International Journal of Management Science
Vol 8, No 1, May 2002

Learning Framework for Robust Planning and Real-Time Execution Control

Gi-Nam Wang*

Dept. of Industrial and Information Systems Engineering,
Ajou University, Suwon, Korea 442-749

Gang Yu**

Dept. of Management Science and Information Systems, McCombs School of Business,
The University of Texas at Austin, Austin, TX78712, USA.

(Received Jan. 2002, Revised Mar. 2002, Accepted May 2002)

ABSTRACT

In this paper, an attempt is made to establish a learning framework for robust planning and real-time execution control. Necessary definitions and concepts are clearly presented to describe real-time operational control in response to plan disruptions. A general mathematical framework for disruption recovery is also laid out. Global disruption model is decomposed into suitable number of local disruption models. Execution pattern is designed to capture local disruptions using decomposed-reverse neural mappings, and to further demonstrate how the decomposed-reverse mappings could be applied for solving disrubtion recovery problems. Two decomposed-reverse neural mappings, *N-K-M* and *M-K-N* are embloyed to produce transportation solutions in real-time. A potential extension is also discussed using the proposed mapping principle and other hybrid heuristics. Experimental results are provided to verify the proposed approach.

1. INTRODUCTION

The e-business environment requires more tightly integrated planning and execution over on-line and off-line application domain. The nature of material acquisition, manufacturing and distribution has also been remarkably changed while

* Gi-Nam Wang's work was partially supported by Korea Ministry of Science and Technology

** Gang Yu's research was supported in part by the National Natural Science Foundation of China under grant number 79928001, and by the FADRC grant and a Special Research Grant from the University of Texas at Austin.

customers demand quick service response to their ordering and to their continuous order changes. The available-to-promise (ATP) and capable-to-promise (CTP) are examples of quick response policies to customers, which require synchronizing planning and real-time execution control. The deployment of real-time decision support systems has increased significantly in recent years. These systems have become prevalent in air and ground transportation, manufacturing, telecommunication, and other industries.

There are a number of technological factors that contributed to the growth of robust planning and real-time execution control systems. Computing speed is now fast enough to solve complex, large-scale, real-time problems, and computer storage is now large enough to accommodate enterprise-wide data. Network is reliable enough to host mission-critical applications, and graphical user interface (GUI) tools are powerful enough to facilitate rapid application development. The advent of modern technology allows for real-time data collection, communication, and information processing and sharing at affordable prices.

In manufacturing, make-to-order (MTO) policy requires a robust and efficient re-planning and re-scheduling in order to provide accurate and rapid order promising to customers. MTO companies should generate optimal or near optimal planning and scheduling in real time as well as a best quotation in order to meet customer orders. They require an efficient execution control such as real time scheduling and operations control using feasible resource capacities and confirmed orders. On the other hand, make-to-stock (MTS) policy focuses on long-term or mid-term planning for future forecasting orders while MTO focuses on short-term scheduling based on received orders. MTS requires a robust planning that is aimed at handling uncertainties or preparing a safety stock for the future. However, many companies adopt MTO and MTS policies simultaneously, thus they require an integrated approach on robust planning and execution control.

The function of ATP and CTP, which aims at notifying the information on product availability and exact delivery date, is critical under the e-business environment. Under the MTO and MTS based bidding situation, customers desire clearly specified delivery date and quantity as well as best quote. However, what commonly occurred is that there might be large gaps between planning and execution environment such as changes in system conditions, parameters, and resource availabilities. Under the situation of occurring dynamic planning change, optimal or near optimal re-planning and re-scheduling are required in real time.

In this paper, a learning framework for robust planning and real-time execution control is discussed. Conceptual definitions and a general mathematical framework are laid out with emphasis on disruption recovery. Disruption model is approximated by an appropriate number of localized execution patterns, which are established at planning stage. To capture localized disruptions, a learning paradigm is designed using two reverse neural mappings. We also demonstrate how the identification of execution patterns and reverse mappings could be applied for solving disruption problems in real-time. A brief discussion is given to extend to general applications using other hybrid heuristics.

Organization of this paper is as follows. Conceptual definitions and terminologies are given in Section 2. In Section 3, a general mathematical framework and solution methodologies for robust planning and real-time execution control are presented. To make our discussion concrete, mathematical models are presented with a specific description problem. Mathematical disruption model is formulated in terms of original planning model and disruption models, and the model is also described approximately by a suitable number of localized execution patterns. Also presented are the basic concepts and principles of the two reverse mappings to solve local disruption problem in Section 4. In Section 5, experimental results are shown and analyzed. Some discussions and concluding remarks are followed in Section 6.

2. DEFINITIONS AND TERMINOLOGIES

In this section, definitions and necessary terminologies are introduced to set up robust planning and real-time execution control. Necessary concepts are discussed in connection with real-time planning and execution control under disruptions. The following definitions are used throughout this paper.

2.1 Real-time operations control:

Given an operational plan (assume optimized or generated with the best effort during planning stage), real-time operations control is the process of making an optimal or near optimal decision in real-time on revised execution of the plan and/or recovery of the plan [G. Yu, 2000].

2.2 Disruption:

Perturbation of the system that may result in plan change. Such perturbation may be caused by changes in system environment, parameters, resource availabilities, and etc. Disruption model is expressed as a deviation from original model in terms of disruption parameters.

2.3 Execution patterns:

A disruption model could be partitioned into a number of localized disruption models. Execution pattern refers to a localized disruption model and the solution mapping capability for a given local disruption model. Generally, a pattern could be a solution model, a well-defined procedure, or a core of solution to a specific problem, which could be repetitively reusable solutions for handling similar range of disruption problem. In this paper, execution pattern includes a mapping capability to produce solutions for a localized disruption model as well as the corresponding mathematical model itself.

2.4 Framework:

A systematically defined process as well as a set of models and model generators producing solution patterns. A framework may contain a set of classes providing customizable solution patterns.

Robust planning and execution control may require a number of different solutions depending on application domains and situations. Knowledge depository (i.e., component depository) is necessary to store and to provide appropriate models and solutions. Real-time operations control is performed effectively and efficiently by selection of appropriate execution patterns from well-organized framework.

2.5 Learning framework:

A framework should be able to generate execution patterns, which could be repetitively usable models, solution procedures, or methodologies. In order to make an optimal or near optimal decision in real-time, it is necessary to store necessary execution patterns into a knowledge depository, and it is also essential to give on-line solutions to specific problems without consuming too much time. Learning characteristics could be helpful for on-line applications. A framework could contains learning capability in order to produce robust execution patterns in real-time.

Robust solutions could be focused on minimizing the worst–case "opportunity cost" or on minimizing maximum regret under uncertainty. Emphasis may also be placed on stochastic approaches under which various scenarios are described with probability distributions and the goal is to find a solution minimizing expected total cost [I. Averbakh and O. Berman, 2000]. However, it is difficult to accurately assess the probability distribution for various scenarios in a given problem. The min-max regret approach, interpreted as minimizing the worst-case cost, could be a viable approach in real world applications [P. Kouvelis and G. Yu, 1997].

In this paper, robustness refers to robust problem solving capability to various disruption problems. Learning framework is designed for providing robust solutions in real time to many different disruption cases.

2.6 Robust planning:

Uncertainty in operational systems is often faced at the planning stage. There is insufficient knowledge about parameters of a system. Given a plan, there might be unpredictable disruptions at the execution phase. To cope with such a perturbation result in plan change, various approaches have been studied such as sensitivity analysis, stochastic modeling approach, and min-max regret approach. In spite of these efforts, there are considerable gap between planning and its realization. An attempt is given here to obtain robust planning solution patterns from a well-organized learning framework.

2.7 Synchronization of planning and execution: robust planning and adaptive execution

Planning and execution control should be tightly integrated. Planning should be conducted while taking into account of smooth operational control, and operational control should also be performed with a planning solution in mind. Realization of planning includes establishing and best performing the current operations control and re-planning with consideration of execution result to next planning. Different approaches have been proposed for solving re-planning and re-scheduling tasks (Lynn Ling X Li, 1999), and especially in the supply chain management area, some models have been presented to perform execution and planning tasks simultaneously (Mark J. Euwe etal., 1998). Mostly employed methods are based on real time computation and data sharing. Major shortcomings are caused by the unmanageable computational complexity for solving large-scale problems. Little effort is given to reduce the number of re-planning and re-scheduling tasks by using learning framework for generating robust planning and adaptive execution solutions simultaneously.

3. THE MATHEMATICAL FRAMEWORK AND SOLUTION METHODOLOGIES

We now present a general framework for real-time planning and execution control Our focus will be on plan recovery under disruptions.

We define the decision space as X, parameter vector of the system as a, and the objective function as $f(\cdot,\cdot)$. The planning optimization model, P, becomes

$$(P)\quad z = \max f(a;x) \qquad \text{subject to} \qquad x \in X$$

Let the optimal solution to the above model be x^*.

Disruptions model, D, can be caused by the following:

- changes in system environment
- change in system parameters
- change in availability of resources
- new external restrictions, new considerations
- uncertainties in system performance, and
- uncontrollable events.

The above disruptions will lead to

$$a \xrightarrow{D} a'$$
$$X \xrightarrow{D} X'$$

A new objective function $g(a', x^*; x)$ is necessary for measuring recovery impact.

Thus, the real-time execution control problem could be expressed as a disruption model as

$$z_R = \max g\ (a', x^*; x) \qquad \text{subject to} \qquad x \in X',$$

The new objective can be modeled as the weighted combination of the original objective and a term that accounts for the deviation from the original plan:

$$g(a', x^*; x) = \alpha f(a', x) + \beta h(|x - x^*|)$$

The operational cost for redirecting resource flows, notifying plan changes, and reflecting the impact caused by recovery are all captured by the last cost term The weights α and β are used to specify the relative importance of operational recovery costs.

Commonly occurred are complexities of real problems. One mathematical model is not enough for describing real planning situations. Multiple objectives exist with different variables, and there are trade-off relationships between objective criteria. The problem might be more complex and harder when we consider disruption model. Mathematically well-defined models and algorithms are not easily developed. In order to handle complicated disruption problem, robust planning based on learning paradigm is presented.

3.1 Main Principle of the Proposed Learning Framework

Two disruption models are introduced: global and local disruption model. Global disruption model includes all ranges of possible plan change while local disruption model is limited to a specific range of plan changes. In order to handle plan changes effectively, the global disruption model is partitioned into a suitable number of local disruption models depending on the degree of disruption changes. A focus is given on solving local disruption models using learning paradigm. Problem solving capability is captured by a simple mapping capability obtained by learning training data set for a given local disruption model, and the mapping capabilities are generalized for solving test data. Generalization capability could be used for solving similar local disruption problem not included in training data. Generalization capability is a general problem solving capability to variant situation from original plan, which is interpreted as a problem solving capability even though there is a gap between learning data and testing data.

Learning process involves capturing mapping knowledge from training data set and it could be applied for handling similar patterns. After learning at planning stage, real-time planning and execution control are performed by identification of appropriate execution pattern having both similar local disruption model and solution mapping capability. Therefore, at the planning phase, global and local disruption models are established with solutions mapping capability. At the execution phase, the best local disruption model with the corresponding mapping is selected from the exiting ones and used for finding best solutions.

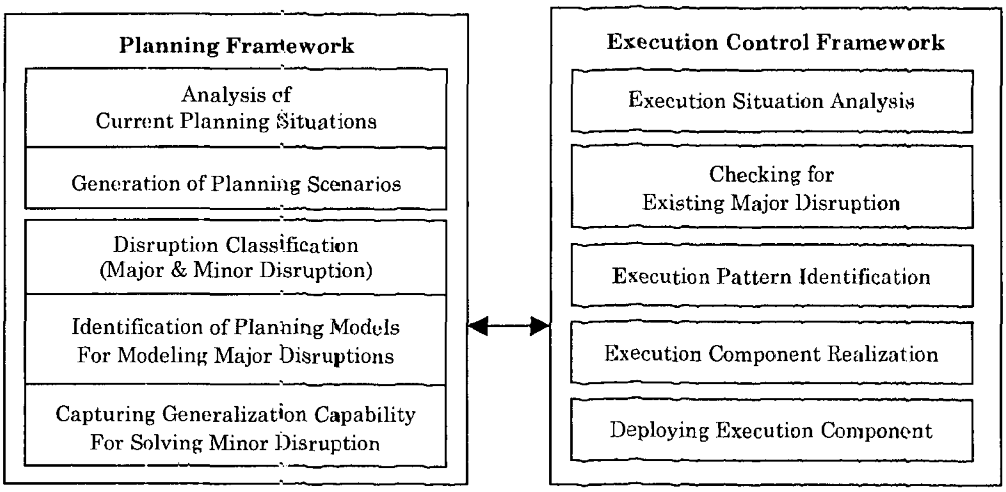

Figure 1. Robust Framework for Robust Planning and Real-Time Execution Control

The above figure describes the overall robust framework for real-time planning and execution control. The robustness of this framework could be connected to the generality of global model and generalization capability contained in local solution patterns.

At the planning stage, the concept of robustness and learning paradigm are employed in order to handle uncertain and complicated disruptions. The execution control is designed for identifying suitable detail disruption pattern and its realization to the current problem in real-time.

3.2 Synchronizing Robust Planning and Real-Time Execution Control

In the following, synchronizing steps are briefly presented, and detailed methods will be given by applying to a specific distribution problem.

Planning Stage: Robust Planning and Capturing Execution Pattern

Step 1. Establish planning model and find optimal or best solutions.
Step 2. Establish disruption model
Step 3. Partition the disruption model into a suitable number of localized models.
Step 4. Generate localized disruption models and find optimal solutions for learning.
Step 5. Learn appropriate mapping capabilities for each local execution pattern.

Execution Stage: Find Best Solutions and Tuning

Step 1. Identify the best execution pattern to the current disruption situation.
Step 2. Apply mapping structure to obtain execution solutions.

Application Problem

We consider a distribution problem with multi-plants and multi-distribution centers. There are dynamically changing orders. The problem is to establish weekly production plans for each plant, and to find daily or hourly distribution schedule from each plant to each distribution center. For simplicity, a focus is given on determining weekly production quantity at each plant minimizing the distribution cost, and also given on determining daily or hourly distribution schedule which is to determine the shipping quantity from each plant to each distribution center minimizing the total transportation cost. At the planning stage, there might be insufficient information on customer order quantities, and there might be gaps between the customer orders and distribution costs at planning phase and those at execution phase. Due to this uncertain customer orders, each plant and distribution center requires robust planning.

In this problem, robust planning could include not only reliably estimating system parameters like customer demands but also determining optimal production quantity to minimize total disruption deviation costs.

With consideration of order priority, above scenario could be extended to more general situations. In reality, the customer orders could have different priority. Daily (or hourly) execution control for distribution should be established considering order priority. Customer differentiation is widely adopted by many companies as a marketing strategy.

A nearest assignment rule might be dominant when we consider only distribution cost. However, consideration of manufacturing costs makes the decision problem complex. Daily (or hourly) scheduling issues are critical to inventory cost, machine or line set-up cost, job change cost, and lost orders.

Considering scenario, we define the following variables, parameters, constraints, and objective function based on the previous mathematical framework.

a. *The planning decision variable (x)*: weekly production plan,
b. *The execution decision variable (x)*: daily (hourly) distribution channels, routes, and schedules; distribution processing sequences; vendor selections, inventory replenishment policy of plant and distribution center, production schedule and job sequence.
c. *The parameters (a)*: demand forecast, worker daily hours and maximum overtime hours, promised delivery dates, inventory capacity, transportation speed, and transportation processing time.
d. *The constraints (X)*: meeting demands, warehouse capacity limitations, delivery deadlines, delivery vehicle availability and capacity, etc.
e. *The original objective function* ($f(\cdot,\cdot)$): maximizing total profit
f. *The disruptions (D)*:
   - Change in system environment: delivery routes and traffic conditions change.
   - Uncontrollable events: power failures and transportation traffic accidents.
   - Change in system parameters: production and inventory capacity.
   - Change in resource availability: product availability by assembly line or machine breakdowns.
   - New external restrictions: price change, environmental change.
   - Uncertainties in system performance: processing of various parts by different parameters.
   - New considerations: new customer orders and changing customer priority.
g. *The real-time planning and operations control objective* ($g(\cdot,\cdot)$): Minimize the total disruption impact including: customer ill-will; lost profit; missed

deliveries; increased inventory cost; increased over-time cost; increased delivery cost; increased production cost; and operational cost for plan changes.

Planning Stage

Step 1: Establish planning model and find optimal solution

From the following established planning model (P), an optimal solution, x^*, could be obtained.

x_{ij} : Production quantity assigned from plant i to distribution center j

C_{ij} : Shipping cost from plant i to distribution center j

W_i : Inventory storage capacity of plant i

d_j : Customer demand in distribution center j at time

$$(P)\ \text{Minimize} \quad \sum_i \sum_j C_{ij} x_{ij}$$

$$\text{Subject to} \quad \sum_i x_{ij} = \sum_j d_j$$

$$x_{ij} \geq 0, \ \forall\ i, j$$

Step 2: Establish Disruption Model

Considering possible plan changes, one general disruption model is not easily established since there could be multiple objective criteria and they could be trade-off relationships. For simplicity, we present one compact model for describing all procedure in planning and execution stage.

C'_{ij} : Changed shipping cost from plant i to distribution center j

d'_j : Changed customer demand in distribution center j

h_{ij} : Production assignment change cost from plant i to distribution center j established at planning phase.

$$(D)\ \text{Minimize} \quad \alpha \sum_i \sum_j C'_{ij} x_{ij} + \beta \sum_i \sum_j h_{ij} \mid x_{ij} - x^*_{ij}$$

$$\text{Subject to} \quad \sum_i x_{ij} = \sum_j d'_j$$

$$x_{ij} \geq 0, \ \forall\ i, j$$

Step 3: Partitioning the Disruption Model

From (*D*), the possible disruption range is decomposed into a suitable number of local disruption ranges. At the planning stage, C_{ij} and d_j are only given. We assume that the ranges of C'_{ij} and d'_{ij} be estimated.

Let $\frac{|\max C'_{ij} - \min C'_{ij}|}{k_1} = \nabla c_{ij}$ and $\frac{|\max d'_j - \min d'_j|}{k_2} = \nabla d_{ij}$. Then the partitioned disruption model is

$$(PD)\ \text{Minimize}\quad \alpha\sum_i\sum_j C_{ij}^{P_m} x_{ij} + \beta\sum_i\sum_j h_{ij}\,|\,x_{ij} - x_{ij}^*$$

$$\text{Subject to}\quad \sum_i x_{ij} = d_j^{P_n}$$

$$Min(C'_{ij}) + (m-1)\nabla c_{ij} \le C_{ij}^{P_m} \le Min(C'_{ij}) + m\nabla c_{ij}$$

$$Min(d'_j) + (n-1)\nabla d_j \le d_j^{P_n} \le Min(d'_j) + n\nabla d_j$$

$$x_{ij} \ge 0, \forall\ i, j$$

where k_1, k_2, m, and n are integers satisfying

$m = 1, 2, \cdots, k_1$ and $n = 1, 2, \cdots, k_2$.

The disruption model (D) is described approximately by a number of local range disruption models (*PD*), and the total number of localized models are determined by incremental ranges of disruption parameters such as ∇c_{ij} and ∇d_{ij}.

Step 4: Generate Localized Disruption Models and Find Optimal Solutions for Learning

Random numbers, within localized range of disruption parameters, are generated for a given $m = 1, 2, \cdots, k_1$, and $n = 1, 2, \cdots, k_2$, and optimal solutions are found for learning the disruption ranges

$$Min(C'_{ij}) + (m-1)\nabla c_{ij} \le C_{ij}^{P_m} \le Min(C'_{ij}) + m\nabla c_{ij}$$

$$Min(d'_j) + (n-1)\nabla d_j \le d_j^{P_n} \le Min(d'_j) + n\nabla d_j$$

Step 5: Capture *N-K-M* and *M-K-N* Mapping Structure

In order to solve localized disruption model which has variable ranged parameters shown in (*DP*), two neural network mappings are designed. General disruption problem is not easy to design a simple mapping scheme having a general solution mapping capability for all possible disruption cases. To overcome this, the disruption model is decomposed into a suitable number of localized disruption models, and two complementarily neural mappings are proposed for solving the decomposed-local disruption problems. Let $\nabla c_{ij}(t)$ and $\nabla d_{ij}(t)$ be the local incremental changes defined in partitioned disruption model *(PD)*. For a given $m = 1, 2, \cdots, k_1$, and $n = 1, 2, \cdots, k_2$, and optimal solutions are found for learning the disruption ranges using two reverse mappings.

$$Min(C_{ij}^{'}) + (m-1)\nabla c_{ij} \le C_{ij}^{P_m} \le Min(C_{ij}^{'}) + m\nabla c_{ij}$$

$$Min(d_j^{'}) + (n-1)\nabla d_j \le d_j^{P_n} \le Min(d_j^{'}) + n\nabla d_j$$

We designed direct mapping relationship between $\sum_i x_{ij}^{DP*}$, $\sum_j x_{ij}^{DP*}$, and $\sum_j D'$. x_{ij}^{DP*} is the optimal solution of *(PD)* and $D_j^{'}$ is the corresponding demand. Two mappings could be described as

$$d \xrightarrow{N_1} x$$

$$x \xrightarrow{N_2} d$$

The first mapping structure N_1 is to determine the total shipping quantity from every plant to each destination (i.e., distribution center). All demands and source supply quantities are utilized as input vector to map total optimal shipping quantity from each plant to each destination. The first N_1 has *M-K-N* mapping structure where *M* is the input vector, *K* is the number of hidden layer, and *N* is the output.

The second mapping is a reverse mapping to the first net N_1. The reverse mapping N_2 has *N-K-M* structure where *N* is input vector, *K* denotes the hidden layer, and *M* is the output. Reverse to the first mapping N_1, the input of second mapping is optimal solution vector for different $d_j^{P_n}$ having the same range, and output is the corresponding demand vector. For given vector *N* and *M*, two reverse mappings are trained using the same training data set, however they use reverse input-output mapping vector.

Execution Stage

Step 1: Identify Best Matching Pattern to the Current Disruption.

For a given real disruption problem of C_{ij}^E and d_j^E, find the best matching localized disruption model and corresponding two phases mapping (defined as execution pattern). The real problem might not be the exactly same as that of learning models. A matching score is applied for finding best execution pattern.

$$P_{m^*} = \min_m \{ | C'_{ij} - C^E_{ij} | \} \quad P_{n^*} = \min_m \{ | d'_j - d^E_j | \}$$

where C^E_{ij} and d^E_j are real parameters at execution stage.

Step 2: Apply mapping structure to obtain execution solutions
As an execution solution, two neural mappings are proposed to capture disruptions within a pattern. Two neural mappings with hybrid tuning procedure are employed for solving the disruption problem in real-time.

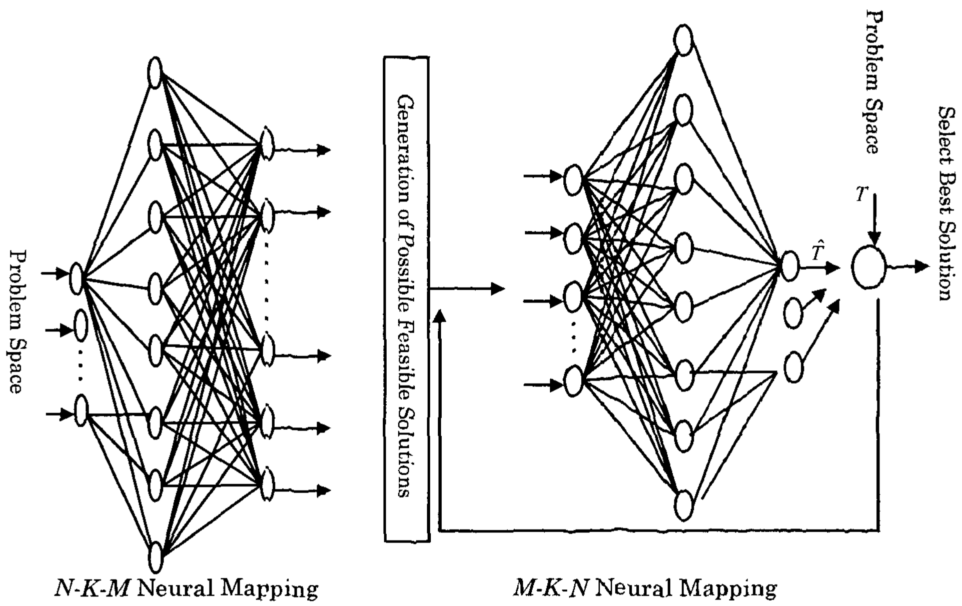

Figure 2. Two M-K-N and N-K-M Neural Mappings

The above figure shows the complement relationship between the first and second neural mapping in finding the best solution. Off-line learning is performed, and all learned mapping knowledge is stored in long-term memory at planning stage. Real-time solution is obtained using the stored long term-memory. As each situation is processed, the long-term memory is continuously updated by off-line learning. The overall procedure in connection with previous framework, shown in Figure 3, could be described as follows.

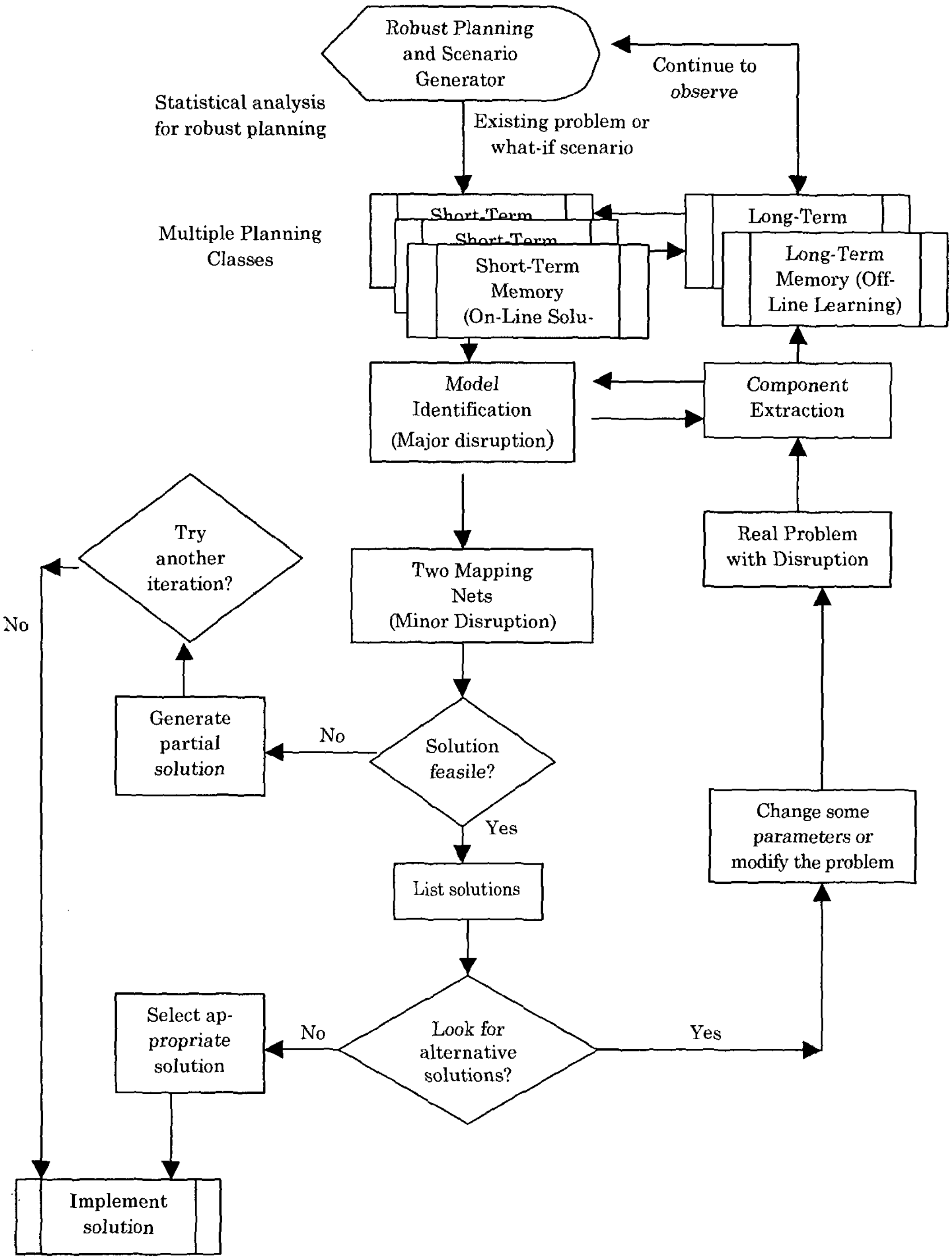

Figure 3. Robust Planning & Real-time Execution Control Framework Flowchart

Approximation of Feasibility and Optimality for Disruption Problem

One of the most difficult tasks in design of mapping relationship is how to find the best appropriate mapping relationship between input and output space. For example, a kind of functional mapping relationship could be well approximated using a feed forward neural network (Lapedes and Farber, Narendra and Parthasarathy). Functional mapping structures are commonly based on *M-K-N* neural net architecture having *M* number of inputs, *K* number of hidden layers, and *N* number of outputs. Compared to stochastic systems, deterministic I-O relationship could be easily captured using appropriate mapping structure. The *M-K-N* neural network mapping could be widely used for approximating deterministic functions as well as stochastic functions. Consider general stochastic function as

$$y_t = f_t\,(y_{t-1},\dots, y_{t-n_y}, u_{t-1}, \dots, u_{t-n_u}, \dots, e_t, \dots, e_{t-n_e}, \Theta_t) + e_t$$

where $\{y_t\}$ is an observable sequence of targets, f_t is an unknown mapping function, u_t is a sequence of inputs, and e_t is a noise process which is independently identically distributed errors. Feed-forward neural networks of multiple input, single output, and single hidden-layer have been shown to be capable of approximating the above nonlinear mapping relationship [Gallant and White 1992, Cybenko 1989, Hecht-Nielsen 1989].

Most common disruption problems are based on complex optimization formulation. The optimal feasible solutions could be described as mapping equations satisfying feasible constraints.

$$
\begin{aligned}
(P^*)\ \text{Minimize} \quad & \sum_i \sum_j C_{ij} x^*_{ij} = Z^* \\
\text{Subject to} \quad & \sum_i x^*_{ij} = \sum_j d_j \\
& x_{ij} \ge 0, \ \forall\, i, j
\end{aligned}
$$

The above optimal solution satisfies two linear equations such as optimality equation and feasibility equation. For a given cost factors, neural network could be trained to satisfy optimal feasible solutions using different demands. For the limited ranges of cost factors and demand variations, neural mapping can capture approximately optimal feasible equation. Two neural mappings, called as feasible neural net (FNN) and optimal neural net (ONN), could be designed for capturing feasible equation as well as optimality equation. FNN

$$d \xrightarrow{FNN} x^*$$
$$x^* \xrightarrow{ONN} z^*$$

The first mapping structure N_1 is to determine the total shipping quantity from every plant to each destination (i.e., distribution center). All demands and source supply quantities are utilized as input vector to map total optimal shipping quantity from each plant to each destination. The first N_1 has *M-K-N* mapping structure where *M* is the input vector, *K* is the number of hidden layer, and *N* is the output.

Disruption is interpreted as deviation from the original problem space. If we could design well-defined mapping structure between the original problem and the solutions, it is possible to apply the mapping scheme for solving disruption problem in real-time. The global disruption model is decomposed into a suitable number of local disruption models. A local disruption could be captured by generalization capability of the neural *N-K-M* and *M-K-N* mappings. A real disruption problem could be overcome by selection of appropriate local disruption model. Decomposition of mapping space into multiple mapping structures could overcome the global disruption problem, and it also improves generalization capability, which is also employed for solving local disruption problem. Generalization capability could also be improved by using complement reverse mapping structure such as *N-K-M*, and *M-K-N* mappings especially when there is not enough training data.

Without loss of generality, general extension could be possible by using proposed mapping principle. Depending on the problem, learning paradigm could be designed based on different mapping structures. Other hybrid learning heuristic may also be required for solving complicate problem.

4. EXPERIMENTAL ANALYSIS

For simplicity, simple transportation problem is employed for verifying proposed scheme. However, the real disruption situation is not easily modeled by one well-defined mathematical model. The simplified transportation problem could be solved efficiently by using the proposed execution pattern identification and reverse mapping scheme. Depending on different cost functions and demands, appropriate two reverse mapping networks, represented by *N-K-M* and *M-K-N* networks, are trained using data set. The data set is randomly generated for a given local range of cost and demand, which is utilized for learning an execution pattern. The number of execution pattern is equal to that of local disruption model. The total number of execution patterns and local disruption models could be reduced to smaller by grouping similar execution patterns and models into one group.

For a local disruption problem, two reverse mappings are trained simultaneously satisfying input and output mapping solution. At planning stage, planning model and global disruption model are established. The global disruption model is decomposed into a suitable number of local disruption models considering different ranges of plan changes. For each local disruption model, a set of training set is generated for capturing two reverse mappings. At execution stage, appropriate local disruption model and the corresponding two reverse mappings are selected using matching scores. From experiments, we could demonstrate how a disruption problem could be solvable by identification of suitable execution pattern and the generalization capability of the selected execution pattern.

The two *N-K-M* and *M-K-N* neural mappings are designed to capture local disruption problem for a given execution pattern. For a given *N-K-M* neural mapping, the input vector N is designed as supply quantities and destination demands, and target vector M is designed as the corresponding optimal solutions. Therefore the vector N and M are determined from problem situation. K denotes the number of hidden layers, which is determined by experimental analysis. The number of nodes for a given hidden layers are also determined by experimental simulation. The reverse *M-K-N* mapping has reverse elements to *N-K-M* mapping, which supply quantities and destination demands are employed as targets, and the corresponding optimal solutions are used as inputs. The number of destinations is chosen from the range of 10 to 100, and the number of suppliers is also chosen from the range of 20 to 50, which is practically reasonable.

The first *N-K-M* mapping net produces coarse solutions for a given input, and the possible solutions are generated from the coarse solutions. For example, if the first mapping net produces distribution solution (12, 7.5) from a plant to certain destination, and if shipping quantities are required to be integers, then the shipping quantities would be (12, 8) and (12, 7). We check feasibility for both solutions, and consider only feasible solution. If there is no feasible solution, we find one feasible solution by incremental moving next feasible solution. In this case the least cost incremental moving is considered. For example, (12, 8) and (12, 7) are infeasible, we consider feasible solution by incremental moving such as (13, 7), (13, 8), (13, 9), (11, 6), (11, 7), and (11, 8). We select the minimum solution among the generated 6 solutions. If there is no feasible solution among the generated feasible solutions, the same procedure is repeated using the generated feasible solution until obtaining feasible-best solutions. The generated feasible solutions are evaluated using the second neural net. The second mapping net produces possible supply quantities and demands using the generated solutions as inputs. Comparing the second network's outputs and the real current supply quantities and demands, and the best solution is determined by selecting the best matching supplying quantities and demands.

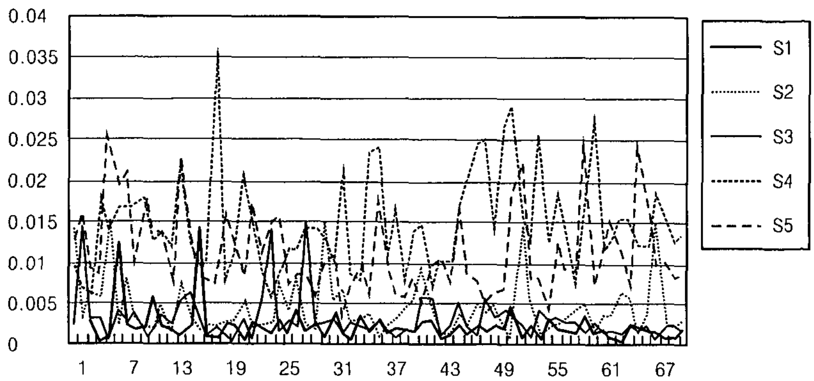

Figure 4.1 MSE between Optimal Solutions and *N−K−M* Mapping Solut ions

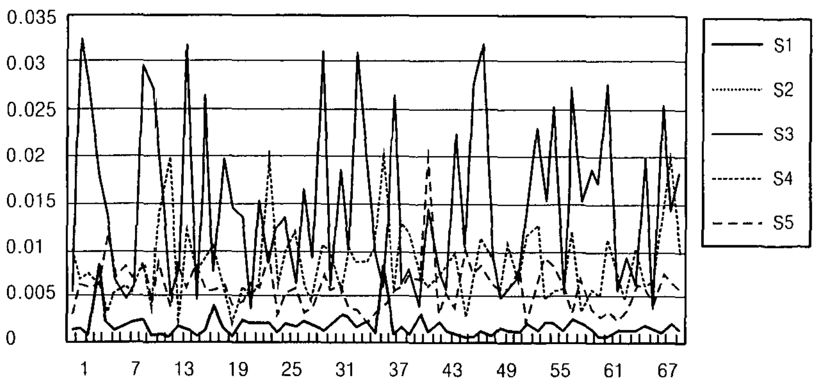

Figure 4.2 MSE between Optimal Solutions and *M−K−N* Mapping Solutions

Figure 4.1 and 4.2 show the *N-K-M* and *M-K-N* mapping capabilities for different costs. Mean Square Errors (MSE) between optimal solutions and mapping solutions are displayed for different supplies and demands. Each trajectory is obtained by averaging 20 different cases. Decomposition by different transportation costs gives a good mapping performance between demands/supplier and the optimal distribution quantities from plant to each distribution centers. Figure 4.3 shows the *N-K-M* and *M-K-N* mapping approximation without decomposition considering different cost functions. The decomposed mapping nets gives considerably better solution than that of the non-decomposed mapping nets. This shows that major disruption could be effectively overcome by execution pattern classification.

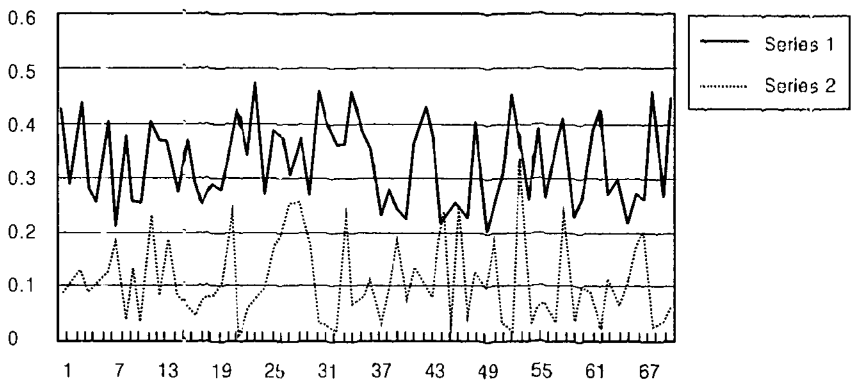

Figure 4.3 MSE between Optimal Solutions and *N–K–M* & *M–K–N* Mapping Solutions

Table 1 shows the possible solutions generated by the first neural mapping. The overall computation is dependent on the number of possible solutions. The more accurate is the first neural mappings, the less number of solution candidates is considered. In our experiment, real data are used for generating possible cases, and all neural mapping values are re-estimated based on approximate rounding-off considering real situations. However, if the output values are in near middle of two distinct adjacent numbers, two adjacent numbers are considered simultaneously. The decomposed first mapping produces considerably good solution, which requires less computation. Output-Mean Absolute Deviation (O-MAD) denotes mean absolute deviation from optimal solutions for the first mapping outputs, expressed by $\sum_i \frac{|x_i^* - output_i|}{x_i^*} \Big/ N$.

Table 1. Comparison of Optimality for the first N−K−M Net

	Training Data			Test Data		
Problem Type	O-MAD	R-MAD	Correct Ratio	O-MAD	R-MAD	Correct Ratio
S 1	0.00132	0	100 %	0.002425	96.2 %	99.2 %
S2	0.003942	0	100 %	0.008254	94.1 %	98.1 %
S3	0.003762	0	100 %	0.006283	97.2 %	99.2 %
S4	0.004044	0	100 %	0.023919	95.4 %	97.4 %
S5	0.001914	0	100 %	0.003789	94.4 %	98.4 %
S6	0.00601	0	100 %	0.030586	95.7 %	97.7 %

Rounded-Mean Absolute Deviation (R-MAD) means the first mapping solutions are rounded based on incremental integers. The correct ratio denotes correct identification ratio of optimal solution after rounded solutions. In training phase, perfect optimal solutions could be found while at testing phase there are few error ratios. However, the optimal solutions could be obtained effectively in the set of generated possible multiple solutions by evaluating rounded solutions.

Table 2 also demonstrates that the proposed two-complement reverse mappings could play an important role in finding the best solution. The first neural mapping produces a coarse solution, while the second net could be utilized for selecting the fine solution. Optimal solutions are compared with the proposed decomposed-reverse mapping solutions. Two reverse mappings could find optimal solutions for most cases. This illustrates that minor disruptions could be handled by the generalization capability of the two reverse mappings.

Table 2. Comparison of Optimality for the second *M−K−N* and Two Reverse *N−K−M*/*M−K−N* mapping Net

	M-K-N Mappings (Testing Data)			Two Reverse Mappings (Testing Data)		
Problem Type	O-MAD	R-MAD	Correct Ratio	O-MAD	R-MAD	Correct Ratio
S 1	0.008222	0.006167	98.2 %	0.003986	0	100 %
S2	0.006437	0.004828	99.1 %	0.003117	0.004828	99.9 %
S3	0.021468	0.016101	98.5 %	0.000421	0	100 %
S4	0.025506	1.643	99.4 %	0.002354	0	100 %
S5	0.030439	0.019129	97.6 %	0.004761	0	100 %
S6	0.030439	0.022829	98.3 %	0.004762	0.022829	99.9 %

The above results show major and minor disruption cases considering different transportation costs as well as different demand and supply quantities. The transportation cost could be proportional to deviational quantities from original optimal plans. Different major disruption cases are generated using different transportation costs, and different minor disruption cases are also generated using different values of supplies and demands

5. CONCLUSION

In this paper, an attempt is given to establish robust framework for real-time planning and execution control. Necessary definitions and concepts are presented to describe real-time operational control in connection with disruption recovery. A general mathematical framework is presented and its application to a simple distribution problem is also illustrated. A learning paradigm is designed using decomposed-reverse neural mappings, and it demonstrates how the decomposed-reverse mappings could be applied for solving disruption problem using generalization capability in real-time.

Optimal solutions for different cost functions, interpreted as local disruption, could be captured by the proposed decomposed two neural mappings. The first neural mapping produces a coarse solution while the second neural mapping is utilized for finding a fine solution. The complementary relationship between the two reverse neural mappings could be utilized efficiently for solving the disruption problems.

We designed robust planning and real time execution control simultaneously in order to reduce the gap between planning and execution. At the planning stage, focus is given on making robust solutions by considering various disruption scenarios and also considering execution pattern having general problem solving capability. As an execution pattern two complementarily reverse mappings are designed for increasing generalization capability, which is suitable for solving local plan change. A lot of planning tasks are required for establishing robust plan based on disruption decomposition. However, most works are based on off-line stage, and an efficient on-line execution is possible using pre-established knowledge.

Even though the proposed methodology is applied to the limited transportation problems, general extensions could be possible by applying the proposed robust planning principle and other hybrid heuristics. However, depending on the problem, different learning paradigms could be designed using different mapping or algorithm structures. Other hybrid learning heuristics could be developed for solving more complex problems, which is our future research in connection with robust disruption prediction. Integrated real-time distribution, production planning, and execution control in connection with various disruption cases will be our further research topic.

REFERENCES

[1] Averbakh, I. and Oded Berman "Algorithms for the robust 1-centered problem on a tree," *European Journal of Operational Research* 123 (2000), 292-302.

[2] Arguello, M. F., J. F. Bard, and Gang Yu, "A GRASP for Aircraft Routing in Response to Groundings and Delays," *Journal of Combinatorial Optimization*, (1997), 211-228.

[3] Arguello, M. F., J. F. Bard, and Gang Yu, "Models ad Methods for Managing Airline Irregular Operations Aircraft Routing," in Operations Research in Airline Industry: Gang Yu (Ed), Kluwer Academic Publisher, Boston, MA, (1997), 1-45,

[4] Cao, J. and A. Kanafani, "Real-Time Decision Support for Integration of Airline Flight Cancellation and Delay Part I: Mathematical Formulations," *Transportation Planning and Technology* 20 (1997), 183-199.

[5] Cao, J. and A. Kanafani, "Real-Time Decision Support for Integration of Airline Flight Cancellation and Delay Part II: Algorithm and Computational Experiments," *Transportation Planning and Technology* 20 (1997), 201-217.

[6] Carrvilla, M. A. and J. P. Sousa, "Hierarchical Production Planning in a Make-to-Order Company: Case Study," *European Journal of Operational Research* 86 (1996), 43-56

[7] Christian H. Timple and Joseph Kallrah, "Optimal Planning in Large Multi-Site Production Networks," *European Journal of Operational Research* 126 (2000), 422-435

[8] Desmond D'Souza, Objects, Components, & Frameworks with UML, Addison-Wesley, Longman, Incorporated, (1998), 1-816.

[9] Euwe, Mark J., Paul A.L. Jansen Christian, and T. H. Veldkamp, "The

Value of Rescheduling Functionality within Standard MRP Packages," *Production Planning and Control* 9, 4(1998), 328-334.

[10] Kouvelis, P., and G. Yu, *Discrete Optimization and its Applications*, Kluwer Academic Publishers, Dordrecht, 1997.

[11] Kouvelis, P., and Gang Yu, *Robust Discrete Optimization and Its Applications*, Kluwer Academic Publishers, 1997.

[12] Lapedes, A. and R. Farber, Nonlinear Signal Processing Using Neural Networks: Prediction and System Modelling, *Preprint LA-UR-87-2662*, 1987.

[13] Lapedes, A. and R. Farber, How Neural Nets Work, *Technical Report Los Almos National Laboratory*, (January 1988), 331-346,.

[14] Lee, I., and M.J. Shaw, "A Neural-net Approach to Real Time Flow-shop Sequencing," Computers & Industrial Engineering 38 (2000), 125-147

[15] Luo, S. and G. Yu, "On the Airline Schedule Perturbation Problem Caused By the Ground Delay Program," Transportation Science 31, 4(1997), 298-311.

[16] Lynn Ling X Li, "Proposing an Architectural Framework of a Hybrid Knowledge-based System for Production Rescheduling," *Expert Systems* 16, 4 (1999), 273-279.

[17] Narendra, K. S. and K. Parthasarathy, Identification and Control of Dynamical Systems using Neural Networks, *IEEE. Transactions on Neural Networks* 1 (1990), 4-27.

[18] Rakshit, A., N. Krishnamurthy, and G. Yu, "Systems Operations Advisor: A Real-Time Decision Support System for Managing Airline Operations at United Airlines," *Interface* 26, 2 (1996), 50-58.

[19] Thengvall, Benjamin G., Jonathan F. Bard and Gang Yu, "Balancing User Preferences for Aircraft Schedule Recovery during Irregular Operations," *IIE Transactions* 32, (2000), 181-193.

[20] Yan, S. and D. Yang, "Decision Support Framework for Handling Schedule Perturbations," *Transportation Research. Part b: Methodology* 30, 6 (1996), 405-419.

[21] Yoshtaka, Kuwata and Hiroki Oohama, "A case study of a real-time problem solving strategy in an air traffic control problems," *Expert Systems with Applications* 12, 1 (1997), 71-79.

[22] Yu, G. "A General Advanced Integrated Real-Time Decision Support Systems (AIRDSS) Framework for Real-Time Operations Control," White Paper for CALEB Technologies Corp. (2000).

[23] Wang, G. "An Adaptive Hybrid Neural Network Approach to Prediction of Nonstationary Processes," *Hybrid Intelligent System Applications*, Edited by Dr. Jay Liebowitz, Cognizant Communication Corporation, (1996), 140-161.