling, trend handling, material tracking, statistical process control and cost reports. MOPS collects data from local or remote computes, process the data and adds values to the data through calculations, analysis and formatting using the above functions. It stores the value added information and makes the results available to the users on the network. These functions help operators as well as managers to check the status of the plant operations quickly, make decisions efficiently, and access the operating conditions for new production of repeated grades.

8.4 IOSS Implementation

We apply a hybrid object-oriented intelligent system building tool, Meta-COOP, which is developed in the Intelligence Engineering Laboratory at the University of Alberta, to implement the IOSS. The distinct characteristics of Meta-COOP make the implementation of the functional modules of IOSS possible.

8.4.1 Introduction to Meta-COOP

Meta-COOP is a hybrid system which permits the combination of a number of problem solving techniques. These techniques include the use of frames, rule, and a powerful programming language, C++.

Meta-COOP distributes its knowledge into a number of knowledge bases. Each knowledge base is basic object within the Meta-COOP environment called a unit. Each unit has an arbitrary number of slots, in which the attributes of the unit are described. Each slot represents one attribute of the unit and has several facets, in which the attributes are specified in more detail. There are two types of units: class units and member units. Member units describe individual objects; class units group several objects with the common attributes into a single class. Therefore, member units are "instances" of the class units. Unlike member units, class units can be defined as a superclass or subclass of another unit. The knowledge units are organized in a hierarchy with inheritance properties. With these characteristics, we can apply process hierarchical decomposition techniques to reduce a complex engineering problem to a number of less complex problems.

As a hybrid system, Meta-COOP allows the integration of various knowledge representations and inference methods, such as frame-based, rule based and method based, external procedures written in any other language, and internal subroutines written in C++.

8.4.2 Hardware layout

Fig. 8.3 illustrates the hardware layout and data flow of the IOSS in plant automation network.

The information collection systems include DCS (Fisher Provox 2000), Power Monitoring System (PAWS) and Bale Finishing System (FORTE). DCS collects the main stream process data and controls most production equipment. PAWS and FORTE are PC based systems. PAWS monitors and controls the energy usage in the main mill equipment. FORTE performs measurement and control of moisture and basis weight for each bale in the finishing line.

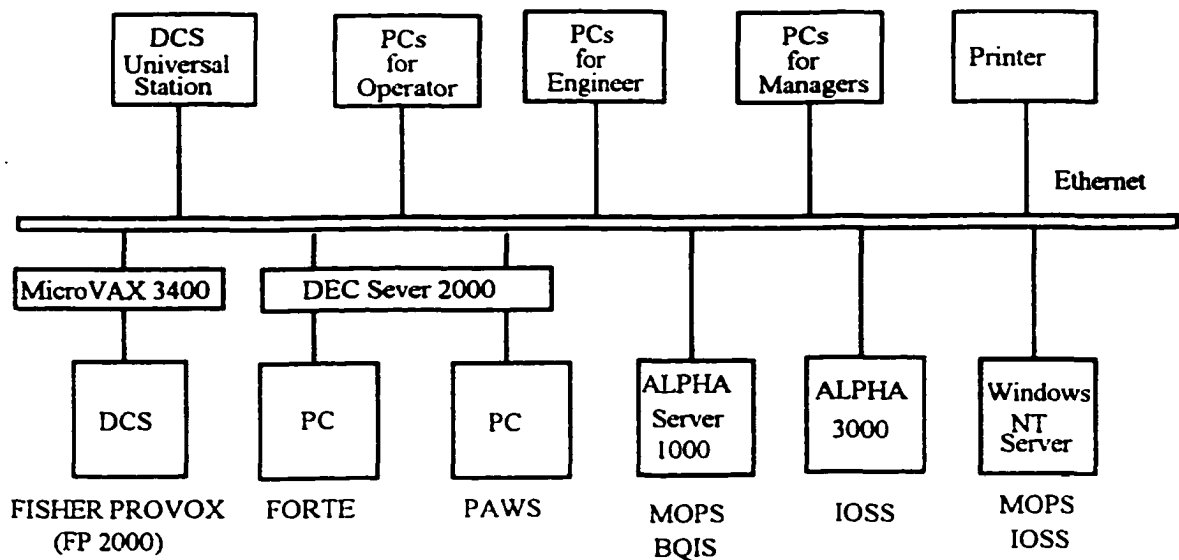


Fig. 8.3. Hardware layout and data flow.

The management information system, MOPS, manipulates and stores data from FP-2000, PAWS, FORTE, and laboratory entry in a current value database (CVD) and a historical database (HDB). The Bale Quality Information System (BQIS) stores bale quality information in Oracle RDB™. MOPS and BQIS provide all the process information that is required for IOSS.

The core of IOSS is written in C++ and resides in a Digital Alpha 3000 workstation running Open VMS 1.5. Some user programs, however, reside in PCs called IOSS clients. The intelligent on-line hypermedia system, IHOM also resides in NT. IOSS in Digital Alpha 3000 communicates with MOPS and BQIS using a DECNet. Since the end users of IOSS are PC-based, IOSS communicates with end users using two different routines. One is achieved by applying MOPS user program. The other is achieved by using IOSS server written in C++ employing TCP/IP. These two routines allow us choose to display operation support results either on user familiar MOPS operator consoles or new IOSS operator consoles. As has been indicated in Chapter 7, IHOM is used as the enhanced operator interface.

Using MOPS user program modules in Open VMS environment, the routines are developed within IOSS to access data from MOPS Current Value Database (CVD) and Historical Database (HDB). IOSS server is executed as a detached process in Alpha 3000.

8.4.3 Interface with MOPS

Since MOPS has been performing important role in the plant operations, a stand-alone intelligent system cannot be used efficiently. The successful intelligent system must be fully connected to MOPS and DCS, and make fully use of the available plant facilities. In this way, the companies' previous investment can be protected. IOSS is therefore designed as a real time intelligent system upon the existing MOPS system. It receives data from MOPS CVD and HDB. The system monitors these value-added data to evaluate the operating conditions and to give suggestions for improving the production. The evaluation results are then sent to MOPS for graphical display and to DCS for necessary automatic handling. The interactions and data flow between IOSS and MOPS are controlled by a system interface.

Since MOPS is installed in VMS but displayed in PCs, IOSS is required to communicate with both VMS and PCs.

IOSS communication within VMS: This is achieved by adding two functional routines to IOSS. The first function is called "getdata", which is responsible for getting requested data from the MOPS database. The second function is called "putdata", which is responsible for putting the requested data to the MOPS database.

IOSS communication with PCs: These are the steps taken to embed IOSS into MOPS:

- Step 1 IOSS is activated in every monitoring time interval or on the request of operators. Then *matdsp* routine gets activated.
- Step 2 MOPS gives the static part of the user displays and returns the control to IOSS by executing *usinit* function.

Step 3 IOSS gets all the required information from MOPS database with *usgepn* routine.

Step 4 IOSS does its reasoning and decision making, then uses *usput* to put the reasoning results in MOPS database and informs MOPS with *uscomp* routine. The results include the dynamic part of the matrix simulator, explanations about the current production status, operation suggestions to operators, etc.

Step 5 MOPS sends these results to PCs and displays them on PCs MOPS displays or in IHOM. Operators are able to browse through IHOM to find out more detailed information as required.

Step 6 IOSS sleeps until next monitoring time comes or a user requests using *usgtcm* command, and then go to step 1.

8.4.4 Implementation results

Fig. 8.4 shows the working principle of IOSS. The blocks with shadow are parts of the intelligent operation support system.

Most modules have been addressed in the previous chapters, except Matrix Simulator. The Matrix Simulator is a process simulation package and graphical tool built in IOSS (Farzadeh et al., 1995). In the Matrix Simulator, process variables are divided into two groups: action variables and result variables. Action variables are those that affect the product quality and can be changed to correct the process conditions. Result variables, which are affected by the action variables, are either product qualities or those used to evaluate the performance of the production processes. The Matrix Simulator provides the relationship between result variables and action variables.

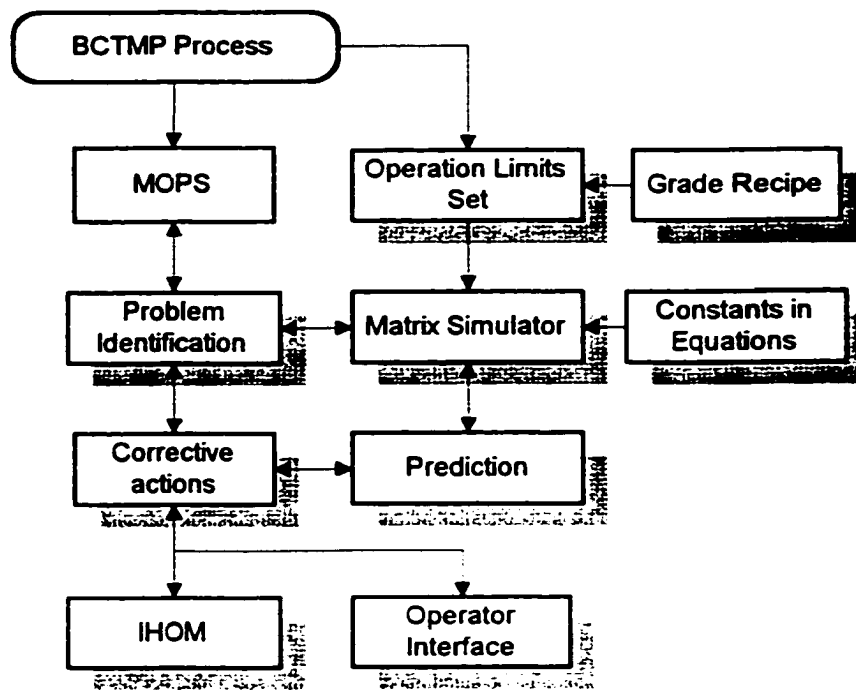


Fig. 8.4. Principle diagram of the IOSS.

The Matrix Simulator is a functional and structural model base of the production process. In order to improve the flexibility of the Matrix Simulator, the BCTMP process is divided into a number of units. The action and result variables are defined for each unit. The Matrix Simulator integrates the relationships in each unit according to material and signal flow. Normal operating conditions are defined using grades and production range. A grade defines the final quality of a particular product. The production range is used to incorporate different relationship between variables when the process operates at different speed and load. For each grade, the target, maximum and minimum limits are also defined. The Matrix Simulator receives process data from MOPS and identifies the normal operating conditions (product grades and production ranges). According to the normal operating conditions, one relationship between the action variables and result variables is selected. Fig. 8.5 gives an illustration of the Matrix Simulator.

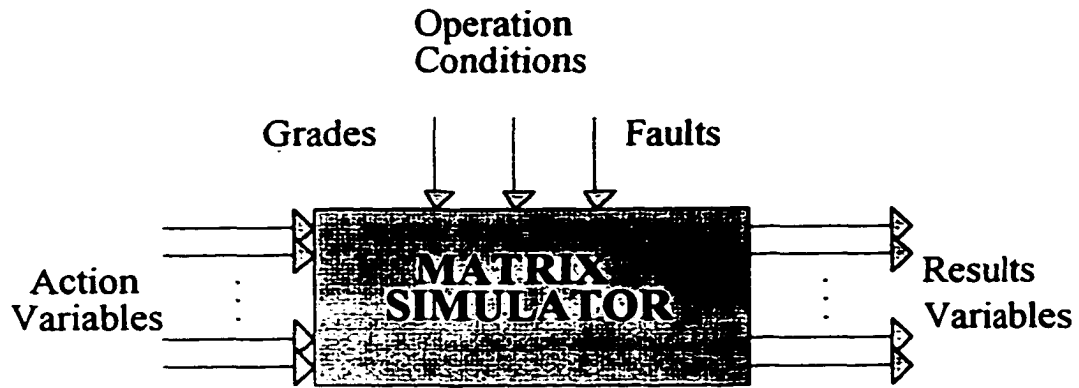


Fig. 8.5. Illustration of the Matrix Simulator.

The operating displays of the Matrix Simulator consist of many squares as shown in Fig. 8.6. The bargraphs display the relationship between action and result variables. If a bargraph is in the positive side of the zero line, an increase in the action variable will cause an increase in the result variable. If the bargraph is negative, an increase in the action variable will push the result variable towards zero line. The other boxes, besides action and result variables, display trend curves, target, and minimum and maximum limits. The other important information on the matrix simulator displays is:

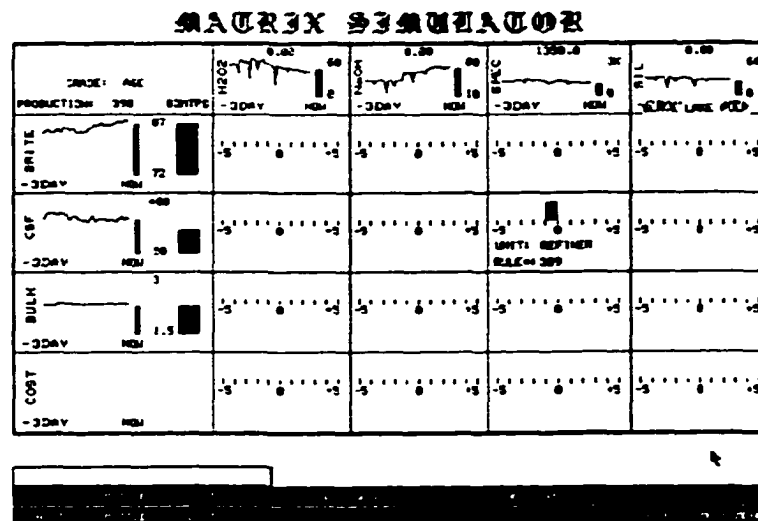


Fig. 8.6. Matrix Simulator user display.

- **Grade information shows the target grade of the final product.**
- **Grade recipe id informs operators which operating conditions are currently selected.**
- **There are three bargraphs for each result variable:**
 - **The current value bargraph displays the current situation. The target, maximum and minimum values are also shown beside this graph.**
 - **The simulated bargraph displays the effect of a change in an action variable on a result variable.**
 - **The dynamics bargraph displays the predicted value of this result variable when the next sample is taken from the process. Hence, it takes into account the dynamics and retention time of the process.**
- **For each action variable, its trend curve and two values are displayed. One value is the actual current value of the action variable. The other one is the value of the change introduced to the variable for simulation.**

The Matrix Simulator is used for two purposes: (1) IOSS utilizes the Matrix Simulator for process simulation in decision making; (2) the operators can introduce changes in action variables to simulate an operation action. By observing the corresponding changes in result variables from the graphical displays, he will be able to decide quantitatively how much change he should introduce to the current operating conditions in order to push the quality variables to the desired targets.

The dynamic case-based reasoning method (Chapter 4) is applied for abnormal problem identification and correction action planning. The cases built in the operation support system cover the following operational problems:

- | | |
|-----------------------------------|--|
| • Poor wood chip quality | • Insufficient bleach chemical supply |
| • Steam mixer blocked up | • Sensor drift or failure |
| • Pulp conveyor blocked up | • Pump failure |
| • Loss of pressates | • Incorrect control loop settings |

- Insufficient steam supply
- etc.

These problems will first affect the following operating variables before they affect pulp quality:

- Low DTPA charge
- Low/high peroxide charge in P1
- Low/high silicate charge in P1
- Low/high alkaline charge in P1
- Low level (retention time) in P1
- Low/high bleaching temperature in P1
- Low/high bleaching consistency in P1
- Low/high peroxide charge in P2
- Low/high alkali charge in P2
- Low/high silicate charge in P2
- Low level (retention time) in P2
- Low/high bleaching temperature in P2
- Low/high bleaching consistency in P2
- Low 3rd wash incoming temperature.

In the above list, P1 is the bleach tower in the interstage washing and bleaching; P2 is that in the third stage washing and bleaching. Table 8.1 illustrates some of the process knowledge collected. The case indexes apply the static and dynamic features of a number of crucial variables, which include:

- Brightnesses incoming and leaving P1
- Residual peroxide in P1
- Residual alkali (pH) in P1
- Brightnesses incoming and leaving P2
- Incoming temperature to P2
- Residual peroxide in P2
- Residual alkali (pH) in P2
- Energy consumption of the pulp conveyors and mixers
- Bleaching temperatures, consistencies, bleach tower levels, chemical flows, etc.
- Setpoint deviations of the relevant regulatory control loops.

EXAMPLE OF PROCESS KNOWLEDGE: CAUSES FOR LOW BRIGHTNESS OUT FROM P2

DISTURBANCE	Poor wood quality	DTPA charge	P1 OPERATION			P2 OPERATION										
			Low px charge	Low alk charge	Low lvl (time)	Low temp.	Low px charge	Low alk charge	Very high alk charge	Low Sil charge	No sil charge	Low lvl (time)	Low temp.	Low bleach conscy	Too high bleach conscy	Incoming temp
		-50-75%	15- 25%	15- 25%	40-50%	-5-10 C	5- 10 %	5- 10 %	10-20%	-10-20	-100	-15-25%	-5-10C			3rd wash
SYMPTOM P1																
Incoming brightness	0; -3	0	0	0	0	0										
Residual peroxide	0	-1; -2	-1; -3	+1; +3	-1; -3	+1; +3										
Residual alkali, pH	0	0	0	-1; -3	+1; +3	+1; +2										
Leaving brightness	-1; -3	-1; -2	-1; -3	-1; -3	-1; -3	-1; -2	0	0	0	0	(-1; -2)	0	0	0	0	0
SYMPTOM P2																
Incoming brightness	-1; -3	-1; -2	-1; -3	-1; -3	-1; -3	-1; -2	0	0	0	0	0	0	0	0	0	0
Incoming temperature							0	0	0	0	0	0	0	0	0	+3; +5
Residual peroxide	0	-5; -6	-1; -2	+1; +3	-1; -3	-1; -2	-1; -5	1; +8	-5; -8	-1; +2	-5; -8	+1; +3	+1; +5	+1; +3	-5; -8	-5; -8
Residual alkali, pH	0	+2; +6	0	-1; -3	-1; -3	-1; -2	1; +5	1; -8	+2; +8	-1; -2	+2; +6	0; +3	0; +5	0; +3	+2; +8	+2; +8
Temperature increase	0	+3; +8	0	(-1)	0	0	0; -2	0; -5	+3; +10	0; -2	+3; +7	-1; -3	-1; -5	-1; -3	+3; +10	+3; +10
Leaving brightness	-2; -5	-4; -8	0; -1	-1; -3	-1; -3	-1; -2	-2; -8	-2; -8	-4; -10	0; -2	-4; -8	-2; -6	-2; -4	-2; -5	-4; -10	-4; -10
		If decomp							High decomp	No decomp	High decomp				High decomp	High decomp
Difference from target			Other causes:													
	-3 = halfway to low limit		Process disturbances													
	-5 = on or just below low limit		Shut down and start third presses													
	-10 = far below low limit		Broke inmix. Broke of low quality													
			Dryer brightness loss													
			Washing / water balance													

Table 8.1. Causes of P2 low brightness.

Summary Pages are designed to show the summary information about the process operations. Fig. 8.7 is a Summary Page for the bleach plant. The Summary Pages show the low, high, target and actual values of all the important operating variables. If a process variable is out of normal limits, the background of the box for its actual value will be in red. With the Summary Pages, operators can have a quick bird view of the current operation conditions, which will aid him in decision making.

PROCESS STATUS BLEACHING
Grade: A6E

		Low	Target	Actual	High				
Impregnation:									
Caustic addition, NAOH	Kg/t	0.00	0.00	0.00	1.00				
BTPA addition, (incl transf)	Kg/t	0.30	0.60	0.60	0.65				
Liquor uptake	m ³ /t	0.65	0.74	0.66	1.15				
BTPA to screen rove	Kg/t	0.35	0.40	0.40	0.50				
Brightness, 1A/B (CC21)	%	48.00	49.20	49.77	55.00				
Bleach plant chemical charges, etc									
		P1				P2			
Total peroxide charge	Kg/t	7.00	7.00	6.00	12.00	46.0	48.0	113.5	55.0
Caustic charge	Kg/t	19.00	20.00	10.00	21.00	33.0	34.0	16.0	35.0
Silicate charge	Kg/t	0.00	0.00	0.00	1.00	24.0	25.0	15.0	26.0
Level	%	80.0	85.0	79.6	100.0	95.0	0.0	99.0	100.0
Temperature, mixer, exh	C	58.0	60.0	64.5	65.0	48.0	49.0	62.0	52.0
top	C					60.0	0.0	66.4	70.0
bottom	C	57.0	0.0	60.9	66.0	73.0	0.0	70.5	77.0
Retention time, NOPS	min	65.0	0.0	60.0	120.0	150.0	0.0	190.4	250.0
Incoming brightness	%	40.0	0.0	49.0	55.0	65.0	0.0	63.0	69.0
Outgoing brightness	%	64.0	0.0	54.3	60.0	64.5	0.0	63.7	65.5
Total process caustic charge									
Sulphuric acid	Kg/t	53.0	54.0	50.0	55.0				
pH, dilution chest						0.50	0.00	0.50	0.00
pH, FST						6.90	0.00	0.00	7.35
Residual peroxide, FST	g/L					6.00	0.00	7.20	7.45
Residual peroxide	Kg/t	0.25	0.00		0.45	1.50	0.00	0.67	3.00
Residual alkalinity	g/L					5.50	0.00		12.00
						275.0	0.0	0.0	305.0

ADVNLC PDET_INFO BLOC WREN MATERIAL PAGE DWI HYP TEST MATRIX PRINT PREV SCRIN
 GRAPH LIB TTRND LIB SUPPORT RECORD PT HELP PT INFO EXPEDIT BATHMENU

Fig. 8.7. Displays for the Summary Pages.

The system compares the new problem features with the indexes of the relevant cases in case memory. The cases whose features are matched will be accessed and the similarity (or confidence) is calculated using the mechanism in each case. The priority of the retrieved cases are ordered according to the confidence, critical rate, last occurrence time and the frequency. The operator interface provides the detail information about all the retrieved cases. Fig. 8.8 presents the diagnostic results and suggested solution to an operation problem.

EXPERT DECISION MAKING			
INCIDENT NUMBER:	CASE012	OCCURENCE TIME:	11:33:45 05/25/88
CONFIDENCE:	0.90	CRITICAL RATE:	IMMEDIATE ACTIONS
ROOT CAUSE:	THE CONTROL LOOP FOR THE TEMPERATURE IN P1 IS OUT OF ORDER		
SYMPTOMS:	THE BRIGHTNESS IN 2A/B IS LOWER THAN NORMAL THE TEMPERATURE AFTER IMPCO STEAM MIXER IS LOWER THAN NORMAL		
VERIFICATION:	CHECK THE MODE OF THE CONTROL LOOP 631TC367 CHECK THE SET POINT OF THE CONTROL LOOP 631TC367 CHECK THE CONTROL VALVE		
CONSEQUENCES:	THE BRIGHTNESS OF THE FINAL PULP WILL BE LOWER THAN NORMAL WILL RESULT IN OFF SPECIFICATION PULP PRODUCT MAY CAUSE EXCESSIVE CONSUMPTION OF BLEACH CHEMICALS		
IMMEDIATE OPERATOR ACTIONS:	CONTROL ROOM: RESET THE TEMPERATURE CONTROL LOOP 631TC367 IF AUTOMATIC CONTROL IS NO MORE IN FUNCTION, USE MANUAL CONTROL		
	FIELD: CHECK THE CONTROL VALVE AND THE STEAM SUPPLY HELP CONTROL ROOM OPERATOR FOR ANY REQUIRED SITE ACTIONS		
<small>ACD44LC ASST_INFO BLOW/WRM MATERIAL PAGE - PAGE + MYP_TEXT MATRIX PRINT PREV SCRN GRAPH LIB TRIM LIB SUPPORT RECORD PT HELP PT INFO EXPERT PRINT/MENU</small>			

Fig. 8.8. Operator interface for problem solution.

The implementation results showed that IOSS was very promising in process operations. For the operational problems covered by the case memory, the operation support system was able to provide an early identification with satisfactory accuracy before the pulp quality went off-specification. By taking the corrective actions, operators were able to reduce the off-spec product, reduce the chemical consumption, and minimize production loss. Knowledge updating was convenient in the operation support system. A new case can be easily created and added to the case memory. With the operation support system, better production can be achieved through more consistent and timely operation.

8.5 Conclusions

The operation support system technology is demanded by process industry. This chapter presented the implementation results of an intelligent operation support system (IOSS) on a BCTMP process. The IOSS is designed based on the analysis of production requirement and human operator's recognition behavior. It is integrated with existing plant automation systems and adds "intelligence" to the existing systems, which makes the IOSS more efficient than stand alone systems. The easy-to-use operator interface and the intelligent hypermedia on-line manual improve the acceptability of the intelligent system. Operators are actively involved in the decision making instead of being replaced by the machine.

Chapter 9

CONCLUSIONS AND RECOMMENDATIONS

9.1 Conclusions

The main contribution of this thesis is a group of new methods for the intelligent operation support system theory and design. These methods provide efficient problem solving, easy knowledge acquisition and good operator acceptability.

The thesis investigates the operation support problems from the operator's recognition behavior and process operation characterization. It emphasizes on the integration of various knowledge representations and reasoning methods to provide improved reasoning efficiency and problem solving flexibility. The results obtained in this research represent one step advance to the success of intelligent operation support system applications to process industry. The main advantages of the proposed methods can be summarized as follows:

1. **INTEGRATED ENVIRONMENT.** The proposed methods provide an effective environment to integrate various methods and technologies. The dynamic case-based reasoning method (DCBR) is integrated with other reasoning methods, including model-based reasoning (MBR) and rule-based reasoning (RBR), to achieve maximum problem solving efficiency. The integration with principal component analysis and Kalman-filter based approaches are also developed.

Multimedia system technology is integrated to the intelligent operation support system to enable multimedia information handling and representation. The system is also integrated with the existing plant automation systems, such as distributed control systems (DCS) and management information systems (MIS), to take full advantage of the existing plant facilities.

2. **ERGONOMICAL AND SYSTEMATIC DESIGN.** The analyses of human recognition behavior in problem solving reveal many psychological issues that are significant in process operations. The new problem solving model, which incorporates not only the human recognition behaviors but also the system requirements, is applied as the basis of operation support system design. The multilayer modularity architecture achieves the decomposition and coordination of functionality, process and methodology, as well as the separation of general and specific knowledge. The results provide a methodology for ergonomical and systematic design of high performance operation support systems.
3. **EFFICIENT PROBLEM SOLVING.** Process operation is a time critical situation. Reasoning efficiency is crucial for an operation support system. The DCBR applies time-tagged indexes, dynamic features and composite features to extend the traditional CBR to the dynamic problems and lead to more accurate and timely problem solving. It uses the case based reasoning as the principal reasoning paradigm and other approaches as the supplemental reasoning paradigms to compensate the limitations of each individual approach. Improved efficiency is also achieved by applying task decomposition and process decomposition to break up a complex operation support problem into a number of simpler problems.
4. **FULL USE OF OPERATION KNOWLEDGE.** Knowledge acquisition has long been identified as the bottleneck in developing intelligent systems. Knowledge in process industry exists in various levels and types. With the hybrid reasoning structure, rich knowledge representations can be used.
5. **PROBLEM SOLVING FLEXIBILITY.** The multiple indexing paths in DCBR significantly improve problem solving coverage and adaptation capability of CBR.

A new problem can be solved by the combination of a few cases that are structurally or functionally similar to it. The problem solving flexibility is also represented by the fact that various reasoning methods can be selected to suite the engineering requirements.

6. **ECONOMIC SOLUTIONS.** The method of actuator and sensor design is based on the new fault distances and objective trees that represent instrumentation requirement of operation support systems. It generates the scheme of actuator and sensor locations that ensure satisfactory system performance and the least instrumentation investment.
7. **MULTIMEDIA INFORMATION.** Multimedia information has been extensively used in process operations. The intelligent hypermedia on-line manual (IHOM) provides the operation support system with the capability of handling multimedia information.
8. **GOOD OPERATOR ACCEPTABILITY.** The proposed intelligent operation support system (IOSS) has an underlying reasoning mechanism that is consistent with operator's problem solving. The IHOM serves as a multimedia operator interface to provide operator with on-line detail information in its original multimedia form. It greatly improves the operator acceptance of the system.
9. **PRACTICAL.** The new methods developed in this thesis are feasible for industrial applications. A prototype intelligent operation support system has been developed for a bleached chemical thermo-mechanical pulp plant. The system have been implemented in the plant computer system and tested with real-time plant operations. The results were very encouraging. With the help of the operation support system, operators were able to identify the undesirable conditions earlier and reduce off-spec product and production cost.

9.2 Recommendations

Intelligent operation support systems have been becoming increasingly important in process industry. The distributed control systems (DCS) and management information systems (MIS) can no longer satisfy the requirements of the modern industrial production. The industry requires a new technology that helps operators with decision making in abnormal or nonroutine operations.

The methods developed in this thesis have many features that are desirable to industrial applications. Future extension with the developed methods can be focused on the following capacities:

1. **ON-LINE LEARNING.** Because of the complexity of the production processes, it is impossible to develop a knowledge base that covers all the abnormal conditions. On-line updating of knowledge base is required. However, on-line learning is a very difficult problem. The DCBR is integrated with MBR for case adaptation and with other types of methods for problem solving. By properly configuring the system and developing a component for automatic knowledge acquisitions, the new case can be created and the knowledge base can be eventually expanded in the course of the operation. If a new process condition is not covered by either the case memory or the model-based knowledge base, a new case can be created and added to the case memory by generating the features from process data with the interaction with human experts.
2. **INTEGRATION WITH OTHER PROVEN METHODS.** The emphasis of this thesis is on the integration of various methods to achieve high problem solving efficiency. The integration of DCBR with a few reasoning methods, including MBR, RBR, PCA and Kalman-filter based methods has been included in this thesis. Other proven methods, such as qualitative simulations, should be also integrated to further improve system performance. Once again, the DCBR provides an effective integration environment with all types of problem solving methods.

BIBLIOGRAPHY

- Alterman, R., 1988, "Adaptive planning", *Cognitive Science*, **12**, 393-422.
- Althoff, K and S. Wess, 1991, "Case-based knowledge acquisition, learning and problem solving for diagnostic real world tasks", *Proc of EKAW*, March 1991.
- Anderson, B.D.O. and J.B. Moore, 1971, *Linear Optimal Control*. Englewood Cliffs, NJ: Prentice-Hall.
- Ashley, K.D. and E.L. Rissland, 1987, "Compare and contrast, a test of expertise", *Proc 6th AAAI*, Seattle, pp. 273-278.
- Bailey, C.W., 1990, "Intelligent multimedia computer systems: Emerging information resources in the network environment", *Library Hi Tech*, **8**, 29-39.
- Bareiss, E., B. Porter and C. Wier, 1988, "PROTOS: An exemplar-based learning apprentice", *International Journal of Man-Machine Studies*, **29**, 549-561.
- Bareiss, E.R., 1989, *Exemplar-Based Knowledge Acquisition: A Unified Approach to Concept Representation, Classification, and Learning*, Boston: Academic.
- Becraft, W.R., D.Z. Guo, P.L. Lee and R.B. Newell, 1991, "Fault diagnosis strategies for chemical plants: A review of competing technologies", *Proc 4th Int. Symp. on Process Systems Engineering (PSE'91)*, Vol. 2, pp.12.1-12.15, Montebello, Canada.
- Bonissone, P.P. and S. Dutta, 1990, "Integrating case-based and rule-based reasoning: the possibilistic connection", *Proc 6th Conf. on Uncertainty in Artificial Intelligence*, Cambridge, MA.
- Borisson, U., 1979, "Self-tuning regulators for a class of multivariable systems", *Automatica*, **15**, 209-215.
- Boy, G.A., 1993, "Integrated human-machine intelligence", *Proc First European Symp. on Computer Aided Process Engineering*, S395-S404.
- Bradshaw, G., 1987, "Learning about speech sounds: the NEXUS project", *Proc Fourth Int. Workshop on Machine Learning*, pp. 1-11.

- Branting, L., 1989, "Integrating generalizations with exemplar-based reasoning", *Proc DARPA Workshop on Case-Based Reasoning*, San Matero, Calif., pp. 80-93.
- Calandranis, J., G. Stephanopoulos and S. Nunokawa, 1990, "DiAD-Kit/boiler: On-line performance monitoring and diagnosis", *Chem. Eng. Prog.*, **86**, 60-80.
- Chen, L.W. and M. Modarres, 1992, "Hierarchical decision process for fault administration", *Computers chem. Engng*, **16**, 425-448.
- Cheung, J.T. and G. Stephanopoulos, 1990, "Representation of process trends part I. A formal representation framework", *Comput. chem. Engng*, **14**, 495-510.
- Clancy, J.E., G.J. Carrette and K.M. Gersten, 1992, "Real-time advisory control applications in the petrochemical industry", *ISA Transactions*, **31**, 59-63.
- Clemons, J.W., 1992, "CIM strategies and the shop coordination system", *ISA Transactions*, **31**, 29-37.
- Conkin, J., 1987, "Hypertext: An introduction and survey", *Computer*, Sept. issue, 17-41.
- Console, L. and P. Torasso, 1991, "A spectrum of logical definition of model-based diagnosis", *Computational Intelligence*, **7**, 133-141.
- Coyne, J., 1990, "The importance of expert systems in a hypermedia environment", *Proc IEEE Conf. on Managing Expert System Program and Projects*, pp. 205-208, Bethesda, MD.
- Cuena, J., 1993, "Knowledge architecture for real time decision support", in *Second Generation Expert Systems*, J. David, J. Krivine, and R. Simmons (Editors), pp. 689-720, Springer Verlag.
- Davis, R., 1984, "Diagnostic reasoning based on structure and behavior," *Artificial Intelligence*, **24**, 347-410.
- De Kleer, J. and S. Brown, 1984, "A qualitative physics based on confluences", *Artificial Intelligence*, **24**, 7-83.
- Deckert, J.C., M.N. Desai, J.J. Deyst and A.S. Willsky, 1977, "F-8 DFBW sensor failure identification using analytic redundancy", *IEEE Trans. Autom. Control*, **AC-22**, 795-803.

- de Silva, C.W., 1989, *Control Sensors and Actuators*, Prentice Hall, Englewood Cliffs, NJ.
- de Silva, C.W., 1995, *Intelligent Control – Fuzzy Logic Applications*, CRC Press, Boca Raton, FL.
- de Silva, C.W. and N. Wickramarachchi, 1998, “Knowledge-based supervisory control system of a fish processing workcell; Part I: system development”, *Engineering Applications of Artificial Intelligence*, **11**, 97-118.
- DeLorenzo, M.L., 1990, “Sensor and actuator selection for large space structure control”, *AIAA Journal of Guidance, Control, and Dynamics*, **13**, 249-257.
- Desouze, E. and S.P. Bhattacharyya, 1981, “Controllability, observability and the solution of $AX-XB=C$ ”, *Linear Algebra Appl.*, **3**, 167-188.
- D'Hulster, F., R.M.C. De Keyser and A.R. Van Cauwenberghe, 1983, “Simulation of adaptive controllers for a paper machine headbox”, *Automatica*, **19**, 407-414.
- Donahue, D., 1989, “OGRE: Generic reasoning from experience”, *Proc a Workshop on Case-Based Reasoning*, pp. 248-252.
- Dong, D. and T.J. McAvoy, 1996, “Nonlinear principal component analysis – based on principal curves and neural networks”, *Computers chem. Engng*, **20**, 65-78.
- Dubois, D. and H. Prade, 1980, *Fuzzy Sets and Systems: Theory and Applications*, Academic Press, New York.
- Duda, R.O. and P.E. Hart, 1973, *Pattern Classification and Scene Analysis*, Wiley-Interscience, New York.
- Dunia, R.H, S.J. Qin and F. Edgar, 1995, “Multivariable process monitoring using nonlinear approaches”, *Proc American Control Conference*, Seattle, Washington, pp. 756-760.
- Durfee, E.H., V.R. Lesser and D.D. Corkill, 1989, “Cooperative distributed problem solving”, *The Handbook of Artificial Intelligence*, Vol. IV, A.B. Barr, P. Cohen, and E. Feigenbaum, Eds. Reading, P.A.: Addison Wesley, pp. 83-147.
- Dvorak, D. and B. Kuipers, 1991, “Process monitoring and diagnosis”, *IEEE Expert*, June, 67-74.

- Dyne, B. and M. Harvey, 1992, "Decision support system for pulp blending strategy", *Pulp and Paper Expert Systems Workshop*, Pointe Claire, Quebec, Nov.
- Erlenbach, S., Y. Ying, M. Rao and K. Danielson, 1994, "An expert system for on-line incident detection and reporting", *Proc 1994 CPPA Western Branch Fall Conference*, Oct., 25-27, Grande Prairie, Canada.
- Eterno, J.S., D.P. Looze, J.L. Weiss and A.S. Willsky, 1985, "Design issues for fault-tolerant restructurable aircraft control", *Proc 24th IEEE Conf. on Decision and Control*, Fort Lauderdale, pp. 900-905.
- Farell, A.E. and S.D. Roat, 1994, "Framework for enhancing fault diagnosis capabilities of artificial neural networks", *Computers chem. Engng*, **18**, 613-635.
- Farzadeh, H., Q. Xia, Y. Ying, M. Rao, C. Henriksson, K. Danielson and J. Olofsson, 1995, "Knowledge integration system for Slave Lake Pulp Corporation and DMI Peace River Division", *Pulp & Paper Canada*, **96**, 25-28.
- Fjeld, M., 1978, "Application of modern control concepts on a kraft machine", *Automatica*, **14**, 107-117.
- Frank, P.M., 1990, "Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy - a survey and some new results", *Automatica*, **26**, 459-674.
- Fu, K.S., 1982, *Syntatic Pattern Recognition and Applications*, Prentice-Hall, Englewood Cliffs, N.J.
- Gallanti, M., G. Guida, L. Spampinato and A. Stefanini, 1985, "Representing procedural knowledge in expert systems: an application to process control", *Proc. 9th Int. Joint Conf. for Artificial Intelligence*, pp. 342-352.
- Geladi, P. and B.R. Kowaksi, 1986, "Partial least squares regression: a tutorial", *Anal Chim Acta*, **185**, 1-17.
- Geromel, J.C., 1989, "Convex analysis and global optimization of joint actuator location and control problems," *IEEE Trans. Automatic Control*, **AC-34**, 711-720.
- Gertler, J.J., 1988, "Survey of model-based failure detection and isolation in complex plants", *IEEE Control Systems Magazine*, **8**, 3-11.
- Gessner, R., 1990, "Building the hypertext system", *Dr. Dobb's Journal*, June.

- Ghosh, D. and C.H. Knapp, 1989, "Measurement selection for linear multivariable control systems," *Automatica*, **25**, 55-63.
- Gilbert, E.G., 1969, "The decoupling of multivariable system by state feedback", *SIAM J. Control*, **7**, 50-63.
- Goel, A. and B. Chandrasekaran, 1989, "Use of device models in adaptation of design cases", *Proc DARPA Workshop on Case-Based Reasoning*, Vol. 2, pp. 100-109.
- Goel, A., 1989, *Integration of Case-Based Reasoning and Model-Based Reasoning for Adaptive Design Problem Solving*, Technical report (PhD Diss.), Dept. of Comp. and Inf. Science, Ohio Univ.
- Goel, A., 1992, "Integrating case-based and model-based reasoning: a computational model of design problem solving", *AI Magazine*, 50-53, Summer.
- Goel, A., and Chandrasekaran, B., 1989, "Use of device models in adaptation of design cases", *Proc DARPA Workshop on Case-Based Reasoning*, Vol. 2, pp. 100-109.
- Golding, A. and P.S. Rosenbloom, 1991, "Improving rule-based systems through case-based reasoning", *Proc. AAAI Conf.*, pp. 22-27.
- Gonzalez, R.C. and M.G. Thomason, 1978, *Syntactic Pattern Recognition*, Addison-Wesley, Reading, MA.
- Grantham, S.D. and L.H. Ungar, 1990, "A first principles approach to automated troubleshooting of chemical plants", *Computers chem. Engng*, **14**, 783-798.
- Gunn, D.J. and A.K. Sinha, 1983, "Nonlinear control of a paper-stock flowbox", *Proc. IEE*, **130D**, 131-136.
- Halasz, F.G., 1988, "Reflections on notecards: seven issues for the next generation of hypermedia systems", *Communication of ACM*, **31**, 836-852.
- Hammond, K. and K. Hurwitz, 1988, "Extracting diagnostic features from explanations", *Proc DARPA Workshop on Case-Based Reasoning*, pp. 169-178.
- Hammond, K., 1989, *Case-Based Planning: Viewing Planning as a Memory Task*, New York: Academic.
- Hastie, T.J. and W. Stuetzle, 1989, "Principal curves", *J. Am. Stat. Assoc.*, **84**, 502-516.

- Henriksson, C., W. Smith, K. Danielson, and J. Olofsson, 1992, "Value-added information--A high payback investment for the mill," *Proc. Tappi Process Control Conference*, pp. 5-19, Atlanta, Georgia, USA.
- Hoskins, J.C. and D.M. Himmelblau, 1988, "Artificial neural network models of knowledge representation in chemical engineering", *Comput. chem. Engng*, **12**, 881-890.
- Hoskins, J.C., K.M. Kaliyar and D.M. Himmelblau, 1991, "Fault diagnosis in complex chemical plants using artificial neural networks", *AIChE J.*, **37**, 137-141.
- Höskuldsson, A, 1988, "PLS regression methods", *J. Chemometrics*, **2**, 211 - 218.
- Huang, H.M., 1996, "An architecture and a methodology for intelligent control", *IEEE Expert*, April, 46-55.
- Iri, M., K. Aoki, E. O'Shima and H. Matsuyama, 1979, "An algorithm for diagnosis of system failure in the chemical process", *Comput. chem. Engng*, **3**, 439-493.
- Isermann, R., 1984, "Process fault detection based upon modelling and estimation methods - a survey", *Automatica*, **20**, 387-404.
- Ishida, Y., H. Tokumaru and N. Adachi, 1987, "Diagnosability and distinguishability analysis and its applications," *IEEE Trans. Reliability*, **R-36**, 531-538.
- Ishida, Y., N. Adachi and H. Tokumaru, 1985, "A topological approach to failure diagnosis of large-scale systems", *IEEE Trans. SMC*, **SMC-15**, 327-333.
- Ishida, Y., N. Adachi and H. Tokumaru, 1986, "A diagnosability analysis by means of fault distance," *Int. J. Systems Science*, **17**, 1105-1120.
- Jang, Y., 1993, *HYDI: A Hybrid System with Feedback for Diagnosing Multiple Disorders*, Technical report, MIT/LCS/TR-576.
- Janusz, M. and V. Venkatasubramanian, 1991, "Automatic generation of qualitative description of process trends for fault detection and diagnosis", *Engng Applic. Artif. Intell.*, **4**, 329-339.
- Jeffreys, S., 1987, "Software simplifies batch control design", *Control Engng*, Sept, 107-109.

- Jones, P.M., R.W. Chu and C.M. Mitchell, 1995, "A methodology for human-machine systems research: Knowledge engineering, modeling, and simulation", *IEEE Trans. SMC*, **SMC-25**, 1025-1038.
- Kaspar, M.H. and W.H. Ray, 1992, "Chemometric methods for process monitoring and high performance controller design", *AIChE Journal*, **38**, 1593-1608.
- Kavuri, S.N. and V. Venkatasubramanian, 1993, "Using fuzzy clustering with ellipsoidal units in neural networks for robust classification", *Computers chem. Engng*, **17**, 765-784.
- Kavuri, S.N. and V. Venkatasubramanian, 1994, "Neural network decomposition strategies for large-scale fault diagnosis", *Int. J. Control*, **59**, 767-792.
- Keel, L.H., S.P. Bhattacharyya and J.W. Howze, 1988, "Robust control with structured perturbations", *IEEE Trans. Automat. Contr.*, **AC-29**, 68-77.
- Kokawa, M., S. Miyazaki and S. Shingai, 1983, "Fault location using digraph and inverse direction search with application", *Automatica*, **19**, 729.
- Kolodner, J. and R. Kolodner, 1987, "Using experience in clinical problem solving: Introduction and framework," *IEEE Trans. SMC*, **17**, 420-431.
- Kolodner, J., R. Simpson and K. Sycara-Cyranski, 1985, "A process model of case-based reasoning in problem solving", *Proc Ninth International Joint Conference on Artificial Intelligence*, pp. 284-290, Menlo Park, California.
- Kolodner, J.L., 1983, "Reconstructive memory: A computer model", *Cognitive Science*, **7**, 243-280.
- Kolodner, J.L., 1987, "Extending problem solver capabilities through case-based inference", *Proc Fourth Int. Workshop on Machine Learning*, pp. 167-178.
- Kolodner, J.L., 1991, "Improving human decision making through case-based decision aiding", *AI Magazine*, Summer Issue, pp. 52-68.
- Konstantinov, K.B. and T. Yoshida, 1992, "Real-time qualitative analysis of the temporal shapes of the (bio)process variables", *AIChE Journal*, **38**, 1703-1715.
- Koton, P., 1989, "A case-based resource allocation and scheduling system", *Proceedings of a Workshop on Case-Based Reasoning*, 285-289, San Mateo, Calif..

- Koton, P., 1989, *Using Experience in Learning and Problem Solving*, Technical Report, MIT/LCS/TR-441.
- Kowalski, A., 1991. "Diagnostic expert system for solving pitch problems", *Pulp and Paper Expert Systems Workshop*, Pointe Claire, Quebec.
- Kramer, M.A. and B.L. Palowitch Jr, 1987, "A rule-based approach to fault diagnosis using the signed directed graph", *AIChE Journal*, **33**, 1067-1078.
- Kramer, M.A., 1992, "Autoassociative neural networks", *Comput. Chem. Engng.*, **16**, 313-328.
- Kresta, J.V., J.F. MacGregor and T.E. Marlin, 1991, "Multivariate statistical monitoring of process performance", *Canadian Journal of Chemical Engineering*, **69**, 35-47.
- Kubrusly, C.S. and H. Malebranche, 1985, "Sensors and controllers location in distributed systems -- A survey," *Automatica*, **21**, 117-128.
- Kuipers, B., C. Chiu, D.T. Dalle Molle and D.R. Throop, 1991, "High-order derivative constraints in qualitative simulation", *Artificial Intelligence*, **51**, 343-379.
- Kuipers, B.J., 1989, "Qualitative simulation", *Artificial Intelligence*, **29**, 289-238.
- Lapointe, J., B. Marcos, M. Veillette and G. Laflamme, 1989, "BIOEXPERT--an expert system for wastewater treatment process diagnosis," *Computers chem. Engng*, **13**, 619-630.
- Lapp, S.A. and G.J. Powers, 1977, "Computer-aided synthesis of fault-trees", *IEEE Trans. Reliability*, **26**, 2.
- Laurent, J., 1986, "Comparative evaluation of three expert system developing tools: KEE, knowledge craft and ART", *The Knowledge Engineering Review*.
- Lebeau B., R. Arrese, S. Bauduin, R. Grobet and C. Foulard, 1980, "Noninteracting multivariable paper machine headbox control: some comparisons with classical loops", *Proc 4th IFAC Conf. on Instr. and Autom. in the Paper, Rubber, Plastics and Polym. Industr.*, Gent, pp. 227-238.
- Leitch, R. and A. Stefanini, 1989, "Task dependent tools for intelligent automation", *Artificial Intelligence in Engineering*, **4**, 126-143.

- Lenoard, J.A. and M.A. Kramer, 1990, "Limitations of the backpropagation approach to fault diagnosis and improvement with radial basis functions", *Presented at the AIChE annual meeting*, Chicago.
- Leung, K.S. and M.H. Wong, 1990, "An expert system using structured knowledge", *Computer*, 38-47.
- Liu, W., Q. Xia, C. Henriksson and H. Farzadeh, 1995, "Intelligent multimedia on-line manual development at the Ranger Slave Lake Pulp Corporation", *Proc Second Research Review Conference of IEL*, University of Alberta, Edmonton, Canada, pp. 235-244.
- Locatelli A., R. Scattolini and N. Schiavoni, 1986, "On the design of reliable robust decentralized regulators", *Large Scale System*, 10, 95-113.
- Long, A.B. ,1984, "Computerized operator decision aides", *Nuclear Safety*, 25, 521-524.
- Looze, D.P., S.M. Krolowski, J.I. Weiss, J.S. Eterno and S.W. Gully, 1984, "An approach to restructurable control system design", *Proc 23rd IEEE Conf. On Decision and Control*, Las Vegas, pp 1392-1397.
- Macchietto, S., G. Stuart, T.A. Perris, and G.R. Dissinger, 1989, "Monitoring and on-line optimization of process using speedup," *Computers chem. Engng*, Vol. 13, pp. 571-576.
- Macchion, D.J. and D.P. Vo, 1993, "A hybrid knowledge-based system for technical diagnosis learning and assistance", *Proc First European Workshop, EWCBR-93*, Kaiserslautern, German, pp. 301-312.
- MacGregor, J., 1994, "Statistical process control of multivariate process", *Control Eng. Practice*, 3, 403-414.
- MacGregor, J.F., C. Jaeckle, C. Kiparissides and M. Koutoudi, 1994, "Process monitoring and diagnosis by multiblock PLS methods", *AIChE Journal*, 40, 826-838.
- Marchionini G. and B. Shneiderman, 1988, "Finding facts vs. browsing knowledge in hypertext systems", *Computer*, pp. 70-79, January.

- Mariton M. and P. Bertrand, 1987, "Reliable flight control systems: components placement and feedback synthesis", *Proc 10th IFAC World Congress*, Munich, pp. 150-154.
- Martin, E.B. and A.J. Morris, 1996, "An overview of multivariate statistical process control in continuous and batch performance monitoring", *Transactions of the Institute of Measurement and Control*, **18**, 51-60.
- Martin, E.B., A.J. Morris and J. Zhang, 1996, "Process performance monitoring using multivariate statistical process control", *IEE Proc.-Control Theory Appl.*, **143**, 132-144.
- Maule, R.W., 1991, "Media integration for an information system", *Information Resources Management Journal*, **4**, 13-20.
- Maybury, M., 1992, "Intelligent multimedia interfaces", *IEEE Expert*, **7**, 75-77.
- Meier, W., R. Weber and H.J. Zimmermann, 1994, "Fuzzy data analysis -- methods and industrial applications", *Fuzzy Sets and Systems*, **61**, 19-28.
- Modarres, M. and T. Cadman, 1986, "A method of alarm system analysis for process plants," *Computers & Chemical Engineering*, **10**, 557-665.
- MoDo Chemetics, 1990, *MOPS: Millwide Information & Operation*, MoDo Chemetics.
- Montgomery, R.C. and A.K. Caglayan, 1974, "A self-reorganizing digital flight control system for aircraft", *AIAA 12th Aerospace Sciences Meeting*, Washington, D.C.
- Murdock, J.L. and G. Hayes-Roth, 1991, "Intelligent monitoring and control of semiconductor manufacturing equipment", *IEEE Expert*, December issue, 19-31.
- Nielsen, J., 1990, "The art of navigating through hypertext", *Communication of the ACM*, **298 -308**, March.
- Nimmo, I., 1995, "Adequately address abnormal operation", *Chemical Engineering Progress*, Sept, 36-45.
- Nomikos, P. and J.F. MacGregor, 1994, "Monitoring of batch processes using multi-way principal component analysis", *AIChE Journal*, **40**, 1361-1375.
- Palus, M. and I. Dvrak, 1992, "Singular-value decomposition in attract reconstruction: pitfalls and precautions," *Physics D*, **55**, 221-234.

- Pan, J. and J.M. Tenenbaum, 1991, "An intelligent agent framework for enterprise integration", *IEEE Trans. SMC*, **21**, 1391-1407.
- Park, S.W. and D.M. Himmelblau, 1987, "Structure design for system fault diagnosis", *Comput. chem. Engng*, **11**, 713-726.
- Paske, R., 1990, "Hypermedia: A brief history and progress report", *T.H.E. Journal*, **18**, 53-56.
- Patel, R.V., M. Toda and B. Sridhar, 1977, "Robustness of linear quadratic state feedback design in the presence of system uncertainty", *IEEE Trans. Autom. Contr.*, **AC-22**, 945-949.
- Patton, R.J., P.M. Frank, R.N. Clark, 1989, *Fault Diagnosis in Dynamic Systems: Theory and Applications*, Prentice Hall.
- Petersen, I.R., 1987, "A procedure for simultaneously stabilizing a collection of single input linear systems using non-linear state feedback control", *Automatica*, **23**, 33-40.
- Petti, T.F., J. Klein and P.S. Dhurjati, 1990, "Diagnostic model processor: using deep knowledge for process fault diagnosis", *AIChE Journal*, **36**, 565-575.
- Pews, G. and S. Wess, 1993, "Combining model-based approaches and case-based reasoning for similarity assessment and case adaptation in diagnostic applications", *Preprints First European Workshop on Case-Based Reasoning (EWCBR-93)*, Vol 2, pp. 325-328.
- Piovoso, M.J., K.A. Kosanovich and J.P. Yuk, 1992, "Process data chemometrica", *IEEE Trans. Instrum. Meas.*, **42**, 262-268.
- Polak E., 1971, *Computational Methods in Optimization: A Unified Approach*, Academic Press.
- Portinale, L., P. Torasso, C. Ortalda and A. Giardino, 1993, "Using case-based reasoning to focus model-based diagnostic problem solving", *Proc First European Workshop (EWCBR-93)*, Kaiserslautern, German, pp. 325-337.

- Ragusa, J.M. and J. Coyne, 1992, "Integrating knowledge-based (expert) systems and digital video: A development and application paradigm shift", *Journal of Interactive Instruction Development*, **5**, 18-23.
- Ragusa, J.M., 1994, "Models and application of multimedia, hypermedia, and intellimedia integration with expert systems", *Expert Systems with Applications*, **7**, 407-426.
- Rao, M., R. Dong and V. Mahalec, 1994a, "Intelligent system for safe process startup", *Engineering Application of AI*, **7**, 349-360.
- Rao, M, Q. Xia and Y. Ying, 1994b, *Modeling and Advanced Control for Process Industries -- Applications to Paper Making Processes*, Springer-Verlag, London.
- Rao, M. and H. Qiu, 1993, *Process Control Engineering*, Gordon and Breach, US.
- Rao, M. and J. Corbin, 1992, "Intelligent system for batch digester operation", *78th Annual Meeting*, Technical Section, CPPA, B81-85, Montreal, Canada.
- Rao, M. and Q. Xia, 1994, "Integrated distributed intelligent system for on-line monitoring and control of pulp process", *Canadian Artificial Intelligence*, **33**, 5-10.
- Rao, M., 1991, *Integrated System for Intelligent Control*, Springer-Verlag, Berlin.
- Rao, M. and Q. Xia, "Intelligent on-line monitoring and control system – a univeristy-industry-government R&D partnership program", *International Journal of Engineering Education*, **13** (6).
- Rao, M., Q. Wang and J. Cha, 1993, *Integrated Distributed Intelligent Systems in Manufacturing*, Chapman & Hall, London.
- Raskin, R., 1990, "Multimedia: The Next Frontier for Business ?", *PC Magazine*, **9**, 151-191.
- Rasmussen, J. and L.P. Goodstein, 1987, "Decision support in supervisory control of high-risk industrial systems", *Automatica*, **22**, 663-671.
- Rasmussen, J., 1993, "Diagnostic reasoning in action", *IEEE Trans. SMC*, **23**, 981-992.

- Renard, F., L. Sterling and C. Brosilow, 1993, "Knowledge verification in expert systems combining declarative and procedural representations", *Computers chem. Engng*, **17**, 1067-1090.
- Rengaswamy, R. and V. Venkatasubramanian, 1995, "A syntactic pattern-recognition approach for process monitoring and fault diagnosis", *Engng Applic. Artif. Intell.*, **8**, 35-51.
- Rich, S.H. and V. Venkatasubramanian, 1987, "Model-based reasoning in diagnostic expert systems for chemical process plants", *Comput. chem. Engng*, **9**, 285-293.
- Rissland, E. and D. Skalak, 1989, "Combining case-based reasoning and rule-based reasoning: a heuristic approach, *IJCAI-89*, Vol. 1, pp. 20-25, August.
- Rodden, G., 1995, "Changing the face of research and development", *Pulp & Paper Canada*, **96**, 19-23.
- Rouse, B. and S.H. Rouse, 1979, "Measures of complexity of fault diagnosis tasks," *IEEE Trans. SMC*, **9**, 720-727.
- Rumph, P.E., 1993, "Partnerships with the government", *Proc 1993 Control Conference*, pp. 311-315.
- Sachs, P.A., A.M. Paterson and M.H.M. Turner, 1986, "ESCORT -- an expert system for complex operations in real time", *Expert Systems*, **3**, 22-29.
- Saelid, S., A. Mjaavatten and K. Fjalestad, 1989, "An object oriented operator support system based on process models and an expert system shell", *Proc First European Symp. on Computer Aided Process Engineering*, S97-S108.
- Schank, R. 1986, *Explanation Patterns: Understanding Mechanically and Creatively*, Hillsdale, N.J., Erlbaum.
- Schank, R., 1975, "The structure of episodes in memory", In *Representation and Understanding*, eds., D.G. Bobrow and A. Collins, New York: Academic, pp. 237-272.
- Schank, R., 1981, "Failure-driven memory", *Cognition and Brain Theory*, **4**, 41-60.

- Selfridge, M. and B. Cuthill, 1989, "Retrieving relevant out-of-context cases: A dynamic memory approach to case-based reasoning", *Proc Workshop on Case-Based Reasoning*, pp. 370-387, San Mateo, Calif.
- Seong, P.H., M.W. Golay and V.P. Manno, 1994, "Diagnostic entropy: a quantitative measure of the effect of signal incompleteness on system diagnosis," *Reliability Engineering and System Safety*, **45**, 235-248.
- Shimenura E. and Fujita M., 1985, "A design method for linear state feedback systems processing integrity based on a solution of Riccati-type equation", *Int. J. Control*, **42**, 887-899.
- Shiozaki, J., H. Matsuyama, E. O'Shima and M. Iri, 1985, "An improved algorithm for diagnosis of system failures in the chemical processes", *Comput. Chem. Engng.*, **9**, 285-293.
- Short, D.D., 1992, "Instruction in a post literate culture: multimedia in higher education", *Journal of Information Systems Education*, **4**, 11-21.
- Shu, Y., H. Yang and H. Qiu, 1995, "INTEMOR--Intelligent multimedia system for on-line real-time applications", *Proc Second Research Review Conference of IEL*, University of Alberta, Edmonton, Canada, pp. 71-94.
- Siljak, D.D., 1980, "Reliable control using multiple control systems", *Int. J. Control*, **33**, 303.
- Simpson, R.L., 1985, *A Computer Model of Case-Based Reasoning in Problem Solving: An Investigation in the Domain of Dispute Mediation*, Ph.D. Thesis, School of Information and Computer Science, Georgia Institute of Technology.
- Sipior, J.C. and E.J. Garrity, 1992, "Merging expert systems with multimedia technology", *Data Base*, **21**, 45-49.
- Slade, S., 1991, "Case-based reasoning: a research paradigm", *AI Magazine*, Spring, 42-55.
- Soucek, B., 1991, "From modules to application-oriented integrated systems", in *Neural and Intelligent Systems Integration: Fifth and Sixth Generation Integrated Reasoning Information Systems*, pp. 1-36, John Wiley & Sons, Inc.

- Stanfill, C., 1988, "Learning to read: a memory-based model", *Proc DARPA Workshop on Case-Based Reasoning*, pp. 402-413, San Mateo, Calif.
- Stephanoploulos, G.; 1990, "Artificial intelligence in process engineering--Current state and future trends," *Computers chem. Engng*, **14**, pp. 1259-1282.
- Stephanopoulos, G. 1991, "Towards the intelligent controller: formal integration of pattern recognition with control theory", *Chemical Process Control -- CPCIV*, (Edited by Arkun Y. and Ray W.H.), Texas, pp 615-645.
- Stephanopoulos, G., J. Johnston, T. Kriticos, R. Lakshmanan, M. Mavrovouniotis and C. Sillet, 1987, "Design-Kit: an object oriented environment for process engineering", *Comput. chem. Engng*, **11** (6).
- Straub, D.W. and Wetherbe, J.C., 1989, "Information technologies for the 1990s: An organizational impact perspective", *Communications of the ACM*, **32**, 1328-1339.
- Sugeno, M. (Ed.), 1985, *Industrial Applications of Fuzzy Control*, North-Holland, Amsterdam.
- Tien, J.M. and M.M. Croke, 1990, "An operators-oriented approach to decision support systems", *Proc 1990 IEEE Conf. on SMC*, Nov. 4-7, pp. 909-914.
- Torasso, P., L. Portinale, L. Console, and M.C. Mont, 1992, "Approximate reasoning in a system combining prototypical knowledge with case-based reasoning." In L.A. Zadeh and J. Kacprzyk, editor, *Fuzzy Logic for the Management of Uncertainty*. John Wiley & Sons.
- Tsuge, Y., J. Shiozaki, H. Matsuyama, E. O'Shima, Y. Iguchi, M. Fuchigami and M Matsushita, 1985, "Feasibility study of a fault-diagnosis system for chemical plants", *Intern. Chem. Engng*, **25**, 660-667.
- Turner, R., 1988, "Organizing and using schematic knowledge for medical diagnosis", *Proc DARPA Workshop on Case-Based Reasoning*, San Mateo, Calif., pp. 435-446.
- Venkatasubram, V. and S.H. Rich, 1988, "An object-oriented two-tier architecture for integrating compiled and deep-level knowledge for process diagnosis", *Comput. Chem. Eng*, **12**, 903-921.

- Venkatasubramanian, V. and K. Chan, 1989, "A neural network methodology for process fault diagnosis", *AIChE Journal*, **35**, 1993-2002.
- Venkatasubramanian, V., R. Vaidyanathan and Y. Yamamoto, 1990, "Process fault detection and diagnosis using neural networks -- I Steady-state processes", *Computers chem. Engng*, **14**, 699-712.
- Vianna, R.F. and C. McGreavy, 1995, "Qualitative modeling of chemical processes - a weighted digraph (WDG) approach", *Computers chem. Engng*, **19**, pp.S375-380.
- Vidyasagar M. and Viswanadhm N., 1985, "Reliable stabilization using multi-controller configuration", *Automatica*, **21**, 599-602.
- Waters, A. and J.W. Ponton, 1989, "Qualitative simulation and fault propagation in process plants", *Chemical Engineering Research and Design*, **67**, 407-422.
- Whiteley, J.R. and J.F. Davis, 1992, "Knowledge-based interpretation of sensor patterns", *Comput. chem. Engng*, **16**, 329-346.
- Whiteley, J.R. and J.F. Davis, 1994, "A similarity-based approach to interpretation of sensor data using adaptive resonance theory", *Computers chem. Engng*, **18**, 637-661.
- Willsky, A.S., 1976, "A survey of design methods for failure detection in dynamic systems", *Automatica*, **12**, 601-611.
- Willsky, A.S., 1986, *Detection of abrupt changes in dynamic systems. Lecture Notes in Control and Information Sciences*, (Ed. M. Basseville and A. Benveniste), Vol 77. Springer-Verlag.
- Wold, S., 1978, "Cross validatory estimation of the number of components in factor and principal components model", *Technometrics*, **20**, 397-404.
- Wold, S., K. Esbensen and P. Geladi, 1987, "Principal components analysis", *Chemometrics Intelligent Lab. Syst.*, **2**, 37-52.
- Xia, Q., M. Rao, X. Shen and Y. Ying, 1993, "Systematic modeling and decoupling design of a pressurized headbox", *Proc Canadian conference on electrical and computer engineering*, Vancouver, pp 962-965.

- Xia, Q., M. Rao, Y. Sun and Y. Ying, 1993 "A new technique for decoupling control", *International Journal of Systems Science*, **24**, 289-300.
- Xia, Q. and M. Rao, 1992, "Fault-tolerant control of paper machine headbox", *Journal of Process Control*, **2**, 171-178.
- Xia, Q. and M. Rao, 1997a, "A Hybrid intelligent system for process operation support", *Proc IEEE International Conference on SMC*, Oct. 12-17, Orlando, Florida, USA.
- Xia, Q. and M. Rao, 1997b, "Incorporating system dynamics in case-based reasoning for process operation support", *Proc. IEEE International Conference on SMC*, Oct. 12-17, Orlando, Florida, USA.
- Xia, Q., H. Farzadeh, M. Rao and C. Henriksson, K. Danielson and J. Olofsson, 1993, "Intelligent operation support system for slave lake pulp corporation," *Proc. 1993 IEEE Conference on Control Application*, Vancouver, B.C., Sept. 13-16, pp. 591-596.
- Xia, Q., M. Rao, C. Henriksson and H. Farzadeh, 1997, "Case-based reasoning for intelligent fault diagnosis and decision making in pulp processes", *Pulp & Paper Canada*, **98** (2), 26-30.
- Xia, Q., M. Rao, X. Shen and G. Sedgiwik, 1994a, "Intelligent design of actuator and sensor for emergency support systems", *Proc IFAC Workshop on Integration of Process Design & Control*, June 27-28, 1994, Baltimore, Maryland, USA.
- Xia, Q., M. Rao, Y. Ying and X. Shen, 1994b, "Adaptive fading Kalman filter with applications", *Automatica*, **30**, 1333-1338.
- Xia, Q., X. Shen, M. Rao and V. Gourishankar, 1995, "Robust estimation and compensation for actuator and sensor failures of linear systems", *International Journal of System Science*, **25**, 1867-1876.
- Yager, T., 1992, "The multimedia PC: high-powered sight and sound on your desk", *Byte*, 217-226.
- Yankelovich, B. J. Haan, N. K. Meyrowitz and S. M. Drucker, 1988, "Intermedia: the concept and the construction of a seamless information environment", *Computer*, 81-96, January.

- Yedavalli, R.K., 1985, "Improved measures of stability robustness for linear state space model", *IEEE Trans. Autom. Contr.*, **AC-30**, 577-579.
- Yu, C.C. and C. Lee, 1991, "Fault diagnosis based on qualitative/quantitative process knowledge", *AIChE Journal*, **37**, 617-628.
- Zadeh, L.A., 1965, "Fuzzy sets", *Information and Control*, **8**, 338-353.
- Zadeh, L.A., 1978, "Fuzzy sets as a basis for a theory of possibility", *Fuzzy Sets and Systems*, **1**, 3-28.
- Zhang, J., Martin, E. Martin and A.J. Morris, 1995a, "Fault detection and classification through multivariate statistical techniques", *Proc American Control Conference*, Seattle, Washington, June 1995, P. 751-755.
- Zhang, J., E.B. Martin and A.J. Morris, 1995b, *Non-linear principal components analysis and accumulated score plots*, Internal Report, Department of Chemical & Process Engineering, University of Newcastle, United Kingdom.
- Zimmermann, H.J., 1987, *Fuzzy Sets, Decision Making, and Expert Systems*, Kluwer Academic Publications.
- Zurcher, J., 1994, *Application of Artificial Intelligence to Improve Pulp Mill Operations*, M.Sc. Thesis, Dept. of Chemical Engineering, University of Alberta.