ADAPTIVE GOAL-SETTING IN TASKS WITH MULTIPLE CRITERIA

by

LEONID CHARNY

M.Sc. in Mechanics and Applied Mathematics (cum laude)
Moscow University, Moscow, USSR
(1975)

Submitted to the Department of
Mechanical Engineering
in Partial Fulfillment of the Requirements
for the Degree of

DOCTOR OF SCIENCE in MECHANICAL ENGINEERING

at the
Massachusetts Institute of Technology
July 1989

Signature of Author . . . [illegible] . . .
Department of Mechanical Engineering
July 1989

Certified by . . . [illegible] . . .
Thomas B. Sheridan
Thesis Supervisor

Accepted by . . . .
Ain A. Sonin
Chairman, Department Committee

Adaptive Goal Setting in Tasks with Multiple Criteria

Leonid Charny

Submitted to the Department of Mechanical Engineering
on July 24, 1989 in partial fulfillment of the
requirements for the Degree of Doctor of Science in
Mechanical Engineering

ABSTRACT

An interactive decision-aiding technique was developed to assist a human operator in selecting a satisfactory goal in a control task. This technique is also applicable for solving traditional multiple criteria decision making problems.

An *attainability set* of a system was introduced as a set of all goals that the system can potentially achieve. A goal was defined as a collection of constraints on a desirable final state of the system. The goal was considered achieved as soon as all these constraints were satisfied. The notion of the attainability set was extended to the case of a system operating in the environment with perturbations. The goal selection problem was modeled as a decision problem of exploration of the attainability set.

The developed decision-aiding technique is based on dynamic approximation of the attainability set and permits the decision maker to scan the desired regions of the attainability set to select a goal that is satisfactory. The technique can be used as a tool for supervisory control of systems whose goals are characterized by conflicting and incommensurable criteria. The proposed technique is especially useful when neither the utility function nor the distribution of the perturbations is available.

The advantage of the proposed technique was demonstrated in experiments with human subjects, performed to compare the proposed technique with conventional control methods. A simple sixth order dynamical system was selected for experimental tasks that required operating the system at the limit of its capabilities. Performance of human subjects using the proposed technique was compared with their performance using conventional manual and automatic controls. Using a computer aid based on the technique developed in this thesis, the subjects achieved significantly better performance than when using traditional methods.

Thesis Committee: Prof. Thomas B. Sheridan (Chairman)
Prof. David C. Gossard
Prof. David H. Marks
Prof. Derek Rowell
Prof. Duvvuru Sriram

Acknowledgements

This work was supported by the NASA Ames Research Center under grant NCC 2-360, the Boeing Military Airplanes Corporation and by the Psychology Board of Naval Research under contract N0014-83-K-0193. I thank them for their financial support.

I would like to thank Professor Thomas B. Sheridan, my thesis adviser, for his continued help and encouragement. I also want to thank him for his patience and understanding of the unusual personal circumstances that coincided with my work on this thesis. I deeply appreciate all the personal help that he has given me in conjunction with these circumstances.

I am grateful to my committee members, Profs. David C. Gossard, David H. Marks, Derek Rowell and Duvvuru Sriram, for their assistance. I am especially thankful to Professor Marks for invigorating my research with his tough questions.

Special thanks to Dr. Mary Hornsby of Boeing Military Airplanes. My research collaboration with her was a truly enjoyable and enlightening experience.

I thank my diligent and patient subjects who put up with long and lonely hours interacting with my software: Chi Cheng, the Great Computer Keeper; Kan Chin, the Mac Wizard; Hari Das, who has made me realize how close the most exotic places of the world really are; and Jim Roseborough, the procreator of the most useful software in the lab. I also want to acknowledge other members of the MMSL: Fo Buzan, one of the few of us Real Americans in the lab; Sam and Mija Landsberger, the warmest and kindest couple; Max Mendel, the High Priest of Bayesianism who has converted the entire lab, myself included, to his religion; Jong Park, my successor as the IRIS Tsar; Juggy Raju, the true guru-in-residence and the most distinguished cook; Jie Ren, the only surviving Fuzzionist in the lab; and Wael Yared, who enhanced my perspective on various subjects. I am thankful to all MMSL members for their eagerness to help and for making life inside the MIT walls more pleasant.

I am deeply grateful to my extended family, especially Jacob, Nina, and Yadviga, who spared no effort in relieving me from many of my family duties so that I could devote my energy to this project.

My special gratitude goes to my brother Benjamin, who not only has been a loving and caring brother, but also has always exemplified to me the prowess and prodigious tenacity of a true scientist. It was my reunion with him after many years of strenuous and bitter struggle that catalyzed the wind-up of this thesis. As on numerous past occasions, my brother's help and advice were vital and enabled me to complete the thesis.

This work would never have been finished - and most likely never started - without the support of my wife Yelena. It was only with her patience, encouragement and loving care that I was able to endure the emotional roller-coaster of my MIT study combined with my public activity. Her contribution to my getting this degree is at least equal to mine. She and our son Isaac have made this whole endeavor even more worthwhile.

Table of Contents

Preface

This thesis consists of 7 chapters.

Chapter 1 describes the goal-control paradigm and its relation to the traditional multiple-criteria decision making problem; defines the attainability set; and discusses some general considerations concerning the design of decision aids.

Chapter 2 reviews the existing research in the area of multiple-criteria decision making.

Chapter 3 contains a formal description of the proposed decision-aiding technique for the goal-control paradigm, the Dynamic Range Tradeoff (DRT).

Chapter 4 discusses the implementation of the DRT human-computer interface and describes the software system developed for discrete static tasks.

Chapter 5 introduces a special class of systems, resource-sharing systems, and describes the implementation of the DRT technique for the goal-control of such systems.

Chapter 6 describes experiments dealing with control of an experimental system, conducted for the purpose of evaluating the DRT technique.

Chapter 7 contains conclusions; research contributions; and suggestions for further research.

Abbreviations

DM	- Decision Maker
MSS	- Maximum Satisficing Set
PCC	- Pareto Cluster Cell
RSS	- Resource-Sharing Systems

Notation

$\overline{1,N}$	- a set of integers from 1 to N
Ω_u	- set of admissible controls
ψ_{t_0,t^*}	- A-attainability set from time t_0 at time t^*
$\overline{\psi}_{t_0,t^*}$	- B-attainability set from time t_0 by time t^*
$\overline{\psi}_{t_0}$ or $\overline{\psi}_{t_0,\infty}$	- (I-) attainability set from time t_0
$\Xi_{l,r}$	- a set of terminal times t^* for which the A-attainability set at time t^* has a non-empty intersection with interval $[\,l,r\,]$
$\mathbf{R}^M$	- an M-dimensional space, or a set of real-valued k-tuples
$\mathbf{y}$	- (bold character) a point (vector) in $\mathbf{R}^M$ with coordinates y_i

Chapter 1

Goal Setting Paradigm

1.1 Introduction

We are considering a class of systems that are *goal* or *target oriented.* A system of this kind performs a task that is described by some goal. The goal is usually specified as a set of conditions on a desirable final state of the system. As soon as all these conditions are satisfied, the goal is considered achieved. The system gets the goal specifications from a human operator/decision-maker (DM). After accepting the goal, the system operates autonomously and independently of the human.

The human however can monitor the system's advancement towards the goal and can intervene if needed. The human's involvement in controlling such a system is best described by the supervisory control paradigm[58][59]. The major feature of a system in consideration is that during its operation the system constantly assesses its current state and uses some state feedback mechanism (algorithm) to generate actions necessary to attain its goal. A system of this kind is also *adaptive* in the sense that it attempts to generate corrective control actions to accomplish the goal, even in the case of unforeseen perturbations.

The corrective control might be able to compensate for some perturbations, such as small disturbances, inaccuracies of computation, etc. Any system, however, is limited in the ability to generate control actions. A large perturbation or an accumulation of small ones may be impossible to compensate with available controls. A system that has been subjected to such perturbations might *never* be able to attain its goal. The task either has to be abandoned or *another goal* must be given to the system. A human decision is usually needed to come up with a new goal.

The present work addresses the problem of selecting an initial attainable goal as well as a new goal when the old goal becomes unattainable. A decision-aiding technique described in this thesis helps the operator to explore different alternatives and make an informed decision as to which new goal to select.

The goal setting paradigm can be applied to a broad number of existing systems - from automatic controllers to AI problem-solving programs. This paradigm also closely relates to the Multiple Criteria Decision Making.

1.2 Goal Selection

Two questions need to be asked by a DM controlling any system:

- How to chose the task goal for the system?
- How to achieve that goal?

The second question in the case of dynamical systems is usually considered by control theory. The first question somehow is presumed to have been answered. In fact, this question is rarely even considered. The issue of how to choose a goal for a system is very important, however, because it is closely connected to the question of how to achieve this goal, or how to control the system.

The basic question - whether the system *can* accomplish its goal - is also rarely addressed in the research literature. In the meantime, every system has its internal limitations, that affect the range of goals that the system can actually achieve. For instance, the system's actuators can generate limited power, the resources for accomplishing the task might be limited, etc. However, the system conventionally is operated with such a margin of capabilities, that the issue of accomplishing the task is usually resolved before the task begins. Only when a system has to operate at the limit of its capabilities and the outcome of the task is critically important, then the issue of choosing a proper goal for the task becomes relevant.

Another situation, however, occurs when the system experiences unexpected perturbations, whose cumulative effect over time on the system could be much greater than it was anticipated at the beginning of the task. At some point during the execution of the task the problem might arise - whether to change the task and have the system do something different, or to scrap the task altogether. This problem can develop due to a gross deviation (as the result of the perturbations) of the system from the initially planned path. This is the situation when it needs to be decided whether the system has enough capability to continue the old task, potentially operating at its limits, or the task should be modified. It might also become evident at some point that the task cannot at all be accomplished as planned.

If the decision is to modify the original task, then the question emerges: how much does the original task need to be modified? In most cases it is desired that the original goal would be replaced with a satisfactory substitute. On the other hand, the new goal must be such that it would not immediately generate the same problems that have plagued the old one - i.e. the new goal must be achievable (or at least look achievable) at the onset.

The problem of selecting a goal for a system is inseparable from the issue of how the system is controlled while working to achieve the goal. It might also be that the system is unable to accomplish its goal not because of its intrinsic limitations, not due to the perturbations, but because it is simply poorly controlled. It is in the realm of control theory to come up with better and more robust controllers. In this thesis we will concentrate on the question of how to help the DM select the goal so that it is both satisfactory to her and in the same time is achievable by the system.

Before proceeding with formal definitions, let us consider several examples.

1.2.1 Examples

Manipulator.

An illustrative example of a system under consideration is a manipulator that automatically generates and executes a trajectory, moving from its initial configuration to the one that corresponds to a given target position of the end-effector.

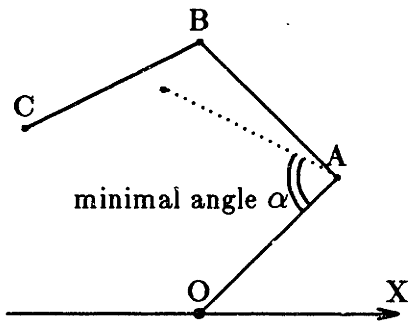


Figure 1.1. 2D 3-link Manipulator.

Consider a 3 degree of freedom manipulator that operates in the upper half-plane (Fig. 1.1) in 2D. A configuration of the manipulator is restricted due to the limitation on the values of the joint angles. The base joint angle $\angle$ AOX can range from $-180°$ to $180°$, while the angle between two adjacent links[1] cannot be greater than some minimal angle α: $\alpha \leq \angle$ BAO (or $\angle$ CBA).

1. All angles in this example are non-oriented angles, so that the angle between two adjacent links never exceeds $180°$.

One can compute a workspace of such a manipulator - i.e. a set of all reachable positions of the end-effector (Fig. 1.2). Given a target point from the workspace (e.g. Target 1), the manipulator would move along a trajectory that positions its end-effector in the target point. The trajectory can be generated by one of the path-generating algorithms.

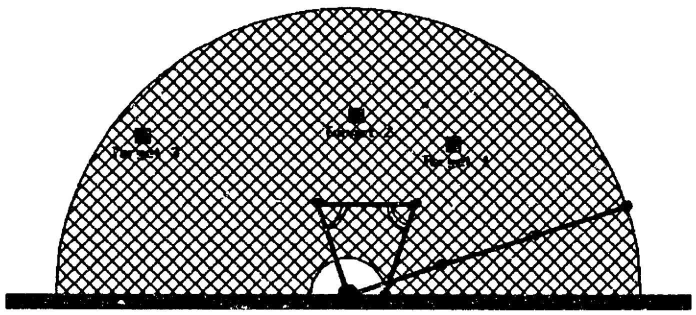


Figure 1.2. Workspace of a normal manipulator (the hatched region).

The outer half-circle boundary of the workspace has a radius equal to the length of a fully extended manipulator when $\angle CBA = \angle BAO = 180^\circ$ (the manipulator joints are labeled on Fig. 1.1). The inner half-circle has a radius equal to the distance from the base joint to the tip of the manipulator in a configuration when both angles between the adjacent links are at their minima α, when $\angle CBA = \angle BAO = \alpha$.

Suppose that a perturbation has manifested itself in the failure of the base joint actuator. After the perturbation has occurred, the base joint ceased to move and got stuck at some position, so that the base joint angle cannot be changed any longer. The workspace of what is now a semi-handicapped manipulator is different from the original workspace (Fig. 1.3). Point Target 1 no longer belongs to the workspace of the manipulator. If the original task was to move the end-effector of the manipulator to point Target 1, then this task can no longer be accomplished.

One of the options is to shut off the manipulator and to repair it. However, there may be additional considerations that require the original task to be accomplished, at least partially. In the latter case a new target point, that does belong to the reduced workspace, might be given to the manipulator.

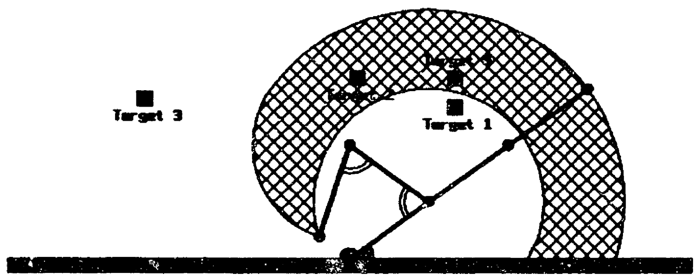


Figure 1.3. Workspace of a broken manipulator (the hatched region).

The lower boundary of the workspace is the arc centered at joint A and having a radius equal to the distance from joint A to the tip of the manipulator in a configuration when both angles between the adjacent links are at their minima α, when $\angle CBA = \angle BAO = \alpha$ (the manipulator joints are labeled on Fig. 1.1). The upper boundary of the workspace is a composition of two arcs. The right part of the boundary is the arc centered at joint A and having a radius of two fully extended links (AB+BC). The left part of the boundary is the arc centered at the rightmost position of link (AB) and having a radius equal to a length of one link.

The rationale for selecting a new target point depends on the original task. If the manipulator was performing spray painting, then the selection of Target 4 as a new goal might be acceptable. Target 4 does belong to the new workspace and is also in close proximity of the old goal Target 1. For some other task there might be no clear need for a proximity of the new goal to old one. Target 2 could be selected as a new goal instead. After a new target has been selected, the task can be resumed, and the DM will be sure that the new task can be accomplished (at least under the present circumstances).

Computer-Aided Design.

A different kind of an application can be taken from the computer-aided design area. In a course of a conceptual design[57], a designer sequentially adds constraints on the designed object. The computer verifies the consistency of those constraints and detects it if an incoming constraint is inconsistent with the previous ones. If an incoming constraint is inconsistent with all others, then there is an impasse. The designer has to make a decision either to change the new constraint or to change some of the earlier entered constraints. Changing a constraint frequently amounts to changing a value of one of the design parameters.

In order to make an informed decision, the designer needs to know how changing values of these parameters affects the consistency of all accumulated constraints. The set of all consistent design parameters is an analogue of the workspace of the manipulator example.

As an example[2] consider a design problem: find unknown values x and y that satisfy a set of constraints. The designer sequentially adds new constraints and the computer checks whether a new constraint is consistent with all the previous ones.

Suppose that at some point of the design process there are only two constraints:

$$\begin{cases} x + y \geq 4 \\ x^2 - y \leq 15 \end{cases} \tag{1.1}$$

There are infinite number of solutions (x,y) that satisfy them. Suppose the designer enters a new constraint:

$$y \leq 0 \tag{1.2}$$

It is easy to verify that this constraint is incompatible with (1.1). Something has to be done to correct the impasse.

Suppose that it is possible to change only one or both constraints (1.1) by changing their right-hand sides.[3] One can consider the following problem: find a and b for which constraints (1.3) are consistent:

2. This problem is an artificial one. Its only purpose it to demonstrate what is meant by a set of all consistent design parameters.
3. These are the so-called soft constraints, while (1.2) is the hard constraint.

$$\begin{cases} x + y \geq a \\ x^2 - y \leq b \\ y \leq 0 \end{cases} \tag{1.3}$$

Parameters a and b can be viewed as parameters of some goal in the space of target parameters (a,b)[4]. The old goal was:

$$a = 4, \quad b = 15 \tag{1.4}$$

The old goal was consistent with the *old* set of constraints (1.5):

$$\begin{cases} x + y \geq a \\ x^2 - y \leq b \end{cases} \tag{1.5}$$

but it is no longer consistent with the new set of constraints (1.3).

If a different constraint were entered instead of (1.2), and if that constraint were consistent with (1.5) and goal (1.4), then the designer could have continued adding more constraints to (1.5) without changing the goal. The addition of (1.2) to (1.5) makes goal (1.4) no longer attainable. This constitutes (as far as the computer is concerned) an irreparable perturbation.

To select a new goal, the designer needs to know for which values of (a,b) inequalities (1.3) are consistent. All such pairs (a,b) form the *feasibility* set of

4. The notion of a goal that is used here is different from the notion of the ***design goal***. It would, however, be appropriate to interpret (a,b) as one of the ***sub-goals*** of the design process.

parameters of the problem (1.3). This set is equivalent to the workspace in the manipulator example.

A feasibility set of constraints (1.5) is the entire 2D plane $(a,b) \in \mathbf{R}^2$. The feasibility set of (1.3) is described by the following relations:

$$a^2 \leq b, \quad \text{when } a \geq 0 \tag{1.6}$$

$$b \geq 0, \quad \text{when } a < 0$$

The old goal (a=4, b=15) is located outside of the new feasibility set (affected by the "perturbation (1.2)) (Fig. 1.4). A new goal (a′,b′) (e.g. (a′=4,b′=17)) could be selected from the set defined by (1.6). Having modified the goal, the designer can resume adding new constraints until another "impasse" occurs.

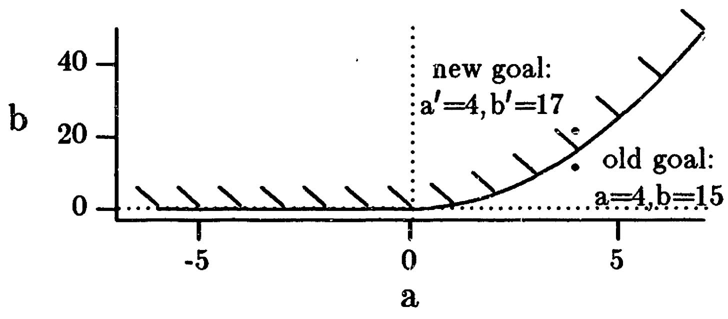


Figure 1.4. Feasibility set (1.6) of inequalities (1.3) (the shaded area).

1.3 Attainability Set

The key element of the presented examples is the computation of a workspace or a feasibility set of a problem. Determining this set is critical for selecting the system's goal and deciding how to replace it if the previously selected one becomes unattainable. Both the workspace and the feasibility set are particular cases of what we will call hereafter the *attainability* set.

We define attainability using the concepts of ***task goal***, and ***perturbations.***

1.3.1 Task Goal

Let us envision a system that evolves in time while performing some task. We assume that the system performance during the task is characterized by several scalar indicators G_i $(i = \overline{1,M})$, which can be calculated at every moment of time. In the most general case the values of these indicators at any instant of time are functions of time, the system's initial state and the entire history of the system's evolution from the beginning up until that instant of time.

The purpose of the task is to achieve some *goal*. The goal is defined by a set of acceptable, or *target* levels, $(g_i^-\ ,\ g_i^+)$, $(i = \overline{1,M})$ that are specified for each of the system's indicators. A goal is achieved as soon as the values of all indicators fall within these levels, i.e. when

$$g_i^- \leq G_i \leq g_i^+ \tag{1.7}$$

for all $i=\overline{1,M}$

The goals considered in this thesis can be characterized as *terminal* goals. The goal is accomplished at the *first instant* of time when the goal conditions (1.7) become true. The goal cannot be something that needs to maintained. For example, a driving task with a goal of *staying within a lane* is not a legitimate task in this interpretation.

A goal specifies a region (a hyper-parallelepiped) in the space of the target function G_i. This region will be called the ***goal box.*** This box can obviously degenerate into a point when $g_i^- = g_i^+$.[5]

The range of goals that the system can reach depends on the system's ability to generate control actions necessary for achieving the goal and on the perturbations the system is subjected to.

1.3.2 Perturbations

The system strives to achieve the task goal in the environment with perturbations. Two types of perturbations are considered in this paradigm: the planned and the unplanned perturbations.

1. The planned perturbations $\mathbf{p}(t)$ belong to a certain, known in advance, set Ω_p. These perturbations are similar to noise terms in control systems. Usually these perturbations are small in some sense, but their cumulative effect on the system over time can be potentially significant. Different models could be put forth to account for their effect on the system. Since that model might not accurately predict the future perturbations, replanning, or selecting another goal might

5. In fact, a goal (g_i^-, g_i^+) in (1.7) can be interpreted as a single point in $\mathbf{R}^{2M}$.

become necessary as the result.

2. The perturbations of the second type are *unplanned perturbations.* It is presumed here that these perturbations occur only at discrete instants of time, but each time they occur they make a very significant impact on the system and its ability to do the task. A breakdown of a system's component or a major change of the environment in which the system operates, are the examples of these perturbations. A perturbation of this kind at time t_1 can be viewed as an instantaneous modifier of the current state of the system, or of the set of admissible control actions, or of the set of admissible trajectories of the system's state over time. These perturbations are not planned for, in the sense that the task goal is selected without expecting these perturbations to occur. However, as they occur, replanning, or selecting another goal might become necessary as the result.

These two types of perturbations are very different in the eyes of the DM. In contemplating the course of actions and selecting the task goal, the DM considers only the planned perturbations. However both type of perturbations affect the system's ability to achieve the goal.

Consider an example of driving a car (the system) to some destination. The goal can be formulated as the desired destination together with the desired range of the times that it would take the car to get to that destination.

A planned perturbation could be the traffic congestion that can slow down the car's advancement towards the goal. This perturbation is to be expected and it can be modeled. The goal is selected with this perturbation in mind. However the traffic congestion can be so bad, that after a while it might become clear, that the goal of getting to the desired destination in the desired time is no longer attainable. Moreover, it might be impossible to accomplish the goal even if the rest of the road is clear from traffic. The car might already have lost so much time, that it cannot accomplish the goal even if it is driven with its maximum possible speed.

A unplanned perturbation could be a road accident making the road completely impassable for a long period of time. Or the road might get closed for repair. Or, for example, the car might get a flat tire, etc. The important thing about the unplanned perturbations is that they are dealt with only as they occur, not at the time when the goal is selected. In many cases the unplanned perturbations might actually not occur at all.

The failure of the actuator in the manipulator example is a manifestation of the unplanned perturbation. The inaccuracy of the positioning and measurement devices are planned perturbations. So are the computational errors of the control algorithm. In the meantime, all perturbations in the CAD/CAM example are unplanned.

The planned perturbations are considered at the time when the goal is selected. The nature of these perturbations depends on the circumstances of the system and the task. However these perturbations are always expected to be present during the task.

The decision problem that the DM faces is to select a goal for the system. In selecting the goal the DM must always be conscious about the question whether that goal can be reached at all. The reachability, or attainability of the goal depends on two factors: the controls that the system generates and the perturbations that obtain during the system advancement towards the goal.

There can be different schemes to account for the effect of the future perturbations when selecting the goal for a task. The following planning scheme is suggested and followed in this thesis: Since the perturbation are not known in advance, during the planning stage they are approximated by a constant. The full treatment of the issue of how to choose that constant is outside of the scope of this thesis.[6]

For example if the values of perturbations are known to be within a certain interval, then the middle value or the extreme values of that interval can be taken as approximating constant. Or if the distribution of the values of the perturbations is known, then the expected value of this distribution can be taken as the constant approximating the perturbations.

The main idea of the goal planning process followed in this thesis can be formulated as follows: since the range of attainable goals depends on both the control actions and the perturbations that obtain in the future, making a deterministic assumption about the future perturbations removes uncertainty from the problem of finding the set of attainable goals. As soon as the perturbations are

6. This problem largely depends on the model of perturbations that is used by the DM.

approximated as a constant, the set of attainable goals becomes a function of control actions only.[7]

To summarize the role of the perturbations in the proposed goal setting paradigm, it can be said that the perturbations are the main reason why a once selected and attainable goal might become unattainable sometimes later. The reason why the perturbations are subdivided into planned and unplanned is due to the fact that only the *planned* perturbations are accounted for in the computation of the set of attainable goals. At the same time, perturbations of both types can make a goal unattainable. Selecting an appropriate class of planned perturbations therefore is a part of the system modeling process.

Another reason why a goal might become unattainable are some *bad control decisions.* Even if the system has a built-in mechanism that helps it to cope with the perturbations (e.g. an automatic feedback controller), it might not be able to cope with the perturbations. Or the system might be manually controlled by the DM, and it is known that the humans are not optimal controllers at all.

Even in the case when at every instant of time the system does attempt to generate a "good" control action with the purpose of achieving the task goal, there might be problems. The control actions that the system can come up with

7. The problem of finding a set of attainable goals as a function of only controls actions is unavoidable regardless of the approach. For instance, if the probability distribution of the perturbations is used during the planning stage, the question of *what goals are attainable given a particular realization of the perturbations* is equivalent to the one we address.

at any instant of time are limited, and it might be physically impossible to bring about an appropriate control action, even if the control algorithm correctly suggests one.

1.3.3 Attainability

Two questions can be posed in this context:

- Given a goal, would the system be able to accomplish the task and achieve that goal, if there were no perturbations at all?

- What goals can the system accomplish in the absence of the perturbations, beginning from its present state?

The latter question is obviously more general than the former. Knowing the answer to the latter question it would be easier to resolve the former. The notion of the *attainability set* is introduced here to address the second question. Simply speaking, the attainability set is a set that contains all accomplishable goals.

Here are the formal definitions:

We consider a system that evolves in time. The system's evolution is described by (1.8):

$$\begin{aligned} &\mathbf{x}(t_1) = \mathbf{H}(t_0, t_1, \mathbf{x}(t_0), \mathbf{u}_{[t_0,t_1]}, \mathbf{p}_{[t_0,t_1]}) \\ &\mathbf{x}(t) \in \Omega_x, \quad \mathbf{p}(t) \in \Omega_p, \quad \mathbf{u}(t) \in \Omega_u \end{aligned} \tag{1.8}$$

Where:

t_0	is a given initial time;

t_1	is current time ($t_1 \geq t_0$);
t	is any time ($t \geq t_0$);
$\mathbf{x}(t_0) \in \mathbf{R}^n$	is the initial system state at time t_0 ($n \times 1$ real-valued vector);
$\mathbf{x}(t_1) \in \mathbf{R}^n$	is the system state at time t_1 ($n \times 1$ real-valued vector);
$\mathbf{u}_{[t_0,t_1]}$	is a vector-function of the entire control during the period from t_0 until t_1;
$\mathbf{p}_{[t_0,t_1]}$	is the entire vector-function of perturbations during the period from t_0 until t_1;
$\mathbf{u}(t) \in \mathbf{R}^k$	is a control vector at time t ($k \times 1$ real-valued vector);
Ω_u	a set of admissible values of the control vector-function of t;
$\mathbf{p}(t) \in \mathbf{R}^r$	is a value of perturbation at a particular time t ($r \times 1$ real-valued vector);
Ω_p	a set of possible perturbations as a vector-function of t.
Ω_x	a set of admissible trajectories in the system's state space.

The system is performing some task that is determined by the task goal. The goal is specified by a set of M pairs of real numbers (g_i^-, g_i^+), $i=\overline{1,M}$. The goal is achieved at the first instant of time $t^* \geq t_0$ when the following M conditions (1.9) hold:

$$\begin{cases} g_1^- \leq G_1(t_0, t^*, \mathbf{x}(t_0), \mathbf{u}_{[t_0,t^*]}, \mathbf{p}_{[t_0,t^*]}) \leq g_1^+ \\ g_2^- \leq G_2(t_0, t^*, \mathbf{x}(t_0), \mathbf{u}_{[t_0,t^*]}, \mathbf{p}_{[t_0,t^*]}) \leq g_2^+ \\ \dots \\ g_M^- \leq G_M(t_0, t^*, \mathbf{x}(t_0), \mathbf{u}_{[t_0,t^*]}, \mathbf{p}_{[t_0,t^*]}) \leq g_M^+ \end{cases} \tag{1.9}$$

Where:

t^*	is an *unknown* terminal time;
$\mathbf{u}_{[t_0,t^*]}$	is a vector of the entire control functions during the period from t_0 until t^*;
$\mathbf{p}_{[t_0,t_1]}$	is the entire history of perturbations from t_0 until t^*.

Functions G_i are called the indicators of the task, or the *target functions.* They are the integral part of the system/task description. The goal $(\mathbf{g}^-,\mathbf{g}^+)$ is specified at the beginning of the task, i.e. at time t_0.

The system might use some type of a state feedback algorithm to select its control action[8] $\mathbf{u}(t) \in \Omega_u$ so that it can achieve the task goal (1.9) at some finite terminal time ($t^* < \infty$). The following information is available at any moment of time t_1 for making the control decision: the current state of the system, the entire history of the system's evolution, and

$\mathbf{u}_{[t_0,t_1]}(t)$ - the obtained system's control function from the beginning,

$\mathbf{p}_{[t_0,t_1]}(t)$ - the encountered perturbation from the beginning.

Suppose that we interested in selecting a goal for the system at some time t_1 ($t_0 \le t_1$). To do the planning we first transform the system using the approximation of the future perturbations with a constant. In other words, we assume that the perturbation terms in (1.8) and (1.9) can be replaced with

$$\mathbf{p}_{t_1,[t_0,t^*]} = \begin{cases} \mathbf{p}_{[t_0,t_1]}(t) & \text{for } t_0 \le t \le t_1 \\ \text{const} & \text{for } t > t_1 \end{cases} \tag{1.10}$$

In other words, the approximation of the perturbation, $\mathbf{p}_{t_1,[t_0,t^*]}$ coincides with already obtained perturbations, $\mathbf{p}_{[t_0,t^*]}$ up until t_1 and is a constant in the future,

8. The system's state is a vector function of time, so is the system control. A singular form for describing the state and the control of the system is used to refer to the entire state vector and the vector of controls.

i.e. after t_1.

The approximation (1.10) is used for defining the attainability set of the system **from** time t_1.

Definition: The **A-attainability set** of system (1.8) executing task (1.9) **from time** t_1 **at time** t^* is set ψ_{t_1,t^*} in the M-dimensional space $\mathbf{R}^M$ that is defined in (1.11):

$$\text{for any member of this set: } \mathbf{y} = (y_1, y_2, \ldots, y_M)^T \in \psi_{t_1,t^*} \subseteq \mathbf{R}^M$$
$$\text{there exists control } \mathbf{u}^{(y)} \in \Omega_u \text{, such that} \qquad (1.11)$$
$$\begin{cases} \mathbf{u}^{(y)}(t) = \mathbf{u}_{[t_0,t_1]}(t) \text{ for } t_0 \leq t \leq t_1 \\ y_i = G_i(t_0, t^*, \mathbf{x}(t_0), \mathbf{u}^{(y)}_{[t_0,t^*]}, \mathbf{p}_{t_1,[t_0,t^*]}), \quad \text{for all } i=\overline{1,M} \end{cases}$$

where $\mathbf{p}_{t_1,[t_0,t^*]}$ is a vector function of the perturbations which coincides on the time interval $[t_0,t_1]$ with the perturbations that have obtained from t_0 until t_1, and is constant from t_1 until t^*.

According to this definition, the A-attainability set of the system/task is a set of all possible values of the target functions G_i at time t^*, assuming that no perturbations occur in the future from t_1 until t^*, and everything relevant about the system is known at t_1 and before. Time t_1 thus is the origination time, while t^* is the terminal time.

Two more definitions:

Definition: The **B-attainability set** of system (1.8) executing task (1.9) **from time** t_1 **by time** t^* is a union of all A-attainability sets from t_1 at times before t^*:

$$\overline{\psi}_{t_1,t^*} = \bigcup_{\substack{t \\ t_1 \leq t \leq t^*}} \psi_{t_1,t} \tag{1.12}$$

Definition: The **I-attainability set,** or simply the ***attainability set*** of system (1.8) executing task (1.9) **from time** t_1 is a union of all A- (or, which is the same, of all B-) attainability sets from t_1 *at* all times:

$$\overline{\psi}_{t_1} = \overline{\psi}_{t_1,\infty} = \bigcup_{\substack{t \\ t_1 \leq t \leq \infty}} \psi_{t_1,t} \tag{1.13}$$

Let's consider an example:

The system in consideration is a unit mass moving along a single dimension and controlled by a force. The mass is also subjected to some exogenous force perturbations. The system is described by an ordinary differential equation:

$$\ddot{x}(t) = u(t) + p(t) \tag{1.14}$$

where x (t) is the position of the mass at time t, u(t) is the control at time t, and p(t) is the perturbation.

The set of admissible controls is:

$$u(t) \in \Omega_u \quad \text{iff} \quad |u(t)| \leq 10 \tag{1.15}$$

and the set of possible perturbations is:

$$p(t) \in \Omega_p \quad \text{iff} \quad 19 \leq |p(t)| \leq 21 \tag{1.16}$$

The initial state of the system at time t_0=0 is

$$x(0) = \dot{x}(0) = 0$$

Let's consider a task that is specified by a goal on the mass' position and velocity. The goal functions of the system/tasks are:[9]

$$\begin{aligned} G_1(t) &= x(t) \\ G_2(t) &= \dot{x}(t) \end{aligned} \tag{1.17}$$

and the the goal is achieved as soon as

$$\begin{cases} g_1^- \leq G_1(t^*) \leq g_1^+ \\ g_2^- \leq G_2(t^*) \leq g_2^+ \end{cases} \tag{1.18}$$

To determine the A-attainability set ψ_{0,t^*} of this system from time $t_0=0$ at some time t^* we first select a constant that approximates the perturbation. Based on (1.16) we select this constant as the middle point of the interval of possible values of the perturbation, i.e. 20.

The system's evolution (1.14) can be transformed by this approximation into (1.19):

$$\ddot{x}(t) = u(t) + 20 \tag{1.19}$$

The A-attainability set of the system from time 0 at time t^* is a set in $\mathbf{R}^2$ of all pairs of the values $(G_1(t^*), G_2(t^*))$ where the values of the target functions G_i are calculated assuming approximation (1.19).

Integrating equations of motion (1.19) from $t=0$ to $t=t^*$, we get:

9. The target functions in this example are equal to the system's state variables. It happened only by coincidence, and this fact plays no role in the computation of the attainability sets for this example.

$$\begin{cases} G_1(t^*) = \int_0^{t^*}\int_0^{t} u(\tau)d\tau dt + 10(t^*)^2 = \int_0^{t^*} (t^* - t)u(t)dt + 10(t^*)^2 \\ G_2(t^*) = \int_0^{t^*} u(t)dt + 20t^* \end{cases} \tag{1.20}$$

From (1.20) and (1.15) it follows that

$$10t^* \le G_2(t^*) \le 30t^* \tag{1.21}$$

To determine the attainability set of the problem at time t^*, let's take some value of G_2 from the interval (1.21). Using the following result (1.22) which can be easily proven using the variational calculus:

$$\text{if } \int_0^{t^*} u(t)dt = y \text{ and } |u(t)| \le U \text{ then:}$$
$$U\left(\left(\frac{t^*+\frac{y}{U}}{2}\right)^2 - \frac{(t^*)^2}{2}\right) \le \int_0^{t^*} tu(t)dt \le U\left(\frac{(t^*)^2}{2} - \left(\frac{t^*-\frac{y}{U}}{2}\right)^2\right) \tag{1.22}$$

we get from (1.20)

$$\begin{cases} 0.025(G_2(t^*))^2-0.5t^*G_2(t^*)+7.5(t^*)^2 \le G_1(t^*) \le -0.025(G_2(t^*))^2+1.5t^*G_2(t^*)-7.5(t^*)^2 \\ 10t^* \le G_2(t^*) \le 30t^* \end{cases} \tag{1.23}$$

The A-attainability set, ψ_{t_0,t^*} described by (1.23) is shown on Fig. 1.5 for $t^*=5$ and $t^*=10$. Each attainability set is confounded inside of two parabolas plotted on these figures.

The expression (1.24) for the B-attainability set can be obtained from (1.23). It is easy to show that this set is contained between two parabolas that are the *envelope* curves for the A-attainability sets:

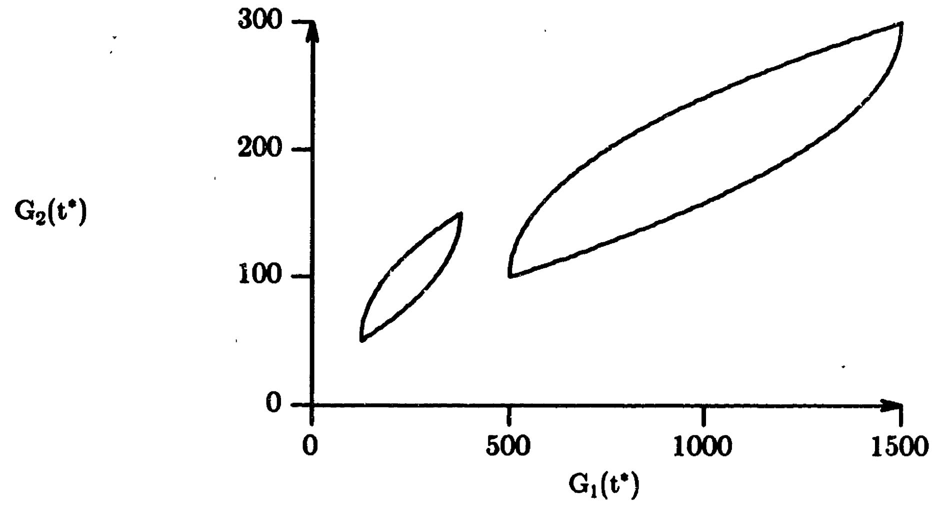


Figure 1.5. A-attainability sets.

The smaller curve confines the A-attainability set $\psi_{0,5}$ for $t^* = 5$,

the bigger curve confines the A-attainability set $\psi_{0,10}$ for $t^* = 10$.

$$\begin{cases} \frac{1}{60}(G_2(t))^2 \le G_1(t) \le \begin{cases} \le 0.05(G_2(t))^2 & \text{for } 0 \le G_2(t) \le 10t^* \\ \le -0.025(G_2(t))^2 + 1.5t^*G_2(t) - 7.5(t^*)^2 & \text{for } 10t^* \le G_2(t) \le 30t^* \end{cases} \\ 10t^* \le G_2(t) \le 30t^* \\ 0 \le t \le t^* \end{cases} \qquad (1.24)$$

The B-attainability set $\bar{\psi}_{0,10}$ is plotted on Fig. 1.6 for $t^* = 10$. The set is confined between two outer curves. The inner curves confine $\psi_{0,5}$ and $\psi_{0,10}$ respectively (the same sets as displayed on Fig. 1.5).

The attainability set of the system is a set between two envelopes of all A-attainability sets:

$$\frac{1}{60}(G_2(t))^2 \le G_1(t) \le \ 0.05(G_2(t))^2$$

This set is shown on Fig. 1.7.

To summarize the distinction between A-, B- and (I-) attainability sets: the A-attainability set spans values of the target functions at some given time, the B-

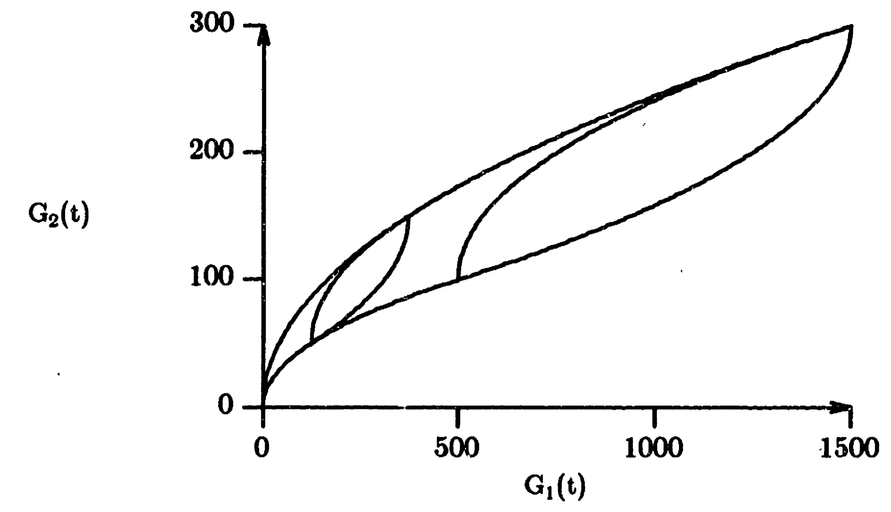


Figure 1.6. B-attainability set.

The outer curve confines $\bar{\psi}_{0,10}$, the B-attainability set.

The inner curves confine the A-attainability sets $\psi_{0,5}$ and $\psi_{0,10}$.

attainability set spans values of the target function for a period of time, the (I-) attainability set spans values of the target function for an infinite period of time.[10]

If an explicit time constraint is imposed as part of the task specification, then this time constraint can be incorporated as another target function of the task $G_{M+1} = t^*$. In this case the I-attainability set coincides with the B-attainability set *by* a sufficiently large time.

10. This definition of attainability corresponds to the accepted in control theory notion of the set of attainability in the case when $G_i = x_i(t^*)$ (e.g. see [38]). In the present paradigm the definition is extended to arbitrary scalar functions of the terminal time and the state variables.

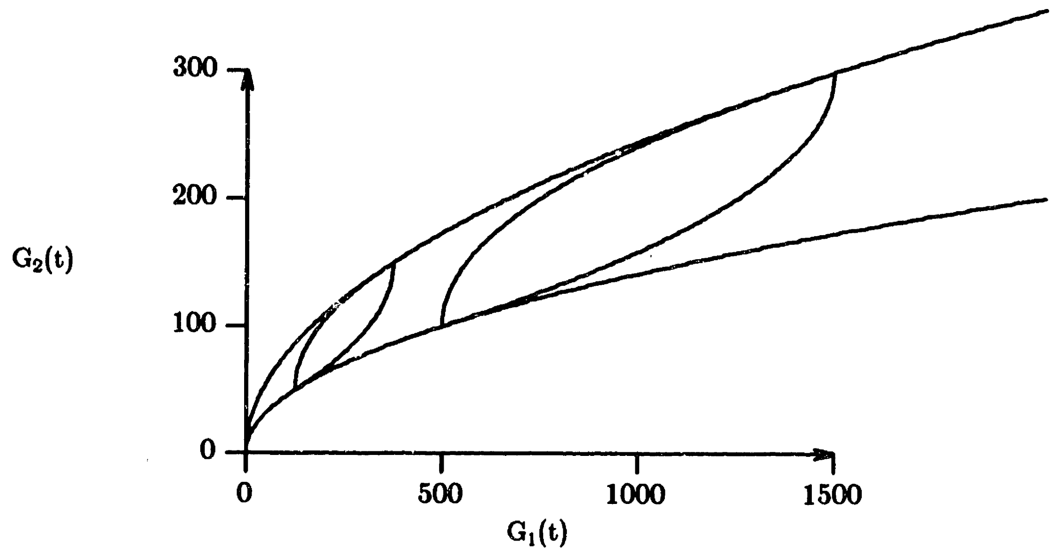


Figure 1.7. The (I-) attainability set.

Using the notion of the attainability set, it is possible to answer a question whether a particular goal is accomplishable. Each goal defines a goal box region, which will be denoted $[\mathbf{g}^-,\mathbf{g}^+]$ in the goal space:

$$\mathbf{y} = (y_1,y_2,...,y_M)^T \in [\mathbf{g}^-,\mathbf{g}^+] \text{ iff}$$
$$g_i^- \leq y_i \leq g^+ \quad \text{for all } i = \overline{1,M}$$

If the goal region has a non-empty intersection with the attainability set of the system from time t_0 then the goal is achievable from time t_0 in the absence of future perturbations:

$$\text{if } \overline{\psi}_{t_0,\infty} \cap [\mathbf{g}^-,\mathbf{g}^+] \neq \varnothing$$
$$\text{then goal } (g_i^-,g_i^+) \text{ is attainable from time } t_0 \tag{1.25}$$

If there where no perturbations, the attainability set would have had to be calculated only once - at the beginning of the task. Any goal that satisfied (1.25) from $t_0 = 0$ would have been an attainable goal. However even in the absence of perturbations, the once selected goal might become unattainable in the future. In the absence of perturbations this might happen only due to a *bad* control action.

An example of a bad control action is to drive a car in a wrong direction on the highway. If the goal was to get to some destination within some desired time, and that goal was accomplishable at the onset, then it might become unattainable after the car has driven for a while in the wrong direction.

A reasonable aid in this situation might be the one that recalculates the attainability set as time goes by. If wrong control actions are being executed, then the intersection of the attainability set with the goal region (1.25) would reduce to the empty set. If this reduction takes place gradually, then just monitoring the change of the attainability set with time might be helpful to correcting control actions. However by the time the intersection of the attainability set with the goal region vanishes, no future control actions could remedy the situation. A new goal must be selected then. The presence of perturbations makes the need for recalculation of the attainability set as the time passes more pronounced.

1.4 Attainability-Based Controller

So far the attainability set was introduced as a feedback monitoring tool for observing the system's progress towards accomplishing the goal. The selection of the control actions was supposed to be done by a controller separate from the attainability set evaluator.

It seems beneficial to combine the controller with the attainability set evaluator in a single control loop. The diagram of this type of a controller and the human-computer interface that could go along with it is shown on Fig. 1.8.

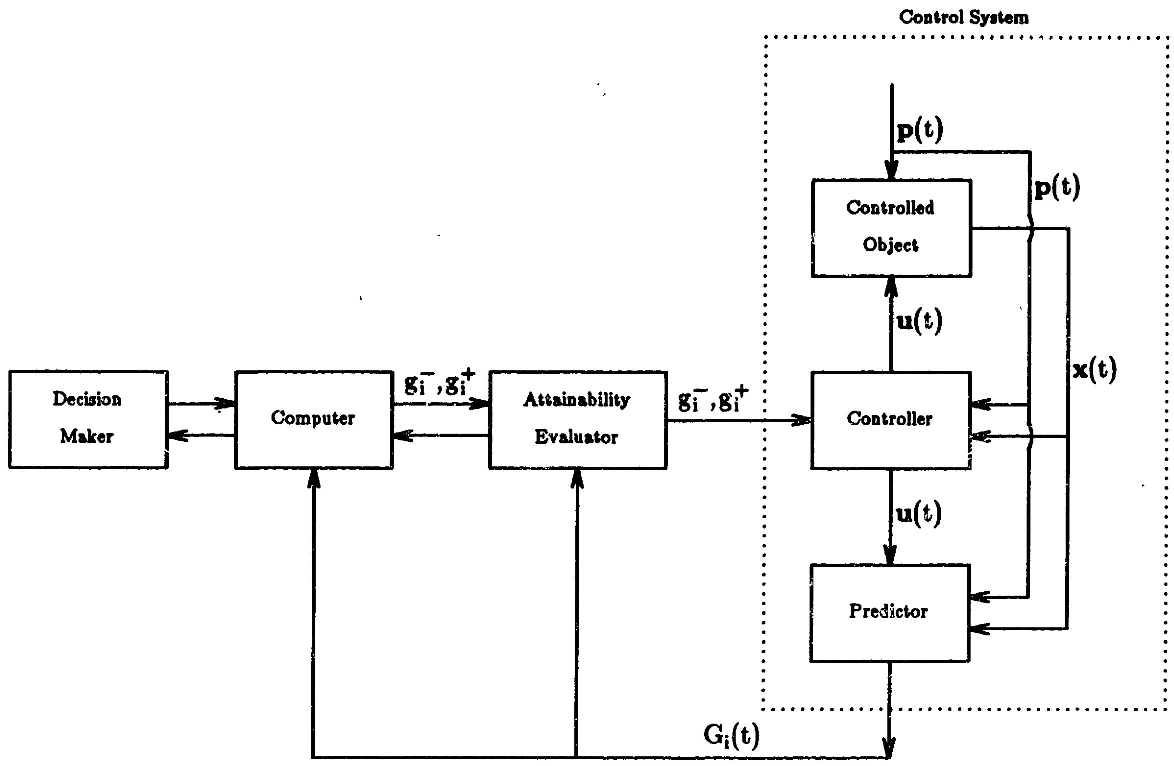


Figure 1.8. Goal-Setting Control Interface.

Control actions at time t_1 are selected based on the same assumption that was used for the estimation of the attainability set. Namely the control is synthesized assuming that the future perturbations *are known*. The constant for approximation of the future perturbations is provided to the controller by the DM.

The control $\mathbf{u}(t_1)$ at time t_1 is selected in such a way that the system *can* reach the goal in the absence of future perturbations. To be consistent with the attainability condition, the control action selected at time t_1 should satisfy the following conditions:

$$\begin{gathered}\text{there exist some terminal time } t^* \geq t_1 \text{ such that:}\\ g_i^- \leq G_i(t_0, t^*, \mathbf{x}(t_0), \mathbf{u}_{t_1,[t_0,t^*]}, \mathbf{p}_{t_1,[t_0,t^*]}) \leq g_i^+ \\ \mathbf{u}_{t_1}(t) \in \Omega_u\end{gathered} \tag{1.26}$$

Where:

$\mathbf{u}_{t_1,[t_0,t^*]}$ is a vector-function of the entire control selected at time t_1. This function coincides with the control that has already been implemented up until time t_1 on the interval $[t_0, t_1)$. In effect this function is selected only for interval $[t_1, t^*)$.

$\mathbf{p}_{t_1,[t_0,t^*]}$ is a vector function of the perturbations which coincides on the time interval $[t_0, t_1)$ with the perturbations that have obtained from t_0 until t_1, and is a constant from t_1 until t^*.

If the constant approximation of the future perturbations were correct, then any control satisfying (1.26) at time t_1 could be used in the open loop. Since perturbations that will occur in the future may be different from the approximation, conditions (1.26) have to be constantly tested. However in any practical (digital) implementation the same control selected at t_1 will stay in effect for some duration of time.

Conditions (1.26) could have a non-unique solution (in regard to t^* and $\mathbf{u}_{t_1,[t_0,t^*]}$ as the unknowns). It is a separate question as to which particular solution is to be executed. One way to go is to have the DM select a particular target function, say G_k, and to execute the solution that optimizes G_k subject to constraints (1.26).[11] An example of the implementation of such a control strategy can be found in Chapter 6.

1.5 Multiple Criteria Decision Making

It is often the case that the DM wishes to get as low as possible (or as high as possible) values of some of the target functions (provided that the values of all other target functions are the same). Without a loss of generality we will restrict ourselves to the case when the lower values of the target functions are preferred to the higher values of the same functions under otherwise equivalent circumstances.[12] A target function for which this preference exists on the values when values of others are fixed is called a ***preferentially independent*** target function.[34]

If it is desired that the goal value of some target function, say of G_k was as low as possible, then it implies that only g_k^+ is needed for the specification of a goal for this target function. It certainly makes sense in this case not to restrict intentionally the values of this goal function from below. It does not matter to the DM how low a value the target function will actually get when the goal is achieved, as long as this value is lower than g_i^+. If similar considerations hold for all indicator functions G_i, then the selection of a new goal is closely related to the

11. The DM also needs to specify whether G_k is to be minimized or maximized.
12. The extension to the case of a target function whose higher values are preferred to the lower ones is obvious.

traditional problem of Multiple Criteria Decision-Making.

The presence of only a single bound g_k^+ on the value of a target function G_i indicates that the DM *prefers* the lower values of G_k to the higher ones. Tightening this constraint, i.e. making it even lower, might be beneficial from the DM's point of view.

Tightening bound g_k^+ reflects the DM's preference ordering "the lower the better" (or at least not worse) for the goal values of G_k. If there are several target functions having this property, then the preferences to have the value of each of those function as low as possible might be in conflict with each other.

In choosing a goal, the DM is constrained by the bounds of the attainability set. Setting some g_i^+ too low would necessitate the choice of a higher value of some other g_j^+, so that the goal is attainable. The tradeoff is to be made between the benefits gained from lowering the values of some g_i^+ and the subsequent losses incurred by the necessity to increase the values of some other g_j^+ to keep the goal attainable. This tradeoff is at the heart of the traditional Multiple Criteria Decision Making paradigm (MCDM). The overview of the existing research in the area of MCDM from the perspective of this thesis is presented in Chapter 2.

A traditional MCDM problem[9][29][34][73] is concerned with concurrent minimization of M scalar functions:

$$\min_{\mathbf{x} \in X} F_i(\mathbf{x}), \ (i = \overline{1,M}) \tag{1.27}$$

In the case of the goal setting paradigm, the functions to be minimized are G_i and the argument $\mathbf{x}$ is a tuple $(\mathbf{u}_{t_1,[t_1,t^*]}, t^*)$.

Every conventional MCDM method prescribes the way of obtaining some solution $\mathbf{x}^{\text{optim}}$ that belongs to the Pareto subset $\mathbf{X}^{\text{par}}$ of the problem (1.27) (see Chapter 2). A solution belonging to the Pareto set, however, might not be the best choice for a new goal. The reason is that any Pareto solution, $g_i^{+(\text{par})}$, (selected at time t_1) is *unstable* to future perturbations. This solution is usually located on the boundary of the attainability set (*from* t_1) and may be attainable for only a short time after t_1. In the case of a Pareto solution there is simply no room for the values of G_i to go any higher to offset the effect of the perturbations, and still remain attainable.

On the other hand, the DM might wish to select the goal sufficiently close to the Pareto boundary. No conventional MCDM allows the exploration of the alternatives *in the vicinity* of the Pareto boundary. A decision-aiding technique proposed in this thesis alleviate these shortcomings. This technique, the Dynamic Range Tradeoff helps the DM to select the goal in also the case when the goal is specified by two parameters g_i^- and g_i^+ for some target functions and by only one parameter g_k^+ for the others. This technique is also effective in solving a traditional MCDM problem, i.e. the problem of selecting a "one-sided" goal.

Another connection to the MCDM paradigm can be observed in a situation where the DM chooses a new goal to replace some goal (g_i^-, g_i^+) that just became unattainable. In choosing a new goal the DM might wish to set parameters $(g_i'^-, g_i'^+)$ of the new goal as close to (g_i^-, g_i^+) as possible. In doing so she also might have to make a tradeoff. Namely, for some target function G_i the new goal parameters might have to be distanced farther away from the old parameters than for some other target function G_j.

Despite of the apparent similarities between the goal setting paradigm and the MCDM, the traditional MCDM techniques are often ill-suited for aiding the DM in choosing an attainable goal. They are useful in helping the DM to explore the Pareto boundary of the problem, but they are insufficient in helping her to select a goal that is located at a distance from the Pareto boundary. In addition to that, a MCDM problem is only a partial case of the goal setting paradigm that is presented in this thesis. Any situation where two bounds g_i^- and g_i^+ need to be specified for some target function is beyond the scope of the MCDM. An example of a problem like that is described in Chapter 6.

1.6 Approximation of the Attainability Set

It is apparent that the DM needs to know and be able to explore the attainability set (not only its Pareto boundary in the case of one-sided target functions) in order to select a reasonable goal (g_i^-, g_i^+). To do so she also needs to be able to establish if a particular goal $\mathbf{g}^-, \mathbf{g}^+$ is attainable.

The attainability set is a set in a multi-dimensional space. It is difficult, if not impossible, for the humans to visualize it. The technique developed in this thesis helps the DM to get a feel for the attainability set, allows her to explore this set interactively, and to select an attainable goal. This technique is based on a causal approximation of the attainability set by adjustable ranges in the goal space and it is described in Chapter 3.

1.7 Consideration in the Design of Decision Aids

1.7.1 Decision Aid Design Problem

The problem of designing decision aids is too broad to be addressed at any depth in this thesis. We only touch upon the issues that are relevant in the context of the goal setting paradigm.

The decision aid design problem consists of specifying a set of characteristics for a decision aid that ensures that the decision making performance will be improved. There is a difference between many types of decision situations where the aid might be needed.

Consider a decision problem of choosing the correct answer in a multiple choice mathematics test. What could be a good aid for this problem? A calculator? A textbook with a similar problem already solved? A prompt from a friend? A throw of a dice? Apparently, all of the above, maybe with the exception of the last one. However it is conceivable that the DM would have to resort to the last decision aid (the dice) if no other aid is available.

On the other end of the decision-making spectrum consider an investment portfolio selection problem. It is well known that many well respected advisers might have opposite opinions about the merits of a particular investment. What aid could be suggested for this type of problem?[13]

These two decision situations reflect two extremes of a range of decision problems. There are decision situations in which the correct answer exists, only the DM may not know it for some reasons. In decision situations of another kind, the correct answer may not exist. Different decision makers might have different views about the correctness of a particular answer. In the meantime the variety of their opinions may not be the result of their ignorance but a consequence of the inherent difficulty of the problem.

It is the decision situations of the latter type that cannot be sufficiently automated and may always require the human presence in the decision loop. So one cannot expect a decision aid to come up with the solution of the problem. The aid only can be expected to help the DM to come up with one.

Among the reasons making the human presence in the decision loop necessary are the following[53]:

1. Best alternative cannot be computed. This includes the situations in which different human experts have different opinions about the best solution;

2. No adequate model of the process is available. The human has extraordinary memory, judgment and reasoning powers that enable her to make decisions even when the process attains some unmodeled state;

13. In fact, there are many aids on the market to help one solve exactly this problem. The usefulness of many of those aids is however questionable.

3. Human's unique sensory abilities cannot be replaced;

4. Political factors prevent replacement of human.

In this thesis we concern ourselves with helping the decision maker facing the decision situation of the first kind. The decision situation in consideration is the goal selection for an evolving system. We specifically consider the type of circumstances where no single expert opinion exist about what the best goal selection is. It is also frequently the case that the very evaluation of performance is not obvious.

The answer to the question *what is a good aid?* largely depends on how the performance is measured with and without the aid. In many domains this question is very difficult to answer. It may require a painstaking and laborious statistical processing of a great deal of observations, and even then the results could be controversial.

A problem of evaluating some decision aids frequently leads to what we call the *Roseborough dilemma* [53]:

This is a dilemma of evaluating a decision aid for which the normative actions cannot be computed. If the decision situation were such that the best actions could be computed, then the normative action should be used in place of the human and the DM is not necessary. However in the decision situations where the presence of the human is indispensable, the best actions cannot be computed, and a single decision cannot be judged as to its value, optimality, or correctness.

In the latter case the decision aid should be studied in the laboratory, in a completely modelled environment. The latter requirements implies that *in the laboratory environment* the evaluation criterion and the normative course of actions must be computable. The last requirements might seem to contradict the previously stated nature of the original decision problem, in which neither the best course of actions nor the evaluation criterion were available.

The *Roseborough dilemma* is that in spite of this ostensible contradiction between the real decision situation (for which the decision aid is designed) and a controlled laboratory environment that is used for testing the aid, the laboratory testing might nevertheless be the best way to verify and test the aid. The problem of navigating the boat (Chapter 6) was chosen as an example decision problem whose elements can be brought into the laboratory and examined carefully.

1.7.2 Types of Decision Aids

If it is impossible to fully model the system about which the decision is to be made, an alternative way is to model the decision maker instead. Among various ways to model the human decision maker, the models based on the utility theory stand out as the most rigorous (mathematically) and the most consistent (conceptually) approach. The utility theory as a theory describing an (economic) behavior of a single decision-maker was first suggested in[17] (see [66]) and later developed as a rigorous mathematical theory that included the foundation of the Bayesian statistics in[55].

Most decision aids based on a model of the human decision maker roughly operate on the same principle. They operate with a model of DM that contains several "slots" (or parameters) characterizing individual decision maker in a decision situation. These slots are filled (i.e. the model is personalized) in the process of (direct and indirect) *solicitation* from the human (which might include observing human actions over time). As soon as the slots are filled, the aid cranks out the correct decision. In effect, this type of aid operates according to the principle: *find who the decision maker is and what is it that she wants, and then produce the best answer that suits her.*

A different approach is to design decision aid as a *decision tool.* The tool approach is not based on any particular model of the DM. The aid based on the tool approach still leaves the decision up to the DM. It however provides what is called a *Decision support environment.* The proliferation of the computers has been accompanied with the spread of this type of tools. In fact, a bulk of the current AI research has turned from the research geared towards reproducing the human decision making process to providing the tools for assisting the human in making decisions. The spread of the expert systems technology attests to that.

The decision aid developed in this thesis belongs to the latter category of decision aids. The aid contains a set of novel tools that are supposed to create a supportive decision environment for the DM. The environment that is created with this decision aid does not preclude the use of any other method, if available. However in the case of a general goal setting paradigm no other way to aid the DM has been suggested in the research literature.

The fact that no normative model of the human is used by a decision aid does not preclude it from effectively aiding the DM. A model of the decision process may be (and should be) eventually created after the decision aid has been established. An analogy can be found in driving a car. We still are in a process of building models of humans driving cars; in the meantime we already have been building cars.[71]

Another aspect of decision aiding is a degree of generality of an aid. It is safe to say that the more general a particular decision aid is, the more it can be modified and augmented to suit a specific task. The goal setting paradigm suggested in this thesis *is* a very general paradigm, and it describes a variety of practical situations.

A decision aid developed in this thesis has been designed to provide the DM with a general purpose tool to deal with a general case of the goal setting situation. It is easy to foresee particular goal-setting situations which can be successfully approached with specific, sometimes ad-hoc methods. The latter should not be considered as ***competitors*** of the general technique offered in this thesis. Instead, the method developed in this thesis provides an approach and guidelines for attacking those problems for which no other decision technique is available.

Chapter 2

Multiple Criteria Decision Making

2.1 Introduction

The goal setting paradigm as it has been mentioned in Chapter 1, has a close connection to the Multiple Criteria Decision Making problems. The work on developing computer decision aids for the latter[10][11] has led us to the former. Though the goal setting paradigm is broader and more general than the MCDM problem, some insights in the latter are useful for the understanding of the former.

In this Chapter we give a brief overview of the existing research on Multiple Criteria Decision Making.

The essence of the Multiple-Criteria Decision Making paradigm is to find **x**, an element of some set Ω that minimizes several functions at the same time:

$$\min_{\mathbf{x} \in \Omega} F_i(\mathbf{x}), \ (i = \overline{1,M}) \tag{2.1}$$

where Ω is a set of potential choices that can be either continuous or discrete.

Problems that can be modelled by this paradigm commonly occur in every-day life. They pervade all that we do and include tasks ranging from engineering design to business and political decision-making. This paradigm and its mathematical formalization have long been subjects of numerous scientific publications. The surveys (e.g. [25][37][39][65]) list several thousand references on the subject.

At the inception of the MCDM as a distinct research area, most methods of solving (2.1) were automated mathematical algorithms that had very little if any interaction with the DM (see e.g.[14]). These methods were rooted in classical optimization techniques. Multiple criteria optimization was actually viewed by some researches as a natural extension of a single-objective optimization. In recent years the emphasis has been gradually shifting from purely mathematical algorithms towards the algorithms that extensively communicate with the DM, i.e. it is recognized that different DMs might prefer different solutions of (2.1). A number of mathematical models have been developed to provide the human decision maker (HDM) with computer assistance in solving the MCDM problem (e.g. [73][9][29][46][68][48]).[1]

It is only natural that many research efforts have been directed towards helping the human to solve the MCDM problem. The application of the MCDM methods has been suggested in many engineering areas (e.g. [64][65][46][48]), including design of aircraft control systems[64], structural mechanical systems[16], space station optimization[3], to name a few. Most of these application however are theoretical in nature, they are designed to demonstrate particular methods, and their results have rarely been implemented in practice.

In spite of a great many methods reported in the literature there is a discernible lack of applications of MCDM techniques in practice, as has been pointed by different authors (e.g. [41]). It was suggested that the most profound reason for a limited acceptance of MCDM methods in practice is a lack of DM's confidence in the obtained "optimal" decision[7]. The decision makers who have the authority to exercise their choice have little understanding of the underlying mathematical models, and therefore have reservations in accepting computer advice. Further, and perhaps more important, existing technologies seldom include more than one way of trading between objectives (while the decision-maker may consider many).

1. In the goal setting paradigm set Ω corresponds to the set Ω_u of all admissible controls in (1.8). Functions F_i of (2.1) correspond to the target functions G_i of (1.9).

2.2 Existing MCDM Methods

To solve (2.1) means to find a point $\mathbf{x}^g$ from the set Ω of all possible alternatives that minimizes all criteria $F_j(\mathbf{x})$ at the same time. Most often this is an impossible task because functions F_j attain their minima at different values of $\mathbf{x}$. Solving (2.1) therefore means finding some compromise solution $\mathbf{x}^g$ that corresponds to the best trade-off among values $F_j(\mathbf{x}^g)$ of all functions. However not all elements of Ω should be considered as possible candidates for this compromise. Ω may be reduced to its *Pareto* subset Ω^P. Those and only those alternatives $\mathbf{x}^p \in \Omega$ belong to the Pareto subset Ω^P, that are better than *any* other alternative in Ω in respect to at least one criterion. Those alternatives that belong to Ω but do not belong to Ω^P may be termed obviously inferior and ignored. Any *rational* compromise solution of (2.1) should be looked for within Ω^P only.[2]

It is also useful to talk about the Pareto set in the space of the functions F_i in (2.1). These functions define a mapping: F_i: $\Omega \longrightarrow \mathbf{R}^M$. The image of the set of all alternatives can be denoted $\mathbf{F}(\Omega)$. Similarly, the Pareto set of this image is the image of the Pareto subset of Ω: $\mathbf{F}(\Omega^P)$. Since the space that contains Ω is

2. Here is a formal definition of the Pareto subset Ω^P of Ω for a problem (2.1). Those and only those members $\mathbf{x}^p$ of Ω belong to the Pareto set Ω^P that satisfy the following condition: for any element $\mathbf{x}' \in \Omega$ there exists some criterion j ($1 \leq j \leq M$), such that $F_j(\mathbf{x}^p) < F_j(\mathbf{x}')$. An alternative $\mathbf{x}^{infer}$ that does not belong to Ω^P should never be selected as a compromise solution of (2.1) because there always exists some alternative from the Pareto set $\mathbf{x}^p \in \Omega^P$ such that $F_j(\mathbf{x}^p) \leq F_j(\mathbf{x}^{infer})$ for all j ($j = 1,M$), and for some criterion k: $F_k(\mathbf{x}^p) < F_k(\mathbf{x}^{infer})$. In other words, $\mathbf{x}^p$ is better that $\mathbf{x}^{infer}$ in respect to criterion k and at the same time it is no worse that $\mathbf{x}^{infer}$ in respect to all other criteria.

the space of the alternatives, $\mathbf{R}^M$, the space of the values of F_i, can be called the space of the criteria.

One can identify two stages of solving a MCDM problem: 1) generating a Pareto set of alternatives; 2) selecting the "best" or "optimal" alternative from the Pareto set. These two problems are very different in nature. The first problem can be solved solely by mathematical means, without any assumption about the decision maker. Finding an "optimal" alternative from within the Pareto set, however, is a mathematically incomplete task. Unless some additional assumptions are made, the latter problem simply does not have a solution and is mathematically ill-posed (as well as problem (2.1) itself).

Different approaches to MCDM are based on different assumptions about the nature of trade-offs that would make problem (2.1) solvable. These assumptions form a foundation of a broad range of models - from normative models of human decision-making, such as the utility theory[34], to "unmodeled" decision making such as satisficing models[70]. It is widely recognized that in order to complete the mathematical model, some information must be extracted from the DM. Approaches differ in the way this is done.

Existing techniques can be classified as either one-step (non-interactive) or multi-step (interactive) methods. In one-step methods, information is elicited from the DM, then passed on to the computer, and the computer in return generates the "optimal" solution of the problem. Classical multiple attribute utility theory is one example of such methods. These methods use the DM as a source of values for parameters of their model. An interaction with the DM is basically a *measurement* procedure. There are strong arguments questioning practical value of these methods (e.g. [71][73]).

Multi-step interactive methods are increasingly recognized as a practical approach to solving MCDM problems. All these methods are based on iterative sessions of human-computer communication. Based on the type of such communication and the models they use, these methods can be classified into three categories:

- **Alternative Set Reduction.** On each iteration the DM answers certain questions that make it possible for the computer to reduce the Pareto set of possible solutions. The most famous method of this type, the ELECTRE method[15], was very popular in Europe and was tried in many areas of application. This method was invented as an alternative to the classical utility theory and it allows intransitivity of DM's preferences. This method was criticized on theoretical grounds, in particular for not "satisfying any system of consistent and appealing axioms"[2].
- **Computer Suggests an Alternative and Asks Questions, Human Answers.** On each iteration of methods of this type, the computer offers the DM a possible solution and then asks some questions pertaining to the DM's liking/disliking of this solution. The answers are then processed according to some mathematical model and a new alternative solution is presented to the DM on the next iteration. Methods of this type usually operate under assumptions about an unknown but implicit utility function. Some of the better known methods include: Surrogate Worth Trade-off method[9] and Zionts and Wallenius method[74].
- **Human Asks, Computer Answers.** Methods of this kind usually do not

assume anything regarding the underlying utility function; they do not even assume the existence of a utility function. The human specifies some desired values of the attributes and the computer generates a feasible alternative in response. The reference point approaches[22][62][70] belong to this category.

2.3 Deficiencies of the Existing Methods

There is a strong disagreement between some authors regarding relative merits of specific methods (e.g. [72][1]). Many techniques have been built, but it is very difficult to determine which one makes more sense than the others[2].

It is tempting to find the structure of the decision-maker's preferences about the domain in order to help her solve (2.1). The only consistent and rigorous way to do it is the utility theory[34][55]. However, practical implementation of this theory runs into several major difficulties, including: 1) prohibitive time required to find out one's utility function; 2) questionable accuracy and sensitivity of the utility measurement procedure, and therefore unknown accuracy of the results; 3) dependency of the utility on the bounds of the set $\mathbf{F}(\Omega)$ of alternative solutions - should this set be moved around, the utility function will have to be recalculated anew.[3]

3. Most procedures calculate multi-attribute utility function inside a box in the criteria space. The larger the box, the less acurate the calculations are. If the set $\mathbf{F}(\Omega)$ is moved outside of the box for which the utility has been calculated, it has to be re-calculated anew.

It was observed that different methods lead to different solutions[7]. If the DM understands and believes in the mathematical model run by the computer, it is very likely that she will accept the results of the interaction with this method. For instance, a creator of a specific MCDM method will most probably use her method herself. The variety of existing methods has thus led to a variety of proponents (and opponents) of those methods.

Some decision-makers, on the other hand, could accept any method as long as it leads to a "reasonable" alternative. Suppose that the computer suggests an alternative that satisfies some given constraints on the values of its criteria. Some DM will readily accept this solution and terminate any further search for an even "better" one. Another DM may want to proceed and explore other choices. In tasks where the "goodness" of the final solution is very important to the DM, she needs to be convinced that she has found something really close to **her** "optimum".

It is our hypothesis that a major source of the user's dissatisfaction comes from her inability to continue to look at the (Pareto) set of alternatives as a whole during the selection process. No existing method provides the DM with this outlook. As a result the decision-maker may have doubts as to whether there is yet a "better" alternative left unseen, or whether a previously abandoned alternative has to be reconsidered.

Existing interactive MCDM methods iteratively gather information about DM's preferences and try to steer her to the best solution. They offer the DM one or several alternative solutions at each iteration. Based on her responses about relative merits of these (and maybe previously seen) alternatives, the computer generates another (or several) new potential solutions. The hope is that after several iterations of this human-computer dialogue the DM will converge to the "best" alternative. All of these MCDM procedures are supposed to be "smart" and selective in choosing next candidate alternatives. They hope to utilize information obtained from the DM's responses at earlier iterations.

Using these methods the DM can never be sure how to interpret the absence of alternatives that she would have liked. It may be that *nothing* satisfactory is available in Ω, or it may be that this particular method has not picked the "right" solution. Existing MCDM methods do not give the decision-maker a global view of the Pareto set because they expose her only to a small selected set of alternative solutions. The DM may have doubts how representative and complete this set is for her to make a decision. The decision may be made out of frustration and a lack of motivation to proceed with more iterations.

It is argued in [73] that it does not matter what the best solution is - it could be satisficing or maximizing, compromising or arbitrary - the only thing that matters is that the DM is confident that the best solution has been obtained.

Another problem with some of the existing methods is the ambiguity of the human-computer interface. The DM may be asked to provide some quantitative data that she does not know well (e.g. marginal rates of substitution or weights). Sometimes the computer exercises great flexibility in interpreting the decision-maker's responses. For instance, some methods use a standard (user-independent) proximity measure on the set of alternative solutions. Or some methods require the DM to perform some tedious and lengthy procedures, such as for example, exhaustive pairwise comparison of many alternatives[67].

It is a daring task to model and to make use a model of human decision-making process. One should realize that though (2.1) seems to be a good model of a decision problem, the MCDM is a *foreign* problem for human beings when the number of alternatives is large. It is not a problem humans can solve easily and it is not a problem they like to solve. Apparently, our heuristics help us to reduce the number of alternatives before actually starting the selection process. Due to a lack of short term memory, we just cannot process sufficient information if the number of alternatives and/or criteria is large[2].

If the human uses a computer-aid, she tends to adjust her decision-making to that specific aid. The decision maker actually begins to think in terms and the "language" of the decision model - be it "satisficing point", "weights", or "lotteries". The model and the input/output language of the computer aid therefore affect the human decision making process and cast it into their mold. Since the human does not know how to solve problem (2.1), a computer model that she uses becomes an integral part of her own decision process.

Very difficult and not yet adequately resolved is the issue of the experimental verification and assessment of various methods for solving the MCDM problems using real human subjects. Many researchers have resorted to assessing their methods using *artificial subjects,* i.e. some mathematical expressions for the utility function that was supposed to adequately replace the real humans (e.g. [27][49]). It is easy to notice that the assessment of a MCDM method is itself a MCDM problem[28], and this poses additional problems for the experimental comparison of different methods.

Some experiments though bring surprising *negative* observations about the merits of the decision aiding methods. Thus the experiments with the Zionts and Wallenius method led to the surprising conclusion: "the DM's don't see the feedback between answers on detailed questions (required in the interactive programming) and their overall strategy. In spite of the acknowledgement of the multiobjective character of a decision process, it seems that DM's tend to make decisions on a basis of one objective/goal at a time"[44].

2.4 Related MCDM Research

The ideas of the DRT method developed in this thesis, have been inspired by several works and ideas from the area of MCDM.

The first relevant idea is the one of *satisficing.* The notion of satisficing as the rational for the DM behavior was first introduced by H. Simon in [61] and was developed into a MCDM technique by Wierzbicki and others (e.g. [22][62][70]).

The idea of satisficing and the so-called *reference point approaches* to the MCDM problem is that the natural way for the DM to solve a MCDM problem is to iteratively set and gradually change the *aspiration levels,* i.e. the maximum acceptable levels of the criteria. A similar idea is behind the so-called *goal programming* methods[30][31].

The reference point algorithms work in the following way: The DM specifies her aspiration levels which are interpreted by the computer as her *goal levels,* or a *reference point* in the criteria space. If these levels are attainable, then another point in the criteria space is generated that at the same time a) belongs to the

Pareto set of the problems; and b) in some way "preserves" the balance of the initial aspiration levels. If the levels are not attainable, then yet another point from the Pareto set is generated that also is expected to preserve as much as possible the initial balance of the aspiration levels.

If the DM is not *satisfied* with the observed solution(s), she can change her aspiration levels and get another suggestion, always selected from the Pareto set of the problem. The process terminates when the DM is satisfied with the solution she has found. These methods differ in the way the selection of the alternative in response to the DM's specified aspiration levels is done. The key idea behind of all of these methods is that the human preferences (and therefore her satisficing levels) dynamically develop in the process of interaction with the alternative solutions.

Another important idea pertaining to the aspiration levels is that the as soon as the DM has found satisficing aspiration levels, she does not care too much whether which alternative to choose as long as it satisfies these levels[63]. This consideration gave rise to a method of solving *any* MCDM problem as a discrete MCDM one. This was achieved by creating a finite "cloud" of equally spread points in Ω and looking for a solution only from among these points. This approach, the search with the so-called *LP-sequences* [63], represent a method of actually approximating the attainability set with a finite mesh of points. In this approach no specific concern was placed on selecting a solution from the Parteo subset.

Another important aspect of a computer aid for solving the MCDM problem, is the issue of the human-computer interface. It is now recognized that computer graphics offer more than just an illustration of an algorithm that is at the core of the decision aid. The graphics can also be a decision aiding tool in its own right[21][35][45][60]. The existing graphics aids range from bizarre "Chernoff faces"[12] to the interactive system for "driving" along the Pareto set, the so-called "Pareto race"[36]. The latter system was reported to have earned high praise from many different users.

The DRT technique developed in this thesis draws from these ideas. Its purpose is not only to help the DM select an attainable solution but also do it with confidence in her choice.

Chapter 3

The Dynamic Range Tradeoff Method

3.1 Range Approximation of a Set

3.1.1 Introduction

The Dynamic Range Tradeoff (DRT) method is a method of interactive exploration and approximation of the attainability set of a decision problem. This method can be used for the exploration of dynamic (i.e. changing with time) as well as static attainability sets. This method helps the DM select an attainable goal.

Under most circumstances the DM is interested in selecting a goal that is located away from the boundary of the attainability set, so that this goal will be robust against future perturbations. However the desire to attain the lowest values of some or all of the target functions drives the choice close to the boundary. As it was mentioned before, existing methods of solving MCDM problems focus exclusively on selecting a point from the boundary of the attainability set, i.e. from the Pareto subset thereof.

It is clear that in all cases the DM needs to know where the boundary of the attainability set is. However very often it is difficult to compute that boundary. Moreover, if the dimension of the attainability set is greater than three, it is difficult if not impossible for the humans even to visualize that set.[1]

The DRT method helps the DM explore an *approximation of the boundary* of the attainability set and select a goal in a controlled proximity from that boundary. This method also is useful in helping the DM solve a traditional MCDM problem, i.e. to select a goal in the case of preferentially independent target functions.

The following analogy illustrates the essence of the DRT method. Suppose one is in a dark room and needs to find and explore an edge of a large object. Suppose that the object is a 2D piece on the wall and the only tool available for the exploration is a spotlight providing a rectangular beam of light (see Fig. 3.1).

1. Regardless of what technique is used to choose a goal, it is always necessary to to compute at least some parts of the attainability set. For instance if the utility function approach is used, the utility function must be maximized over the entire attainability set. In the DRT approach we deal exclusively with the approximation of the attainability set and its presentation to the DM in such a way that it indirectly facilitates the selection of the attainable goal that is acceptable to the DM.

The spotlight can be moved right, left, up and down. The size of the rectangle that the spotlight illuminates on the wall also can be changed.

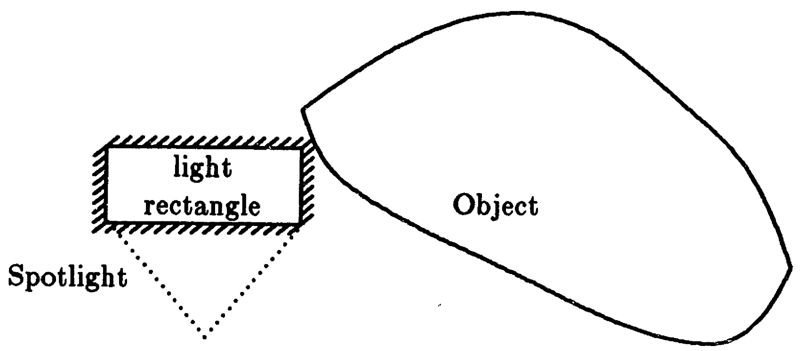


Figure 3.1. The object and the spotlight.

Looking at the area illuminated by the spotlight one can see either: 1) the wall (Figs. 3.1 and 3.2 a)); or 2) the object (Fig. 3.2 b)); or 3) a piece of the wall and a piece of the object (Fig. 3.3 a)).

The third case out of these three is the most informative one. In this case a piece (AB) of the boundary of the object can be seen (the spotlight illuminates rectangle KLNM). Suppose however that the only curves that can be distinguished are the straight lines. So one can not see the exact boundary of the object on Fig. 3.3 a), but can see only its approximation by straight line segments (AA′ and A′B). At the same time the piece of the object that is illuminated by the spotlight is approximated with a rectangle ALBA′. This approximation is shown on Fig. 3.3 b). The hatched area inside of the light rectangle KLNM approximates the wall, i.e. the area outside of the object but still inside of the light rectangle.

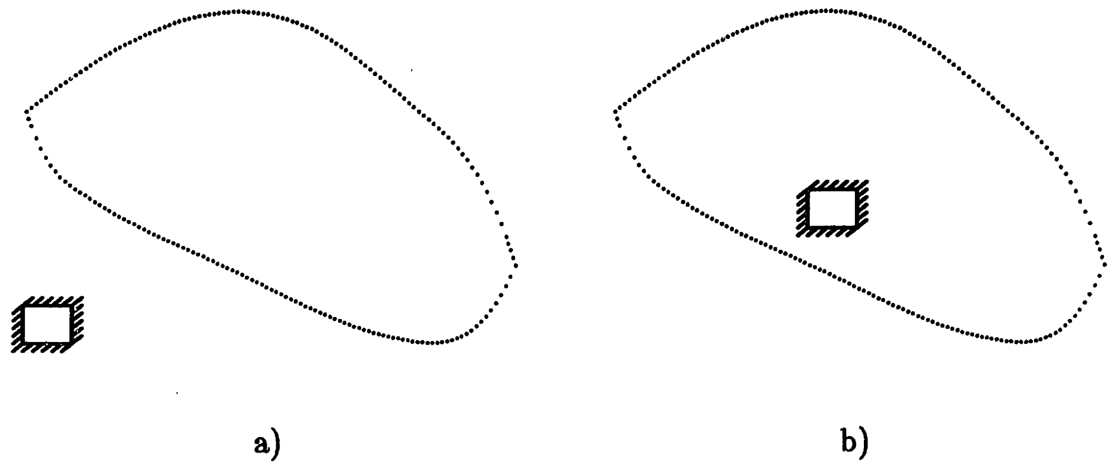


Figure 3.2. The spotlight illuminates a) the wall; and b) the object.

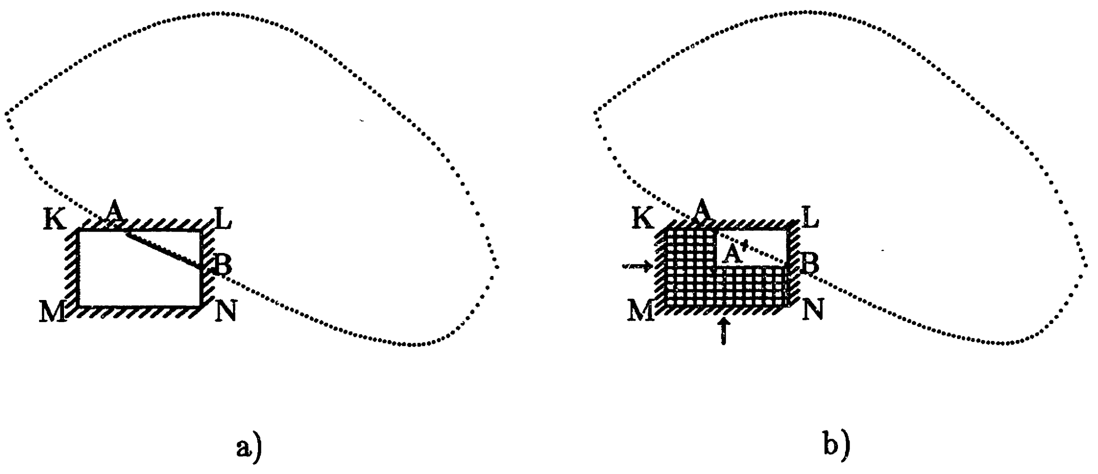


Figure 3.3. The object's boundary a); and its approximation b).

This last is obviously a very rough approximation of the actual boundary and of the object, however when the size of the spotlight rectangle gets smaller, the error introduced by this approximation becomes smaller as well. To reduce the error and still not loose the sight of the boundary one can push the left side of the light rectangle to the right and the bottom side of the rectangle upwards. At some point the top side of the rectangle can be moved downward, etc. In effect these manipulations with the boundaries of the light rectangle amount to "trapping" a point on the object's boundary. An intermediate stage of the boundary "trapping" is shown on Fig. 3.4.

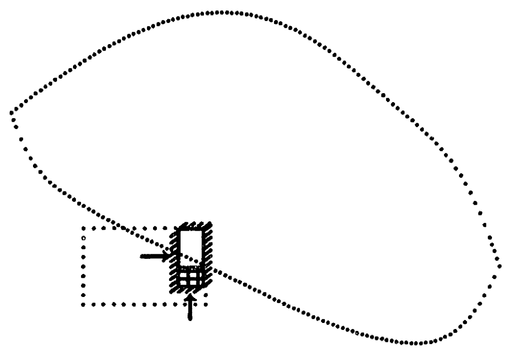


Figure 3.4. The boundary of the object is being *trapped.*

If the respective boundaries of the light rectangle are moved closer and closer to each other, but so that a piece of the object's boundary is still illuminated, then eventually the light rectangle will converge to a point. It is important to note that this point will be focused on the object's boundary, and that a piece of the boundary (the point itself) will be approximated exactly. What makes this whole process of "trapping" the boundary so useful, is that at each moment of time one does not need to know the exact boundary (AB). The entire "trapping" process can be guided solely by the boundary's approximations (A′B) (Fig. 3.3 b)).

The basic idea of the DRT method is that the DM explores the attainability set by specifying some *desired region* in the goal space (an analog of the light rectangle), while the computer in turn estimates the area inside of the specified region, that contains the attainability set (using the same straight line approximation of the boundary as in the spotlight example).

The desired area specified by the DM is interpreted by the computer as the system's *goal*. If the the system is controlled by the DRT controller, then this desired region is literally interpreted as the system's goal and the controller tries to achieve that goal. If the system is controlled by some other means, then the computer only monitors the system's evolution towards the goal.

If the attainability set changes with time, then the computer response is constantly updated, accounting for the evolution of the attainability set (and of the system). The estimation of the attainability set inside of the goal region continues even if the DM does not change the specifications of the region.

As soon as it is detected that no part of the attainability set belongs to the declared goal region, the goal is determined unattainable. The DM is immediately warned about that.

Below is a formal description of the DRT technique.

3.1.2 Associated Range of a Set

The basic notion of the DRT method is the notion of a *range*. Range is a construction that is used in the DRT for approximating and exploring arbitrary sets in finite-dimensional spaces. The use of ranges has several advantages - the range construction is natural and intuitively clear, ranges are flexible to use, and they can be relatively easily computed. Some other useful mathematical properties of the ranges are discussed below.

A range can be visualized as two boxes (hyper-parallelepipeds), one inside of another, in the M-dimensional space. The outer box (called the *active bounds* box) is controlled by the DM: each of its sides can be moved independently of all other sides. The inside box (called the *passive bounds* box) approximates the piece of the attainability set that is contained inside of the outer box. This approximation is similar to the straight line approximation of the object in the spotlight example. Below is a formal definition of a range of a set in the M-dimensional space $\mathbf{R}^M$:

Definition: An *associated range* of a closed set $\Psi \subseteq \mathbf{R}^M$ is a group of $4 \times M$ real numbers, or *bounds* , L_i^A, L_i^P, R_i^A, R_i^P $(i=\overline{1,M})$, that satisfy the following conditions:

$$\begin{aligned} L_i^P &= \min y_i \\ R_i^P &= \max y_i \end{aligned} \quad (i=\overline{1,M}) \tag{3.1}$$

subject to:

$$\begin{cases} L_j^A \le y_j \le R_j^A, \quad (j=\overline{1,M}) \\ \mathbf{y} = (y_1,y_2,\ldots,y_M)^T \in \Psi \end{cases} \tag{3.2}$$

(where $\mathbf{y} = (y_1,y_2,\ldots,y_M)^T \in \mathbf{R}^M$ denotes a point in the M-dimensional space).

This range is denoted

$$(\mathbf{L}^A,\mathbf{R}^A \xrightarrow{\Psi} \mathbf{L}^P,\mathbf{R}^P) \tag{3.3}$$

The bounds with a superscript "A" (i.e. $\mathbf{L}^A$ and $\mathbf{R}^A$) are called the *active* bounds, or constraints. The bounds with a superscript "P" (i.e. $\mathbf{L}^P$ and $\mathbf{R}^P$) are called the *passive* bounds. The bounds can also be separated into two groups: the left bounds $(\mathbf{L}^A,\mathbf{L}^P)$, and right bounds $(\mathbf{R}^A,\mathbf{R}^P)$.

If conditions (3.2) are inconsistent then the range is called a *null* or *infeasible* range. Since it is presumed that set Ψ is closed, the finite minima and maxima in (3.1) do exist (as long as the active bounds are finite). A range whose corresponding active and passive bounds are equal to each other ($L_i^P = L_i^A$, $R_i^P = R_i^A$ for all $i=\overline{1,M}$) is called a *cleared* range.

The active bounds of a range bind a *box* $[\mathbf{L}^A,\mathbf{R}^A]$ in the M-dimensional space. The passive bounds also bind a box, $[\mathbf{L}^P,\mathbf{R}^P]$ in the same space.

Definition: A **box** $[\mathbf{a},\mathbf{b}]$ ($\mathbf{a}$ and $\mathbf{b} \in \mathbf{R}^M$) in $\mathbf{R}^M$ is an M-dimensional hyper-parallelepiped, i.e. a set containing all those points $\mathbf{x} = (x_1,\ldots,x_M)^T \in \mathbf{R}^M$ that satisfy (3.4):

$$a_i \leq x_i \leq b_i \tag{3.4}$$

for all $i=\overline{1,M}$.

The definition of a range establishes a causal relationship between the sets of active and passive bounds, expressing the latter as the functions of the former. This causal relationship is reflected in the notation (3.3).

It is clear from the definition of a range that the box corresponding to the passive bounds is the *minimal box* containing the intersection of set Ψ with the box corresponding to the active bounds. In other words, the passive bounds are the bounds of a piece of Ψ, that is located inside of the box determined by the active bounds of the range. A range in 2D can be easily visualized, Figs. 3.5 - 3.6 show examples of two ranges associated with the same bounded set in $\mathbf{R}^2$.

To declare a range (associated with some set Ψ) it is enough to specify its active bounds. It follows from the definition, that given two ranges $(\mathbf{L}^A,\mathbf{R}^A \xrightarrow{\Psi} \mathbf{L}^P,\mathbf{R}^P)$ and $(\mathbf{L}'^A,\mathbf{R}'^A \xrightarrow{\Psi} \mathbf{L}'^P,\mathbf{R}'^P)$, such that

$$\begin{matrix} L_i^A \le L_i'^A \le L_i^P \\ R_i^P \le R_i'^A \le R_i^A \end{matrix} \quad (i=\overline{1,M}) \tag{3.5}$$

then

$$\begin{matrix} L_i'^P = L_i^P \\ R_i'^P = R_i^P \end{matrix} \quad (i=\overline{1,M})$$

In other words, if the active bounds of some range (marked with $'$) are located between the respective passive and active bounds of another range, then these two ranges have the same passive bounds.

As the result, the range with the "tightest" active bounds out of a set of all ranges that have given passive bounds $\mathbf{L}^P$ and $\mathbf{R}^P$, is a cleared range $(\mathbf{L}'^A,\mathbf{R}'^A \xrightarrow{\Psi} \mathbf{L}'^P,\mathbf{R}'^P)$ such that:

$$\begin{matrix} \mathbf{L}'^P = \mathbf{L}'^A = \mathbf{L}^P \\ \mathbf{R}'^P = \mathbf{R}'^A = \mathbf{R}^P \end{matrix} \quad (i=\overline{1,M}) \tag{3.6}$$

If set Ψ is bounded, then there exists a minimal cleared range containing the entire set Ψ. This range is called the *minimal containing* range of a set, and its bounds are defined by (3.7):

$$\begin{aligned} \mathbf{L}^P &= \mathbf{L}^A = \min_{y \in \Psi} y_i \\ \mathbf{R}^P &= \mathbf{R}^A = \max_{y \in \Psi} y_i \end{aligned} \quad (i=\overline{1,M}) \qquad (3.7)$$

The bounds of the minimal containing range simply define the global bounds of set Ψ.

In the process of exploring Ψ, the DM interactively selects a sequence of ranges by defining their active bounds. The computer in turn generates and displays to the DM the corresponding passive bounds. Interactively trying out different sets of active bounds and observing the evolution of the corresponding passive bounds provides the DM with valuable insight into the global structure of Ψ. If the attainability set changes with time, the passive bounds are recomputed as frequently as possible to keep up with the changing attainability set, even when the DM does not change the active bounds.

It is important to highlight a particular group of ranges that consists of the ranges with fixed values of some of the active bounds. A class of this ranges has a special significance to decision-aiding. These are the ranges with some of the active bounds infinite and unchangeable.

For example, consider some range (3.3), such that:

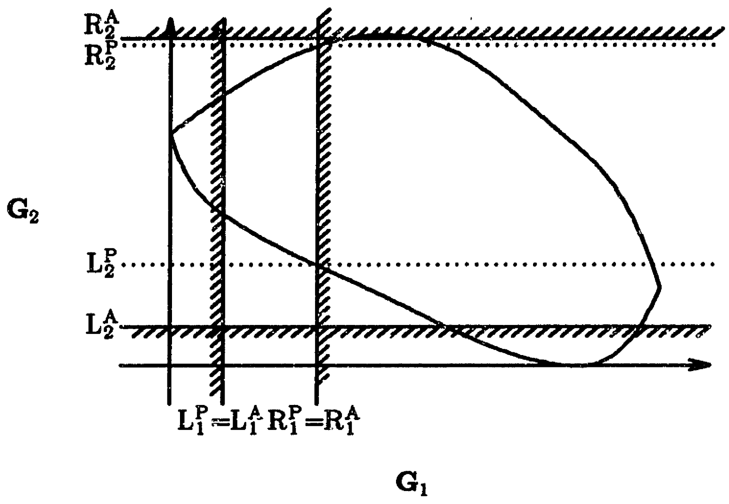


Figure 3.5. Example of a range in $\mathbf{R}^2$.

The lines with shadings correspond to the active bounds. The passive bounds of G_1 coincide with the corresponding active ones.

$$\begin{aligned} &L_k = -\infty \quad (k = i_1^-,\ldots,i_k^-) \\ &\text{and} \\ &L_j^A = \infty \quad (j = i_1^+,\ldots,i_r^+) \end{aligned} \tag{3.8}$$

where k spans a set of indexes corresponding to the infinite unchangeable left bounds, and j spans a set of indexes corresponding to the infinite unchangeable right bounds.

It is clear from (3.1) that these infinite-valued bounds can be entirely neglected since they do not affect the passive bound computations. In effect, the presence of an unchangeable infinite active bound for some dimension, i, is equivalent to having just one active bound for that dimension. A range with some of the active bounds being infinite and unchangeable is called a *mixed range*. A range $(\mathbf{L}^A,\mathbf{R}^A \xrightarrow{\Psi} \mathbf{L}^P,\mathbf{R}^P)$ that has an infinite unchangeable bound for every of its dimensions, is called a *directional range*. Without a loss of

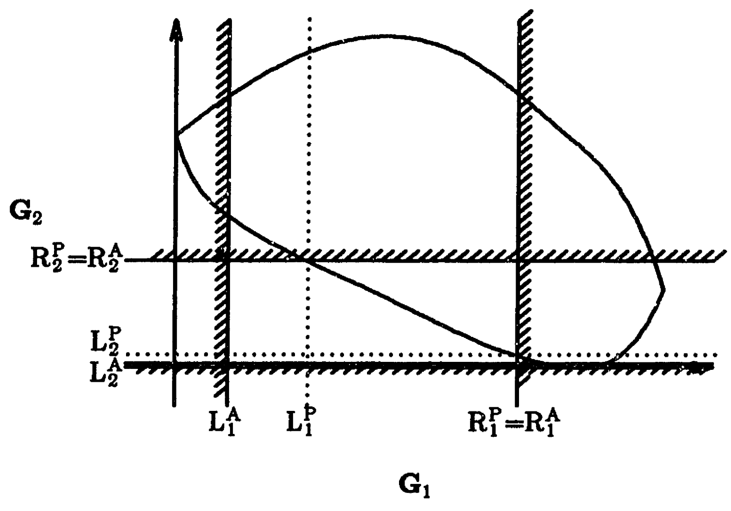


Figure 3.6. Example of a range in $\mathbf{R}^2$.

The lines with shadings correspond to the active bounds. The right passive bounds coincide with the corresponding active ones.

generality it will be assumed hereafter that these are the left bounds of a directional range that are infinite and unchangeable:

$$L_i^A = -\infty \quad (i=\overline{1,M})$$

The directional ranges are useful for helping the DM solve a MCDM problem. Specifically, if all target functions are preferentially independent, the DM needs to specify only one goal value, g_i^+, for each of the target functions. This corresponds to a range with only right active bounds, i.e. to a directional range. The goal values thus serve as the right active bounds of the ranges used for the exploration of the attainability set, i.e. $R_i^A = g_i^+$.

To account for the presence of the dimensions with one and two active bounds, the following notation is used. We introduce the *type of a range* as a sequence of M numbers corresponding to the M dimensions of $\mathbf{R}^M$ containing set Ψ. Each number in the sequence is either ± 1, or 2. The i-th number of the range type corresponds to the number of the active bounds used for the i-th dimension. If the i-the number is equal to -1 then it indicates that only the left active bound, L_i^A, is specified for the i-th dimension and the right bound is unchangeable and infinite ($R_i^A = \infty$). Similarly if the i-th number is equal to +1 then $L_i^A = -\infty$ and the i-th dimension has only the right active bound, R_i^A.

3.1.3 Range Properties

The usefulness of the ranges for the approximation and exploration of the attainability set comes from several properties of this construction. The following two lemmas reflect some of the properties. The first lemma has direct implications for the design of the range-based computer aid for a MCDM problem. The second lemma impacts a smaller class of applications, namely the ones with a convex attainability set.

Lemma 1 (Pareto Convergence): Lower passive bound L_i^P of (3.1) of a directional range with finite active upper bounds are the same whether the minima (3.1) are taken over the entire set Ψ or over its Pareto subset $\Psi^{Par} \subseteq \Psi$.

Proof: We consider set Ψ and some range *Range1* = $(\mathbf{L}^A, \mathbf{R}^A \xrightarrow{\Psi} \mathbf{L}^P, \mathbf{R}^P)$ associated with it. Let's consider another range, *Range2* = $(\mathbf{L}^A, \mathbf{R}^A \xrightarrow{\Psi^{Par}} \mathbf{L}^{Par,P}, \mathbf{R}^{Par,P})$ associated with Ψ^{Par} and having the same active bounds as *Range1*.

From the definition of a range:

$$L_i^{Par,P} = \min_{y \in \Psi^{Par}} y_i \tag{3.9}$$

subject to conditions (3.2). Minimization (3.9) over a subset, Ψ^{Par}, cannot produce a result lower than the result of the minimization (3.1) over the entire set Ψ. Therefore

$$L_i^{Par,P} \geq L_i^P$$

To show that $L_i^{Par,P} = L_i^P$, we assume the contrary, i.e. that for some target function k:

$$L_k^{Par,P} > L_k^P \tag{3.10}$$

This implies that a set of points from Ψ which actually deliver the minimum to L_k^P does not intersect with the Pareto subset Ψ^{Par}.[2] In other words, any point $\mathbf{y}' \in \Psi$ such that

$$y'_k = L_k^P \tag{3.11}$$

does not belong to the Pareto subset: $\mathbf{y}' \notin \Psi^{par}$. This contradicts the definition of the Pareto subset of Ψ, since every point $\mathbf{y}'$ satisfying (3.11) is better than any other point of Ψ in respect to the k-th criterion. Therefore at least one of the

2. We must assume that there can be more than a single point from set Ψ that delivers each minimum (and maximum) in (3.1).

points satisfying (3.11) *must* belong to the Pareto set. This contradiction proves the Lemma.

The Lemma has an important implication for design of the human-computer interface for aiding the DM in solving a MCDM problem. Let's imagine that the values of all active bounds of some directional range are continuously decreased. Suppose that the active bounds are being decreased with some constant speed (which might be different for different target functions). As the consequence of this decrease, the passive bounds increase. At some point of time the active bounds will collide with the corresponding passive bounds and the range will degenerate into a single point **S** in Ψ:

$$L_i^P = R_i^P = S_i$$

for all $i=\overline{1,M}$.

It follows from Lemma 1 that this point, **S**, must belong to the Pareto boundary of Ψ. It is also easy to show that any point from the Pareto boundary can be obtained as a limiting location of the passive bounds of a range, when its active bounds are lowered appropriately.

In light of this result, the DM can actually scan the Pareto set, by interactively "sliding" the active bounds up and down. The computer in turn would display the obtaining passive bounds. This technique allows a dynamical scanning of the Pareto subset of the problem, similar to the scanning of an object with a spotlight.

If the passive bounds of a range have not degenerated into a point, there is a gap between some of the left and right passive bounds. It is clear that for every passive bound, L_j^P (or R_j^P), $(j=\overline{1,M})$ there is always a point from the attainability set, $\mathbf{y} \in \Psi$, located "on the j-th side" of the passive bound box: $y_j = L_j^P$ (or $y_j = R_j^P$). Since it is often desirable that the goal defined by the active bounds was robust to the future perturbations, the question is whether the passive bounds really confine other points of the attainability set, i.e. whether there are any points of the attainability set that are located ***strictly inside*** of the passive bound box.

To clarify this point consider the following example. Set $\Psi \subset \mathbf{R}^2$ consists of only 2 points A and B with coordinates (1,1) and (2,2) respectively, and the active bounds of the range are:

$$\begin{array}{ll} L_1^A = 0 & L_2^A = 0 \\ R_1^A = 3 & R_2^A = 3 \end{array}$$

The passive bounds of this range are respectively:

$$\begin{array}{ll} L_1^P = 1 & L_2^P = 1 \\ R_1^P = 2 & R_2^P = 2 \end{array}$$

see (Fig. 3.7).

It is clear that though the passive bound box contains two points of the set, Ψ, this box has no points of Ψ strictly inside of it. The question therefore is: when does the passive bound box contain points of the attainability set that are located away from the boundary and strictly inside of the box?

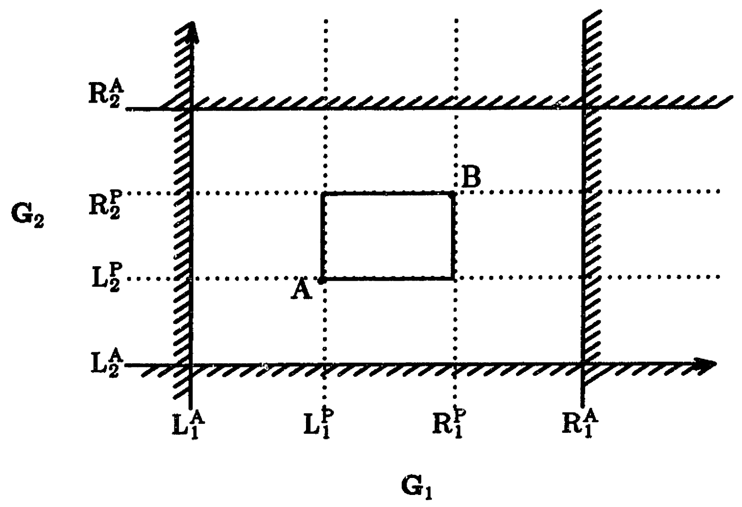


Figure 3.7. An example of a range with no internal points inside of the passive bound box.

The following Lemma gives the answer to this question under some special circumstances:

Lemma 2 (Passive Box Boundary Proximity): Suppose all passive bounds L_i^P, and R_i^P of a range are finite, and Ψ is convex. Then there is at least one point $\mathbf{f} = (f_1,...,f_M)^T \in \Psi$ inside of an M-dimensional box with the following bounds:

$$\begin{aligned} L_i^P + \Delta_i \leq f_i \leq R_i^P - \Delta_i \\ \text{where: } \Delta_i = \frac{1}{M}(R_i^P - L_i^P) \end{aligned} \tag{3.12}$$

for all $i = \overline{1,M}$.

It is clear that there are always some points of Ψ that are located on the sides of the passive bound box. Lemma 2 establishes the existence of at least one strictly *internal* point of Ψ inside of the passive bound box.

The Lemma states that if a convex closed set in $\mathbf{R}^M$ is contained inside of a (passive bound) box and intersects with every side of the box, then there is a point of the set, that is located inside of the box, at a distance from every side. The minimum distance of that point from some side of the box is equal to $\frac{1}{M}$th of the distance from that side to the opposite side of the box, i.e. to the side that is parallel to the first one.

If $M = 2$, then the Lemma is equivalent to the following statement: any convex closed 2D set that intersects with all sides of a rectangle, always contains the middle point of that rectangle (see Fig. 3.8).

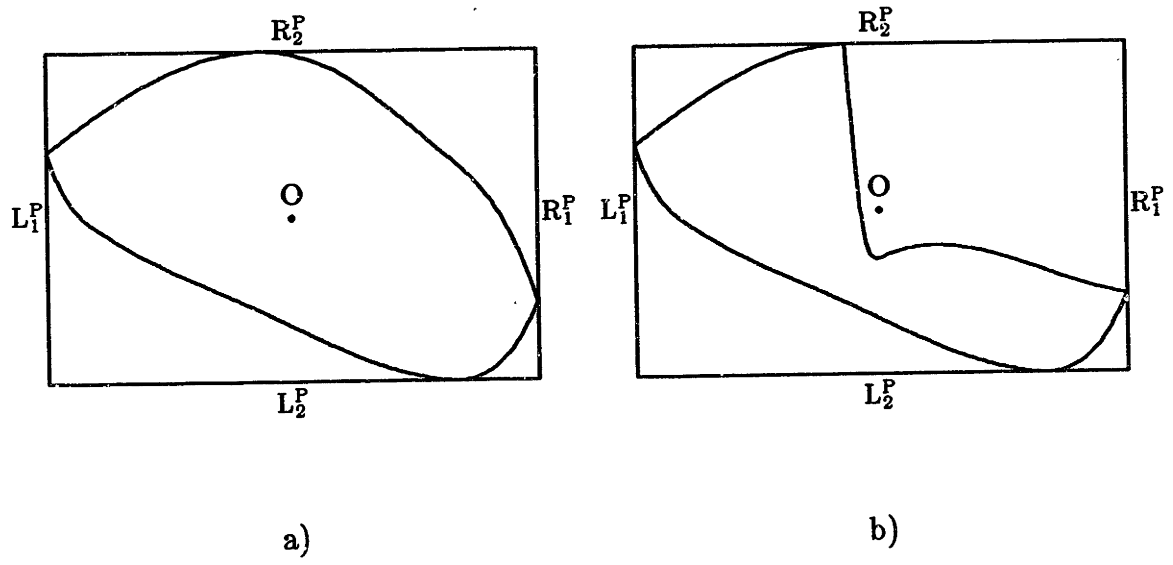


Figure 3.8. 2D illustration of Lemma 2.

a) Convex set contains the middle point O of the surrounding rectangle.

b) Non-convex set does not contain the middle point O.

The proof of this Lemma can be found in Appendix A.

3.1.4 Range Approximation of the Attainability Sets

The discussion about the ranges applies to any set in $\mathbf{R}^M$. Implicitly it was assumed that Ψ actually was the attainability set. However, expressions (3.1) - (3.2) can be made explicit using the notation of Chapter 1.3.

Suppose that there is some system described by equations (1.8) and performing a task defined by goal $[\mathbf{g}^-,\mathbf{g}^+]$. We can construct a range $(\mathbf{L}^A,\mathbf{R}^A \xrightarrow{\Psi} \mathbf{L}^P,\mathbf{R}^P)$ associated with the attainability set of the task, whose active bounds are defined by the goal specifications:

$$\begin{cases} \mathbf{g}^- = \mathbf{L}^A \\ \mathbf{g}^+ = \mathbf{R}^A \end{cases} \tag{3.13}$$

The passive bounds of such a range are calculated using (3.14), the analogues of (3.1):

$$\begin{aligned} L_i^P &= \min_{t^*,\mathbf{u}_{t_1,[t_0,t^*]}} G_i(t_0,t^*,\mathbf{x}(t_0),\mathbf{u}_{t_1,[t_0,t^*]},\mathbf{p}_{t_1,[t_0,t^*]}) \\ R_i^P &= \max_{t^*,\mathbf{u}_{t_1,[t_0,t^*]}} G_i(t_0,t^*,\mathbf{x}(t_0),\mathbf{u}_{t_1,[t_0,t^*]},\mathbf{p}_{t_1,[t_0,t^*]}) \end{aligned} \quad (i=\overline{1,M}) \tag{3.14}$$

subject to:

$$\begin{cases} L_j^A \leq G_i(t_0,t^*,\mathbf{x}(t_0),\mathbf{u}_{t_1,[t_0,t^*]},\mathbf{p}_{t_1,[t_0,t^*]}) \leq R_j^A \quad (j=\overline{1,M}) \\ \mathbf{u}_{t_1,[t_0,t^*]}(t) \in \Omega_u \end{cases} \tag{3.15}$$

and the evolution equations (1.8).

These conditions can be written in a shortcut form:

$$\left.\begin{aligned} L_i^P &= \min_{\chi \in \Omega_\chi} G_i(\chi) \\ R_i^P &= \max_{\chi \in \Omega_\chi} G_i(\chi) \end{aligned}\right. \quad (i=\overline{1,M}) \tag{3.16}$$

subject to:

$$L_j^A \leq G_i(\chi) \leq R_j^A \quad (j = \overline{1,M}) \tag{3.17}$$

Where the argument, χ, is a tuple $(t^*, \mathbf{u}_{t_1,[t_0,t^*]})$ and Ω_χ is a set in the functional space that encapsulates the set of all admissible controls Ω_u satisfying (1.8). This shortcut form (3.16) - (3.17) will be used throughout the rest of this chapter.

The passive bounds (3.14) determine the approximation of the attainability set inside of the goal bounds. It is evident that no part of the attainability set is located inside of the gap between the active and the passive bound boxes. The implementation of the human-computer decision-making interface that utilizes the ranges is discussed in Chapter 4.

3.2 Maximal Satisficing Sets

If the attainability set changes with time, then this set might "slip out" of the active bound box. When that happens the goal (3.13) becomes unattainable and the set of conditions (3.17) becomes inconsistent. As the consequence, range (3.16) - (3.17) becomes infeasible. At this point the DM needs to change the active bounds so that they would define another, an attainable goal.

The DM can find an attainable range by reducing some of the left bounds of the infeasible range while increasing some of the right bounds. The question is:

how much should the active bounds of the existing range be changed so that the new range will be feasible (and therefore will define an attainable goal)?

In most cases the DM would want to change the original (infeasible) range as little as possible. However, to find out what minimum changes the infeasible range has to undergo to become feasible, one needs to have a measure of proximity of one range from another. In the DRT method the following measure of the range proximity is used: ***the number of the bounds of an infeasible range*** which must be changed to make the range feasible. All feasible ranges that preserve the maximum number of the active bounds of the original infeasible range are automatically generated and presented to the DM. These ranges are called the *maximal satisficing ranges* and they are closely connected to the *maximal feasible subsets* of constraints.

Below is a definition of the maximal feasible subset of constraints associated with some set Ψ in $\mathbf{R}^M$ and range $(\mathbf{L}^A, \mathbf{R}^A \xrightarrow{\Psi} \mathbf{L}^P, \mathbf{R}^P)$.

Definition: Given a system of M pairs of inequalities (3.18):

$$\begin{cases} L_i^A \leq G_i(\chi) \leq R_i^A \quad (i=\overline{1,M}) \\ \chi \in \Omega_\chi \end{cases} \tag{3.18}$$

any subset of k of these pairs of inequalities $(j = i_1,..,i_k)$ is called a *maximal feasible subset of constraints* (3.18) iff:

1. there exists a solution of the reduced system (3.19) of inequalities:

$$\begin{cases} L_j^A \le G_j(\chi) \le R_j^A, & (j = i_1, \ldots, i_k) \\ \chi \in \Omega_\chi \end{cases} \tag{3.19}$$

2. the addition of any pair of constraints (3.20) from (3.18)

$$L_\nu^A \le G_i(\chi) \le R_\nu^A \quad \text{where } \nu \notin (i_1, \ldots, i_k) \tag{3.20}$$

   to (3.19) makes the augmented set of constraints ((3.19) + (3.20)) infeasible.

Thus for any infeasible range with active bounds L_i^A, R_i^A there is a feasible range $L_i'^A, R_i'^A$ that retains the maximum number of the original active bounds[3]. The active bounds of this range are composed of the bounds comprising the maximal feasible subset of constraints of (3.18), i.e. of the retained bounds of the original range. The rest of the active bounds are omitted or, which is the same, are set to ±infinity. In other words, if (3.19) is the maximal feasible set of constraints, then $\mathbf{L}'^A, \mathbf{R}'^A$ can be interpreted as active bounds of a feasible range (3.21) - (3.22):

3. This definition and the subsequent discussion can be changed if each pair of constraints $L_i^A \le G_i(\chi) \le R_i^A$ were considered as two independent constraints. Then the total number of constraints in (3.18) would have been 2M. The increase of the number of constraints could result in some increase of the computational time. It is yet to be studied whether the change from considering pairs of constraints to considering individual constraints would be beneficial to the decision aid. It is however preferred in this thesis to consider these constraints always in pairs, since it is a pair of constraints that defines a range of goal values of a target function. Obviously this problem does not arise in the MCDM problems where all target functions have only one active bound.

$$\begin{cases} L_j^{\prime A} = L_j^A \\ R_j^{\prime A} = R_j^A \end{cases} \quad \text{for } j \in (i_1,\ldots,i_k) \tag{3.21}$$

and

$$\begin{cases} L_\nu^{\prime A} = -\infty \\ R_\nu^{\prime A} = \infty \end{cases} \quad \text{for } \nu \notin (i_1,\ldots,i_k) \tag{3.22}$$

The retained bounds $j \in (i_1,\ldots,i_k)$ are the *activated* bounds of the maximum feasible set (and the corresponding target functions are the activated target functions), while those bounds (with indexes $\nu \notin (i_1,\ldots,i_k)$) that are not retained, are the non-activated bounds.

Using active bounds $\mathbf{L}^{\prime A}$ and $\mathbf{R}^{\prime A}$ defined in (3.21) - (3.22), one can compute the passive bounds $L_i^{\prime P}, R_i^{\prime P}$ of another range, $(\mathbf{L}^{\prime A}, \mathbf{R}^{\prime A} \xrightarrow{\Psi} \mathbf{L}^{\prime P}, \mathbf{R}^{\prime P})$, and then use these passive bounds as the active bounds instead of the non-activated bounds of the original range, $(\mathbf{L}^A, \mathbf{R}^A \xrightarrow{\Psi} \mathbf{L}^P, \mathbf{R}^P)$, thus defining a new range (3.23) - (3.24):

$$\begin{cases} L_j^{\prime\prime A} = L_j^{\prime A} = L_j^A \\ R_j^{\prime\prime A} = R_j^{\prime A} = R_j^A \end{cases} \quad \text{for } j \in (i_1,\ldots,i_k) \tag{3.23}$$

$$\begin{cases} L_\nu^{\prime\prime A} = L_\nu^{\prime P} \\ R_\nu^{\prime\prime A} = R_\nu^{\prime P} \end{cases} \quad \text{for } \nu \notin (i_1,\ldots,i_k) \tag{3.24}$$

The last range is called the *maximal satisficing range* or the *maximal satisficing set* (MSS) of the active bounds of the original infeasible range. This range has a special meaning: it reflects the minimum necessary modifications that must be done to a goal (defined by the active bounds of the original infeasible range) to make it attainable.

There could be more than one maximal satisficing set corresponding to the same active bounds. In fact, any subset of the pairs of inequalities (3.18) potentially can form a maximal feasible set of constraints and therefore define a maximal satisficing set. However, though the total number of different subsets of the set of M pairs of inequalities (3.18) is 2^M, the total number of the maximal satisficing sets cannot exceed $\binom{M}{K}$, where $K = [\frac{M}{2}]$.[4] A proof of this fact can be found in Appendix B.

Though the maximal possible number number of the maximal satisficing sets can be quite big, in practice there are much fewer alternative MSSs than the maximal number. All of these MSSs can be effectively generated and presented to the DM for her consideration.

Maximal satisficing sets provide a useful tool to the DM in her search for an attainable goal. Maximal satisficing ranges can be generated every time when

4. Here $\binom{m}{k}$ is a binomial coefficient, $\binom{m}{k} = \frac{m!}{k!(m-k)!}$, and $[n]$ denotes the maximal integer not exceeding n.

the range (3.14) - (3.15) defined by the task goal (3.13) becomes infeasible. When that happens and the goal becomes unattainable, the DM is given the option to scan all maximum satisficing ranges (3.23) - (3.24) and select any of them as a baseline for selecting a new goal. By providing the DM with all MSSs the computer effectively enunciates: "the present goal is unattainable, the range is infeasible. In order to select an attainable goal, some of the present goal specifications have to be changed. These are all maximal combinations of the old specifications that still can be kept". An effective recursive algorithm for generating all MSSs is described in the next section.

Let's consider a few examples:

1. Consider set Ψ shown on Fig. 3.9 and the goal region ABCD. This goal is clearly unattainable and there is just one MSS, whose active bounds are defined by rectangle KLMN. (The bounds of the ranges on this picture are denoted in the same way as in (3.23) - (3.24).) In this example the 2-nd pair of inequalities in (3.18) has to be changed. The only activated constraint of the maximal satisficing set in this example is $L_1^A \leq G_1 \leq R_1^A$, while constraint $L_2^A \leq G_2 \leq R_2^A$ is non-activated and is replaced with $L_2^{\prime\prime A} \leq G_2 \leq R_2^{\prime\prime A}$.

2. A different goal, such as for example the one on Fig. 3.10, has two maximum satisficing sets, whose active bounds are KLMN and TUVW. The former maximal satisficing set corresponds to the case when the 1-st pair of constraints in (3.18), $L_1^A \leq G_1 \leq R_1^A$, is activated. The latter MSS corresponds to the case when the 2-nd pair of constraints, $L_2^A \leq G_2 \leq R_2^A$, is activated.

3. Consider a simple analytical example for a case M=3.

Suppose that the the set of controls are variables $(x,y) \in \mathbf{R}^2$, the system does not explicitly depend on time, and the target functions are:

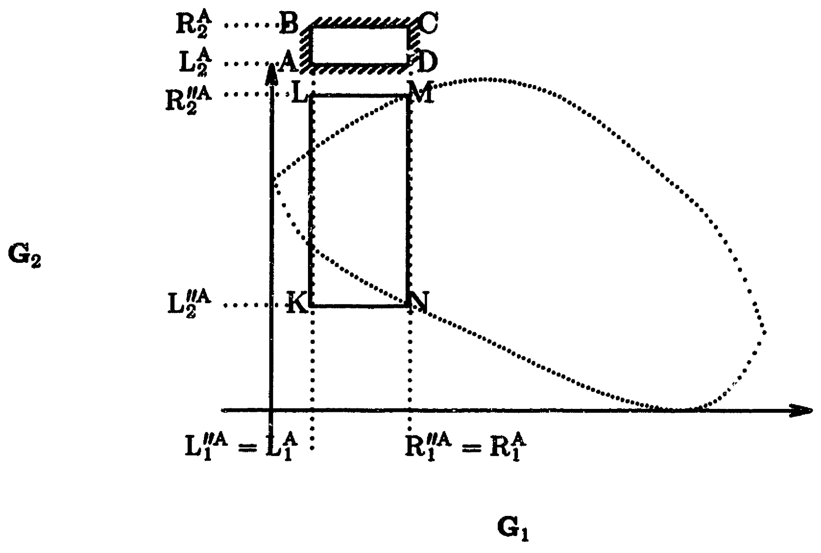


Figure 3.9. Goal region ABCD and the maximal satisficing range KLMN

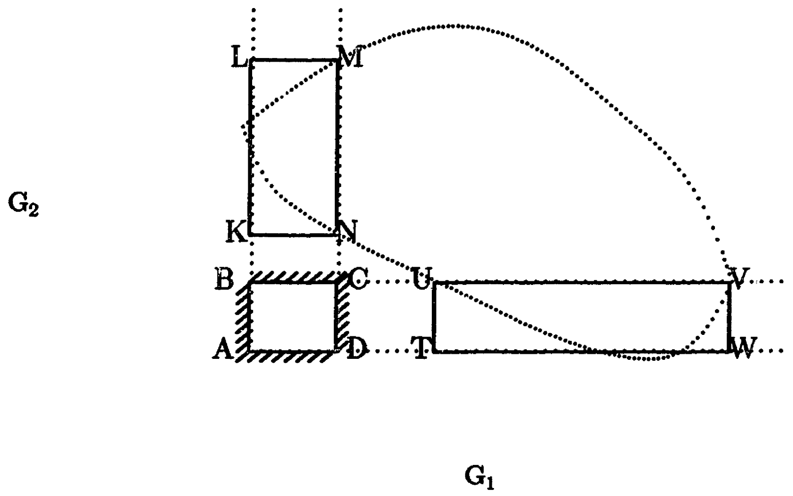


Figure 3.10. Goal region ABCD and its two maximal satisficing ranges KLMN and TUVW.

$$\begin{cases} G_1(x,y) = x+y \\ G_2(x,y) = x \\ G_3(x,y) = y \end{cases}$$

Suppose that the infeasible goal is:

$$\begin{array}{ll} L_1^A = 12 & R_1^A = 15 \\ L_2^A = 0 & R_2^A = 5 \\ L_3^A = 4 & R_3^A = 6 \end{array}$$

and conditions (3.18) become

$$\begin{array}{rcl} L_1^A = 12 \leq & x + y & \leq 15 = R_1^A \\ L_2^A = 0 \leq & x & \leq 5 = R_2^A \\ L_3^A = 4 \leq & y & \leq 6 = R_3^A \end{array} \tag{3.25}$$

In this example there are 3 MSS:

The first one corresponding to the 2-nd and 3-rd pairs of constraints of (3.25) activated:

$$\begin{array}{ll} L_1^A = 4 & R_1^A = 11 \\ L_2^A = 0 & R_2^A = 5 \\ L_3^A = 4 & R_3^A = 6 \end{array}$$

the second one corresponding to the 1-st and 3-rd pairs of constraints of (3.25) activated:

$$\begin{array}{ll} L_1^A = 12 & R_1^A = 15 \\ L_2^A = 6 & R_2^A = 11 \\ L_3^A = 4 & R_3^A = 6 \end{array}$$

and the third one corresponding to the 1-st and 2-nd pairs of constraints of (3.25) activated:

$$\begin{array}{ll} L_1^A = 12 & R_1^A = 15 \\ L_2^A = 0 & R_2^A = 5 \\ L_3^A = 7 & R_3^A = 15 \end{array}$$

3.2.1 Algorithm for Computation of All MSSs

The problem of finding the biggest MSS, i.e. a set that satisfies the largest number of inequalities from (3.18), is a well known problem (see e.g.[33]). An new method presented here finds **all** MSSs of the inequalities (3.18). Below is the description of the computational algorithm for finding all MSSs.

Let's consider all subsets of the pairs of inequalities (3.18). There are a total of 2^M of such subsets. Each subset is uniquely defined by a list of indexes, $(j = i_1,..,i_k)$, of those pairs of inequalities from (3.18) that belong to that subset. Let's introduce for every subset an M-digit long binary number J, that will be referred to as the *subset indicator number:*

$$J = \{n_1,...,n_M\} \tag{3.26}$$

The j-th digit of the subset indicator number is 1 $(n_j = 1)$, if constraint $L_j^A \leq G_j(\chi) \leq R_j^A$ is present in the subset and $n_j = 0$ otherwise. Let's call a *cardinality,* $|J|$, of an indicator number J (3.26) the total number of digits of J that are equal to 1 (this is the same as the cardinality of the subset itself):

$$|J| = \sum_{i=1}^{M} n_i \tag{3.27}$$

We will hereafter freely interchange all statements about subsets of inequalities and their corresponding indicator numbers. For instance, we would say that one number *is a subset* of another number, if a subset of inequalities corresponding to the former number belongs to the subset of inequalities corresponding to the latter number.

A set of all subsets of inequalities comprise a partially ordered set. The partial ordering is naturally induced by the inclusion $\subseteq$ operation. This partial ordering of the subsets of (3.18) can be translated into a partial ordering of the indicator numbers. Specifically, $J_1 \geq J_2$ if J_1 has digit 1 in all positions (in the expression for the indicator number (3.26)) where J_2 does. Or in other words, $J_1 \geq J_2$ if all pairs of inequalities that belong to the subset corresponding to J_2 also belong to the subset corresponding to J_1.

The algorithm for finding all MSS's is based on a recursive search through all subsets of (3.18) with a concurrent bookkeeping to avoid redundant operations. The search is organized in a form of a tree, Γ, each node of which contains some indicator number. We will refer to this number as the node's number. We will also refer to the node in the same fashion as to the subsets of inequalities (3.18), i.e. a node will be referred at as *feasible*, *infeasible*, or *maximal.*

The root of Γ contains $\{0...0\}$. Any node with the node's number having 1 as its last digit is a terminal node. All other nodes have children nodes. A child's cardinality is 1 larger than the cardinality of its parent. Children of a node are generated by successively replacing vacant 0, in positions to the right of the rightmost 1 in the parent node, with 1. Fig. 3.11 shows an example of tree Γ corresponding to $M = 4$.

Nodes of the tree Γ are recursively traversed in a depth-first order starting from the root. A list of already found MSSs is maintained during the execution. Each node can be entered several times: first time it is entered on the way from its parent, other times it is re-entered returning from its children. Every return from a child is accompanied by a message indicating whether that child is feasible or infeasible. The algorithm proceeds in the following order:

1. Initialization. A list of MSSs is initialized as empty. The root is entered as the first node.

2. Upon entering a node for the first time (from the parent) the following operations are performed:

   - The node is compared against the list of already found MSSs to find out whether it is a feasible but non-maximal node (a subset of some already found MSS).[5] Based on this comparison the following is done:

     — If it is a terminal feasible non-maximal node - then "FEASIBLE" is returned to the parent node.

     — If it is a non-terminal feasible non-maximal node - then the next child is entered.

   - If no information about the node's feasibility can be obtained from the list of already found MSSs, its feasibility is explicitly tested (the test itself is discussed below). Based on the feasibility test the following is done:

     — If this node is feasible and terminal - then it is an MSS. It is added to the list of MSSs and then "FEASIBLE" is returned to its parent.

     — If it is a feasible but not a terminal node - then the next child is entered.

     — If the node is infeasible - then "INFEASIBLE" is returned to the parent node.

3. Upon re—entering a node from its child the following is performed:

- If all children have not been explored yet - then the next child is entered.
- If all children have been explored and all of them returned "INFEASIBLE", then the current node is an MSS. It is added to the list of MSSs and "FEASIBLE" is returned to the parent.

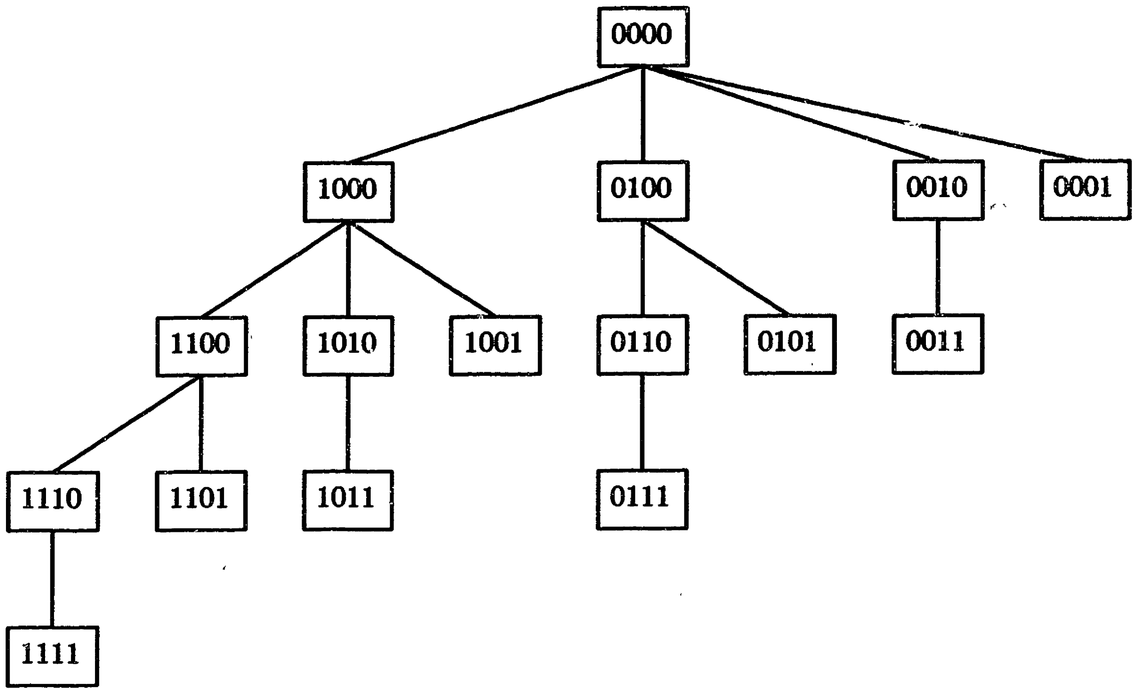


Figure 3.11. An MSS search tree for M=4.

5. A node J′ is a feasible non-maximal node if there exists some already established MSS node J_m (i.e. J_m belongs to the list of already found MSS nodes) such that $J' \wedge (\neg J_m) = 0$. Here $\neg$ is a bitwise *negation* operation on the digits of a number, and $\wedge$ is a bitwise *and* operation.

The tree Γ is a sorted tree (a sorted oriented graph) of the partial ordering of the set of all indicator numbers. The algorithm is based on the following, easily established facts: 1) if some node is infeasible then all of its children are also infeasible; 2) if a node is a maximal feasible node, then all of its children are infeasible; 3) the maximal feasibility of a node can be determined solely based on the feasibility of its children and the comparison with the list of previously found maximal nodes. The nodes from other, not yet explored branches of Γ, do not affect the current node's status.

The most costly (timewise) procedure in the algorithm is establishing whether a particular node is feasible. However the sequential nature of the algorithm permits substantial reduction of the computations involved in determining the node's feasibility.

To demonstrate this computational advantage, let's suppose that some node J' has been determined feasible, i.e. a solution, χ', has been found that satisfies the feasibility condition (3.28) for this node:

$$\begin{cases} L_j^A \leq G_j(\chi') \leq R_j^A \quad (j \in I_{J'}) \\ \chi' \in \Omega_\chi \end{cases} \tag{3.28}$$

where

$$j \in I_{J'} = \{i_1, i_2, \ldots, i_k\} \tag{3.29}$$

and $I_{J'}$ is the set of all positions in which binary number J' has 1. (I.e. $I_{J'}$ is a set of the indexes of all activated pairs of inequalities in node J'.)

Let's denote J″ - some child of J′. This child has just one additional pair of constraints, say the m-th, that makes it different from J′. In other words, the constraints for J″ include all constraints (3.28) plus the m-th pair of constraints of (3.18):

$$\begin{cases} L_j^A \le G_j(\chi) \le R_j^A \quad (j \in I_{J'}) \\ L_m^A \le G_m(\chi) \le R_m^A \\ \chi \in \Omega_\chi \end{cases} \tag{3.30}$$

where $m \notin I_{J'}$.

To establish whether (3.30) is consistent, we first test if χ', the solution of the parent node, satisfies the new constraint, i.e. we test if

$$L_m^A \le G_m(\chi') \le R_m^A \tag{3.31}$$

If (3.31) is true, then (3.30) is obviously consistent and the child J″ is feasible. If (3.31) is not satisfied, then either

$$G_m(\chi') < L_m^A \tag{3.32}$$

or

$$G_m(\chi') > R_m^A \tag{3.33}$$

If (3.32) is true, then consider the solution of the following maximization problem (3.34):

$$G^+ = \max_{\chi \in \Omega_\chi} G_m(\chi)$$
$$\text{subject to:} \qquad (3.34)$$
$$\begin{cases} L_j^A \leq G_j(\chi) \leq R_j^A \quad (j \in I_{J'}) \\ G_m(\chi) \leq R_m^A \end{cases}$$

where $j \in I_{J'}$.

The following fact makes finding the solution of (3.34) convenient from the computational point view. Since χ' already is known to satisfy the conditions of the maximization problem (3.34), then any iterative process of solving (3.34) can be started from $\chi_1 = \chi'$. Moreover, the iterative optimization process might not have to be carried out all the way, since if at some iteration we get a feasible (as far as the conditions of optimization (3.34) are concerned) point χ_n such that $G_m(\chi_n) \geq L_m^A$, then this signals that the child, J'', is feasible and χ_n itself can be used in place of χ' for testing (if needed) the children of J''.

If on the other hand, the optimization has to be carried all the way[6] and the solution of (3.34) is such that

$$G^+ \geq L_m^A$$

then the node J'' is feasible, otherwise it is not.

6. The problem of establishing the necessary accuracy of the optimization process might be a rather difficult one. This issue is outside of the scope of this thesis.

Similarly, if (3.33) is true, the following minimization problem (3.35) can be solved:

$$G^{-} = \min_{\chi \in \Omega_\chi} G_m(\chi)$$
$$\text{subject to:} \tag{3.35}$$
$$\begin{cases} L_j^A \leq G_j(\chi) \leq R_j^A \quad (j \in I_{J'}) \\ G_m(\chi) \geq L_m^A \end{cases}$$

Minimization (3.35) can be solved in a similar iterative manner, starting from a feasible point $\chi_1 = \chi'$. As in the case of maximization (3.34), the existence of an intermediate solution χ_n such that $G_m(\chi_n) \leq R_m^A$ signals the feasibility of J''. If on the other hand, the solution of (3.35) is such that $G^{-} > R_m^A$, then the node J'' is infeasible.

The just described procedure is directly applicable to the case of a directional range. In this case each pair of constraints (3.18) degenerates into a single constraint $G_i(\chi) \leq R_i^A$. As the result, the only optimization problem that is necessary to solve in this case is (3.36):

$$G^{-} = \min_{\chi \in \Omega_\chi} G_m(\chi)$$
$$\text{subject to:} \tag{3.36}$$
$$G_j(\chi) \leq R_j^A \quad (j \in I_{J'})$$

If $G^{-} \leq R_m^A$, or an intermediate solution, χ_n, exists such that $G_m(\chi_n) \leq R_m^A$, then the node is feasible. Otherwise it is not.

The just described algorithm finds all MSSs for the general case of Ω_χ and $G(\chi)$. A special case of resource-sharing systems is described in Chapter 5. The only time consuming part of the algorithm are the optimizations (3.35) and/or (3.34). The algorithm takes a reasonable care to minimize the number of these optimizations. The use of the parent's solution in a node as the starting point of the numerical optimization process adds a significant computational advantage. No redundant optimizations are performed since it is attempted to establish the feasibility of a child based on straightforward criteria, such as comparison with the list of already found MSSs.

The computational complexity of this algorithm is presently being studied. This complexity however is closely related to the maximum possible number of the MSSs. The latter number is obtained in Appendix B.

3.3 Pareto Cluster Cells

3.3.1 Introduction

The mechanism of the maximal satisficing sets is useful for helping the DM find an attainable goal if some goal becomes unattainable. The Pareto cluster cells technique on the other hand is a method of exploration of the entire attainability set in the case when all target functions are preferentially independent, i.e. in the case of a MCDM problem.

The idea of the method is to partition the goal space into a partially ordered set of *cells,* then to determine which cells have non-empty intersection with the attainability set (these cells are called *activated*), and finally select the Pareto subset of the activated cells.

The Pareto cluster cells (PCC) technique provides the DM with a compact snapshot of the location of the attainability set inside of the goal space. Having this snapshot, the DM can concentrate on exploring smaller regions in the goal space, each of the size of a cell. A cell then can be used as a starting point for the range exploration of its inside.

The PCC technique can be viewed as a generalization of the Maximal satisficing sets technique for directional ranges.

3.3.2 The Basics

Suppose that the DM is solving a MCDM problem. This implies that the values of each of the target functions are preferentially ordered: the DM prefers lower values of any target function to its higher values, provided that all other functions remain the same.

Suppose that a range of values of the k-th target function (i.e. the real axis) is partitioned by the points $\gamma_{k,1}$, $\gamma_{k,2}$,..., $\gamma_{k,N-1}$ into N non-overlapping adjacent intervals, so that these points are the endpoints of the intervals (Fig. 3.12).

$\gamma_{k,1}$
$\gamma_{k,2}$
...
$\gamma_{k,N-2}$
$\gamma_{k,N-1}$
G_k

Figure 3.12. Partitioning the range of values of a target function into intervals.

Each interval can be assigned a label reflecting the "goodness" to the DM of the values of the target function within that interval. Obviously, the farther an interval is located to the left, the better it is. For example, in the case of N=3, the intervals can be labeled respectively from left to right as *Good*, *Average*, and *Bad* (Fig. 3.13).

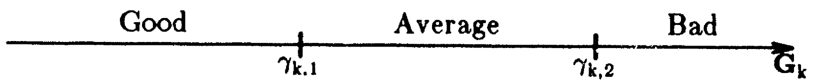


Figure 3.13. Labeling the intervals according to their *goodness*.

The partitioning of the values of each of the target functions induces a partitioning of the entire goal space into non-overlapping boxes, or *cells*. Specifically, suppose the range of values of the k-th target function is partitioned into N intervals with the endpoints $\gamma_{k,i}$, $(i = \overline{1,N-1})$. To unify the notation, we can add the infinite bounds to the list of the endpoints: $\gamma_{k,0} = -\infty$ and $\gamma_{k,N} = \infty$. Then the goal space becomes partitioned into N^M cells, where M is the number of the target functions, or the dimensionality of the goal space. Each cell is uniquely defined by a vector of endpoints, with indexes $(i_1,...,i_M)$, $(1 \leq i_k \leq N)$ of the corresponding partitions

$$\gamma = (\gamma_{1,i_1}, \gamma_{2,i_2}, ..., \gamma_{M,i_M}) \tag{3.37}$$

that bind the cell *from above.* In other words a cell that corresponds to the vector of endpoints (3.37) is described by (3.38):

$$\begin{cases} \gamma_{1,i_1-1} \leq y_1 \leq \gamma_{1,i_1} \\ \gamma_{2,i_2-1} \leq y_2 \leq \gamma_{2,i_2} \\ \dots \\ \gamma_{M,i_M-1} \leq y_M \leq \gamma_{M,i_M} \end{cases} \tag{3.38}$$

where $\mathbf{y} = (y_1,\dots,y_M)^T \in \mathbf{R}^M$ are the points of the goal space that belong to the cell. Each cell can be marked with M labels composed of the labels of the intervals of the target functions that create the cell. For instance, in the case of N=3 and M=2, there are 9 cells that can be respectively labeled as:

<good, good>	<good, average>	<good, bad>
<average, good>	<average, average>	<average, bad>
<bad, good>	<bad, average>	<bad, bad>

Cell (3.38) is called a *activated* cell associated with the attainability set Ψ, if it has a non-empty intersection with Ψ. Each activated cell can be viewed as a cluster of points in the goal space. Together all activated cells provide a *clustered* representation of the attainability set. If we imagine that all activated cells are colored while the non-activated cells are not, then the colored cells compose a cluster of cells in the goal space, that can be viewed as a cluster of cells approximating the attainability set (Fig. 3.14).

The preference ordering of the target functions values induces the preference (partial) ordering on the set of all cells. Namely, some cell (3.38) defined by the partition endpoints vector with indexes $(i_1,\dots,i_M)$ $(1 \leq i_k \leq N)$ is *preferred* to another cell defined by the endpoints vector $(j_1,\dots,j_M)$ $(1 \leq j_k \leq N)$, iff:

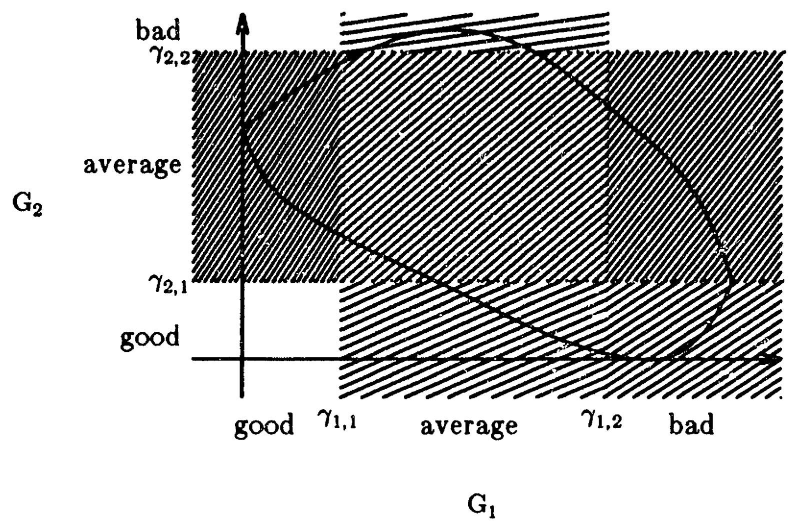


Figure 3.14. Cluster cell partitioning of the attainability set. The activated cells are shaded.

$$i_k \leq j_k \quad \text{for all } k = \overline{1,M} \tag{3.39}$$

and at least one inequality (3.39) is a strict inequality, "<".[7]

This preference ordering of the cells is a clustered approximation of the preference ordering of the points of the goal space. Evidently, as the size of the cells approaches to zero, so that each cell contains exactly one point of the goal space, then the ordering (3.39) approaches to the regular ordering of the goal space.

7. Conditions (3.39) are equivalent to $\gamma_{k,i_k} \leq \gamma_{k,j_k}$.

Using that preference ordering of the cells, one then can select a *Pareto subset of the set of all activated cells.* This Pareto subset of the cells is a cluster cell approximation of the Pareto set of the original problem.[8] An example of the Pareto cluster cells for the set shown on Fig. 3.14 is shown on Fig. 3.15. Out of a total of 9 cells, 6 cells are activated and only 2 of them are the Pareto cluster cells in this example. Fig. 3.15 differs from Fig. 3.14 in that only PCCs are shaded on Fig. 3.15. The PCCs that obtain in this example can be labeled as <good, average> and <average, good>[9].

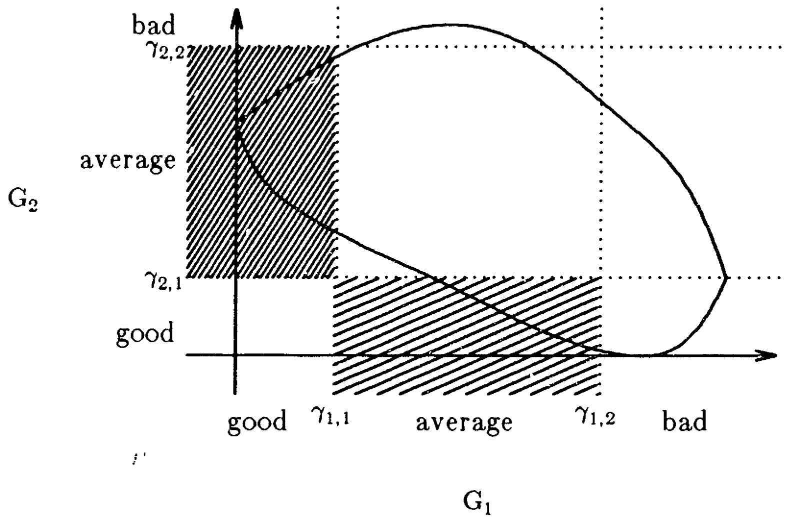


Figure 3.15. The Pareto cluster cells.

8. Note that the the cluster cells have a Pareto subset even if the original attainability set Ψ is unbounded.

The PCCs give a compact way of exploring the attainability set. The DM can interactively select the partitioning of the target function values and observe in turn the resulting PCCs. Any of the generated PCCs can be selected for a further exploration if its bounds are used as the active bounds of some range.

The partitioning of the values of the target functions is a natural and easy task for the DM, especially if this partitioning is done using lexical labels reflecting the goodness of the respective interval. For instance, partitioning the values of a target function into 5 intervals can be done using the labels: <very good>, <good>, <average>, <fair> , <bad>. The DM can also independently control the number of partitions of each of the target function. For example one target function can be partitioned into 3 intervals, while another target function can be partitioned into 5 intervals.

Since the DM can control both the number of the partitions of each of the target functions as well as the positions of the endpoints of these partitions, the PCC technique offers a flexible tool for a snapshot analysis of the attainability set. The example of the human-computer interface is presented in Chapter 4.

9. As it was noted earlier, the maximal satisficing sets can be viewed as a partial case of the Pareto cluster cells when the number of endpoints, N=1, and only two labels are used with partitioning: <good> and <bad>. However the problem of determining the maximum possible number of PCCs is a more difficult problem that the one about the maximum possible number of MSSs. A solution of this problem can be found in Appendix C.

3.3.3 Algorithm for Computation of All PCCs

The algorithm for computation of all PCCs is similar to the one used for finding all MSSs. However more bookkeeping is required for determining all PCCs as compared to computing all MSSs.

Suppose that the values of the k-th target function have been partitioned into N_k intervals.[10] Each cell is then uniquely defined by a vector $(i_1,\ldots,i_M)^T$ $(0 \leq i_k \leq N_k)$, composed of the indexes of the partitioning endpoints that bind the cell from above (in accordance with conditions (3.37)). This vector,

$$J = (i_1,\ldots,i_M)^T, \quad 0 \leq i_k \leq N_k \tag{3.40}$$

is called a *cell number.*

The algorithm uses the notion of *feasibility* of a cell.

Definition: Cell (3.40) is called feasible if a hyper-quadrant in $\mathbf{R}^M$ defined by (3.41)

10. I.e. the algorithm deals with the cases when different target functions are partitioned into a different number of intervals.

$$\begin{cases} y_1 \le \gamma_{1,i_1} \\ y_2 \le \gamma_{2,i_2} \\ \dots \\ y_M \le \gamma_{M,i_M} \end{cases} \tag{3.41}$$

(where $\mathbf{y} = (y_1,...,y_M)^T \in \mathbf{R}^M$ is a point of the quadrant) has a non-empty intersection with Ψ. Otherwise the cell is called infeasible.[11]

The search of the PCCs is a recursive algorithm combining depth-first and lexicographical traversing of all cell numbers. The lexicographical nature of the algorithm permits effective pruning of the search tree using the information about the feasibility of already tested cells. The algorithm is designed to reduce the number of feasibility tests. Though the algorithm is similar in the spirit to the one used for finding all MSSs, it is more difficult to visualize it as a tree search. We present a formal description of the algorithm.

We use a symbolic notation which is in most parts self-evident. The algorithm is centered around a recursively invoked procedure. Every time this procedure is invoked it is being passed some parameter. Similarly, upon the return from every call to this procedure some parameter is returned to the calling subroutine.

11. Compare (3.41) with (3.38). The region described by the former spans a whole quadrant of the cells defined by the latter.

Three lists of cell numbers are maintained during the execution of the algorithm: a list of already found Pareto numbers, a list of *sure feasible* numbers, and a list of *sure infeasible* numbers.

The algorithm operates with the following variables:

$(i_1,...,i_M)$	- current cell number;
F	- a flag that is used to keep track of which component of the current cell number was added from the parent;
ParetoFlag	- a flag that is used to decide if the current cell is a Pareto cell;
S	- a flag that stores the parameter passed upon entering the invocation of the recursive procedure;
ListPCC	- a list of all found Pareto cells;
ListFeas	- a list of all sure feasible cells;
ListInFeas	- a list of all sure infeasible cells;
LevelRec	- the recursion level;
R	- a flag returned upon returning from the recursive procedure.

The algorithm proceeds in the following order:

```
(0) [Initialization] $i_k \leftarrow N_k$ $(k = \overline{1,M})$;   ListPCC $\leftarrow$ EMPTY;  ListFeas $\leftarrow$
        EMPTY;
        ListInFeas $\leftarrow$ EMPTY;  LevelRec $\leftarrow$ 0

(A) [First entry into recursive procedure] <Enter (B) passing 0>

******************** START OF RECURSION ********************
```

```
(B) [Entering Recursion] <Receive F>; ParetoFlag $\leftarrow$ TRUE; S $\leftarrow$ F

(C) [Test if this is the first entry] If $S = 0$ then go to (H)

(D) [Test if the current cell is feasible non-Pareto cell] <Look up ListPCC>;
        if <cell is feasible non-Pareto> then
            ParetoFlag $\leftarrow$ FALSE; if $F = M$ then <return FEASIBLE >
                                    else go to (H)

(E) [Test if it is the first level of recursion] if LevelRec = 1 and $F < M$ then go to
        (H)

(F) [Test if the end of branch] if LevelRec = 1 then <Add the current cell to
        ListPCC>

(G) [Test feasibility of the current cell]
        if <The current cell is FEASIBLE> then
            if $S = M$ and ParetoFlag = TRUE then
                <Add the current cell to ListPCC>; <return FEASI-
        BLE >
            else go to (H)
        else <return INFEASIBLE >

(H) [Form the next (child) cell] LevelRec $\leftarrow$ LevelRec + 1;
        if $F = S$ or if $\{F \neq S$ and $i_F = N_F\}$ then $F \leftarrow F + 1$; $i_F \leftarrow 0$

(I) [Entry into recursive procedure] <Enter (B) passing F>
```

```
(J) [Receive return from recursive call] <R is returned from the call>

(K) [Adjust current cell number] LevelRec $\leftarrow$ LeveRec $-$ 1; if $i_F < N_F$ then
        $i_F \leftarrow i_F+1$

(L) [Check returned flag] if R = FEASIBLE then ParetoFlag $\leftarrow$ FALSE

(M) [Do next cell] if F < M then go to (H)

(N) [Process terminal node] if $i_M < N_M$ and R = INFEASIBLE then go to (H)

(O) [Correction] $i_M \leftarrow N_M$

(P) [Test if found Pareto cell] If ParetoFlag = TRUE then <Add cell to
        ListPCC>

(Q) [Return] <return FEASIBLE>
```

********************* END OF RECURSION *********************

This algorithm generates all PCCs. The only time-consuming procedure in this algorithm is the feasibility testing of the current cell (G). The algorithm is implemented in such a way that the number of such testing is minimized.

The feasibility testing of a cell (3.40) is done in the following way. First it can be noticed from the algorithm that by the time when cell (3.40) is tested for feasibility (i.e. becomes the current cell), $i_S = 0$ and it is already established that cell

$$(i_1,\ldots,i_{S-1},N_M,i_{S+1},\ldots,i_M)$$

is feasible. In order to test the feasibility of the current cell the following minimization problem, whose feasibility is assured, is solved (the notation is the same as in (3.36)):

$$G^- = \min_{\chi \in \Omega_\chi} G_S(\chi)$$

$$\text{subject to:} \qquad (3.42)$$

$$\begin{cases} G_1(\chi) \leq \gamma_{1,i_1} \\ \dots \\ G_{S-1}(\chi) \leq \gamma_{1,i_{S-1}} \\ G_{S+1}(\chi) \leq \gamma_{1,i_{S+1}} \\ \dots \\ G_M(\chi) \leq \gamma_{1,i_M} \end{cases}$$

After G^- has been determined, the two closest endpoints $\gamma_{S,r-1}$ and $\gamma_{S,r}$ of the S-th goal function's partitioning are found such that:

$$\gamma_{S,r-1} < G^- \leq \gamma_{S,r} \qquad (3.43)$$

If r in (3.43) is such that $r \neq N_S$ then cell

$$(i_1,\dots,i_{S-1},r,i_{S+1},\dots,i_M)$$

is added to the list of *sure feasible* cells (ListFeas).

Similarly, if $r \neq 0$ then cell

$$(i_1,\dots,i_{S-1},r-1,i_{S+1},\dots,i_M)$$

is added to the list of *sure infeasible* cells (ListInfeas).

The lists of the *sure feasible* and *sure infeasible* cells are maintained in order to minimize the number of optimizations (3.42).

Chapter 4

Human-Computer Interface and DRT Implementation for Discrete Tasks

4.1 Human-Computer Interface

4.1.1 Introduction

The details of the DRT implementation and the particulars of the human-computer interface may vary from one application to another. In Chapter 6 we describe an implementation for controlling a dynamical system that was used in the experiments. In the present Chapter we outline the features that must be present in any DRT implementation and illustrate these features on a software system that handles discrete tasks.

4.1.2 Basic Interface

The basic interface permits the DM to interactively set and adjust active bounds, while observing the corresponding change of the passive bounds. The basic display of a single range is shown on Fig. 4.1. At each instant of time, t, this display exhibits some range, $(\mathbf{L}^A,\mathbf{R}^A \xrightarrow{\Psi(t)} \mathbf{L}^P,\mathbf{R}^P)$ associated with the *current* attainability set, $\Psi(t)$. (As it was explained in Chapter 3, superscript "A" denotes the active bounds and superscript "P" denotes the passive bounds of the range.)

The display is composed of several rectangles. These rectangles correspond to the target functions, G_i $(i = \overline{1,M})$, and there are as many rectangles on the screen as there are target functions. Each rectangle stretches along the axis that represents the values of the corresponding target function. The vertical sides of the rectangles correspond to the active bounds of the respective target function. The left side of the i-th rectangle corresponds to L_i^A, and the right side corresponds to R_i^A.

The left and/or right sections of the rectangles are shaded. The ends of the shadings correspond to the passive bounds of the respective target function. Thus the right end of the shaded area of the left section of the i-th rectangle corresponds to L_i^P, and the left end of the shaded area of the right side of the i-th rectangle corresponds to R_i^P.

The active bounds are set by the DM, while the passive bounds are calculated by the computer. Since the attainability set might change with time, even when the boundaries of the rectangles remain unchanged, the shaded areas could change with time, reflecting the changes of the attainability set. The unshaded,

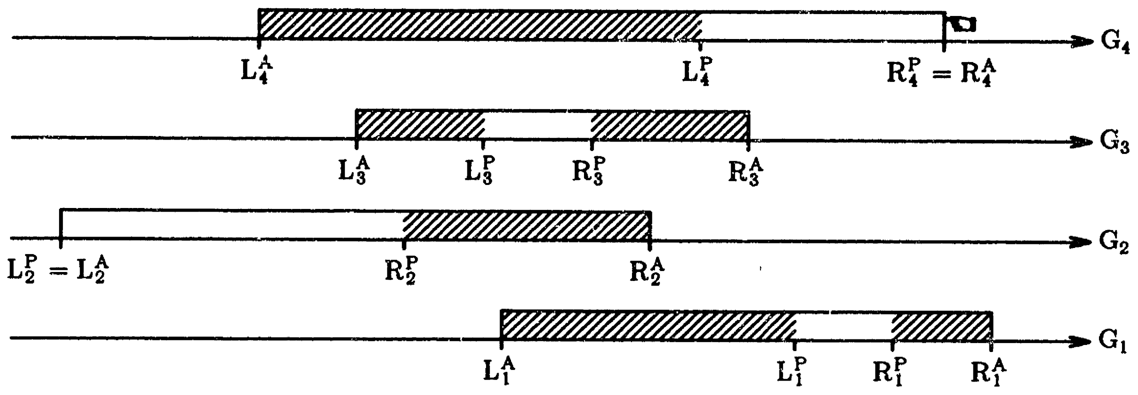


Figure 4.1. Range display.

clear portions of the rectangles represent an approximation of the attainability set at any instant of time.

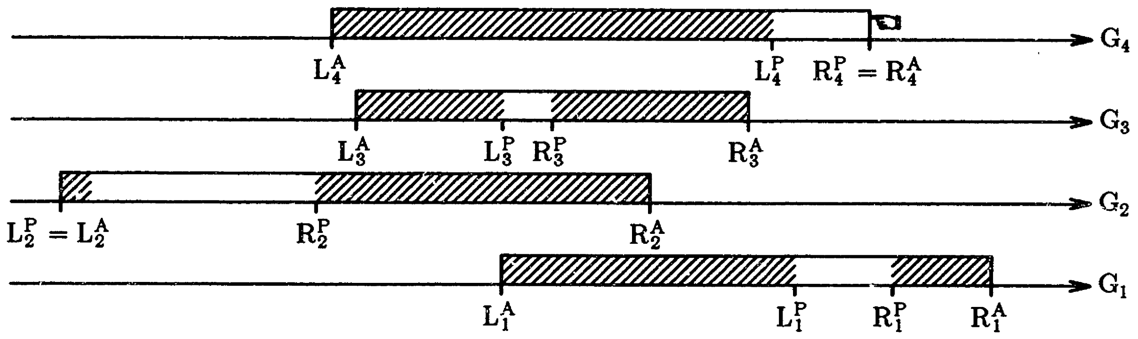


Figure 4.2. One of the active bounds of the original range (Fig. 4.1) has been changed.

The DM can independently move the boundaries of any rectangle, thus changing the values of the corresponding active bounds. A change of a single active bound might case a corresponding change of some or all passive bounds. For instance, if the right active bound, R_4^A of G_4 on Fig. 4.1 is reduced, then the range corresponding to the new active bounds might look like the one on Fig. 4.2. Since the change involved the reduction of one of the right active bounds, the passive bounds can only get closer to each other, so that the shaded area can only become larger. The "clear" attainable area of the range on Fig. 4.2 has indeed become smaller, while the shadings have expanded.

In a similar fashion the DM can change all active bounds, one by one, and observe the corresponding changes of the passive bounds. The active bounds correspond to the desired values of the goal, while the passive bounds show the approximation of the set of attainable goals. If at some point the goal becomes unattainable and the range becomes infeasible, the shaded areas spread and completely fill all rectangles. A feasible range then can be found using the Maximal satisficing sets technique. In the case of a MCDM problem the Pareto cluster sets technique can also be used. This interactive exploration of the approximation of the attainability set with the passive bounds is the key component of the DRT technique.

In the case when the attainability set of the problem is convex, in addition to knowing that the attainability set is located inside of the unshaded areas, the DM also is assured, according to Lemma 2 of Chapter 3, that the attainability set has a non-empty intersection with the dotted boxes displayed on Fig. 4.3. The dotted boxes do not explicitly appear on the display, however the convexity of the attainability set assures the existence of at least one of its point inside the dotted boxes. This information is useful for selecting a goal that is located at a distance from the boundary of the attainability set.

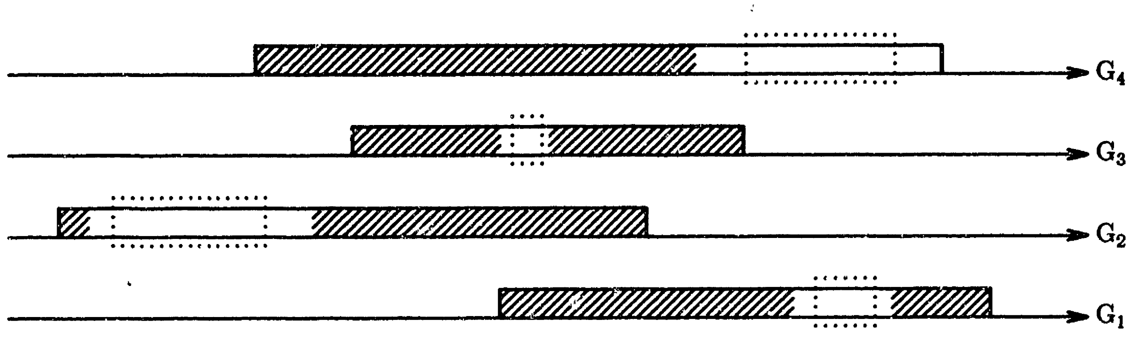


Figure 4.3. Convex attainability set intersects with the dotted intervals.

4.2 Interface for Discrete MCDM Tasks

4.2.1 Task Description

A discrete MCDM task is a task of choosing a single alternative out of a finite number of alternatives. Each alternative is characterized by M, usually incommensurable, criteria. These criteria are assumed to be preferentially independent, meaning that the DM prefers lower values of a criterion to the higher values of this criterion, provided that the values of all other criteria remain unchanged. The task is to select an alternative that has the minimal value of all criteria, or to find a compromise alternative that has reasonably low values of all criteria.

The problem can be extended to include criteria that the DM wishes to *maximize* instead of minimizing. The criteria whose higher values are preferred to the lower ones are called the ***positive*** criteria, while the criteria whose lower values are preferred to the higher values are called the ***negative*** criteria.

The task can be formulated as an MCDM problem of finding index i (the alternative) that delivers the following M maxima and minima at the same time:

$$\begin{cases} \min\limits_{i=\overline{1,N}} a_j^i \quad (j = \overline{1,M^-}) \\ \max\limits_{i=\overline{1,N}} a_k^i \quad (k = \overline{M^-+1,M}) \end{cases} \tag{4.1}$$

where N is the total number of alternatives and each alternative is characterized by M criteria. The values of the criteria corresponding to the i-th alternative are respectively $a_1^i, a_2^i, \ldots,$ and a_M^i. It is desired to minimize criteria from 1 to M^- and at the same time maximize criteria from M^-+1 to M.

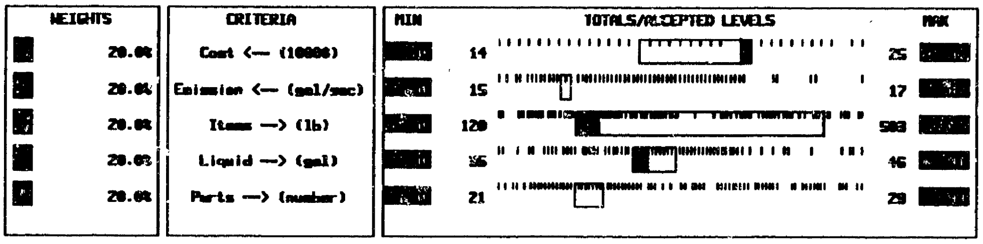
This task, as any MCDM problem, is ill-defined. A number of methods have been proposed to help the DM solve this problem. Some of these methods were discussed in Chapter 2. Here we describe a DRT decision-aiding system for this type of problems. It must be emphasized that in the case of a static discrete task, as well as in the case of a traditional MCDM problem, the DRT approach provides yet another method of helping the DM solve the problem. As it was mentioned in Chapter 2, it is practically impossible to compare the merits of different methods of solving MCDM problems. This applies to the DRT method as well. Though it can be asserted that the DRT method provides the DM with a new and insightful look at the set of alternatives, we however avoid claiming this method's superiority over the already existing ones *for this particular type of problems.*

The discrete MCDM task however provides an easy ground for the demonstration of the basic DRT techniques, though the full advantages of the DRT method may not yet be evident.[1]

This task (4.1) can be rephrased employing the terminology used throughout this thesis. The *system* in this case is static, i.e. time-independent, and having just a single state. The control u is an integer from 1 to N (an alternative). There are M target functions that depend only on the value of the control and these functions can have only N discrete values. Specifically $G_j(u) = a_j^u$. The attainability set is a set of N points in $\mathbf{R}^M$, where the i-th point is $(a_1^i, \dots, a_M^i)^T$.

Fig. 4.4 provides a typical display that appears in the process of the DM's interaction with the computer while solving (4.1). There are a total of 5 criteria. 3 of them are positive criteria (*Items*, *Liquid*, and *Parts*) whose values are wished to be maximized (this is reflected in the direction of the arrow next to the criterion name). 2 criteria are negative and their values are wished to be minimized (*Cost* and *Emission*). The range appearing in this problem is a directional range of the type -1-1111.

1. The experiments described in Chapter 6 do demonstrate the advantages of the DRT method in a situation where traditional methods do not work.



WEIGHTS	CRITERIA	MIN	TOTALS/ACCEPTED LEVELS		MAX
20.0%	Cost <— (1000$)	14		25	
20.0%	Emission <— (gal/sec)	15		17	
20.0%	Items —> (lb)	120		503	
20.0%	Liquid —> (gal)	36		46	
20.0%	Parts —> (number)	21		29	

Figure 4.4. A DRT display for a discrete task with preferentially independent criteria.

Each criterion is assigned a rectangle. Since a MCDM problem requires a directional range, only one side of each rectangle corresponds to an active bound. The opposite side of the rectangle corresponds to a passive bound. Specifically, the left side of a positive criterion rectangle corresponds to the left active bound of this criterion and the right side of a negative criterion rectangle corresponds to the right active bound of the criterion. Similarly, the right side of a positive criterion rectangle corresponds to the right passive bound of this criterion and the left side of a negative criterion rectangle corresponds to the left passive bound of the criterion.

There are four columns of numbers on the screen: two columns to the left of the rectangles and two columns to the right. The numbers in the columns marked **MIN** and **MAX** display respectively the absolute minima and maxima of the values of the critera. The other numbers show the values corresponding to the location of the left and right sides of the rectangles.

The small bars inside of the criteria region on the display show all discrete values of the criteria that the N available alternatives have.

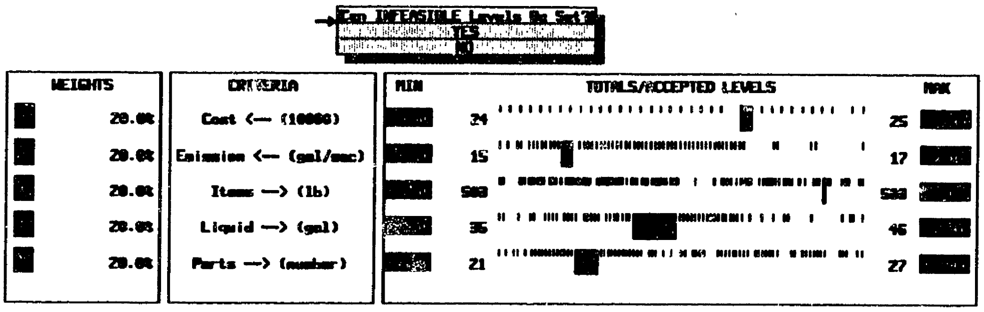
The are three modes of interaction with the system: regular, MSS and PCC. The DM can select the mode using a pop-up menu interface.

In the regular mode the DM can interactively move the sides of the rectangles that correspond to the active bounds. A side of a rectangle can be moved using the computer mouse. By adjusting these rectangles, the DM can change the left active bounds of positive criteria and the right active bounds of negative criteria.

The values of the active bounds can be interpreted as the *worst acceptable* levels of the criteria. By interactively changing the active bounds the DM effectively explores the trade-offs between different criteria. The interaction with the computer can also be seen as the exploration of the trade-off between *the worst acceptable* levels of the criteria (defined by the active bounds) vs. *the best possibly achievable* levels of the criteria (determined by the passive bounds).

The DRT establishes a *what if* dialogue between the human and the computer. The human in effect asks "what happens if I tighten my acceptability requirements on the acceptable levels of some criterion?". The computer answers in response: "then you will have to sacrifice the achievement of certain levels of some or all other criteria".

The active bounds can be freely moved as long as they are not attempted to be positioned beyond the corresponding passive bounds. If the DM does attempt to move an active bound past the corresponding passive bound, the computer would inquire whether the DM wishes to enter the Maximum satisficing sets mode (Fig. 4.5). A pop-up menu interface is used to exchange questions and answers between the human and the computer.



WEIGHTS	CRITERIA	MIN	TOTALS/ACCEPTED LEVELS	MAX
20.0%	Cost <— (10000)	24		25
20.0%	Emission <— (gal/sec)	15		17
20.0%	Items —> (lb)	503		533
20.0%	Liquid —> (gal)	35		46
20.0%	Parts —> (number)	21		27

Figure 4.5. Entering the MSS mode.

If the DM answers "no" then the range is collapsed into a single point (as on Fig. 4.5) and the only way to change this range is to relax some of the active bounds.

If the DM enters the MSS mode, it becomes possible to set all active bounds independently of one another, thus defining a potentially infeasible range. The passive bounds are not displayed in this mode. As soon as the active bounds are set, the DM can request to see all Maximal satisficing sets (Figs. 4.6 and 4.7)^2. There are three MSSs corresponding to the active bounds on Fig. 4.6 and four MSSs on Fig. 4.7.

A shaded bar corresponding to some criterion on these pictures represents the *undesirable* region of the values of the criterion. If the black-bordered rectangle is drawn inside of the shaded area it means that the corresponding criterion is non-activated in that MSS.

Display on Fig. 4.6, for instance, shows that given the desired active bounds on the values of the criteria (displayed in the lower right section of the screen)[3], only three maximum groups of these bounds can be simultaneously satisfied. These groups are displayed in the upper section of the screen and they are: <Cost, Emission> (upper left); <Items, Parts> (lower left); and <Items, Liquid> (lower right).

The DM can select any MSS and return to the regular mode using this range as a starting feasible range. As soon as some MSS is selected, the program gets back to the regular mode.

Using the pop-up menus the DM can enter the Pareto cluster cells mode. In this mode she can place marks inside of each criterion rectangle, thus splitting the range of values of each criterion into up to five intervals. After the marks are set all PCCs are generated and displayed (Figs. 4.8 and 4.9). After studying the PCCs the DM can get back to the regular mode.

On Figs. 4.8 and 4.9 three intervals of goodness are specified for each criterion: <GOOD>, <Average> and <bad>. There are six PCC on Fig. 4.8 and nine on Fig. 4.9. It can be seen from these pictures, than there are no alternatives with more than two criteria being in a <GOOD> range.

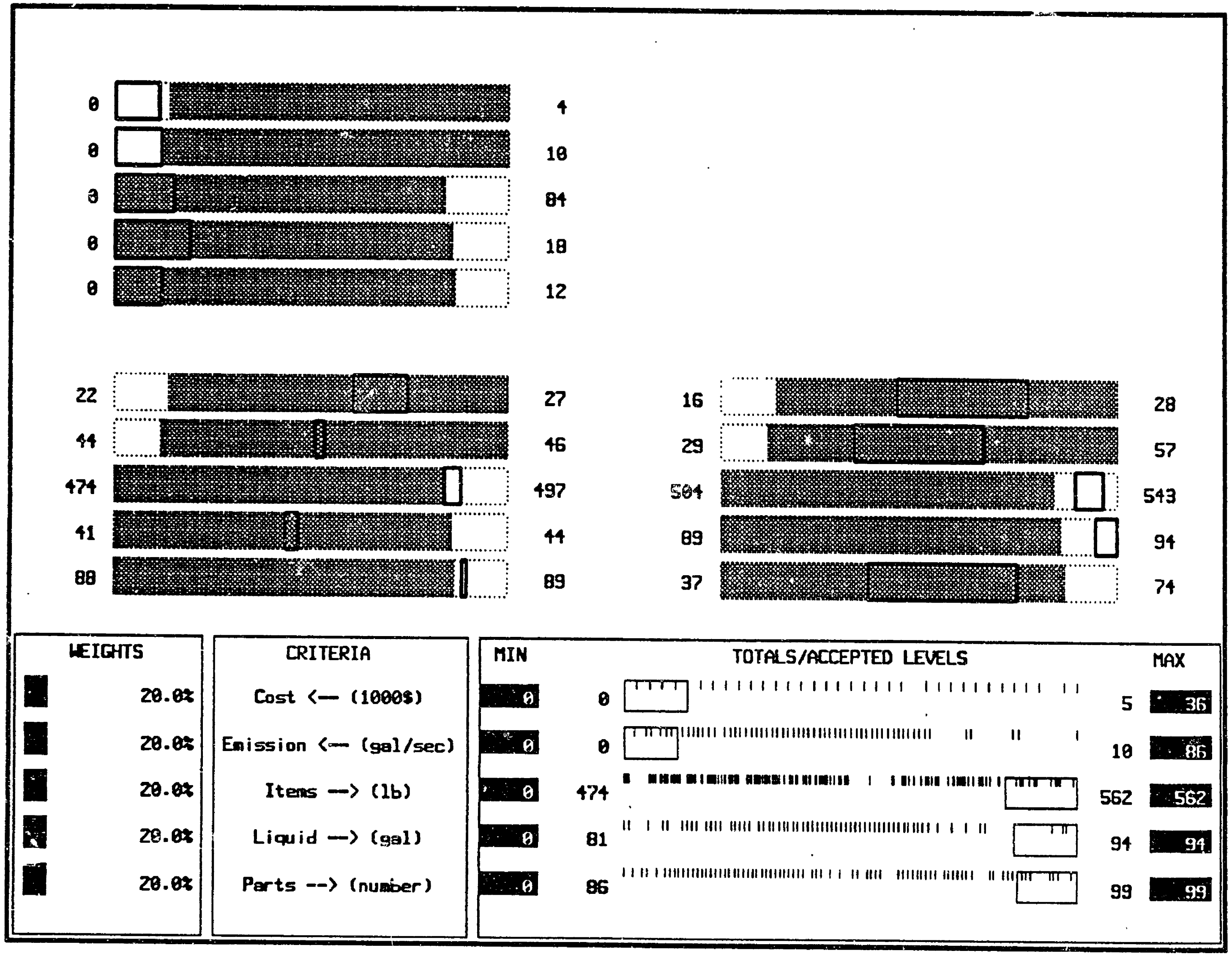
3. These values are from top to bottom: (0-5), (0-10), (474-562), (81-94), and (86-99).
3. The graphical size of the sets is automatically scaled up/down so that all MSSs can fit one one screen.

The values of the passive bounds for a PCC can be obtained using the regular mode and setting the active bounds on the appropriate marks of the cluster cells.

The described interface is generic in nature. It can be used with a variety of applications, and it can be connected to a spreadsheet program.

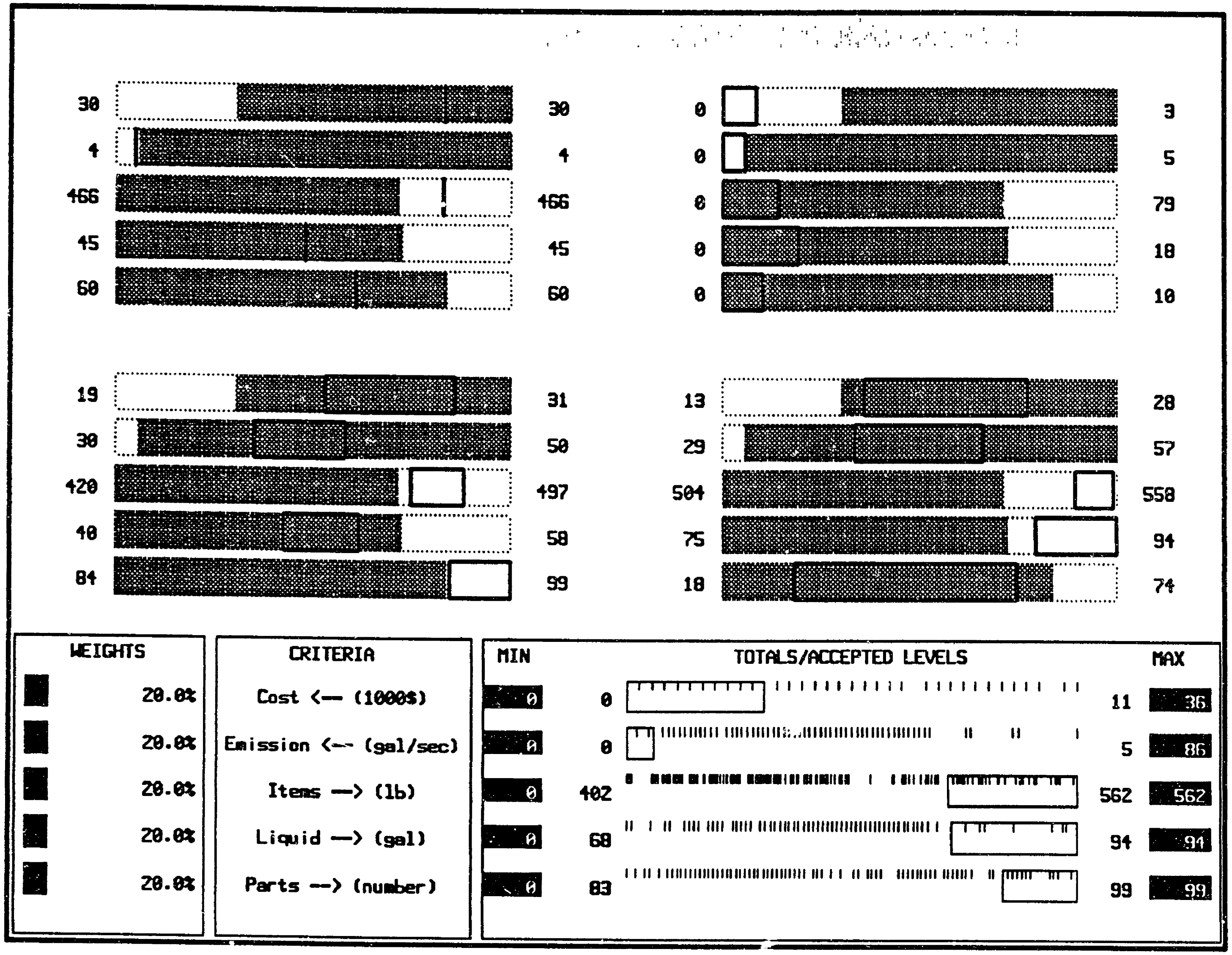
For the demonstration purpose this interface was incorporated into a tactical mission planning decision-aiding system[4]. An example of the DRT display overlaying this application's display is shown on Fig. 4.10. This display exhibits the Pareto Cluster Cells for a problem of choosing the best route for a tactical aircraft mission. Each alternative route is characterized by 6 attributes. A geographic map of the region where the mission is planned appears on the display's background.

4. The decision-aiding approach and the original, non-DRT interface is described in [10].



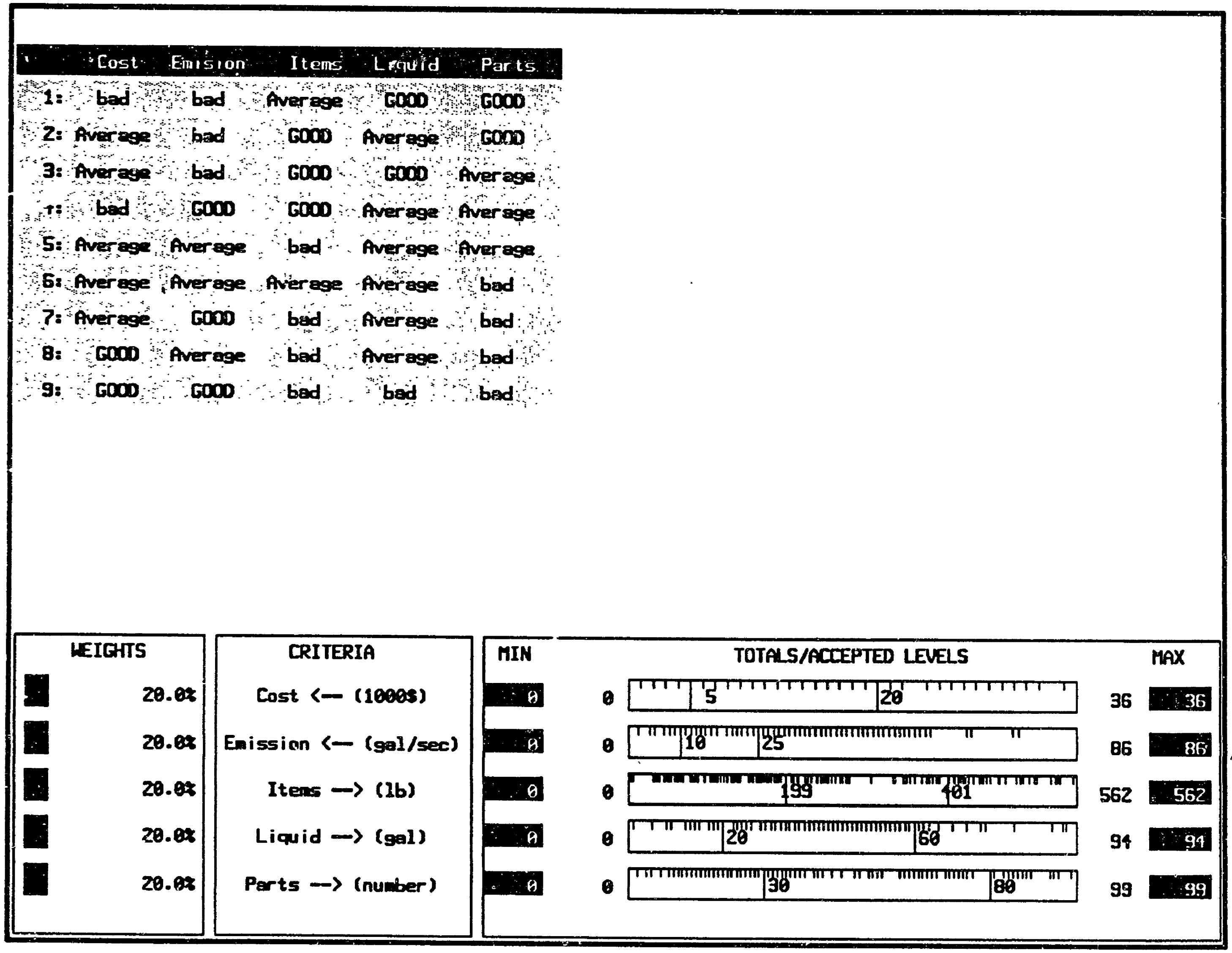
WEIGHTS	CRITERIA	MIN	TOTALS/ACCEPTED LEVELS		MAX
20.0%	Cost <— (1000$)	0	0	5	36
20.0%	Emission <— (gal/sec)	0	0	10	86
20.0%	Items —> (lb)	0	474	562	562
20.0%	Liquid —> (gal)	0	81	94	94
20.0%	Parts --> (number)	0	86	99	99

Figure 4.6. Maximal Satisficing Sets display #1.



WEIGHTS	CRITERIA	MIN	TOTALS/ACCEPTED LEVELS		MAX
20.0%	Cost <— (1000$)	0	0	11	36
20.0%	Emission <-- (gal/sec)	0	0	5	86
20.0%	Items —> (lb)	0	402	562	562
20.0%	Liquid —> (gal)	0	68	94	94
20.0%	Parts —> (number)	0	83	99	99

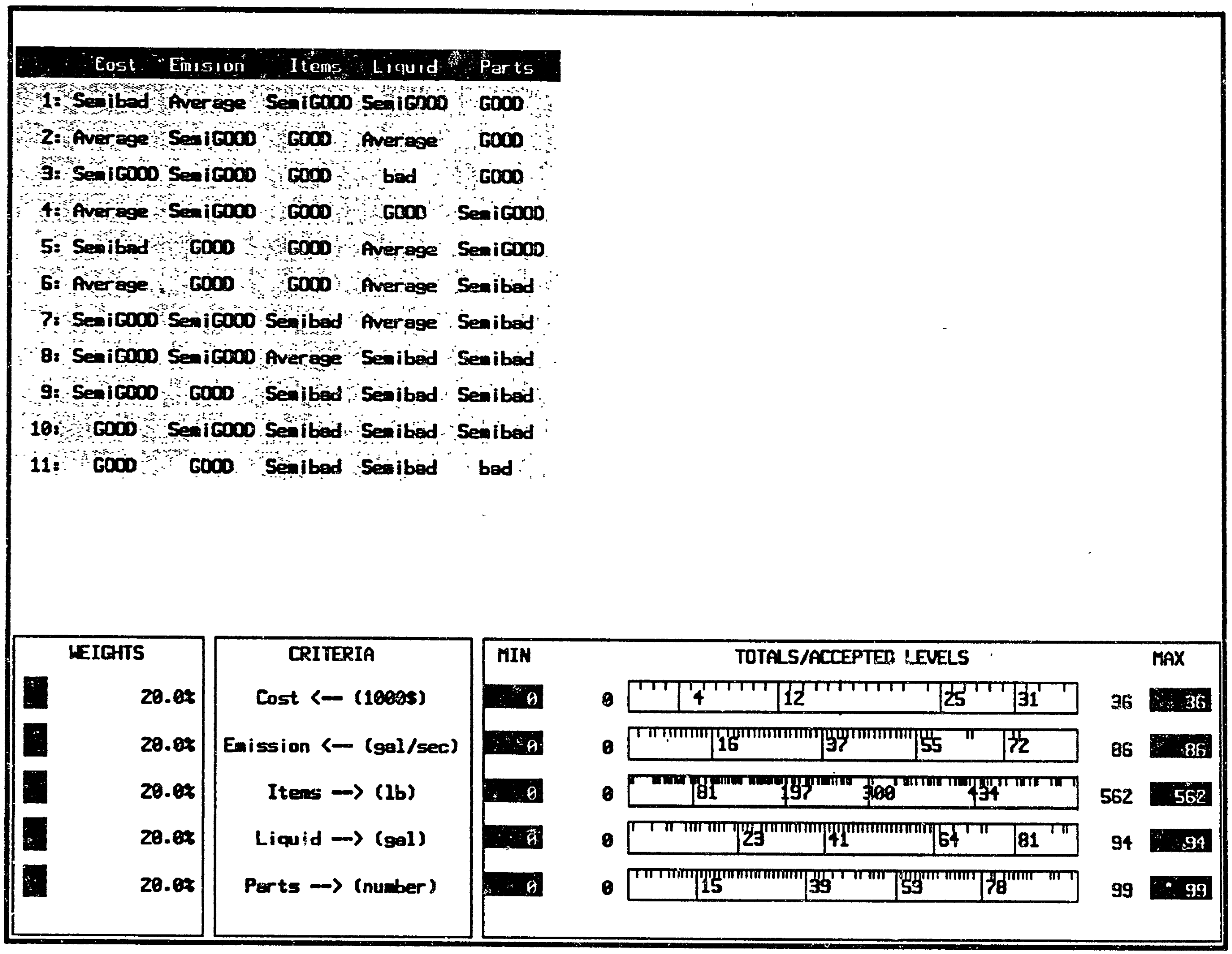
Figure 4.7. Maximal Satisficing Sets display #2.



	Cost	Emision	Items	Liquid	Parts
1:	bad	bad	Average	GOOD	GOOD
2:	Average	bad	GOOD	Average	GOOD
3:	Average	bad	GOOD	GOOD	Average
4:	bad	GOOD	GOOD	Average	Average
5:	Average	Average	bad	Average	Average
6:	Average	Average	Average	Average	bad
7:	Average	GOOD	bad	Average	bad
8:	GOOD	Average	bad	Average	bad
9:	GOOD	GOOD	bad	bad	bad

WEIGHTS	CRITERIA	MIN		TOTALS/ACCEPTED LEVELS		MAX
20.0%	Cost <— (1000$)	0	0	5 20	36	36
20.0%	Emission <— (gal/sec)	0	0	10 25	86	86
20.0%	Items —> (lb)	0	0	199 401	562	562
20.0%	Liquid —> (gal)	0	0	20 60	94	94
20.0%	Parts —> (number)	0	0	30 80	99	99

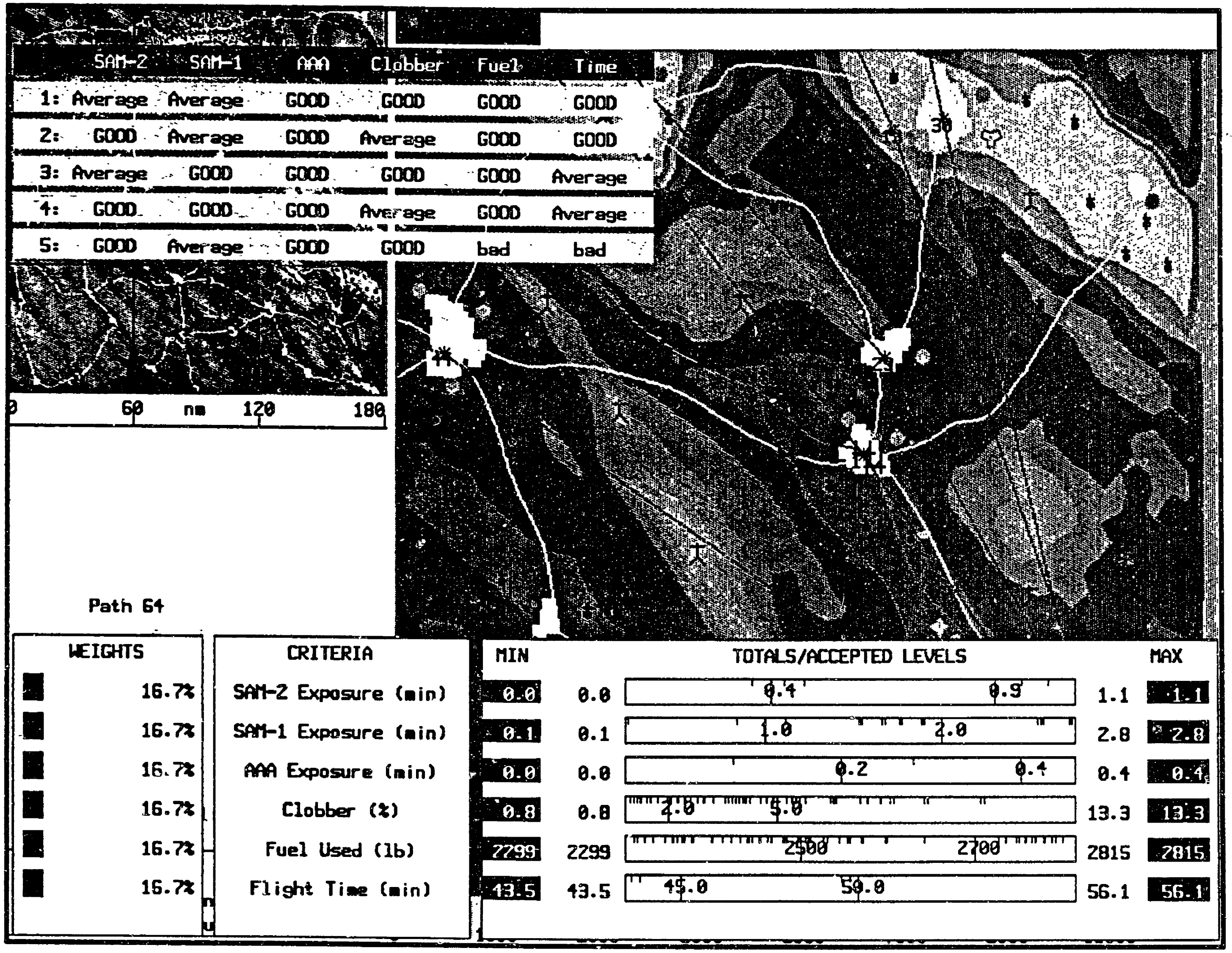
Figure 4.8. Pareto Cluster Cells display #1.



	Cost	Emision	Items	Liquid	Parts
1:	Semibad	Average	SemiGOOD	SemiGOOD	GOOD
2:	Average	SemiGOOD	GOOD	Average	GOOD
3:	SemiGOOD	SemiGOOD	GOOD	bad	GOOD
4:	Average	SemiGOOD	GOOD	GOOD	SemiGOOD
5:	Semibad	GOOD	GOOD	Average	SemiGOOD
6:	Average	GOOD	GOOD	Average	Semibad
7:	SemiGOOD	SemiGOOD	Semibad	Average	Semibad
8:	SemiGOOD	SemiGOOD	Average	Semibad	Semibad
9:	SemiGOOD	GOOD	Semibad	Semibad	Semibad
10:	GOOD	SemiGOOD	Semibad	Semibad	Semibad
11:	GOOD	GOOD	Semibad	Semibad	bad

WEIGHTS	CRITERIA	MIN		TOTALS/ACCEPTED LEVELS		MAX
20.0%	Cost <— (1000$)	0	0	4 12 25 31	36	36
20.0%	Emission <— (gal/sec)	0	0	16 37 55 72	86	86
20.0%	Items —> (lb)	0	0	81 197 300 434	562	562
20.0%	Liquid —> (gal)	0	0	23 41 64 81	94	94
20.0%	Parts —> (number)	0	0	15 39 59 78	99	99

Figure 4.9. Pareto Cluster Cells display #2.



	SAM-2	SAM-1	AAA	Clobber	Fuel	Time
1:	Average	Average	GOOD	GOOD	GOOD	GOOD
2:	GOOD	Average	GOOD	Average	GOOD	GOOD
3:	Average	GOOD	GOOD	GOOD	GOOD	Average
4:	GOOD	GOOD	GOOD	Average	GOOD	Average
5:	GOOD	Average	GOOD	GOOD	bad	bad

Path 64

WEIGHTS	CRITERIA	MIN		TOTALS/ACCEPTED LEVELS		MAX
■ 16.7%	SAM-2 Exposure (min)	0.0	0.0	0.4 0.9	1.1	1.1
■ 16.7%	SAM-1 Exposure (min)	0.1	0.1	1.0 2.0	2.8	2.8
■ 16.7%	AAA Exposure (min)	0.0	0.0	0.2 0.4	0.4	0.4
■ 16.7%	Clobber (%)	0.8	0.8	2.0 5.0	13.3	13.3
■ 16.7%	Fuel Used (lb)	2299	2299	2500 2700	2815	2815
■ 16.7%	Flight Time (min)	43.5	43.5	45.0 50.0	56.1	56.1

Figure 4.10. DRT overlaying the application display.

Chapter 5

Resource Sharing Systems

5.1 Introduction

The DRT technique can be naturally applied to problems related to control of a large class of systems that will be called here *Resource Sharing Systems* (RSS). Several systems are called to be resource sharing if their only inter-dependence comes about from sharing the same *limited resource.* Each of these systems has its own state, control variables, and goal functions. The limitation of the shared resource can be expressed by a single condition relating together every individual system's control variables. This condition will be called a *resource limitation* condition.

To distinguish individual systems in a group of resource sharing systems, hereafter each system in the group will be referred to as a *subsystem* and the entire group of resource sharing subsystems will be referred to as the *system.*

To control each subsystem is to specify the desired ranges (goals), of its goal functions. A subsystem could possibly reach very good[1] levels of its goal functions if it had enough *resource* to do that. However, if only a limited amount of the resource is available to be shared by all the subsystems, it might be impossible to assign goals to each subsystem independently.

To control a group of such subsystems is to set individual goals for each subsystem in the group. If the total amount of the resource is limited and the individual goals cannot be set independently, some tradeoff has to be made in doing the goal assignment. Setting a higher goal for one of the subsystems might effectively cut down on the other subsystem's potentials. This situation is typical for a MCDM paradigm. The purpose of the decision-aid is to help the DM set individual goals for all subsystems, in light of their competition for the same limited resource.

In addition to sharing a limited resource, each subsystem is exposed to some perturbations (different for each subsystem). Due to the effect of these perturbations, the same goal that was assigned to the subsystem earlier might become

1. Assuming that the "goodness" is somehow defined for each subsystem.

unattainable later. Thus in controlling a group of resource sharing systems, the same attainability issues arise, that have been discussed before. This time the problem is to find a *combined attainability set of a group* of resource sharing subsystems.

To start one needs a formal definition of a *resource.* Different definitions can be recommended. Common sense however suggests that the resource is "something that is better to have in larger quantities". The following definition (which will be used throughout this thesis) agrees with the intuitive understanding that *given a larger amount of the resource, the system can generate a larger variety of control actions.*

Let's consider N subsystems (each marked with a super index i(=1,...N)) that evolve in time. The evolution of each subsystem is described by equations (5.1) - (5.2) that are similar to (1.8). (The nomenclature hereupon corresponds to the one in (1.8) - (1.9).)

$$\mathbf{x}^i(t_1) = \mathbf{H}^i(t_0, t_1, \mathbf{x}^i(t_0), \mathbf{u}|_{[t_0,t_1]}, \mathbf{p}|_{[t_0,t_1]}) \tag{5.1}$$

Each subsystem is exposed to its own perturbations:

$$\mathbf{p}^i(t) \in \Omega_p^i$$

and has its own controls belonging to the subsystem's set of admissible controls:

$$\mathbf{u}^i(t) \in \Omega_u^i \tag{5.2}$$

The resource is expended by all the subsystems. We shall call a scalar non-negative function ϕ^i a *resource function* of the i-th subsystem, if the amount of the resource expended by that system at time t is equal to $\phi^i(\mathbf{u}^i(t))$. The total amount of the resource expended by all subsystems at time t is therefore:

$$\sum_{i=1}^{N} \phi^i(\mathbf{u}^i(t))$$

Let's denote with $F^i(t_1,t_2)$ the amount of the resource spent by the i-th subsystem for a period from t_1 until t_2:

$$F^i(\mathbf{u}^i,t_1,t_2) = \int_{t_1}^{t_2} \phi(\mathbf{u}^i(t))dt$$

and $F(t_1,t_2)$ - the total amount of the resource spent by the entire group of the subsystems:

$$F(t_1,t_2) = \sum_{i=1}^{N} F^i(\mathbf{u}^i,t_1,t_2)$$

The *resource limitation* condition then can be expressed as

$$F(t_1,t_2) \leq R \tag{5.3}$$

where R is the total available amount of the resource.

ϕ^i can be any non-negative scalar function of the subsystem's control. However this function should satisfy the following additional important requirement: *For any time period* $[t_1,t_2]$ *there should exist an admissible control* $\mathbf{u}' \in \Omega_u^i$ *such that* $F(\mathbf{u}',t_1,t_2) = 0$. In other words, the subsystem should be able to generate some control action even if it is provided with no resource. In most cases this control action is $\mathbf{u}'(t) = 0$.[2]

5.2 Attainability Set of a Group of RSS

5.2.1 Combining Individual Attainability Sets

The attainability set of a group of resource sharing systems can be looked at from the point of view of the individual subsystems' attainability sets. The case of the subsystems each having only a single goal function will be treated here.

Let's consider a group of N resource sharing subsystems, each described by (5.1) - (5.2). Suppose that each of these subsystems has only a single goal function G^i:

$$G^i(t_0, t^*, \mathbf{x}^i(t_0), \mathbf{u}|_{[t_0, t^*]}, \mathbf{p}|_{[t_0, t^*]}) \tag{5.4}$$

A goal for the i-th subsystem is specified by two parameters g^{i-} and g^{i+}:

$$g^{i-} \leq G^i \leq g^{i+} \tag{5.5}$$

The usual assumption is made about the subsystem's perturbations: they are is assumed to be zero Chapter 1):

$$\mathbf{p}_{[t_0, t^*]}{}^i = 0$$

2. This last requirement reflects a rather narrow view about the resource that is explored in this thesis. The resource is treated as means to generate control actions rather than a necessary component of the subsystem's existence. Also time is not considered to be a resource according to this definition. A resource here is viewed as something actually enabling "more control actions" during any period of time.

A group of these subsystems can be viewed as a *single system* with a combined state vector $\mathbf{x} = (\mathbf{x}^{(1)},...,\mathbf{x}^{(N)})$, a combined control vector $\mathbf{u} = (\mathbf{u}^{(1)},...,\mathbf{u}^{(N)})$ This system has a total of N goal functions: $G^1,...,G^N$. The evolution of this combined system is described by the merged set of all equations (5.1) for i=1,...,N. The attainability set of the combined system thus is a subset of the N-dimensional space $\mathbf{R}^N$.

The attainability set of this group of resource sharing subsystems is a function of the total amount of the resource. The most useful property of the group of resource sharing systems (which has prompted a separate consideration of these systems) is the fact that knowing the attainability sets of the individual subsystems, one can calculate the attainability set of the whole group.

First let's define a *distribution* of the ρ amount of the resource as a set of N non-negative numbers $\rho^1,...,\rho^N$, such that their sum equals to ρ: $\sum_{i=1}^{N}\rho^i = \rho$.

Any distribution $\rho^1,...,\rho^N$ corresponds to the assignment of the amount ρ^i of the resource to the i-th subsystem. We now can consider an attainability set of an i-th subsystem, provided that its resource is limited by ρ^i.

To determine an attainability set of the i-th subsystem from time t_0 at time t^*, given the resource limitation ρ^i, the set of admissible controls is modified to incorporate the resource limitation. Those and only those controls $\mathbf{u}(t)$ are considered in determining the attainability set that are both admissible and do not overspend the resource:

$$\begin{cases} \mathbf{u} \in \Omega_u^i \\ F^i(\mathbf{u}^i(t), t_0, t^*) \le \rho^i \end{cases}$$

The following Lemma shows the relation of the A-attainability set of the group of the subsystems to the individual attainability sets.

Lemma 5.1. Given ρ, the total amount of the resource available to all subsystems, the A-attainability set of the group of the subsystems is a union of the Cartesian products of the individual subsystems' A-attainability sets, taken over all possible *distributions* of the resource to the individual subsystems.

Proof. This Lemma is intuitively obvious. Suppose some point $\sigma = (\sigma^1, \sigma^2, ..., \sigma^N) \in \mathbf{R}^N$ belongs to the attainability set of the combined system. This implies that that there exist such controls $\mathbf{u}_\sigma^i \in \Omega_u^i$ of the individual subsystems that

$$G^i(t_0, t^*, \mathbf{x}_\sigma^i(t_0), \mathbf{u}_\sigma^i) = \sigma^i$$

The group resource limitation is satisfied by these controls:

$$\sum_{i=1}^{N} F^i(\mathbf{u}_\sigma^i(t), t_0, t^*) \le \rho$$

If we define ρ^i as

$$\begin{cases} \rho^j = F^j(\mathbf{u}_\sigma^i(t), t_0, t^*) \text{ for } j=1,...,N-1 \\ \rho^N = F^{(N)}(\mathbf{u}_\sigma^{(N)}(t), t_0, t^*) - \sum_{i=1}^{N} F^i(\mathbf{u}_\sigma^i(t), t_0, t^*) \end{cases} \tag{5.6}$$

then one can see that $(\rho^1, \ldots, \rho^N)$ is indeed a distribution of ρ amount of the resource and σ^i does indeed belong to the A-attainability set of the i-th subsystem under resource limitation ρ^i. Thus any point of the A-attainability set of the combined system does belong to the Cartesian product of the individual systems' A-attainability sets for some resource distribution $\rho^1, \rho^2, ..., \rho^N$, i.e. the one that is determined by (5.6).

Conversely, given some resource distribution $\rho^1, ..., \rho^N$ let's take *any* set of values σ^i so that σ^i belongs to the A-attainability set of the i-th subsystem, provided its resource limitation is ρ^i. The σ^i's can be viewed as a single N-dimensional point $\sigma = (\sigma^1, ..., \sigma^N)$ belonging to the Cartesian product of the individual systems' A-attainability sets for the resource distribution $\rho^1, ..., \rho^N$. It is obvious that this point, σ, does also belong to the attainability set of the combined system with a total resource limitation ρ. This concludes the proof of the Lemma.

It must be noted that this Lemma is true only for the A-attainability sets. Here is an example showing that the Lemma is not true for the B-attainability sets.

Suppose that the group consists of two subsystems, each described by a first-order linear differential equation: $\dot{x}^1 = u^1 + 100$ (first subsystem), and $\dot{x}^2 = u^2$ (second subsystem). The goal functions are $G^i(t^*) = x^i(t^*)$ (i=1,2), the admissibility set of the i-th subsystem contains all such u(t) that $-1 \leq u^i(t) \leq 1$. The resource functions $F^i(u^i) = |u^i|$. Suppose that at time t_0 the initial conditions are $x^i(0) = 0$, the terminal time is $t^* = 10$, and the total amount of the resource is $\rho = 3$.

Consider the following resource distribution: ($\rho^1 = 1$, $\rho^2 = 2$). It is easy to see that the attainability set for the first subsystem is interval $\psi_{0,10} = [999,1001]$ (A-attainability), and interval $\bar{\psi}_{0,10} = [0,1001]$ (B- attainability); and for the second system $\psi_{0,10} = \bar{\psi}_{0,10} = [-2,2]$ (A- *and* B- attainability).

The Cartesian product of the B-attainability sets thus is: [0,1001] $\times$ [-2,2]. Let's consider (0,2) - a point from this Cartesian product. This point obviously ***does not*** belong to the attainability set of the combined systems of the first and the second subsystems. Specifically, the second system can reach level $G^2 = x^2(t') = 2$ *only* when time $t' \geq 2$. By that time system 1 will be far away from 0, since $G^1 = x^1(t') \geq 199$ for $t' \geq 2$. In other words, the goal functions of subsystems 1 and 2 can never reach levels 0 and 2 respectively at the same time. Therefore point (0,2), though belonging to the union of the B-attainability sets of subsystems 1 and 2, does not belong to the B-attainability set of the combined system.

As far as the A-attainability set of the group is concerned, it is indeed a product of two intervals [999,1001] $\times$ [-2,2].

5.2.2 Resource as a Decision Variable

In specifying a goal for a group of resource sharing systems, the total amount of the expended resource frequently becomes a decision variable itself. When the DM assigns goals to the subsystems she always has in mind the total amount of resource that is going to be expended to reach those goals. It is natural to assume that the DM always has a preference associated with the amount of the expended resource: ***the less resource is spent - the better.***

The amount of the resource the DM is willing to spent can be interpreted as part of the overall goal for the combined system. To incorporate this amount into the system's goal, we can consider the resource itself as another, N+1th subsystem. This system, however is not independent from the other subsystems.

The evolution of the ***resource subsystem*** is described by (5.7):

$$\dot{r}(t) = \sum_{i=1}^{N} \phi^i(\mathbf{u}^i(t)) \tag{5.7}$$

and the goal function for the resource subsystem is:

$$G^r(t_0, t^*, r(t_0)) = r(t^*) \tag{5.8}$$

where r(t) is the amount of the resource expended from time t_0 until t.

The resource subsystem has no controls and is not subjected to any perturbations. The resource subsystem could be made independent of the initial time t_0 with the addition of the initial condition at time $-\infty$:

$$r(-\infty) = 0$$

Since the DM always wishes to minimize the amount of the expended resource, the goal for the resource is specified by a single number, limiting the resource expenditures. Thus to specify a goal for a combined system consisting of all N subsystems and the resource subsystem, one needs to specify 2N+1 numbers:

$$\mathbf{g} = (g^{1-}, g^{1+}, \ldots, g^{N-}, g^{N+}, g^{r}) \tag{5.9}$$

The goal of the combined system is achieved at the first instance of time t^*, when the target conditions (5.10) simultaneously hold:

$$\begin{cases} g^{1-} \leq G^1(t^*) \leq g^{1+} \\ g^{2-} \leq G^2(t^*) \leq g^{2+} \\ \cdots \\ g^{N-} \leq G^N(t^*) \leq g^{N+} \\ r(t^*) \leq g^r \end{cases} \tag{5.10}$$

After the resource has been incorporated into the system as a subsystem, the resulting combined system can be viewed as a system with N+1 goal functions. The attainability set of such a combined system thus is a set in the N+1 dimensional space. The fact that this combined system was obtained by the aggregation of the resource sharing subsystems and the resource subsystems is taken advantage of during the DRT calculations.

5.3 Examples of Resource Sharing Systems

The resource-sharing systems occur in a multitude of practical applications. Let's consider several practical examples of the resource-sharing systems. A detailed treatment of one particular system of this kind is given in Chapter 6.

5.3.1 Economics

A problem of the allocation of limited resources is a key problem in economics. As Milton Friedman, recipient of the 1976 Nobel prize in economics science wrote in[20]:

> An economic problem exists whenever *scarce* means are used to satisfy *alternative* ends. If the means are not scarce, there is no problem at all; there is Nirvana. If the means are scarce but there is only a single end, the problem of how to use the means is a technological problem. No value judgments enter into its solution; only knowledge of physical and technical relationships.

An example of goal-setting problem to a resource-sharing system is a task of budget planning. At present the budget planning involves allocating a limited resource (money) to different agencies and letting the agencies run on its own. Each agency can be viewed as a subsystem sharing resource with other subsystems (other agencies). Each agency has its own goals and tries to reach these goals using its own budget. Presuming that the agencies operate independently of each other, they can be seen as comprising a perfect example of a resource-sharing system. The attainability set of this system is the set of all goals attainable by all the agencies on a combined limited budget.

It is obvious that the agencies operate in the environment with lots of unknowns, which can be interpreted as perturbations. The attainability set of such a system changes with time and the goals might have to be changed during the course of operation, i.e. in the process of the sub-systems already trying to achieve their original goals.

5.3.2 Project Management

Managing a project involves managing groups of people and giving goals to these groups together with the deadlines for their accomplishment.

Each group develops its own strategy (controls) of how to achieve its goal. The resource in this case could be money or personnel that can be moved from one project to another, depending on which *attainable* goals are pursued.

5.3.3 Space Stations

Planning operations initiated from space installation involves taking care of potentially limited and critical resources - oxygen, fuel, food etc.

A non-futuristic example might involve doing several independent and resource-intensive experiments on the same space station. Depending on the unexpected perturbations, these experiments might have to be replanned on the fly - some of them might be curtailed, while others might be given more resource to proceed. Each experiment in this case can be viewed as an independent subsystem, and the issue of the attainability of all these experiments is important in planning the resource allocation.

5.3.4 Military Supply

It is well known that the problem of supplying advancing military units with fuel and other necessary commodities is very important in military planning. Viewing the units advancing on different fronts as independent subsystems, special care must be exercised in allocating potentially limited resource - the fuel - among these units. This allocation might be better done after studying the attainability set of the combined system consisting of all units involved.

5.4 DRT Computations for a Group of RSS

5.4.1 Algorithm for Computation of the Passive Bounds

When the DM specifies goal (5.9) at some time t_0 it is equivalent to specifying an active range in the N+1 dimensional space:

$$\begin{aligned} &L_i^A = g^{i-}, \quad R_i^A = g^{i+} \qquad (i=1,\dots,N) \\ &L_{N+1}^A = 0, \quad R_{N+1}^A = g^r \end{aligned} \tag{5.11}$$

At time t_0 the combined system has initial conditions:

$$\mathbf{x}^{(1)}(t_0), \;\cdots\; ,\mathbf{x}^{(N)}(t_0),\, r(t_0) \tag{5.12}$$

To calculate the passive bounds of range (5.11) we will proceed in the following order. First, the passive bounds of the A-attainability set will be calculated at some fixed terminal time t^* from the initial time t_0. Then the terminal time t^*, will be varied as a parameter, and the passive bounds of the I-attainability set of the whole system will be obtained.

The resource sharing structure of the underlying subsystems allows effective decomposition of the computations. This decomposition is based on the same idea as Lemma 5.1. This idea is that provided its own share of the resource, a subsystem becomes completely independent of the others.

The results presented in this chapter will be illustrated on a simple example. The development of the example will proceed in parallel with the development of the general approach.

The example deals with a group of two resource sharing systems, each described by a first-order linear differential equation:

$$\begin{aligned} &\dot{x}^1(t) = u^1(t) + 100 && \text{(the first subsystem)} \\ &\dot{x}^2(t) = u^2(t) && \text{(the second subsystem)} \end{aligned}$$

Here $x^i(t)$, ($i=1,2$) is the i-th subsystem state, $u^i(t)$, is the i-th subsystem control. The subsystems' sets of admissible controls Ω_u^i are:

$$\begin{aligned} &\Omega_u^{(1)}: \ |u^1(t)| \le 1 \ \text{(the first subsystem)} \\ &\Omega_u^{(2)}: \ |u^2(t)| \le 2 \ \text{(the second subsystem)} \end{aligned} \tag{5.13}$$

The goal functions are:

$$\begin{aligned} &G^1(t) = x^1(t) \ \text{(the first subsystem)} \\ &G^2(t) = 2x^2(t) \ \text{(the second subsystem)} \end{aligned}$$

and the resource functions are:

$$\begin{aligned} &\phi^1(u^1(t)) = |u^1(t)| \ \text{(the first subsystem)} \\ &\phi^2(u^2(t)) = 2|u^2(t)| \ \text{(the second subsystem)} \end{aligned}$$

The initial time $t_0 = 0$ and the initial conditions are:

$$\begin{aligned} &x^1(0) = 0 \ \text{(the first subsystem)} \\ &x^2(0) = 0 \ \text{(the second subsystem)} \end{aligned}$$

Let's introduce the following 3 functions, Δ_-^i, Δ_+^i, Λ^i, for each subsystem in the group:

$$\begin{cases} \Delta^i_-(\rho,t^*) = \min_{u \in \Omega^i_u} G^i(t_0,t^*,\mathbf{x}^i(t_0),\mathbf{u}_{[t_0,t^*]},\mathbf{p}^i) \\ \text{subject to:} \quad F^i(\mathbf{u}(t),t_0,t^*) \leq \rho \end{cases} \tag{5.14}$$

$$\begin{cases} \Delta^i_+(\rho,t^*) = \max_{u \in \Omega^i_u} G^i(t_0,t^*,\mathbf{x}^i(t_0),\mathbf{u}_{[t_0,t^*]},\mathbf{p}^i) \\ \text{subject to:} \quad F^i(\mathbf{u}(t)t_0,t^*) \leq \rho \end{cases} \tag{5.15}$$

$$\begin{cases} \Lambda^i(l,r,t^*) = \min_{u \in \Omega^i_u} F^i(\mathbf{u}(t)t_0,t^*) \\ \text{subject to:} \quad l \leq G^i(t_0,t^*,\mathbf{x}^i(t_0),\mathbf{u}_{[t_0,t^*]},\mathbf{p}^i) \leq r \end{cases} \tag{5.16}$$

The domain of these functions is $\rho \geq 0$, $t^* \geq t_0$, and roman $l \leq r$. Each optimum in (5.14) - (5.16) must be found also considering the evolution equation (5.1) as another constraint.

Functions Δ^i_- and Δ^i_+ express correspondingly the minimum and the maximum values of the i-th goal function, as a function of the subsystem's resource limitation ρ. These functions express the limits that the goal function can reach, under a given limitation of the resource.

For the example at hand, for any terminal time $t^* \geq 0$, these functions can be found from the closed form expression for G^i:

$$\begin{aligned} G^1(t) &= x^1(t) = \int_0^t u^1(\tau)d\tau + 100t \\ G^1(t) &= 2x^2(t) = 2\int_0^t u^1(\tau)d\tau \end{aligned} \tag{5.17}$$

and from the admissibility constraints (5.13). These functions are:

$$\Delta^1_-(\rho,t^*) = \begin{cases} 100t^* - \rho & \text{if } \rho \le t^* \\ 99t^* & \text{if } \rho \ge t^* \end{cases} \quad \Delta^2_-(\rho,t^*) = \begin{cases} -\rho & \text{if } \frac{\rho}{4} \le t^* \\ -4t^* & \text{if } \frac{\rho}{4} \ge t^* \end{cases}$$

$$\Delta^1_+(\rho,t^*) = \begin{cases} 100t^* + \rho & \text{if } \rho \le t^* \\ 101t^* & \text{if } \rho \ge t^* \end{cases} \quad \Delta^2_+(\rho,t^*) = \begin{cases} \rho & \text{if } \frac{\rho}{4} \le t^* \\ 4t^* & \text{if } \frac{\rho}{4} \ge t^* \end{cases} \tag{5.18}$$

Function Λ^i expresses the minimum amount of the resource necessary to allocate to the i-th subsystem, so that the goal function G^i can reach values inside of interval $[\ L,R\ ]$. Whereas the domain of $\Delta^i_\pm$ is any t^*, such that $t^* \geq t_0$, the domain of Λ^i might be more complicated. This follows from the consideration that the interval $[\ L,R\ ]$ might be so "far away" from the value of the goal function at time t_0, that the goal function's value at some other time t^* would never reach that interval, regardless of the amount of the resource the subsystem can use.

This would happen if the interval $[\ L,R\ ]$ is located outside of the A-attainability set at time t^* of the same subsystem, that had no resource limitation. To be exact, the domain of $\Lambda^i(l,r,t^*)$ are all l,r,t^* for which the following *reachability condition* (5.19) holds:

$$\overline{\psi}^i_{t_0,t^*} \cap [l,r] \neq \varnothing \tag{5.19}$$

where $\overline{\psi}^i_{t_0,t^*}$ is the A-attainability set of the i-th subsystem with no resource limitation.

The domain of Λ^i could be interpreted as a set of terminal times as a function of l and r. We will denote this set $\Xi^i_{l,r}$, so that the optimization domain of (5.16) is not empty for those and only those terminal times t^*, that belong to this set: $t^* \in \Xi^i_{l,r}$ (i.e. for which (5.19) holds). Condition

$$t^* \in \Xi^i_{l,r} \tag{5.20}$$

will be referred to as the *consistency* condition for the optimization problem (5.16)

Consider, for example, the second system. From the closed form expression for G^2, (5.17), one can conclude that at time $t^* = 10$ the value of the goal function can be anywhere within interval [- 20,20]. This interval is the B-attainability set $\bar{\psi}^{(2)}_{0,10}$ of the second system from time $t_0=0$ by time $t^*=10$. Obviously, if l and r are selected so that, for example, $20 < l$ then $A^2(l,r,10)$ would not exist.

The set of "good" terminal times therefore is determined from the relations

$$\begin{cases} l \leq 4t^* \\ r \geq -4t^* \end{cases}$$

so that $\Xi^{(2)}_{l,r}$ is an interval of times $[\max(\frac{l}{4},-\frac{r}{4},0),\infty)$.

Similarly, for the first subsystem the consistency condition for the terminal time is

$$\begin{cases} l \leq 101t^* \\ r \geq 99t^* \end{cases}$$

and $\Xi^{(1)}_{l,r}$ is an interval of times $[\frac{l}{101},\frac{r}{99}]$.

For the consistent values of t^* function A^i can be now expressed (based on (5.17)) as:

$$\Lambda^1(l,r,t^*) = \begin{cases} 1 - 100t^* & \text{if } \frac{1}{101} \le t^* \le \frac{1}{100} \\ 0 & \text{if } \frac{1}{100} \le t^* \le \frac{r}{100} \\ 100t^* - r & \text{if } \frac{r}{100} \le t^* \le \frac{r}{99} \end{cases} \tag{5.21}$$

$$\Lambda^2(l,r,t^*) = \begin{cases} l & \text{if } l \ge 0 \\ 0 & \text{if } l \le 0 \text{ and } r \ge 0 \\ -r & \text{if } r \le 0 \end{cases}$$

Note that Λ^1 is defined only when $r \ge 0$.

Using these functions, $\Delta^i_\pm$ and Λ^i, the algorithm for calculation of the passive bounds of the range that corresponds to the active bounds (5.11) proceeds in the following order:

1. Find a set of terminal times, Ξ, that is consistent with all the subsystems' active bounds (5.11). This means that this set contains all those and only those terminal times t^*, for which it is possible to find appropriate controls for each subsystem, such that the subsystem would achieve its goal with these controls. The controls in the meantime are to be chosen with no regard to the resource limitation. In other words, for each subsystem i, there exist controls $\mathbf{u}^i \in \Omega^i_u$ such that

$$g^{i-} \le G^i(t_0,t^*,\mathbf{x}^i(t_0),\mathbf{u}|_{[t_0,t^*]}) \le g^{i+} \tag{5.22}$$

for all i=1,..,N.

It is easy to see that this set of terminal times Ξ is the *intersection* of the individual subsystems' consistency sets:

$$\Xi = \Xi^{(1)}_{L_1^A,R_1^A} \cap \Xi^{(2)}_{L_2^A,R_2^A} \cap \cdots \cap \Xi^{(N)}_{L_N^A,R_N^A} \tag{5.23}$$

where $\Xi^{i}_{L_i^A,R_i^A}$ is the consistency set of the i-th subsystem[3] that corresponds to the active bounds of the range (5.11). We will call this set, Ξ, the *consistency set* of the range (5.11).

For any active range the consistency sets of each subsystems are intervals of the time axis:

$$\Xi^{(1)}_{L_1^A,R_1^A} = [\frac{L_1^A}{101}, \frac{R_1^A}{99}]$$
$$\Xi^{(2)}_{L_2^A,R_2^A} = [\max(\frac{L_2^A}{4}, \frac{R_2^A}{4}, 0), \infty)$$

The consistency set of the whole system is the union of these sets.

Suppose that the goal for this system (the analog of (5.9)) is

$$g^{1-} = 1000,\ \ g^{1+} = 1200,\ \ g^{2-} = 40,\ \ g^{2+} = 60,\ \ g^{r} = 45$$

This goal is translated into the active bounds of a range

$$\begin{array}{ll} L_1^A = 1000 & R_1^A = 1200 \\ L_2^A = 40 & R_2^A = 60 \\ L_3^A = 0 & R_3^A = 45 \end{array} \tag{5.24}$$

The individual consistency sets are:

3. The latter set contains all those t^* that belong to the consistency domain of $A^i(L_i^A,R_i^A)$. This is a set of terminal times for which the i-th subsystem *can* reach its goal (L_i^A,R_i^A).

$$\Xi^1_{1000,1200} = [9.9, 12.12] \quad \text{and} \quad \Xi^2_{40,60} = [10, \infty)$$

and the consistency set of the whole system is time interval

$$\Xi = \Xi^1_{1000,1200} \cap \Xi^2_{40,60} = [10, 12.12] \tag{5.25}$$

2. Find such a subset $\Xi' \subseteq \Xi$ of the consistency set Ξ, that contains terminal times t^*, for which there is enough resource, specified by the bounds (5.11), to achieve all the subsystems' subgoals.

To find out if a particular terminal time $t^* \in \Xi$ belongs to Ξ', we add up all minimum amounts of the resource that are needed for each subsystem to achieve its own goal, which is specified by the bounds (5.11) (i.e. L_i^A, R_i^A). Then this sum is compared with $R_{N+1}^A = g^r$, i.e. with the *maximum available* amount of the resource, which is also specified by the the range (5.11).

It is easy to see that if

$$\Lambda(t^*) = \sum_{i=1}^{N} \Lambda_i(L_i^A, R_i^A, t^*) \geq R_{N+1}^A \tag{5.26}$$

then there is enough resource to achieve all the subsystems' goals. On the other hand, if t^* is such that (5.26) does not hold, i.e. $\Lambda(t^*) < R_{N+1}^A$, then the subsystems' goal cannot be met at that terminal time t^*. Function $\Lambda(t^*)$ is the *minimum resource* function of the system at time t^*.

To conclude: Ξ contains those and only those terminal times t^*, for which (5.26) is true. This set will be referred to as the *resource sufficiency* set.

For the range (5.24), using (5.21), we get (recalling the consistency set (5.25)):

$$\Lambda_1(1000,1200,t^*) = \begin{cases} 0 & \text{if } 10 \le t^* \le 12 \\ 100t^* - 1200 & \text{if } 12 \le t^* \le 12.12 \end{cases}$$

$$\Lambda_2(40,60,10) = 40$$

and

$$\Lambda(t^*) = \Lambda_1(1000,1200,t^*) + 40$$

The resource goal is specified as 45 (from (5.24)). Hence we can conclude that $\Lambda(t^*) \le 45$ only when $t^* \le 12.05$. Therefore the *resource sufficiency* set Ξ' is time interval [10,12.05].

3. At this stage we get the passive bounds of the range (5.11) by optimizing some functions over all terminal times t^* in the *resource sufficiency* set. Specifically, we calculate the passive bounds of the range associated with the I-attainability set of the whole system from time t_0, using (5.27):

$$\begin{aligned} L_i^P &= \max \{ L_i^A, \min_{t^* \in \Xi'} \Delta_-^i(\Lambda^i(L_i^A,R_i^A,t^*) + \rho(t^*), t^*) \} \\ R_i^P &= \min \{ R_i^A, \max_{t^* \in \Xi'} \Delta_+^i(\Lambda^i(L_i^A,R_i^A,t^*) + \rho(t^*), t^*) \} \end{aligned} \tag{5.27}$$

where

$$\rho(t^*) = R_{N+1}^A - \Lambda(t^*) \tag{5.28}$$

is the *resource surplus* at time t^*, i.e. the amount of the resource that remains available after the goals of all N subsystems have been met. $\Lambda(t^*)$ is the minimum resource function defined in (5.26).

In effect, calculations (5.27) amount to optimization over all possible terminal times (i.e. over those times that belong to the sufficiency set of the range) of the following function of each subsystem i: This function is the minimum (maximum), that a goal function of the i-th subsystem can reach, provided that the i-th subsystem is given as much resource as is available, *after* all other subsystems have been allocated their "bare" minimum of the resource necessary for them to reach their own goals.

The passive bounds of the resource itself, (i.e. the minimum and the maximum amounts of the resource, that will remain available after all N subsystems have reached their own goals) are calculated in (5.29):

$$\begin{aligned} L^P_{N+1} &= R^A_{N+1} - \max_{t^* \in \Xi'} \Lambda(t^*) \\ R^P_{N+1} &= R^A_{N+1} - \min_{t^* \in \Xi'} \Lambda(t^*) \end{aligned} \tag{5.29}$$

Let's calculate the passive bounds for our example.

From (5.21) we can calculate the amount of the *surplus resource* (5.28) which is:

$$\rho(t^*) = \begin{cases} \text{if } 10 \le t^* \le 12: & 45 - (0 + 40) = 5 \\ \text{if } 12 \le t^* \le 12.05: & 45 - (100t^* - 1200 + 40) = 1205 - 100t^* \end{cases}$$

and using (5.21) we get:

$$\Lambda^1(1000,1200,t^*) + \rho(t^*) = 5$$

$$\Lambda^2(40,60,t^*) + \rho(t^*) = \begin{cases} 45 & \text{if } 10 \le t^* \le 12 \\ 1245 - 100t^* & \text{if } 12 \le t^* \le 12.05 \end{cases}$$

and substituting this in (5.18), we get:

$$\begin{aligned} \min_{10 \le t^* \le 12.05} \Delta^1_-(\Lambda_1(1000,1200,t^*) + \rho(t^*),t^*) &= \min_{10 \le t^* \le 12.05} (100t^* - 5) &= 995 \\ \max_{10 \le t^* \le 12.05} \Delta^1_+(\Lambda_1(1000,1200,t^*) + \rho(t^*),t^*) &= \max_{10 \le t^* \le 12.05} (100t^* + 5) &= 1210 \end{aligned}$$

$$\min_{10 \le t^* \le 12.05} \Delta^2_-(\Lambda_2(40,60,t^*) + \rho(t^*),t^*)$$
$$= \min\{\min_{10 \le t^* \le 12} \Delta^2_-(\Lambda_2(40,60,t^*) + \rho(t^*),t^*), \min_{12 \le t^* \le 12.05} \Delta^2_-(\Lambda_2(40,60,t^*) + \rho(t^*),t^*)\}$$
$$= \min\{\min_{10 \le t^* \le 12} \Delta^2_-(45,t^*), \min_{12 \le t^* \le 12.05} \Delta^2_-(1245 - 100t^*,t^*)\}$$
$$= \min\{-45, \min_{12 \le t^* \le 12.05} (100t^* - 1245)\} = -45$$

$$\max_{10 \le t^* \le 12.05} \Delta^2_+(\Lambda_2(40,60,t^*) + \rho(t^*),t^*)$$
$$= \max\{\max_{10 \le t^* \le 12} \Delta^2_+(\Lambda_2(40,60,t^*) + \rho(t^*),t^*), \max_{12 \le t^* \le 12.05} \Delta^2_+(\Lambda_2(40,60,t^*) + \rho(t^*),t^*)\}$$
$$= \max\{\max_{10 \le t^* \le 12} \Delta^2_+(45,t^*), \max_{12 \le t^* \le 12.05} \Delta^2_+(1245 - 100t^*,t^*)\}$$
$$= \max\{45, \max_{12 \le t^* \le 12.05} (1245 - 100TS)\} = 45$$

and therefore the passive bounds are obtained from (5.27):

$$L_1^P = \max(1000,995) = 1000 \qquad L_2^P = \max(40,-45) = 40$$
$$R_1^P = \min(1200,1210) = 1200 \qquad R_2^P = \min(60,45) = 45$$

So only one passive bound, R_2^P is different from the corresponding active bound.

5.4.2 Algorithm for Computation of the Maximum Satisficing Sets

The algorithm for computation of the maximum satisficing sets in the case of an infeasible range (5.11) takes advantage of the resource-sharing structure of the system. The basic idea of the computational algorithm is to relax *just enough* active bounds, so that the problem becomes feasible.

The algorithm follows the same recursive path, as the general algorithm described in Chapter 3. The only difference is in the computation of the feasibility of a node of the tree.

The node number in this case is N+1 digits long. The N+1-th digit corresponds to the resource constraint. This constraint is treated differently from all others constraints for the following reason. The resource constraint that is reflected in R^A_{N+1} in the active range (5.11), is a *soft* constraint, like the subsystems' goal constraints. In most practical problems, however there is another, *hard* resource constraint. The latter constraint expresses the "maximum possible limit" on the total amount of the resource. We denote this *hard* limit by R^{max}.[4] The introduction of this hard constraint does not affect the generality of the algorithm, since this limiting value may well be infinite, i.e. it might be that $R^{max} = \infty$.

Before proceeding with the recursive part of the algorithm, the following items need to be computed. Compute the *terminal time consistency sets* for all subsystems, i.e. compute sets $\Xi^i_{L_i^A, R_i^A}$ for i=1,...,N. All these sets are some subsets of the half-axis $t \geq t_0$ of the real numbers. In most practical cases each set is a union of non-overlapping intervals of the real axis, so these sets allow easy computer representation[5].

Define a set of *activated constraints* of any node whose number is

4. Similar *hard* limitations could exist for other subsystems as well. They can be easily incorporated into the algorithm.
5. A union of non-overlapping intervals can be stored, for example, as an array of the end points of the intervals.

$$J = \{n_1, \ldots, n_N, n_{N+1}\} \qquad (5.30)$$

(where n_j is either 1 or 0). Consider a set of integers ξ_J such that it contains integers from 1 to N that are the numbers of all those positions of the node number J that contain 1. This set is called a set of ***activated constraints*** [6]. This set also can be interpreted as a set of the ***activated subsystems*** of the node.

$$j \in \xi_J \text{ iff } n_j = 1 \qquad (5.31)$$

Then we proceed to the recursive part of algorithm described in Chapter 3. For each node of the tree the following algorithm is used to establish the node's feasibility.

To determine if node (5.30) is feasible, the following computations are performed:

1. Check if the resource constraint is activated in the node, i.e. check if $n_{N+1} = 1$. If it is not, then replace the resource active bound with the hard resource constraint, i.e. set $R^A_{N+1} = R^{max}$. Otherwise retain the user specified value, i.e. $R^A_{N+1} = g^r$.[7]

2. Find the intersection of all those terminal time consistency sets that correspond to the activated subsystems' constraints in the node, i.e. form

6. For example, if J = { 1001101 } then set $\xi_J = (1,4,5,7)$.
7. g^r is defined in (5.9)).

set Ξ_J (subindex indicates the connection to the node (5.30)), such that:

$$\Xi_J = \bigcap_{j \in \xi_J} \Xi^{j}_{L_j^A, R_j^A} \tag{5.32}$$

If this set is empty, i.e. $\Xi_J = \varnothing$, then the node is INFEASIBLE, and no further computations are needed for this node. If this set is not empty, then proceed with the next step.

3. At this stage we perform computations that are similar to the ones of the previous section of this chapter. Namely, we compute Λ, that is the minimum over all possible terminal times of the amount of the resource needed for the activated subsystems to meet their goals:

$$\Lambda = \min_{t^* \in \Xi_J} \{ \sum_{i \in \xi_J} \Lambda_i(L_i^A, R_i^A, t^*) \} \tag{5.33}$$

If $\Lambda \leq R_{N+1}^A$ then it means that the node is FEASIBLE, otherwise it is INFEASIBLE.[8]

This concludes the identification of the Maximum Satisficing Sets.[9]

After all MSSs have been identified, the passive bounds are computed for each MSS separately. These computations are similar to the ones described in the previous chapter. The only difference is that the non-activated constraints of

9. As it was noted in Chapter 3, the computations in a node can be performed by starting the iterative process from the solution of the parent node.

the MSS are not taken into account in the resource computations.

To summarize this step, the following computations are done for each MSS. Suppose that an MSS is characterized by some node number J (5.30).

Then the passive bounds of the activated subsystems (the ones that correspond to the activated constraints) $j \in \xi_J$ are:

$$
\begin{aligned}
L_j^P(J) &= \max\left(L_j^A,\ \min_{t^* \in \Xi_J} \Delta_-^j(\Lambda^j(L_j^A, R_j^A, t^*) + \rho(t^*),\ t^*)\right) \\
R_j^P(J) &= \min\left(R_j^A,\ \max_{t^* \in \Xi_J} \Delta_-^j(\Lambda^j(L_j^A, R_j^A, t^*) + \rho(t^*),\ t^*)\right)
\end{aligned}
\tag{5.34}
$$

and the passive bounds of the non-activated constraints (for all such k that $k \in 1,\dots,N$ and $k \notin \xi_J$) are:

$$
\begin{aligned}
L_k^P(J) &= \min_{t^* \in \Xi_J} \Delta_-^k(\Lambda^k(L_k^A, R_k^A, t^*) + \rho(t^*),\ t^*) \\
R_k^P(J) &= \max_{t^* \in \Xi_J} \Delta_-^k(\Lambda^k(L_k^A, R_k^A, t^*) + \rho(t^*),\ t^*)
\end{aligned}
\tag{5.35}
$$

where Ξ_J is defined in (5.32), and

$$\rho(t^*) = R_{N+1}^A - \Lambda(t^*)$$

where

9. Note that minimization (5.33) does not have to be performed with too high an accuracy. If this minimization is done iteratively (and that is the case for most applications), then the iterative process may be stopped as soon as $\Lambda' \leq R_{N+1}^A$, where Λ' is the value of the objective function in (5.33) at some iteration. Obviously, if $\Lambda > R_{N+1}^A$ then the minimization process has to be concluded to the end, with some beforehand selected accuracy.

$$\Lambda(t^*) = \sum_{i \in \xi_J} \Lambda_i(L_i^A, R_i^A, t^*) \}$$

and all other functions are the same, as defined in the previous section of this chapter.

Finally, the limits of the resource constraint are computed according to (5.36):

$$\begin{aligned} L_{N+1}^P(J) &= R_{N+1}^A - \max_{t^* \in \Xi_J} \Lambda(t^*) \\ R_{N+1}^P(J) &= R_{N+1}^A - \min_{t^* \in \Xi_J} \Lambda(t^*) \end{aligned} \tag{5.36}$$

5.4.3 Algorithm for Computation of the Pareto Cluster Cells

Computation of the PCC in the case of the resource sharing systems with preferentially monotonic goal functions is a blend of the recursive algorithm of the Chapter 3 and the techniques of the previous two sections of this chapter.

The details of the computational algorithm can be easily reconstructed from these sources.

Chapter 6

Case Study of the DRT and Experiments

6.1 The System and the Experimental Tasks

6.1.1 Description of The System

A system for the experimental case study of the DRT should be expected to have the following characteristics:

Generality - the system and the experimental tasks should exhibit all general features of an evolving goal-oriented system described in Chapter 1. The tasks should be formulated in the terms of the *goal* paradigm. The solution of a task should have an inherent multiple-criteria structure arising from the nature of the task and the presence of unknown perturbations.

Complexity - the system and the experimental tasks should be sufficiently complex that they are not amenable to simple and obvious solutions.

Computability - the computational complexity of the tasks, however, should be such that they could be computed relatively quickly, in a matter of seconds. A quick computability is a necessary prerequisite of a task that is to be used in experiments with human subjects and therefore has to be solved repeatedly many times.

Tractability - the utilization of the DRT method should be relatively easy and straightforward, so that the essence of the DRT method can be clearly demonstrated.

Usability - the task does not have to model a real application; however it should be easy to perform, so that the human subjects can master it and get a feel for it in a relatively short period of time (in order of 2-3 hours at the most).

The following system has been selected for the case study and the experimental testing of the DRT method. This system is an aggregation of three resource sharing systems. It was chosen as a system that satisfied all of the above guidelines.

The system consists of a boat and an energy-consuming refrigerating unit that sits aboard the boat.

The boat moves in a 2D plane. It is controlled with two independent thrusters propelling it in X- and Y- directions. A force generated by a thruster is controlled by the amount of fuel flowing into that thruster.

The boat travels in an ocean with unknown ocean currents and varying ambient temperatures. The boat's dynamics are described by two decoupled non-linear second order systems (6.1):

$$\begin{cases} m_b\ddot{x} + k_b\dot{x} = \mu C_x(x,y) + \nu_x f_x \\ m_b\ddot{y} + k_b\dot{y} = \mu C_y(x,y) + \nu_y f_y \end{cases} \tag{6.1}$$

where:

$x(t)$ and $y(t)$ - are the coordinates of the boat;

m_b - is the mass of the boat;

$C_x(x,y)$, and $C_y(x,y)$ - are the X- and Y- components of the ocean current;

f_x and f_y - are the controls: the fuel flows into X- and Y- direction thrusters;

ν_x and ν_y - are *positive* fuel-flow/force constants;

k_b - is a *positive* drag coefficient.

For the computational reasons the following was assumed about the direction of the ocean currents and the system's controls:

- the current is always directed from the upper right corner of the screen down to the lower left corner;
- the X- direction thruster can propel the boat only to the right;
- the Y- direction thruster can propel the boat only in the upward direction.

These assumptions translate into:

$$C_x(x,y) \leq 0, \quad C_y(x,y) \leq 0 \tag{6.2}$$

and

$$0 \leq f_x \leq U_x, \quad 0 \leq f_y \leq U_y \tag{6.3}$$

where U_x and U_y are the maximum values of the controls.

These assumptions have been introduced to speed up the computation of the DRT bounds. As it was noted earlier it was necessary to compute the DRT bounds fast, in a matter of 1-1.5 seconds. This, in fact, was the actual speed achieved by the system in the experiments described below.

The boat carries on board some cargo that is stored inside of the refrigerator unit. The cargo is preserved by a chemical that is being continuously absorbed inside of the refrigerator. The temperature of the cargo (and that of the chemical) depends on the fuel flow into the freezing unit of the refrigerator. This flow is of the same order of magnitude as the flows into the thrusters.

The fuel flow into the freezer unit can be controlled, thus affecting the temperature inside of the refrigerator. The temperature inside of the refrigerator affects the rate of absorption of the chemical. The latter is important, since the total amount of the chemical on board is limited. The lower the temperature of the cargo, the slower is the rate of the absorption of the chemical.

The rate of absorption of the chemical inside of the refrigerator, and the temperature inside of the refrigerator are related to the fuel flow into the freezer unit through the following second order system (6.4):

$$\begin{cases} \dfrac{dM(t)}{dt} = -\phi T + \psi \\ C_v \dfrac{dT(t)}{dt} = h(T_a - T) - \lambda f_r \end{cases} \tag{6.4}$$

where:

$T(t)$	- is the temperature inside of the refrigerator;
$T_a(x,y)$	- is the ambient temperature;
$H(t)$	- is the amount of the chemical;
f_r	- is the freezer control: the fuel flow into the freezer unit $(0 \leq f_r \leq U_r)$;
ϕ, h, λ, C_v	- are some *positive* constants,
ψ	- is some constant.

6.1.2 Experimental Tasks

The goal of all experimental tasks was to navigate the boat in the ocean and to collect as high a score as possible. The boat was navigated from its original position to the destination selected by the DM. The time allotted for each task was limited only by the supply of the fuel and the chemical. The task was automatically terminated when either of these two resources ran out.

A chain of islands was marked on the ocean map. It was displayed on the map as a straight line. The performance score accrued during a task was calculated based on the boat's proximity from this island chain. Two kinds of tasks dealing with reaching various types of destinations along the island chain were considered in the experiments.

1. *Target Approaching.* The goal of a task of this kind was to navigate the boat from its original position to the one that was ***as close as possible*** to a given destination on the islands (Fig. 6.1). The destination was marked on the map with a big circle. The achievement score for this task was the closest distance from the destination attained during a trial.

2. *Line Crossing.* The goal of a task of this kind was to cross a chain of islands ***as far along the chain as possible.*** The numbers along the chain (Fig. 6.2) reflected the score achieved after each crossing of the islands. The performance score for a trial was a maximum score of all achieved crossing scores.

A subject always collected some score doing a Target Approaching task, however she could completely fail the Line Crossing task and collect no score at all. The overall scoring scheme however was such that it was to the subject's advantage to continue the task as long as possible, since in both kinds of tasks the score could only go higher.

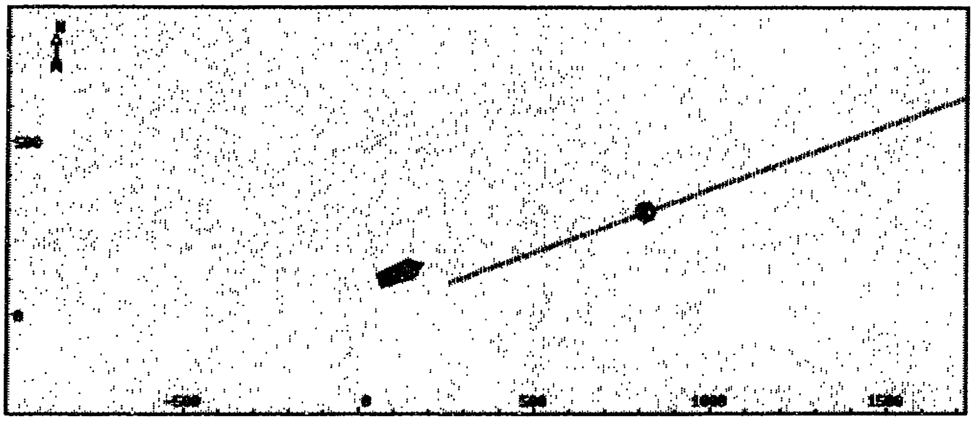


Figure 6.1. Target Approaching Task.

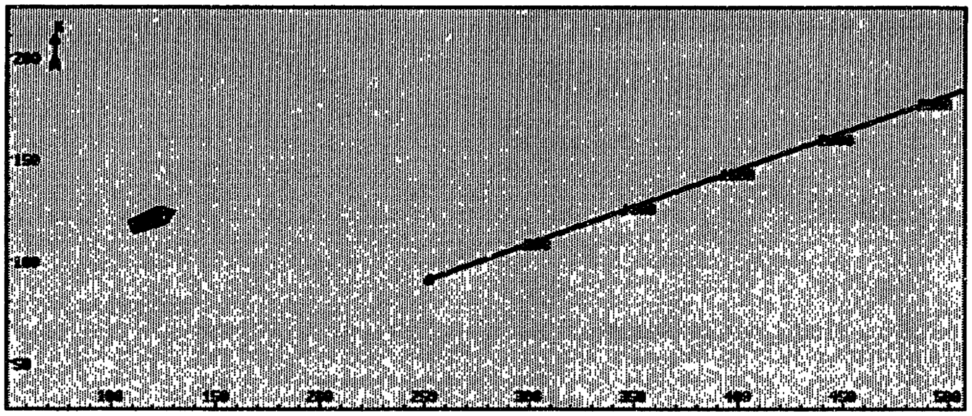


Figure 6.2. Line Crossing Task.

Perturbations in the experimental tasks manifested themselves in two ways. The first type of perturbation was the ocean current and the second one was an *unplanned failure.* While the subjects knew the direction and the relative intensity of the current, they had to cope with only a partial knowledge of the ocean current. An unplanned failure was a sudden loss of some fuel and/or chemical that occurred at random. The subjects were prepared for these failures but did not know whether and when they would take place.

The following parameters varied from one task to another: the constants in the dynamics equations (6.1) - (6.4), the model of the ocean currents (the perturbations), the location of the islands, and the initial conditions:

the boat's initial position and velocity:

$$x(0), \quad y(0), \quad \dot{x}(0), \quad \dot{y}(0)$$

the initial supply of the fuel:

$$\int_0^{\infty}(f_x(t) + f_y(t) + f_r(t))dt = F_0$$

and the initial supply of the chemical:

$$M(0) = M_0$$

The details of the experimental design are described in Chapter 6.2.

6.1.3 Control of the System

6.1.3.1 General Considerations

The system (6.1) - (6.4) was simulated on a Silicon Graphics 2400 Turbo graphics workstation. The DM entered all information into the computer using a computer mouse and a keyboard. The type of information she could exchange with the computer depended on the type of the control mode.

Three modes of controlling the system (boat+refrigerator) have been tested and compared in the experiments: the Manual, the Automatic and the Decision (DRT) control modes. The Manual mode provided the DM with a conventional "hand on" manual control of the system. The Automatic mode was equipped with an Autopilot for navigating the boat towards a specified target. The Decision mode was the implementation of the DRT control technique.

A special feature was added to the interface in order to reduce the penalty for prolonged decisions. This was a *stop time* option. The DM could press a special button and "stop" the process clock, so that she could make her decision without an impending time pressure.

The reasoning behind the adding of this option was the fact that in real control systems the overall time of the process is far greater than the decision-making time[1]. In our experiments one trial had to be finished in a matter of 1-2 minutes. It is obvious that only few complex decisions could be made in such a short time span. The Decision mode required substantially more time for making decisions than the other two modes. (The time was also longer due to a larger amount of information the DM needed to exchange with the computer).

Another consideration was to remove from the control process the element of a skill-based control and to have the DM operate the system on a model-based level.[50] The addition of this feature was also warranted by the fact that the interaction with a more sophisticated decision aid could take more time than the interaction with a simpler one. The stop time feature permitted one to compare different decision aids solely on the basis of their merits thus allowing the DM to take advantages of the capabilities offered by each control mode.

1. As it was noted earlier, the DRT method was intended for systems of that type. Control of a fast system or a system in a quickly changing environment (e.g. a fighter plane) requires a different type of decision aiding.

In all control modes the DM was provided with the following options/information:

Map Display

An area of the screen was designated as a *map window.* The motion of the boat and the position of the islands were displayed in the Map window. The DM could change the scale of the map displayed in the map window (shrink and expand it) and move it up-down-left-right. The DM also could see the direction and the intensity of the ocean currents, as well as the entire past (from the beginning of the trial) trajectory of the boat. The boat was displayed on the computer screen with a vector reflecting the boat's instantaneous velocity. The magnitude and the direction of the instantaneous velocity was calculated relative to the islands (a stationary frame of reference). The direction of the instantaneous velocity was displayed by a dotted line extending from the boat (Fig. 6.3).

Instantaneous Predictor Display

The following *instantaneous* information was displayed to the DM when the boat was heading towards the islands (see Fig. 6.3 as an example):

Proxim -	a distance from the boat to the closest location on the islands (a distance from a point to a line);
Distance -	a distance from the boat to the islands *along the dotted line* (along the instantaneous direction of motion);
Score -	a score of the point on the island chain the boat is heading towards (along the dotted line), if the task was a *Line crossing task*;

Destin -	a distance from the destination on the island chain, if the task was a *Target approaching* task;
Cross. Time -	(*Crossing Time*) the time left before the boat crossed the islands *if it continued to travel with the same speed and in the same direction* as it did at the moment;
Fuel Time -	the time for how long the fuel would last *if the fuel flows in the thrusters and the freezing unit continued at the same rate* as it did at the moment;
Chem. Time -	(*Chemical Time*) the time for how long the chemical would last if it continued to be absorbed at the same rate as it was at the moment.

The DM was also provided with the boat speed and the temperature gauges, as well as with the gauges that showed the total remaining amounts of the fuel and the chemical on board (Fig. 6.3). The last two gauges are marked "Fuel (left)" and "Chemical (left)". The shaded bars inside of these gauges show the actual quantities of these resources left on board.

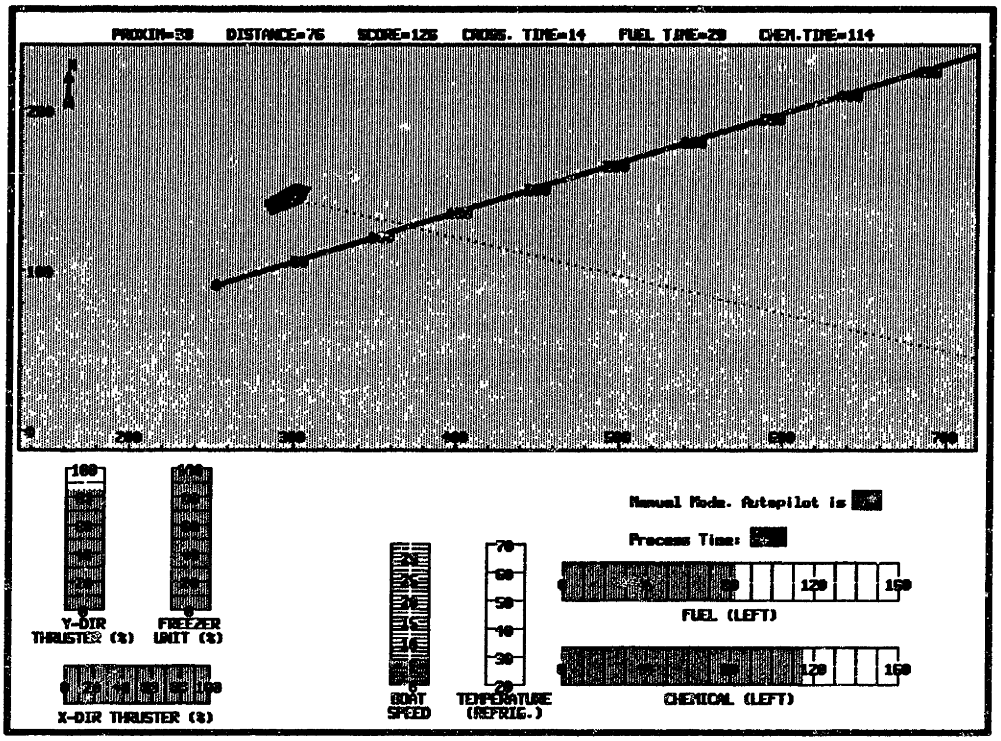


Figure 6.3. A Typical Task Display

6.1.3.2 Manual Control Mode

The basic control mode was the Manual Mode. In this mode the DM was able to adjust manually all 3 controlled values of the fuel flows into the thrusters and into the freezer. This could be done by changing (with a cursor) a length of the corresponding control bar on the display (Fig. 6.3).

The shaded bars inside of the thruster control rectangles (marked "X-Dir Thruster", "Y-Dir. Thruster", and "Freezer Unit") show the percentage of the maximum fuel flow that actually goes into the corresponding thruster.

The computer provided the DM with the Map display and the Instantaneous predictor data described above. Fig. 6.3 shows a typical computer screen during the Manual Control mode.

The Manual Mode was always available to the DM and it could be entered and re- entered from any other mode.

6.1.3.3 Automatic Control Mode

In the Automatic Mode the DM could run the system with an Autopilot. The DM could give the computer a *goal* and the Autopilot would navigate the boat towards that goal. A goal was defined as a 4D vector

$$\mathbf{g} = (x_g, y_g, M_g, F_g). \tag{6.5}$$

The goal (6.5) was achieved at the first instance of time t_g when the coordinates of the boat and the remaining amounts of the fuel and of the chemical on board satisfied the following conditions:

$$\begin{cases} x(t_g) = x_g \\ y(t_g) = y_g \\ M(t_g) \geq M_g \\ F(t_g) \geq F_g \end{cases} \tag{6.6}$$

where

$$F(t) = F_0 - \int_0^t (f_x(\tau) + f_y(\tau) + f_r(\tau))d\tau$$

is the total amount of the fuel left at time t.

The DM entered the specifications (6.5) into the computer using the mouse. These specifications could be changed as often as the DM wished, either while the boat was in motion, or when the system time was "frozen" with the *stop time* option.

Having received the goal specification (6.5) from the DM, the Autopilot navigated the boat synthesizing an optimal control strategy, i.e. adjusting the fuel flows into the thrusters and the freezer so that *the overall fuel expenditures were minimized.* Conditions (6.6) served as the terminal conditions for the optimal control problem. In other words, the solution of the following optimal control problem was synthesized in real time at every instance of time t_0:

$$\min_{t_g,\ f_x(t),\ f_y(t),\ f_r(t)} \int_{t_0}^{t_g} (f_x(t) + f_y(t) + f_r(t))dt \tag{6.7}$$

subject to the dynamical conditions (6.1) - (6.4), terminal conditions with an *unknown terminal time* t_g (6.6) and the initial conditions at time t_0:

$$\begin{gathered} x(t_0) = x_{t_0}, \quad y(t_0) = y_{t_0} \\ \dot{x}(t_0) = v_{xt_0}, \quad \dot{y}(t_0) = v_{yt_0} \\ M(t_0) = M_{t_0} \end{gathered} \tag{6.8}$$

Details of the solution of this optimal control problem can be found in Appendix D.

The optimal control problem (6.7) was solved under the same assumptions about the perturbations (the ocean currents) as the one that are made in the definition of the attainability set, i.e. the constancy of the perturbation. In this example it was assumed that

$$C_x(x,y) = C'_x = \text{constant}, \quad C_y(x,y) = C'_y = \text{constant} \tag{6.9}$$

These assumptions were justified in this case by the following considerations:

- the actual values of the ocean currents were varying within a reasonably *small* interval [2]: $C_x^{min} \leq C_x(x,y) \leq C_x^{max}$, $C_y^{min} \leq C_y(x,y) \leq C_y^{max}$;
- the DM was able to adjust C'_x and C'_y (with a mouse) thus fine-tuning the performance of the automatic controller. It will be discussed later how effectively this option was used by the subjects who participated in the experiments.

The DM needed to provide the Autopilot with the values of C'_x and C'_y. However in the beginning of each session these constants were automatically set to their default values - the average intensity of the current. Two bars marked "X-dir current" and "Y-dir current" on the Automatic Mode display (Fig. 6.4) show the actual values of C'_x and C'_y. The DM could adjust them by moving the corresponding shaded bars on the display with the cursor.

An typical example of the Automatic mode display can be found on Fig. 6.4. A cross on the map corresponds to the goal values of the boat's position ($x_g = 350, y_g = 100$), while the fuel and the chemical goal values are set to 30 and 30 respectively.

The DM knew a priori that the largest currents were to be encountered near the island line, and the magnitude of the currents decreased with the distance from the islands. It was also known to the DM in the experiments that the current streamlines were parallel to the island line[3].

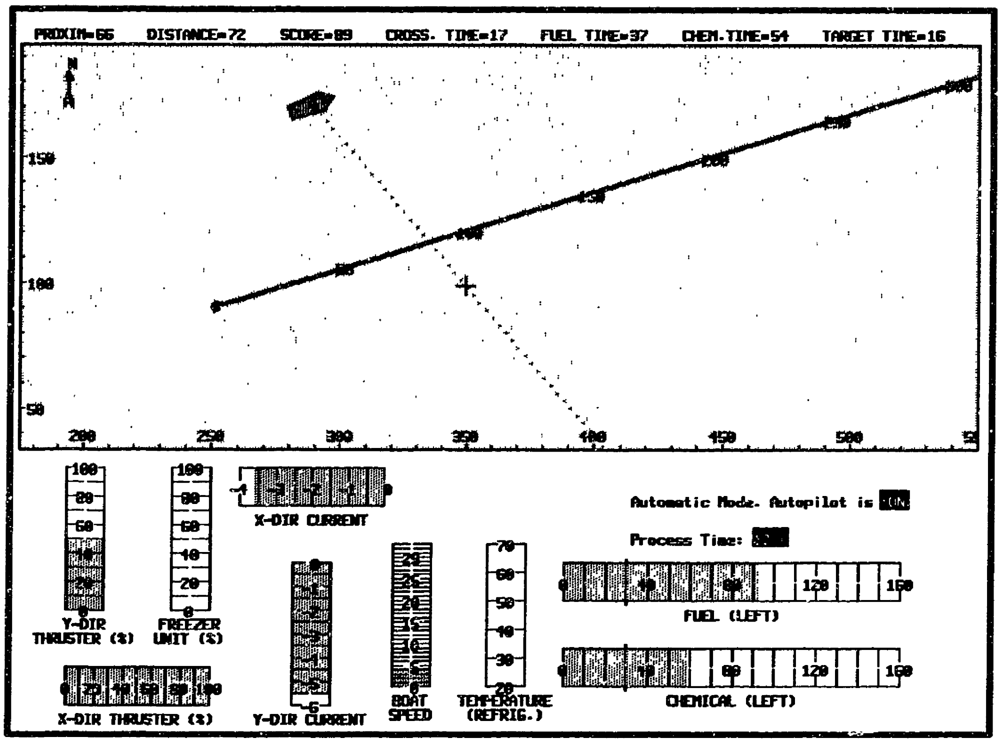
If the optimal control problem (6.7) had a solution, this solution was synthesized and realized by the automatic controller. This corresponded to a case of an *attainable goal.*

On the other hand, if the optimal control problem was inconsistent, and the goal (6.5) was unattainable, then the automatic controller realized a *target pursuit* strategy. This was a non-optimal (as far as the fuel expenditure was concerned) strategy, and it is described in Appendix E. The essence of this strategy was to keep the instantaneous velocity of the boat directed towards the boat position target, while maintaining the freezer fuel flow in such a way that the boat would run out of the chemical and the fuel at about the same time.

The same goal that was determined (by the computer algorithm) to be attainable at some moment of time could become unattainable some time later.

2. If the magnitude of the ocean current varied within a wide range of values, then without some prior knowledge about the underlying functions $C_x(x,y)$ and $C_y(x,y)$ as functions of (x,y), no reasonable planning algorithm could help to navigate the boat. In our system, the constants were used as the simplest ***approximation*** of the currents, in line with the argument about the perturbations in the definition of the attainability set.
3. A current with parallel streamlines does not conform to the 2D conservation law, unless it has a constant magnitude. However, such a current may be viewed as a 2D slice of a 3D current, while the latter is conforming to the 3D conservation law.

The DM would detect this change from a noticeable shift in the behavior of the automatic controller. The change of the controller's behavior can be understood from the fact that the system (6.1) - (6.4) becomes linear with assumptions (6.9). Therefore any *optimal* control strategy is a bang-bang strategy with the thrusters either full open or totally shut. The target pursuit strategy, on the other hand, in most cases generated a partially-open boat thruster.


PROXIM=66 DISTANCE=72 SCORE=89 CROSS. TIME=17 FUEL TIME=37 CHEM.TIME=54 TARGET TIME=16

Automatic Mode. Autopilot is ON

Process Time:

Figure 6.4. A Typical Automatic Mode Display

When the goal became unattainable, the DM could either change the specifications (6.5) of the goal, or adjust the values approximating the disturbance (6.9). The DM could also adjust the parameters of the goal when it was attainable, in the attempt to "push" the goal in the desired direction. The DM could also move the goal position of the boat expecting the boat to track that position.

6.1.3.4 Decision Control Mode

The Decision Mode was designed to help the DM to select a goal. This mode was in the essence a DRT incarnation of the Automatic Mode. The goal specification for this mode was a 4D 2211 active range:

$$\mathbf{g} = (x_g^-, x_g^+, y_g^-, y_g^+, M_g, F_g). \tag{6.10}$$

The goal (6.10) was achieved at the first instance of time t_g when the following conditions became true (compare them with (6.6):

$$\begin{cases} x_g^- \leq x(t_g) \leq x_g^+ \\ y_g^- \leq y(t_g) \leq y_g^+ \\ M(t_g) \geq M_g \\ F(t_g) \geq F_g \end{cases} \tag{6.11}$$

The goal specifications imposed at time t_0 defined the active bounds of a 4D range:

$$\begin{array}{llll} L_1^A = x_g^-, & L_2^A = y_g^-, & L_3^A = M_g, & L_4^A = F_g \\ R_1^A = x_g^+, & R_2^A = y_g^+, & R_3^A = M(t_0), & R_4^A = F(t_0) \end{array} \tag{6.12}$$

The active bounds confined the desired area for the system's state at some time t_g. The corresponding passive bounds were computed and displayed to the DM in almost real time. The passive bounds approximated the attainability set inside the active bounds. Fig. 6.5 shows a typical computer screen for the Decision Mode.

A big rectangle (with the sides specified by the DM) in the map area confines a region of potential goal points with coordinates (x,y) such that:

$$\begin{cases} x_g^- = 490 \leq x \leq x_g^+ = 540 \\ y_g^- = 80 \leq y \leq y_g^+ = 160 \end{cases} \tag{6.13}$$

A smaller (tall and narrow) inner *unshaded* rectangle

$$\begin{cases} L_1^P = 490 \leq x \leq 495 = R_1^P \\ L_2^P = 80 \leq y \leq 130 = R_2^P \end{cases} \tag{6.14}$$

corresponds to the passive bounds of the range. This rectangle shows where the boat *can* actually reach. The *shaded* area between the two rectangles is the area outside of the attainability set, and therefore out of the reach of the boat.

The active bounds for the chemical and the fuel are displayed by rectangles superimposed on the corresponding gauges (Fig. 6.5). Horizontal sides of these rectangles are drawn outside of the gauge bars. A right vertical side of any of these two rectangles coincides with the end of the shaded bar on the gauge, i.e. with the total amount of the resource left. The left side of the active bound resource rectangle corresponds to the goal value of the resource (M_g or F_g). Each of these rectangles thus confines an area on the corresponding resource gauge where the goal value should fall at the time when (6.13) are satisfied. On the display on Fig. 6.5 these values are:

$$\begin{cases} L_3^A = 10, & R_3^A = 80 \\ L_4^A = 60, & R_4^A = 110 \end{cases} \tag{6.15}$$

The passive bounds (6.14) have been computed given the active bounds on the boat's position (6.13) as well as the active bounds for the fuel and the chemical (6.15).

The passive bounds for the fuel and the chemical are also displayed as inner rectangles superimposed with the corresponding gauge bar. A passive bound rectangle is located *inside* of the resource bar (and inside of the corresponding active bound rectangle). On Fig. 6.5 the passive resource bounds are:

$$\begin{cases} L_3^P = 10, & R_3^P = 12 \\ L_4^P = 60, & R_4^P = 78 \end{cases} \tag{6.16}$$

Together passive bounds (6.14), (6.16) provide the DM with an approximation of the attainability set of the system.

The DM could change a goal specification by moving a corresponding side of the appropriate rectangle with a cursor on the screen. The computer prevented the DM from specifying a left active bound higher than a corresponding right active bound and conversely a right active bound lower than a corresponding left active bound.

In the Automatic mode the Autopilot navigated the system towards the target by executing the fuel conservation strategy. The Decision mode offered a choice of this strategy as well as the other one, designed to conserve the chemical.

The selected strategy currently in effect was indicated on the computer display. In the lower-right section of the screen there are two rectangles marked "Optim". These rectangles are located to the right of the resource gauges. The colored rectangle indicates that the resource corresponding to the gauge to the left is currently being conserved (on Fig. 6.5 it is the fuel). The DM could switch from one resource conservation strategy to another by moving the cursor inside of the appropriate rectangle and clicking a mouse button. This could be done while the system was running, as often as desired.

As in the Automatic Mode, if the optimal control problem (6.7) (this time with constraints (6.11)) had a solution, that solution was realized by the Autopilot. If conditions (6.11) were inconsistent (for every t_g), then the computer turned on the alarm and implemented a target pursuit strategy (similar to the one in the Automatic Mode), this time pursuing a target on the map with coordinates $(\frac{x_g^- + x_g^+}{2}, \frac{y_g^- + y_g^+}{2})$. This target was located in the middle of the rectangle formed by the active bounds of the range and described by (6.13).

As in the Automatic Mode, the the DM could adjust the estimates of the ocean current. The default values were automatically set to the average values of the current intensity. On Fig. 6.5 these values are $C'_x = -3.5$ and $C'_y = -5$.

As opposed to controlling the system in the Automatic Mode, the DM could foresee that a presently attainable goal might soon become unattainable. This could be concluded from the "spreading" of the shaded area inside of the active bounds of the range. The shaded area would expand and eventually overtake the region inside the active bounds by the shaded area. As soon as the shaded area filled the inside of the active bound rectangles, the system generated a sound alarm that also alerted the DM to the fact that the goal had become unattainable.

As in the Automatic Mode, when a goal became unattainable the DM had 4 distinct choices:

- to select another goal by adjusting some of the values of the active bounds (6.12).
- to adjust the estimated current values (6.9), in the expectation that this alone would make the same goal attainable;
- to continue running the system in the target pursuit mode;
- to switch to the manual mode and to run the system manually.

These options could be exercised on the fly when the system was running, as well as when the system was stopped and the time was frozen.

The DM also had an option to explore all Maximum Satisficing Sets of the problem, when the goal was unattainable. The computer generated all MSS's and the interface was set up so that the DM could scroll all the MSS's and select a desired one. The computation of the MSS's was done in accordance with the algorithm outline in Chapter 5. If none of the MSS's was satisfactory to the DM, she could go back to the original unattainable range and change some of its specifications.

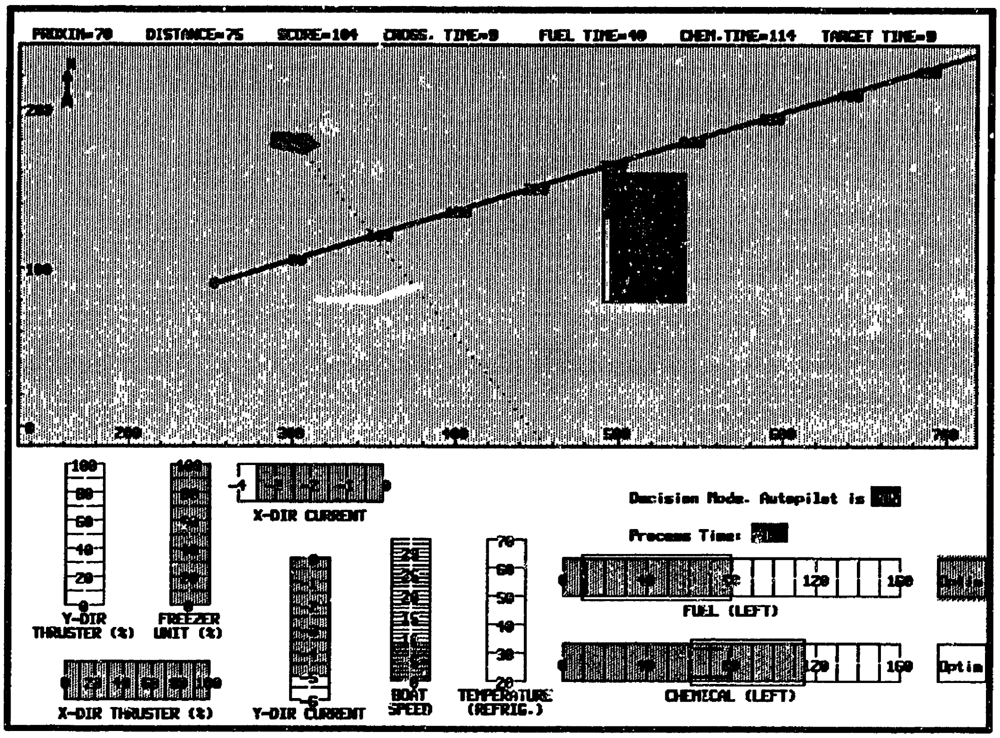


Figure 6.5. A Typical Decision Mode Display

6.2 Design of the Experiments

Twenty tasks were selected for the experiments. Ten of those tasks were the Target Approaching tasks and ten were the Line Crossing tasks. Four subjects participated in the formal data-collection experiments and two subjects participated in the preliminary informal experiments. All of the subjects had either Mechanical Engineering or Mathematical background and were either graduate or former graduate students of MIT.

The data for the experimental tasks were generated using three data bases: one containing the coefficients of ten different dynamical systems (6.1) and (6.4), another containing the data for ten different models of the ocean currents, and the third one containing a list of ***unplanned failures.*** The contents of these data bases is presented in Appendix F.

There were two types of experimental tasks. These types varied in the direction of the island line: in ten tasks the angle between the island line and the X- axis was less than $\frac{\pi}{2}$, in the other ten tasks this angle was greater than $\frac{\pi}{2}$ (Fig. 6.6). The angle of the island line was important because it affected how the ocean current was incorporated into the control strategy of the DM[4].

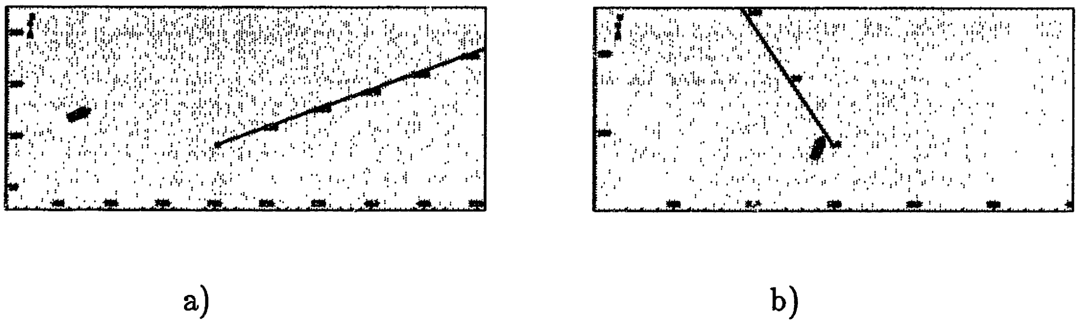


Figure 6.6. Two directions of the island line:

a) angle smaller than $\frac{\pi}{2}$; b) angle greater than $\frac{\pi}{2}$

The *unplanned failures* were built in the experiments to make the DM handle more uncertainty than the unknown current alone. An unplanned failure manifested itself in a sudden loss of some fraction of the available supply of the chemical and/or fuel. The subjects were told that there could be up to three separate sudden failures, with the combined loss of up to 20% of the initial supplies of the fuel and the chemical. The subjects also were aware that the failures could happen only in the first 35 seconds of the trial.

In the course of an experiment each subject performed all twenty tasks, each of the tasks three times - using the Manual, the Automatic, and the Decision modes, thus doing sixty tasks altogether. The subjects were explicitly told that *all* sixty tasks were distinct, so that they would not try to recognize the same task while performing it in different modes.

The experiment was designed in a random block fashion[43]. A choice of a control mode was randomly assigned to each of the twenty baseline tasks and these tasks were presented to the subjects in a random order (different for each subject). A special precaution was taken to avoid a close (sequential) appearance of the same task with different control modes. All in all, each subject expected to solve sixty *distinct* (as she was told) tasks.

4. There are many ways to account for the effect of the ocean currents. To see the difference between two cases on Fig. 6.6 one can notice for example, that in a case a) the boat might initially go against the X-component of the current, and then take advantage of the current and drift with the current (with the X- thruster shut down) towards the goal. In a b) case the boat always has to battle the X- component of the current (i.e. the X- thruster has always to be on).

Each subject was given two trials to do each task, one right after another. The subject also was told that only *the best* score out of the two trials would be credited to her. The addition of the second trial increased the forgiveness of the task[53]. The two trials scheme was implemented for these reasons:

- the first trial was expected to allow the DM to explore the task's environment. During the first trial a subject was expected to get a feel for the system's dynamics, the ocean currents and the sudden failures. It was expected that as the result of the experience gained during the first trial, the subject would act in the second trial with more expertise and the knowledge about that particular task;
- while a target approaching task was a fail proof one, a subject could fail in the line crossing tasks, thus collecting no score at all (if the boat run out of one of the resources before ever crossing the island line). The two trial scheme also was designed to reduce the failure rate in the line crossing tasks.

The experiment had five phases and proceeded in the following order:

1. *Introduction* - A subject was given written instructions in operator's manual describing the system, the tasks and the three control modes. The instructions were illustrated with pictures of the computer screens. The subject read the instructions before going to the computer. This phase took from 45 min. to 1 hour.
2. *Preliminary Practice* - The subject was given eight trial tasks for practicing actual computer skills. The tasks covered all basic permutations of the experimental tasks and included: 4 target approaching tasks, 4 line crossing

tasks; 4 tasks with the island angles below $\frac{\pi}{2}$, and 4 tasks with the angles greater than $\frac{\pi}{2}$. Each task could be repeatedly performed using every control mode. The subject could practice these tasks for 2-3 hours or as long as she felt necessary for mastering the system. The control mode for each of the tasks was selected by the subject herself. The subject could try the same mode performing the same task as many times as she wished.

3. *Instruction / Demonstration* - The subject was shown a demonstration of the system and was given some practical advice as to how to handle different situations arising in the course of performing the tasks.

4. *Expert Practice* - The subject practiced the tasks for another 1-2 hours, this time with the full knowledge of the system and its capabilities.

5. *Data Collection* - The subject performed all 60 experimental tasks assigned to her in a random order and the experimental data was collected automatically by the computer. Some qualitative observations about the subjects' performance were recorded during this stage as well. This phase took about 5-7 hours.

The following data was automatically collected during each experimental trial: the achieved score, the boat's trajectory, the overall process time.

6.3 Results of the Experiments

The following hypothesis was tested in the experiments: the Decision (DRT) control mode was superior to the Automatic and the Manual modes. The scores achieved by the same subjects doing the same tasks were used for the comparison of different modes.

The individual scores achieved by the subjects in each trial during the experiments can be found in Appendix G. The results presented in this chapter are based on the best scores for each of the subject/task/mode triplets (best out of two trials). The performance of the subjects is summarized in tables 6.1-6.2. For each of the control modes the tables show the number of the tasks that received better score using one mode vs. the other.

	Manual	Automatic	Decision
Manual mode better than	*	10	0
Automatic mode better than	10	*	1
Decision mode better than	20	19	*

Table 6.1. Relative performance using the average-best scores.

Tables 6.1 and 6.2 summarize the qualitative comparative performance results taken over all subjects. Table 6.2 shows combined subject's performance using *the average best* scores for each mode/task pair, taken across the subjects. The numbers in Table 6.2 were obtained using *the best* scores for each mode/task pair, taken across the subjects. These tables reflect the performance of a *combined* subject who has performed each task/mode pair 8(=2×4) times.

The combined performance of the subjects is also plotted on Figs. 6.7 and 6.8. On these plots three vertical bars are drawn for each of the 10 tasks (Fig. 6.7 for Target approaching tasks and Fig. 6.8 for Line crossing tasks).

	Manual	Automatic	Decision
Manual mode better than	*	13	3
Automatic mode better than	7	*	2
Decision mode better than	17	18	*

Table 6.2. Relative performance using the best-best scores.

Each bar corresponds to a particular control mode. The bottom of the bar corresponds to the minimum score across the subjects for a particular task using the corresponding control mode. The top of the bar corresponds to the maximum score for the task/control mode combination. The intermediate point on the bar marks *the average* score for the task.

The data for each task, as they are plotted on on Figs. 6.7 - 6.8 are offset by the minimum score for that task (across the subjects and the control modes). This is done in order to fit the results of all tasks on the same plot[5].

On Fig. 6.7 the *lower* values correspond to better performance, while on Fig. 6.7 the opposite is true: the ***higher*** values indicates better performance. These plots reflect the data in Table 6.1. An immediate observation can be drawn from the plots and the table: the average scores using the Decision mode were consistently better that the average scores using the other two modes. These plots also show that the spread of the scores was much tighter when the subjects used the Decision mode than when they used any other mode. In fact, no subject ever failed twice in a line crossing task when using the Decision mode, while there were failures when they used either the Automatic or the Manual modes.

5. The range of the scores achieved for each of the experimental tasks varied from one task to another. The plots preserve the scale of each task's data, but the data are offset by some constant, different for each task.

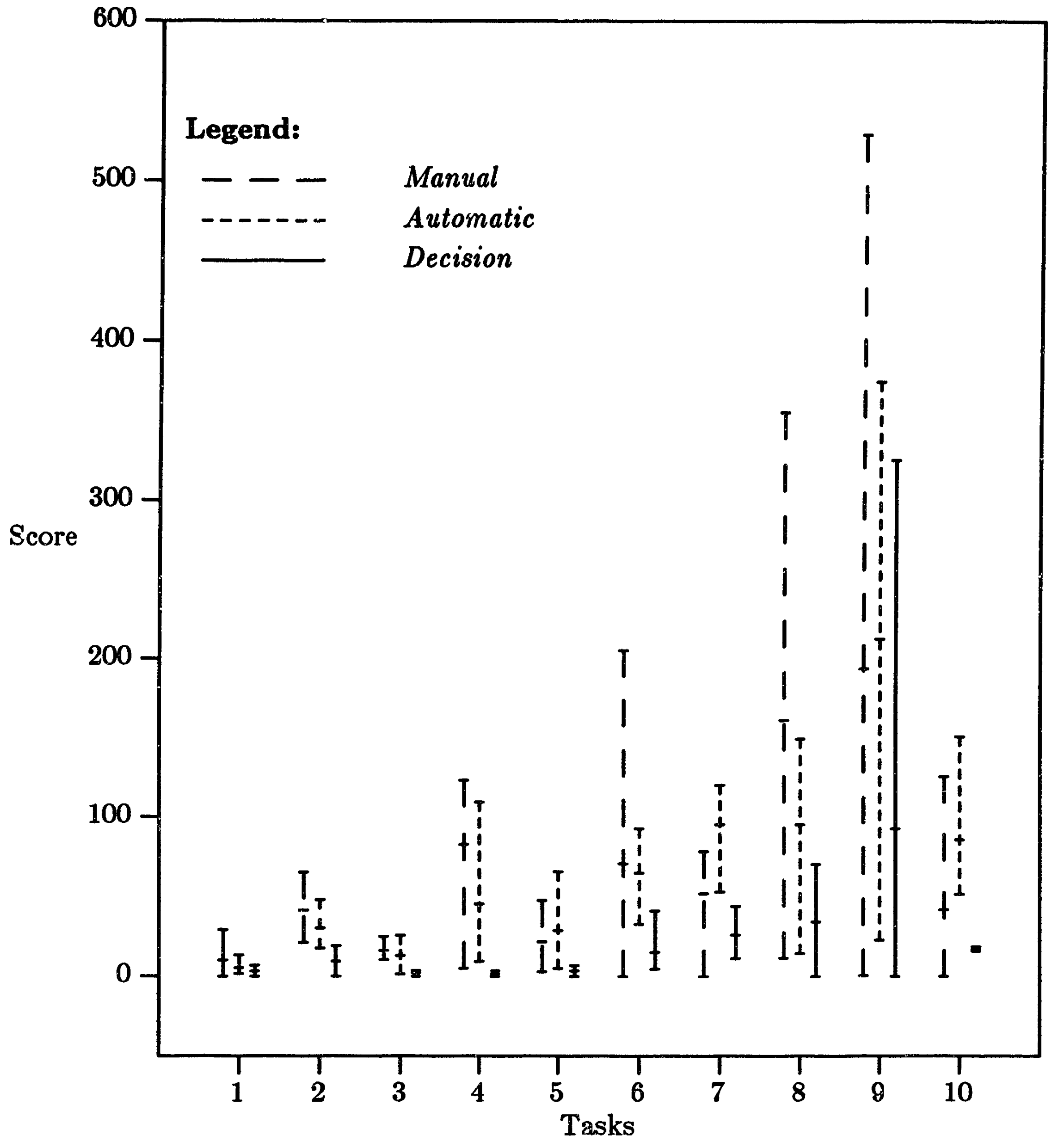


Figure 6.7. Experimental Results: Target Approaching Tasks.

Lower scores reflect better performance.

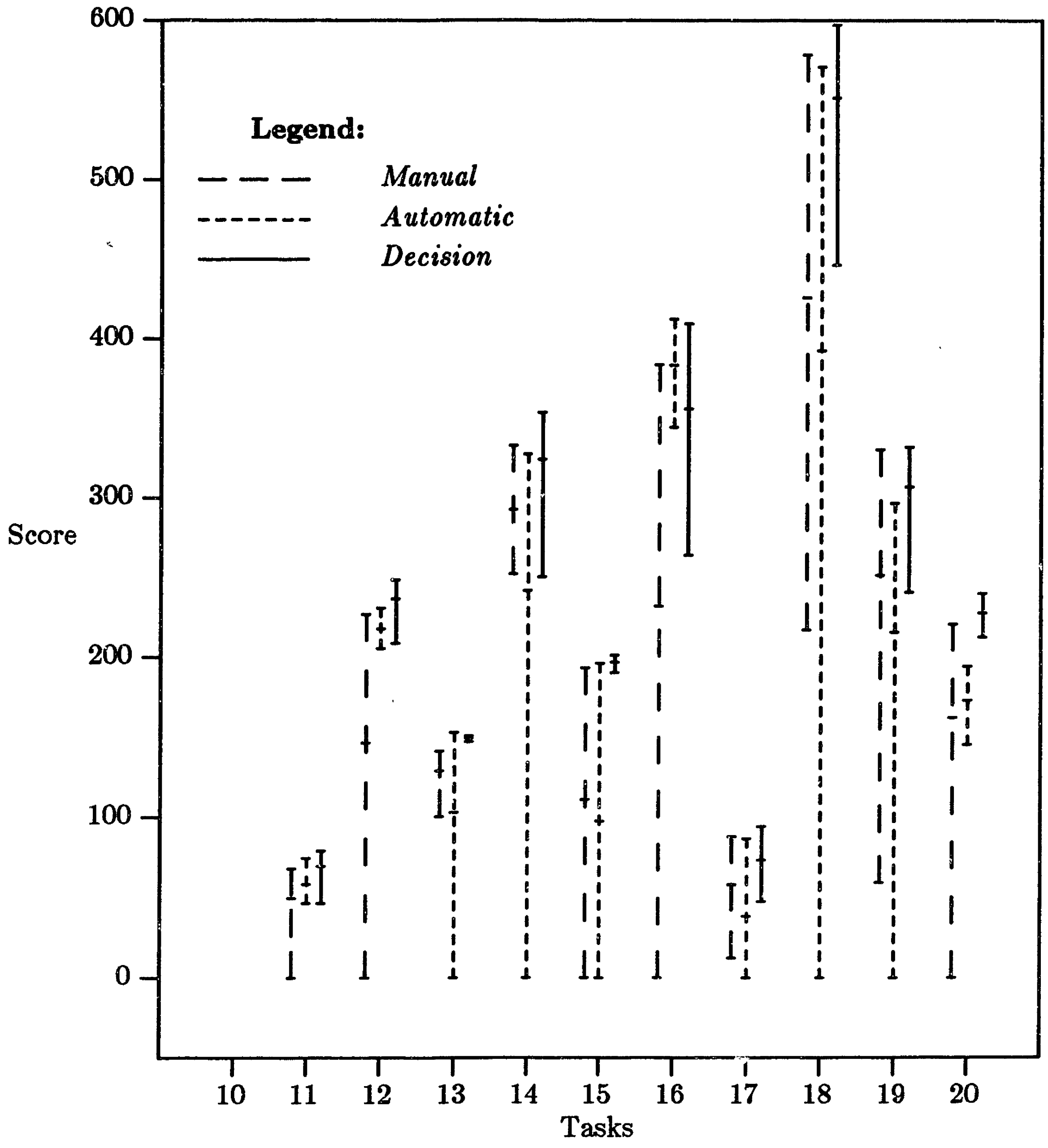


Figure 6.8. Experimental Results: Line Crossing Tasks.

Higher scores reflect better performance.

From these results one can conclude that the Decision mode was far superior across the subjects than the other two modes. Even considering that sometimes the advantage of one mode over the other was marginal, the overwhelming superiority of the Decision mode is clear from the presented results.

Some additional observations concerning individual subject's performances and the merits of the Manual vs. the Automatic modes for each subject could be drawn from Table 6.3.

Subject	*Manual* better than *Automatic*	*Manual* better than *Decision*	*Automatic* better than *Manual*	*Automatic* better than *Decision*	*Decision* better than *Manual*	*Decision* better than *Automatic*
1	4	2	16	7	18	13
2	11	1	9	1	19	19
3	15	3	5	1	17	19
4	15	8	5	2	12	18
Tot:	45	14	35	11	66	69

Table 6.3. Individual subject's performance.

It could be seen from the table that subject 1 had a distinctly better performance using the Automatic mode than the Manual one. On the other hand, subject 4 performed much better using the Manual mode than the Automatic mode. In fact, subject 4 performed extremely well using the Manual mode even compared to the Decision mode.

It was observed during the experiment that subject 4 really tried to perfect the Manual mode while doing just fairly well when using the other two modes. The subject used the *freeze time* option much more frequently when he operated the system in the Manual mode than when he used other modes of control. He would really try to perfect the Manual mode performance by stopping the system, running it for a fraction of a second, and then re-tuning the controls.

It could be seen from Tables A6.4.1 and A6.4.2 that he achieved the best performance using the Manual mode doing the tasks with the island angle greater than $\frac{\pi}{2}$ (6 times out of 10 his Manual mode performance was better). It was observed during the experiments, that when he used the Decision mode doing those tasks, he frequently selected *intermediate* goals for the system, rather than a final goal. This was a very inefficient strategy of using the Decision mode. Due to this inefficiency in using the Decision mode combined with creative use of the Manual mode, the advantage of the Decision mode was not so pronounced in this subject's performance.

6.4 Discussion of the Experimental Results

The "boat" task was selected as a dynamic decision-making task requiring the DM to make tradeoff decisions. Though the tradeoffs varied from one control mode to another, their nature was the same: balancing the reserves of the fuel and the chemical v.s. targeting a *higher* goal.

This tradeoff was especially pronounced in the case of a line crossing task. When the DM performed such a task, she always had in mind some target point on the islands, where the boat would finally cross the island line. The DM was aware that the best strategy to do the job was to go out into the ocean as far as possible and then make a swing and come back to the islands. All subjects realized very quickly that the farther away from the island they would go, the higher goal they could reach.

Going too far away from the islands was, however, risky. The boat could spend too much fuel and/or chemical on the way out, so that there would be not enough of either one of these resource to make it back to the islands. Failing to balance the objectives of going too far (and striving for a higher score) v.s. saving enough resources for the way back, 3 out of 4 subjects failed to accomplish some line crossing tasks in both trials (double failure).

This situation is typical for critical (in which an "optimal" performance is needed) resource-limited tasks. The DRT method was designed to help the DM in this type of decision-making situation. The experiments demonstrated that it has helped. No double failures took place while the system was controlled in a DRT control mode.

The advantages of the DRT method have been demonstrated for the paradigm of controlling a system facing unknown future perturbations. The experimental tasks exhibited features of a typical MCDM problem. Specifically, the preferences for the values of each *attribute* (X-direction, Y-direction, Fuel, and Chemical) where monotonic functions of the attribute values. These attributes were preferentially independent to some degree: for any given target (x,y) the DM preferred to get there with the minimal expenditures of the chemical and the fuel. However for given limits on expenditures of the chemical and the fuel, and some value of x, just bigger (or smaller, depending on the direction of the island line) value of y was ***not better.*** In other words, x and y were not preferentially independent.

The conventional MCDM techniques cannot handle problems of this type. Firstly, they cannot account for unknown perturbations. Secondly, all conventional methods deal with finding a point (not a range) *on* the Pareto set of a problem. In the boat task, the Pareto set was formed by the boundary of the attainability set. This boundary frequently had a rather peculiar shape.

Figs. (6.7) - (6.10) show a few examples of the actual attainability sets in the (x,y) plane for some fixed values of the resource reserves. The outer rectangle in the map window specifies the active bounds of some range, while the inner rectangle shows the passive bounds of the same range. The curve inside of the passive bounds rectangle confines the actual attainability set (more exactly, its projection into $\mathbf{R}^2$: (x,y) plane).

It took the Motorola 68020 based system with a floating point accelerator about 4-5 min. to compute each of these curves. Comparison with 1-1.5 seconds that it took on the average (on the same computer) to compute a set of the passive bounds, shows that a real-time implementation of any algorithm that requires the knowledge of the full Pareto set will be very difficult.

The requirement to know the *entire* Pareto set is indeed a handicap of many conventional methods, including the methods based on the Utility function theory. Using the DRT technique, the DM specifies a very narrow area in the decision space (confined within the active bounds) and all decisions are concentrated in that area. The active bounds on the other hand, are selected after observing the *decision feedback* provided by the corresponding passive bounds. The knowledge of the passive bounds *together* with the understanding of where the target should be located drive the DM's specification of the target region.

The experiments have also shown that it was not to the DM's advantage to specify a very narrow active range (with left bounds close to the corresponding right bounds). Narrow ranges were most frequently used at the very final stages of the task, where the outcome of the fine-tuned steering of the boat was more pronounced. In the beginning and a half-way into the task, narrow ranges proved to be very short-lived and irritating to the DM (by becoming unattainable too quickly).

In all three control modes the DM had some clues whether a particular goal was attainable. In the case of the Manual mode the DM could determine the attainability of the goal by comparing numbers on the predictor display. If the *Crossing* time was bigger than either the *Fuel* or the *Chemical* time, then it was a good indication that *if things went the same way* the target (the crossing point) would be reached. It was obviously in the interest of the DM to keep these times as close to each other as possible, but still having the *Crossing* time larger than the other two. In the case of the Automatic control mode the DM could see right away if the goal was attainable from the behavior of the control system.

It is important to note that just knowing whether a particular goal belonged to the attainability set was not enough for making good decision. The DM needed to know the proximity of the goal from the boundary of the attainability set.

However the *proximity* of the goal to the attainability set could not be sensed in either the Manual or the Automatic control modes. The predictor data in the Manual mode provided some clues about the proximity. However that information was available only at the last stages of the task, when the boat was already heading towards the islands.

Trying to "grope" the attainability boundary using the Automatic mode, the subjects frequently moved the target position of the boat, while the boat continued to move. The boat was expected to track the desired position, while the latter was "pushed" as far as possible in the desired direction (usually along the island chain) before becoming unattainable.

All subjects confirmed that the knowledge of the proximity of the goal from the attainability set was another important advantage of the Decision (DRT) control mode. No conventional MCDM technique that looks for a solution *on* the Pareto boundary provides the proximity information either.

The experimental tasks required dynamic decision making, i.e. the DM had to make *many similar* decisions in the course of one task. Each individual decision had a tradeoff nature and was similar to the decisions encountered in the MCDM paradigm.

The DRT technique can be used for both dynamic and static tasks. The experiments have confirmed several advantages of using the dynamic experimental task (v.s. a static task):

- In the course of performing a single dynamic task the same decision-aid is used many times, thus the aid's usefulness could be determined from a larger testing sample;

- In the static 1 decision-per-task experiment it is very difficult to introduce uncertainty in such a way that the DM could on the one hand remain uncertain about the perturbations and on the other hand acquire some *experience* in dealing with them;

- The evaluation criteria are very difficult to come by for a static tradeoff decision-making task. The very existence of tradeoffs in human decisions arises from the DM being *uncertain* about something connected the task, so the decision *cannot* be easily converted into a single criterion optimization problem. It is an essential component of the MCDM paradigm that different people come up with different solutions. The problem of evaluation of the goodness of those solution, or of the usefulness of alternative decision aids, is a very difficult one. It is my contention that this problem has not been adequately addressed in the literature. The abundance of competing MCDM techniques attests to that.

It is evident that in the conducted experiments the evaluation of alternative decision aids tradeoff was addressed in a novel and a useful way. The evaluation criterion was given to the DM from the very beginning of the task. The same criterion was used by the subjects doing the experiment and by the experimenter evaluating their results. Yet the decisions involved in the experimental tasks had a tradeoff nature, and were similar to the decisions that are part of the framework of the traditional MCDM paradigm.

- The selection of the goal in the Decision mode was a relatively easy task. The DM did not have to agonize about making a terribly wrong choice. This fact shows that the DRT was also useful in *convincing* the DM that she made a good decision. On the one hand this was due to the dynamic nature of the task, that tended not to penalize the DM for a single bad decision, since it could be corrected later (to some degree) in the course of the same task. On the other hand, as it was confirmed by the subjects, the DRT indeed made making a choice easier. This fact is important in light of the issue raised in [6][7] (see Chapter 2) about the inability of the conventional MCDM techniques to convince the DM in the rightness of her choice.

It may be argued that a decision problem arising in any of the experimental tasks is equivalent to a simple one-objective optimization, and that it could be solved by synthesizing an optimal control solution of the following problem (the case of a Line crossing task):

$$\gamma_g = \max_{t_g,\ f_x(t),\ f_y(t),\ f_r(t)} \gamma \tag{6.17}$$

subject to dynamical conditions (6.1) - (6.4), initial conditions (6.8) and the terminal conditions (6.18) with an unknown terminal time t_g:

$$\begin{cases} x(t_g) = X(\gamma_g) \\ y(t_g) = Y(\gamma_g) \end{cases} \tag{6.18}$$

where

$$x = X(\gamma); \quad y = Y(\gamma) \tag{6.19}$$

is a parametric representation of the island line (a straight line in the experiments), and γ a distance along the island line.

It may seem at first glance that problem (6.18) is sufficiently well-defined to be solved *without any human* intervention at all. This assertion however is not correct. The nature of the perturbations (ocean current and unexpected failures) is such, that aiming at the highest possible score (which is defined by the solution of (6.18)) the boat will very likely be unable to reach the islands because it will lack sufficient reserves of the resources. If one planned to allocate these reserves in advance, then the problem of ***how much reserves*** of the chemical and the fuel was enough, would bring one back from the well-defined formulation (6.18) to the one involving tradeoff decision-making.

Another consideration that makes DM's involvement necessary *during* (not only before the start of) the task is connected to the relative simplicity of (6.18) in the case when the islands stretch along a *straight line.* If the island extended along some complex curve, not known to the DM before the start of the task, then problem (6.18) would have been extremely difficult if not impossible to solve. Problem (6.18) would have to be solved in this case for *a general* shape of the island curve, and that is a very difficult problem, especially for curves that do not render a simple analytical representation (6.19). The DRT method would work in this case with the same ease because it is not attached to any particular form or shape of the set of potential terminal states (the island line in this case).

Several issues, though addressed in the experiments, still require further investigation. These issues include:

Approximation of the Perturbations - Only one subject has mastered how to change the values approximating the ocean current. Other subjects made only one or two changes in the course of one task. Though the flexibility of the DRT method compensated to a degree some inaccurate approximations, further study of how the DM could be aided in this is warranted.

Strategic Planning - It has been noted that most of the experimental tasks could be solved by setting and amending a terminal goal, and that setting intermediate goals by the human was inefficient. It is easy to see how the same tasks after just a slight modification would make this *terminal* strategy impossible. Imagine some obstacles in the ocean added to the picture. One could easily see that navigating the boat through the environment with obstacles may render the terminal strategy useless. If there are too many obstacles, the control synthesis problem might become extremely difficult to solve. The option would be to use the same DRT-type decision aid, but setting and changing the intermediate goals instead of the terminal one. The viability of this approach is worth investigating.

6.5 Conclusions from the Experiments

The Decision control mode was shown to be a much better way to accomplish the experimental tasks. In spite of the variability of its advantage from one subject to another, on the average it was clearly the best of all control modes.

The average scores achieved by the subjects using the Decision mode were better than the scores achieved with any other mode in 19 out of 20 tasks. The best scores achieved using the Decision mode were better in 17 out of 20 tasks.

The individual differences in performance using different modes were to be expected. All subjects verbally stated after the experiment that the Decision mode was not only the best mode of controlling the system, but also the easiest one to use. The subjects' however differed in their assessment of the relative merits and the ease of use of the other two control modes.

In the course of the experiments it was observed that the Decision mode was the most effective when used by the subjects who did not aim to profess any particular mode of control, or actually tried to use the Decision mode most efficiently. These were subjects 2, who tried to accomplish each task using each mode equally carefully, and subject 3 who performed all tasks with a lesser degree of attention, but without giving a special attention to any particular control mode.

The most difficult aspect of using the Decision mode was the specification of the values approximating the ocean currents. No prescription was given to the subjects as to how this had to be done. The subjects developed their own strategies of doing that. The most sophisticated use of this option was achieved by subject 2, who scored the best in the overall use of the Decision mode control.

The following observations could be drawn from the experiments. These observations describe the circumstances when the Decision mode control performed better, but yet not *spectacularly* better than the other control modes (in the case of the same subject). It is important to note that the same considerations are valid in a variety of other tasks where the DRT (in my opinion) could be effectively used.

The DRT computer aiding method was not used up to its full advantages under the following circumstances:

Intermediate Goal Setting - This was the case when the subject tried to use the DRT not to set a final goal, but instead mentally planned the strategy of the task and set intermediate goals using the DRT as the aid. This occurred especially often in the tasks with the island angle over $\frac{\pi}{2}$. If the intermediate goal was badly chosen, the overall performance in the task could be adversely affected.

Bad Final Goal Selection - Even when the subject selected a final goal, the goal could be chosen badly. This was particularly pronounced in the case of Target Approaching tasks. In these tasks it was not so simple to decide *where* to select the goal that would be the closest one to the destination. On the other hand, in the case of the Line Crossing tasks the goal selection was relatively straightforward.

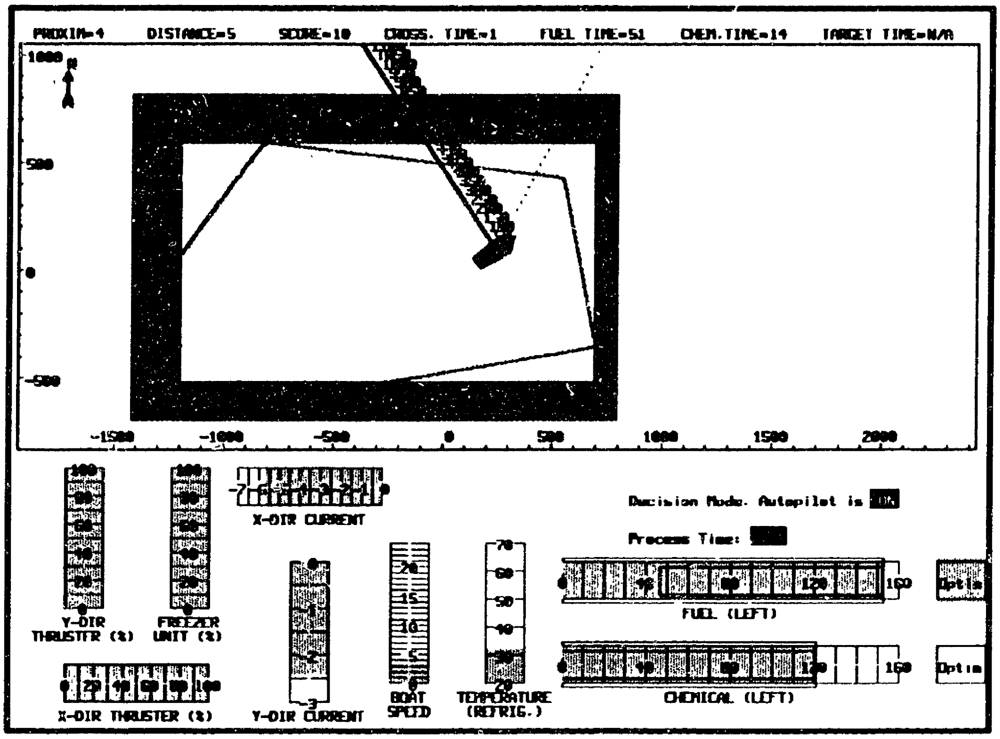
Runaway Decision Aid - The subject could sometimes get carried away and not react fast enough to the signals that the goal became unattainable. The advantages of the DRT faded away rather quickly if the system was operated in the target pursuit mode for too long a time. The prolonged use of the target pursuit mode was, in fact, one of the major handicaps of the Automatic Mode. However in the Automatic mode the subject had no clues as to how to select another, an attainable goal, while these clues where explicitly provided in the Decision mode.

Simple Tasks - In some tasks, such as in task 1 for example, the difference in performance using either mode was not very pronounced. It was observed that the subjects frequently tried to reproduce the bang-bang nature of the Decision mode control (that prevailed in the experimental tasks) when they controlled the system in the Manual mode. As one could see (Appendix A) the optimal control strategy synthesized in the Decision mode had only up to three switching points. This reflected a relative simplicity of the tasks, making it possible for the human to achieve a marginally close performance using a good manual control strategy. Subject 4 actually perfected this type of control.

Effect of Learning - All subjects admitted that they continued to learn how to use the system well into the experiments. The effect of learning was very difficult to compensate because the subjects already had to spend long hours doing the experiment. Further prolonging the experiment was dangerous in that it could adversely affect the subjects' motivation to achieve a better performance. In fact the learning element served as one of the major factors motivating the subjects to perform better. The presence of two trials per task reduced the risk of trying out new alternative strategies and it was an important ingredient in keeping the subjects interested.

To summarize the results of the experiments:

- The performance of the subjects using the Decision mode was consistently better that when using either the Automatic or the Manual modes;
- The spread of individual performance scores was tighter for the Decision mode;
- The failure rate was much smaller (non-existent) when the subjects used the Decision mode;
- The individual performances using the Decision mode ranged from far superior to fairly better compared with either of the other two modes;
- All subjects explicitly confirmed that the Decision mode was the best and the easiest mode (out of the three modes they have been offered) of controlling the experimental system.


PROXIM=4 DISTANCE=5 SCORE=10 CROSS. TIME=1 FUEL TIME=51 CHEM.TIME=14 TARGET TIME=N/A

Decision Mode. Autopilot is ON

Process Time:

Figure 6.7. Example of attainability set #1: boat moving *against* the current.

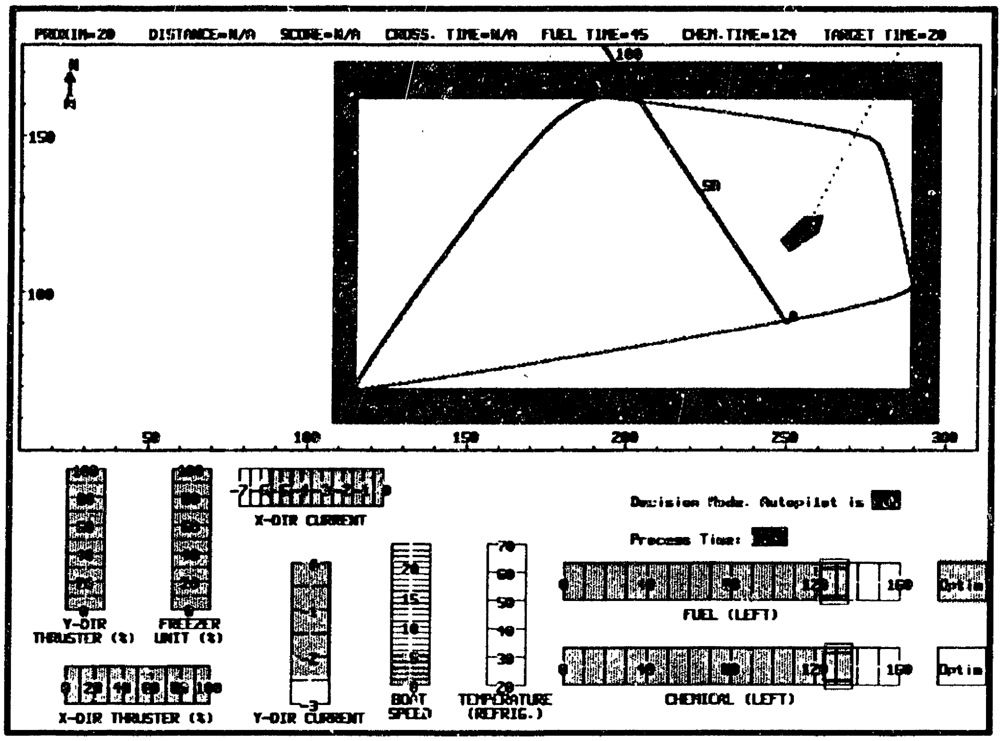
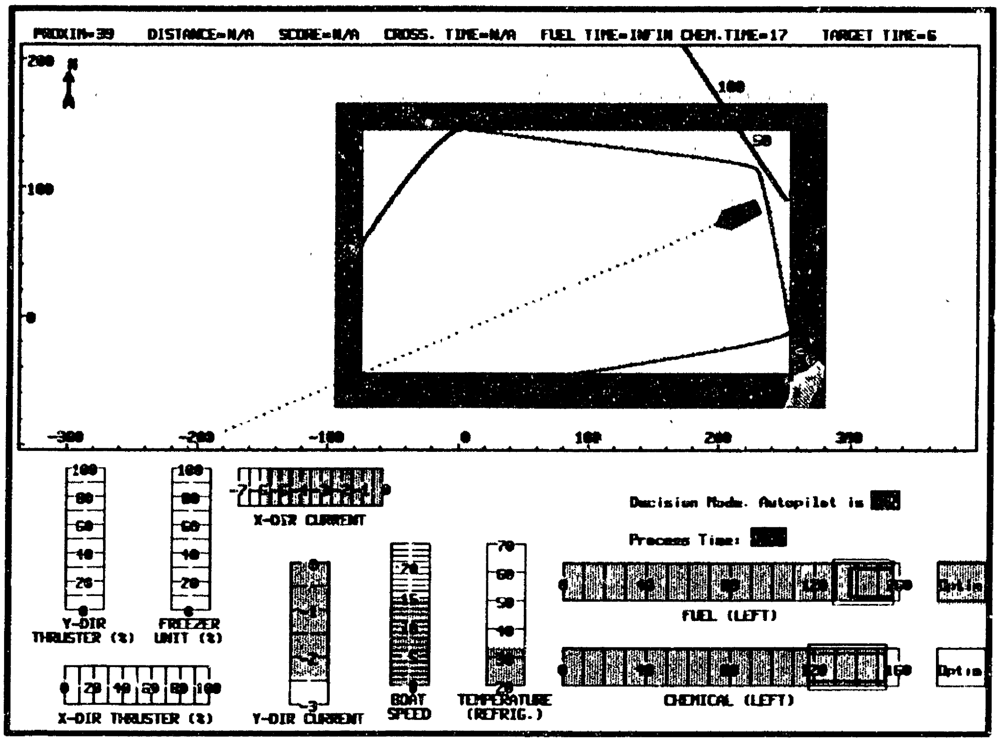


Figure 6.8. Example of attainability set #2: boat moving *against* the current.


PROXIM=39 DISTANCE=N/A SCORE=N/A CROSS. TIME=N/A FUEL TIME=INFIN CHEM.TIME=17 TARGET TIME=6

Decision Mode. Autopilot is

Process Time:

Figure 6.9. Example of attainability set #3: boat moving *with* the current.

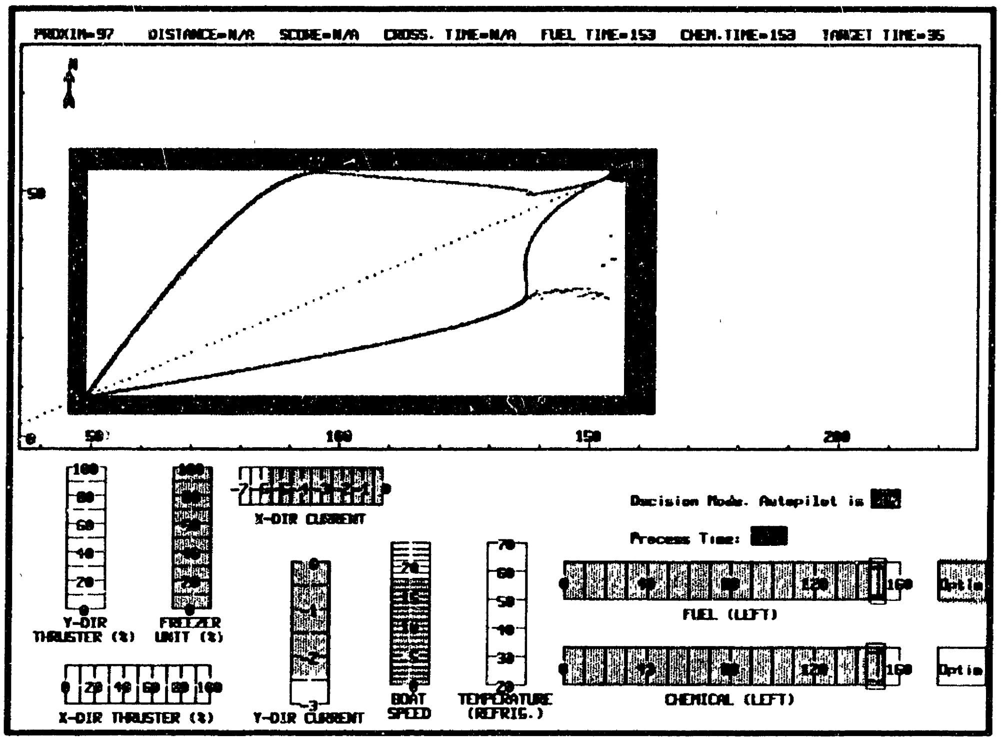


Figure 6.10. Example of attainability set #4: boat moving *with* the current.

Chapter 7

Conclusions

7.1 Summary

Traditionally a control problem is viewed as the problem of finding the ways to accomplish a given task. This thesis addresses a juxtaposed problem of *how* to select the task for a given control system.

The tasks considered in this thesis can be characterized as terminal goal tasks. It is presumed that the control system is able to generate controls for achieving the goal, *if the goal is achievable* in terms of the system's resources. Under the ideal circumstances it is enough to specify a known-to-be achievable goal for the system only once, and then rest assured that the system would accomplish the task. However if the system operates in an environment with perturbations, a goal that seemed to be achievable in the beginning of the task might become unattainable sometime later. When this happens another attainable goal has to be given to the system. Thus a system subjected to perturbations might need to be assigned a goal *more than once* in the course of the same task.

The goal assignment has to be done by a human operator. The operator must always choose an attainable goal. Frequently some other considerations in addition to the mere attainability of the goal guide the operator in her selection of the goal. The key factor is the operator's ability to explore the attainability set of the system.

The following list provides a summary of the major contributions of this thesis:

- This problem of assigning an attainable goal to a system was formulated as a decision task to be solved by the operator of the system. The task centers around the notion of the attainability set of a system evolving in the environment with perturbations.

- A decision-aiding approach to help the human decision maker select an attainable goal was proposed. This approach is based on having minimum information about the perturbations and is especially useful when the circumstances do not permit a comprehensive analysis of the decision situation.

- A decision-aiding technique, the Dynamic Range Tradeoff (DRT), was developed to assist the operator in the goals selection task. This technique provides the operator with a set of tools for exploration of the attainability set of the problem. The technique is based on a causal approximation of the attainability set and facilitates a quick scanning of the desired regions of the attainability set. The technique enables the decision-maker to direct the exploration of the boundary of the attainability set and provides the decision-maker with a global, as opposed to a point-by-point, view of the set. The Maximal satisficing sets and the Pareto cluster cells methods were developed as parts of this technique.

- A class of systems, called resource-sharing systems, whose performance can be improved by the use of the proposed technique, was suggested. The implementation of the proposed technique to a system of this kind was developed.

- An experimental study of the proposed approach was performed. The experiments were conducted on a relatively simple non-linear dynamical system. This sixth order system was directed to perform tasks that required it to operate at the limit of its capabilities. The experimental situation had a distinct feature: though the measure of performance in the task was clearly defined, the decision task had a multi-criteria nature. Performance of human subjects using the proposed approach was compared to their performance using conventional manual and automatic controls.

- The experiments demonstrated that, using a computer aid based on the technique developed in this thesis, the subject achieved significantly better performance than when using traditional methods. The experiments showed that the lack of knowledge by the decision-maker of the boundaries of the attainability set was the major handicap of conventional automatic control. The experiments demonstrated the effectiveness of two key elements of the DRT technique: 1) approximation of the future perturbations with a deterministic function of time; and 2) approximation of the attainability set with simple rectangular shapes.

- The proposed technique was suggested as a useful tool for solving traditional multiple criteria decision making problems, static as well as dynamic. As such a tool this technique provides yet another method of solving these problems and it can be used in conjunction with any other existing method. The technique was implemented in a software system for solving generic discrete multiple criteria decision making problems. The software enables a decision maker to scan a multi-dimensional data-base of discrete data. This software was used as a part of a computer-based decision-aid for tactical mission planning.

7.2 Possible Applications

The increased automation of the workplace more and more often places the human into a position of a supervisory controller[58][59]. The human plans the tasks, directs them, monitors and intervenes when needed. She is no longer involved in the nitty-gritty of the actual execution of the tasks.

It is natural to foresee the situation when the human operator would tell the system *what* she wants to have done and the system would figure out itself *how* to do it. In giving such an assignment to a system, the human should clearly understand the limitations of the system, i.e. to be *realistic* in assigning the task and expecting it done.

The DRT technique can be added to the supervisory control toolbox. It can be used in any task where performance is characterized by multiple interdependent criteria and the issue of finding goal points from the attainable set of criteria is relevant.

A partial list of applications that might benefit from the use of the DRT technique includes:

Remote Vehicle Control - Control of a remote vehicle frequently involves long transmission delays and requires operating the system in an environment with perturbations. Vehicles of this type include unmanned space probes as well as underwater submarines. A goal-control of these vehicles seems to be a natural continuation of the present trend towards their automation. It can be easily foreseen that the issue of a human operator defining and redefining the missions for these vehicles becomes critically important for their successful operation.

Manipulator Control - The simplest form of a goal-control of a manipulator is a (world coordinates) position control of the manipulator's tip. To specify an appropriate tip position it is enough to know the workspace of the manipulator. The attainability set as a 3D position workspace can be relatively easily handled by a human operator. However if the task involves specifying the kinematic, dynamic, energy and time restrictions on the final position of the tip, the dimension of the attainability set grows dramatically. The DRT technique could help an operator to explore this set and select appropriate specifications for the task.

Constraint Negotiations in CAD/CAM - The DRT technique can be useful in deciding which *soft* constraint to relax if during the process of computer-aided design the system becomes overconstrained. The DRT technique can be directly incorporated into the constraint management system similar to the one developed in[57].

Project Management - Managing a project usually involves giving tasks to groups of people and coordinating the performance of these tasks. It is critically important that the tasks are well coordinated and at the same time accomplishable. The DRT approach can help the manager to set realistic goals and change them in the future, if unexpected snags develop.

Military Systems - Military missions are usually planned to be performed at the limit of capabilities of the equipment and personnel. Often a military mission is specified by a set of terminal conditions. Unexpected snags might necessitate quick replanning or even calling off the operation. To make an informed decision the planner must be able to explore the set of attainable missions.

Autopilots - Autopilots are designed to withstand and reject some perturbations. For instance, the aircraft autopilot has a range of winds it can successfully handle. An unexpected burst of wind may however make the whole task unattainable. The pilot should be able to detect it happening and respond to that.

Air traffic control - Controlling many aircraft at the same time means specifying goals to several, potentially interfering with each other, subsystems. Knowing which goals can be safely attained requires knowing the attainability set of the entire system that involves all aircraft.

Resource Management - The management of a limited resource was discussed in Chapter 5. It seems that the performance of systems of this type can benefit from the DRT application.

7.3 Suggestions for Future Research

In addition to implementing the DRT control technique to practical applications, the following issues directly related to the DRT need more research:

Modelling Perturbations - More sophisticated probabilistic models of the perturbations could be used in the assessment of the attainability set. If the distribution of the perturbations is known, this distribution induces a probability distribution over the attainability set. The Bayesian approach[42] potentially can be used for assessing parameters of this distribution.

Discrepancies Between Human and Computer Models - The human operator makes her assessment about the attainability set of the system based on her own model of perturbations. The algorithm driving the automatic control system which controls the actuators might be based on a different model of the perturbations. Special care should be put into aligning these models and having one under control of another. In the experiments described in this thesis, the human operator was determining the parameters defining the perturbation for the automatic controller. However some other schemes of "aligning" the control system's and the operator's models of perturbations can be investigated. For example the control system could select the model itself and feed it to the attainability evaluator used by the operator.

Parallel Processing Algorithms - Since the DRT technique requires computation of the passive bounds of several target functions independently, the computations could be done using parallel computer hardware. Parallel algorithms could be developed for the computation of the Maximal satisficing sets and Pareto cluster cells.

Cluster Cells for Non-monotonic Preferences - The developed algorithm for computation of the Pareto cluster cells requires that the values of the target functions are *monotonically preferential,* i.e. the lower (higher) values of a target function are preferred to the higher (lower) values of the same function if all other target functions remain the same. The cluster cells however can be defined for any group of non-overlapping intervals which jointly form the real axis, as for example, in the case of a unimodal utility. It seems useful to develop a Pareto cluster cells algorithm for this case.

Group Decision-Making - The use of the DRT technique for aiding group decision-making poses an interesting problem. The maximal satisficing sets and Pareto cluster cells techniques could be useful as a group constraints negotiation tool.

Appendix A

Proof of Lemma 2

Proof:

1. First we prove the Lemma for M=2. The Lemma is proved assuming that the contrary is true. Let's assume that there exists a convex set Ψ intersecting with all four sides of rectangle ABCD and not containing the center of the rectangle, point O (Fig. A.1). Since point O is located outside of Ψ, and Ψ is a convex set, there exists a straight line, MN, passing through O, not intersecting Ψ, and separating the entire plane into two half-planes, such that one of the half-planes contains Ψ in its entirety (this follows from a separation theorem for convex sets[43]).

However it can be seen from the picture on Fig. A.1 that any such line cuts the rectangle into two pieces so that there are always two sides of the rectangle that belong to different half-planes. For example, on Fig. A.1 sides AC and BD belong to different half-planes. Since Ψ is located only in one of the half-planes it cannot intersect with both AC and BD at the same time. For instance, if Ψ is located in the same half-plane as AC, then Ψ *cannot* intersect with BD, and vice versa. This contradicts the assumption that Ψ does intersect with all four sides of the rectangle.

There is another limiting case that must be considered. That is if the separating line passes through two vertexes of the rectangle, as line AD for

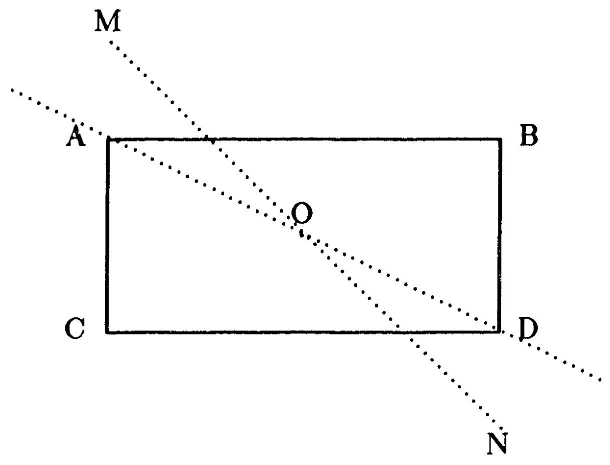


Figure A.1. Illustration to the proof of Lemma 2.

Line MN cuts rectangle ABCD into 2 pieces

Sides AC and BD belong to different pieces.

example. In this case it is clear that Ψ must contain both vertexes A and D, since otherwise it would not intersect with all four sides. But from convexity of Ψ it follows that Ψ must also contain point O which is a point inside of segment AD. This contradiction proves the Lemma for the case M=2.

The proof for M > 2 follows the same idea as the proof for the case when M = 2. This proof is done analytically.

We note that the passive bound box defined by (A.1)

$$L_i^P \leq y_i \leq R_i^P \quad (i = \overline{1,M}) \tag{A.1}$$

can be transformed using the affine transformation

$$x_i' = \frac{y_i - L_i^P}{R_i^P - L_i^P}$$

into a hypercube with one vertex at the origin, $(0,..,0)^T$.

So without a loss of generality it can be assumed that Lemma 2 can be stated in the following form:

If a convex set, Ψ, intersects with all sides of the unit hypercube

$$0 \leq x_i \leq 1 \quad (i = \overline{1,M}) \tag{A.2}$$

then this set has a non-empty intersection with hypercube Π:

$$\begin{gathered} \frac{1}{M} \leq f_i \leq 1 - \frac{1}{M} \quad (i = \overline{1,M}) \\ (f_1, \ldots, f_M)^T \in \Pi \end{gathered} \tag{A.3}$$

The fact that Ψ intersects with all sides of the hypercube means[1] that

$$\begin{gathered} \text{for any } k \text{ there exist} \\ l_1,\ldots,l_{k-1},l_{k+1},\ldots,l_M \text{ and } r_1,\ldots,r_{k-1},r_{k+1},\ldots,r_M \\ \text{such that } 0 \leq l_j \leq 1 \;;\; 0 \leq r_j \leq 1 \text{ for all } j = \overline{1,M} \text{ and } j \neq k \\ \text{so that } (l_1,\ldots,l_{k-1},0,l_{k+1},\ldots,l_M)^T \in \Psi \\ \text{and } (r_1,\ldots,r_{k-1},1,r_{k+1},\ldots,r_M)^T \in \Psi \end{gathered} \tag{A.4}$$

1. Conditions (A.4) follow directly from the definition of the passive bounds (3.1).

To prove the Lemma we assume the contrary, i.e. that $\Pi \cap \Psi = \emptyset$.

Since both Π and Ψ are convex, there exists a hyperplane $\mathbf{a}^T * \mathbf{x} = \mathbf{b}$ that separates[2] them [43][52]:

$$\begin{array}{ll} \mathbf{a}^T * \mathbf{f} \geq \mathbf{b} & \text{for } \mathbf{f} \in \Pi \\ \mathbf{a}^T * \mathbf{y} < \mathbf{b} & \text{for } \mathbf{y} \in \Psi \end{array} \tag{A.5}$$

Without a loss of generality we can assume that a_1 has the greatest absolute value among all a_i:

$$|a_1| \geq |a_j| \quad \text{for all } j = \overline{2,M} \tag{A.6}$$

Let's denote I^+ a set of all indexes $\alpha \in \overline{2,M}$ such that $a_\alpha \geq 0$, and I^- - a set of all indexes $\beta \in \overline{2,M}$ for which $a_\beta < 0$.

Let's consider a case when $a_1 > 0$ and $b \geq 0$.[3] Let's take the following point $\xi \in \Pi$:

$$\xi_i = \begin{cases} \frac{1}{M} & \text{if } a_i \geq 0 \\ 1 - \frac{1}{M} & \text{if } a_i < 0 \end{cases}$$

From (A.5) it follows that

2. Here $\mathbf{x} * \mathbf{y}$ is a regular Euclidian scalar product: $\mathbf{x}^T * \mathbf{y} = \sum_i^M =1 x_i y_i$.

3. It is obvious that $a_1 \neq 0$.

$$a_1 + \sum_{\alpha \in I^+} a_\alpha + (M-1) \sum_{\beta \in I^-} a_\beta \geq M b \tag{A.7}$$

Consider a side of the original hypercube which is described by equation $x_1 = 1$. For *any* point $\mathbf{y} = (1, y_2, \ldots, y_M)^T$ from that side the following estimate holds:

$$\begin{aligned} \mathbf{a}^T * \mathbf{y} &= a_1 + \sum_{\alpha \in I^+} a_\alpha y_\alpha + \sum_{\beta \in I^-} a_\beta y_\beta \\ &\geq a_1 + \sum_{\beta \in I^-} a_\beta \end{aligned} \tag{A.8}$$

since $0 \leq y_\alpha \leq 1$ and $0 \leq y_\beta \leq 1$.

It follows from (A.7) that

$$a_1 + \sum_{\beta \in I^-} a_\beta \geq b \tag{A.9}$$

Estimate (A.9) can be obtained from (A.7) by considering two distinct cases: 1) $|I^-| = 0$ and $|I^-| > 0$.[4] In the first case (A.7) becomes

$$a_1 + \sum_{\alpha \in I^+} a_\alpha \geq M b \tag{A.10}$$

and since in this case $|I^+| = M-1$, we get from (A.10) and (A.6) that $a_1 \geq b$. The last expression is equivalent to (A.9) for this case.

4. $|I^-|$ denotes the number of elements in I^-.

In the second case we have $|I^+| \leq M-2$ and from this, (A.7) and (A.6) we get

$$(M-1)a_1 + (M-1)\sum_{\beta \in I^-} a_\beta \geq M b \tag{A.11}$$

and from (A.11) we get (A.9).

Estimates (A.9) together with (A.8) and conditions (A.5) show that any point belonging to the side $x_1 = 1$ of the unit hypercube also belongs to the same hyper-halfspace as Π, i.e. this side is separated from Ψ by the hyperplane (A.5). The last fact contradicts the assertion that Ψ intersects with all sides of the hypercube, i.e. to (A.4). This contradiction proves the Lemma in the case $a_1 > 0$, $b \geq 0$. The proofs for all other cases (($a_1 > 0$ and $b < 0$), ($a_1 < 0$ and $b \geq 0$), ($a_1 < 0$ and $b < 0$)) are almost identical to this one.

Note: The statement of the Lemma cannot be made any stronger. There are cases when Ψ and Π have only a single point in common. For example if Ψ is described by (A.12)

$$\sum_{i=1}^{M} x_i \leq 1 \tag{A.12}$$

then this set has only point, $(\frac{1}{M}, \frac{1}{M}, \ldots, \frac{1}{M})^T$, in common with Π.

Appendix B

Maximal Number of the Maximal Satisficing Sets

Theorem: the maximum number of maximal satisficing sets is equal to $\binom{M}{K}$, where $K = [\frac{M}{2}]$.

Proof of the Theorem: We are going to establish the maximum possible number of the maximal satisficing sets of a set of M pairs of inequalities (3.18). We will freely interchange statements about *subsets* and their *indicator numbers,* always understanding that we refer to ***subsets of M pairs of inequalities*** (3.18). There are a total of 2^M subsets of the set of these M pairs of inequalities. Any of these subsets can potentially be an MSS, however if it is determined that a particular subset indeed is an MSS, then this fact rules out many other subsets as possible MSSs.

Consider an example of M=4 and subset 0101 being an MSS. In our terminology this means that the maximal feasible set of constraints contains two pairs of constraints (i.e. constraints (3.19) now become (B.13)):

$$\begin{cases} L_2^A \leq G_2(\chi) \leq R_2^A \\ L_4^A \leq G_4(\chi) \leq R_4^A \\ \text{and} \quad \chi \in \Omega_\chi \end{cases} \tag{B.13}$$

with $k = 2$ and $i_1 = 2$ and $i_2 = 4$.

From (B.13) being a maximal feasible set of constraints, we can immediately conclude that subsets 1101, 0111 and 1111 are not feasible; and subsets 0001, 0100 and 0000 are not maximal. So a single MSS, 0101, takes a chance from several other subsets (in this case 6 other subsets) to be an MSS. We will use this idea, that one MSS generates a "chain" of non-MSSs, in establishing the maximal number of MSSs.

We begin by sorting all subsets of (3.18) into M+1 groups[5], numbered from 0 to M. The groups are organized in such a way that all subsets belonging to the k-th group have the same cardinality k. We can picture this arrangement of the subsets into groups by drawing all subsets from one group above the subsets of another group, placing the groups in the order of increased cardinality. All members of one group are placed at the same level on the picture. An example of the picture for M=4 is shown of Fig. B.2.

A natural partial ordering can be established on the set of all considered subsets. This ordering is induced by the *inclusion* ($\subseteq$) operation.[69] We can connect the subsets on the picture to reflect this ordering, thus turning the picture into an ordered graph. Each subset becomes a node of the graph. Any two subsets on the neighboring levels are connected if one of them includes the other. We denote this graph Γ^M. Fig. B.3 shows an example of the graph for the case when M=4.

5. The reference to groups has no group-theoretical bearing.

Group (level) 0:			0000			
Group (level) 1:		0001	0010	0100	1000	
Group (level) 2:	0011	0101	1001	0110	1010	1100
Group (level) 3:		0111	1101	1011	1110	
Group (level) 4:			1111			

Figure B.2. Arrangement of the subsets into the groups of equal cardinality.

We now partition the graph into a union of non-intersecting subgraphs[18] so that this partitioning satisfies the following properties:

1. each subgraphs contains no more than one element from the same level;

2. each subgraph contains an element from the $[\frac{M}{2}]$-th level.

where [n] denotes the maximum integer not exceeding n.

It is clear that each subgraph in such a partitioning can be viewed as a "string" running from some level downwards. An example of such a partitioning for the case M=4 is shown on Fig. B.4.

Assume for a moment that such a partitioning is possible for any M. Then we can observe that if some subset, J', is an MSS, then all subsets that are the members of the same partition (or the same "string") as J' can be ruled out as potential MSSs. This is due to the fact that all subsets in the partition that are located above J' are non-maximal, and all subsets located below J' are infeasible. For instance, if $J' = 0110$ on Fig. B.4, then members of the same partition, i.e. 0010 and 1110 are definitely not MSSs. In addition to that, there are some

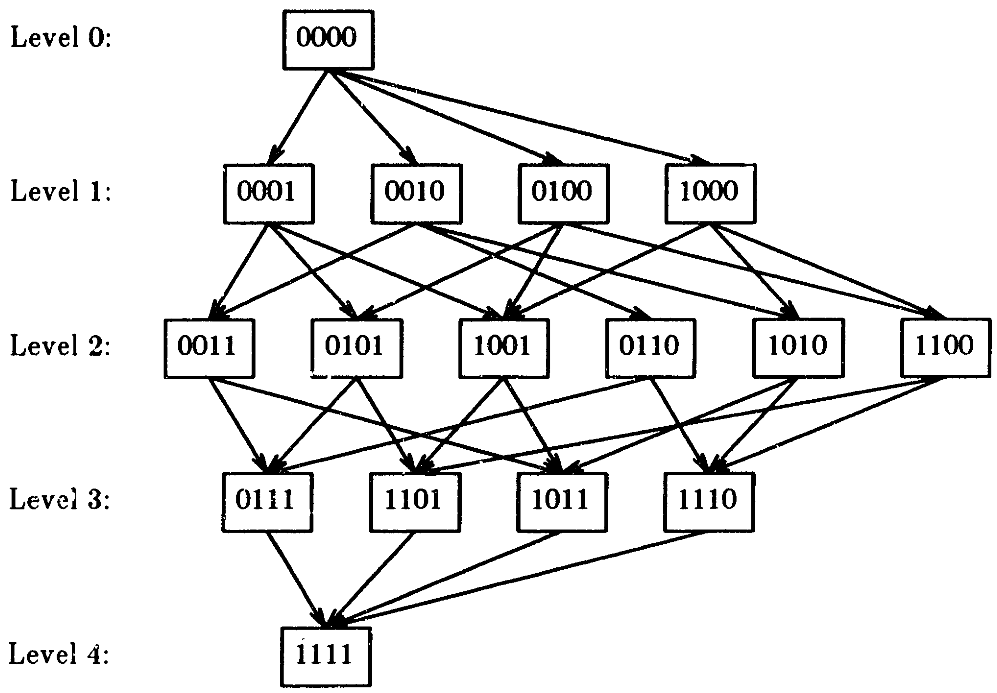


Figure B.3. Partial ordering of the subsets sorted according to their cardinality. Graph Γ^M for M=4.

members of other partitions that are ruled out as potential MSSs. (The "chain" reaction of an MSS generating a string of non-MSSs was explained in the beginning of this Appendix.)

From this we conclude that the number of MSSs cannot be greater than the number of partitions of the graph Γ^M. On the other hand, the total number of the partitions is equal to the number of elements in the group of subsets with cardinality $[\frac{M}{2}]$, since it was required that each partition contained an element from that group. From this we can conclude that the maximum number of MSSs cannot exceed the number of elements in the latter group. Since any group of cardinality K contains exactly $\binom{M}{K}$ members[6], then it follows that the maximum

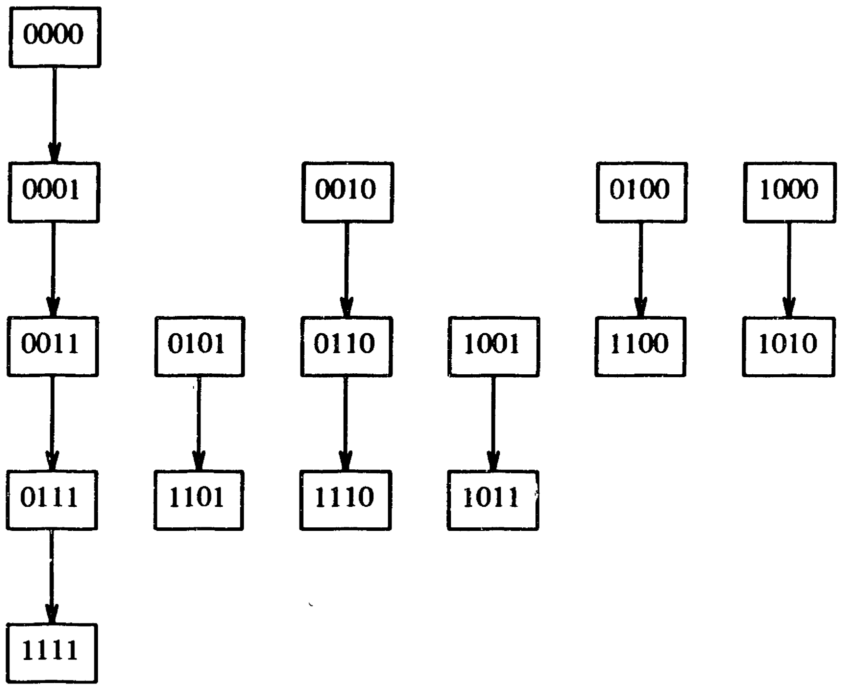


Figure B.4. Partitioning of the graph into subgraphs.

number of MSSs cannot exceed $\binom{M}{[M/2]}$. On the other hand it is possible that *all* subsets belonging to the same group are the MSSs at the same time. Subsequently all subsets of cardinality $[\frac{M}{2}]$ can be the MSSs at the same time. Therefore we can conclude that the maximum possible number of MSSs is exactly equal to the number of substes in the latter group, i.e. to $\binom{M}{[M/2]}$.

6. $\binom{M}{K}$ is a binomial coefficient, $\binom{M}{K} = \frac{m!}{k!(m-k)!}$.

It still remains to be proven that the described partitioning of the graph of all subsets Γ^M is possible for any M. To prove that we first prove the following

Lemma 1: For any M and K, such that $K < \frac{M}{2}$ there exists a complete matching of the members of the K-th group (containing all subsets of cardinality K) into the K+1th group (containing all subsets of cardinality K+1).[7]

Proof of Lemma 1: The Lemma directly follows from the so-called "marriage theorem"[40]. The theorem states that there exists a complete matching of one subset of the nodes of a graph into another subset of the nodes of the graph if every node of the first subset is connected to L or more nodes of the second subset and every node of the second subset is connected to L or less nodes of the first subset.

In our case each member of the group of cardinality K is connected exactly to $M-K$ members of the group of cardinality $K+1$ (the number of vacant 0's in the K-th group nodes). At the same time each element of the group of cardinality $K+1$ is connected to exactly $K+1$ elements of the group of cardinality K (see Fig. B.3 for an illustration).

7. A subgraph of an oriented graph is called a ***complete matching*** of some subset S_1 of the nodes of the graph into another subset, S_2, of the nodes of the same graph, if for any element $s_1 \in S_1$ there is a single edge connecting s_1 to some element $s_2 \in S_2$ and no two elements of S_1 are connected to the same element in S_2 (e.g. see [13]).

Therefore, according to the *marriage theorem,* if

$$M - K \geq K + 1 \tag{B.15}$$

then a complete matching exists. (A.7) is equivalent to $K < \frac{M}{2}$. Q.E.D.

An almost identical considerations can be used to prove a similar result, Lemma 2:

Lemma 2: For any M and K, such that $K > \frac{M}{2}$ there exists a complete inverse matching of the members of the K-th group (containing all subsets of cardinality K) into the K-1th group (containing all subsets of cardinality K-1).

To prove the theorem we construct the partitioning of graph Γ^M into subgraphs in the following way. First we establish the complete matchings from level 0 to level 1, from level 1 to level 2, etc. If M is even, then the last matching would be from level $\frac{M}{2} - 1$ to level $\frac{M}{2}$. If M is odd, then the last matching would be from level $\frac{M-1}{2}$ to level $\frac{M+1}{2}$. Let's call the last level to which the matchings extend (this level corresponds to the group of subsets that has the maximum cardinality out of all groups), the *equator* level.[8] Then we establish complete inverse matchings from all groups from the level M up to the equator level (i.e. to the $\frac{M}{2}$th level if M is even and to the $\frac{M+1}{2}$th level if M is odd).

8. The term *equator* is used to reflect that this is the middle level. For instance, on Fig. B.2 this level is $\frac{M}{2} = 2$.

Next we create the partitions by "threading" elements of the ₒ.aph according to just established matchings starting from the first available element on the levels above the equator level and going down from that element until we get to the equator level. We start with {00...0} on the 0-th level and proceed down to the equator level, collecting all elements in the same partition as {00...0}. Then we pick the first available uncollected element from the highest level and collect members of its partition. We continue this procedure until there are no uncollected elements in the levels above and including the equator. As the result all elements on the levels above and including the equator level will be collected into as many partitions as there are elements in the group corresponding to the equator level.

Then we "connect" to each of the established threads a similar thread coming up from the levels below the equator. The latter threads are created in the same manner, starting from the very bottom level and proceeding along the inverse matchings. After a thread is connected to the equator level, a new thread is started from the first uncollected element on the lowest available level.

As the result the entire graph will be partitioned into non-overlapping "threads" (or subgraphs) so that each "thread" runs through a unique element on the equator level.

This concludes the proof of the Theorem.

Appendix C

Maximal Number of the Pareto Cluster Cells

Theorem: the maximum number of the Pareto cluster cells is equal to the number of the cells with cardinality[9]

$$[\frac{1}{2}\sum_{i=1}^{M} N_i]$$

Proof: A proof of this theorem is similar to the proof of the formula for the maximum number of the maximal satisficing sets (Appendix B). Below is a sketch of the proof.

The proof consists of the following steps: First, all cell numbers are sorted into groups of numbers of equal cardinality. There are a total of $1 + \sum_{j=1}^{M} N_j$ such groups. Second, oriented graph Γ^M is constructed so that: 1) the nodes of the graph correspond to all cell numbers; 2) the nodes are located at different levels: nodes located at the same level have the same cardinality; 3) the cardinality of the levels increases from the bottom level, where it is 0, up to the top level where it is $\sum_{j=1}^{M} N_j$; 3) any node, $J^{(k+1)}$ of the (k+1)-th level is connected to all nodes of

9. Cardinality of a cell whose cell number is (3.40), is a sum of all of its number's components: $|J| = \sum_{j=1}^{M} i_j$.

the k-th level whose cell number differ from the cell number of $J^{(k+1)}$ in only one component.

It is easy to see that if some node of this graph is a Pareto cluster cell, then all nodes connected to it (through the graph) are definitely not Pareto cluster cells.

The third and the last step of the proof is partitioning graph Γ^M into "threads" identical to the ones described in Appendix B. This time however the existence of such a partitioning follows from the results of Hsieh and Kleitman[23] for products of partial orders.

After the graph has been partitioned, it is clear that the maximum number of the Pareto cluster cells is equal to the number of cells at the largest level of the graph, i.e. at the $[\frac{1}{2}\sum_{i=1}^{M} N_i]$-th level. This concludes the proof of the Theorem.

Appendix D

The Boat Example: Computation of the DRT bounds

Below are the details to the DRT techniques applied to the Boat example. A general approach of dealing with resource sharing systems outlined in Chapter 5 is followed here. A virtue of the boat example is that it was possible to complete almost all calculations in the closed form and the optimal control problems arising in the DRT could be reduced to the variational ones.

In the experiments the DRT bounds were computed in real time with a variable sampling time step. The sampling time was determined by the operating system upon entering the computational loop. The computations of the DRT bounds were multiplexed with the simulation of the system dynamics and the human-computer interface. On the average, the DRT computations took 3-4 cycles of the system simulation updates. On the average, the frequency of the DRT bound computations was 0.8-1.2 Hz.

Below is the derivation of how to compute the DRT bounds from some instance of time t_0.

Suppose that state of the system at t_0 is $x(t_0)$, $y(t_0)$, $F(t_0)$. The active bounds are determined by the DM- specified goal $(x_g^-, x_g^+, y_g^-, y_g^+, M_g, F_g)^1$:

$$
\begin{aligned}
L_1^A &= x_g^-, & R_1^A &= x_g^+ \\
L_2^A &= y_g^-, & R_2^A &= y_g^+ \\
L_3^A &= M_g, & R_3^A &= M(t_0) \\
L_4^A &= F_g, & R_4^A &= F(t_0)
\end{aligned}
\qquad (61.1)
$$

The *passive bounds* approximating the attainability set are to be found as the solutions of the following optimization problems:

$$
\begin{aligned}
L_1^P &= \min_{t^*, f_x, f_y, f_r} x(t^*), & R_1^P &= \max_{t^*, f_x, f_y, f_r} x(t^*) \\
L_2^P &= \min_{t^*, f_x, f_y, f_r} y(t^*), & R_2^P &= \max_{t^*, f_x, f_y, f_r} y(t^*) \\
L_3^P &= \min_{t^*, f_x, f_y, f_r} M(t^*), & R_3^P &= \max_{t^*, f_x, f_y, f_r} M(t^*) \\
L_4^P &= \min_{t^*, f_x, f_y, f_r} F(t^*), & R_4^P &= \max_{t^*, f_x, f_y, f_r} F(t^*)
\end{aligned}
\qquad (61.2)
$$

where the optima (61.2) are searched for subject to the system dynamics equations (6.1) and (6.4) and constraints (61.3) imposed by the active bounds of the range:

$$
\begin{cases}
x_g^- \le x(t^*) \le x_g^+ \\
y_g^- \le y(t^*) \le y_g^+ \\
M_g \le M(t^*) \le M(t_0) \\
F_g \le F(t^*) \le F(t_0)
\end{cases}
\qquad (61.3)
$$

1. Effectively in this example we have a four-dimensional 2211 range, since only the lower bounds on the chemical and the fuel are of concern.

The domain of the unknowns in the optimization problems (61.2) is determined by the limitation on the maximum flows into the thrusters and the freezer unit (61.4):

$$\begin{cases} 0 \leq f_x(t) \leq U_x \\ 0 \leq f_y(t) \leq U_y \\ 0 \leq f_r(t) \leq U_r \\ t^* \geq t_0 \end{cases} \tag{61.4}$$

The boat+refrigerator might seem to be a system with two resources, however it can be viewed as an aggregation of *three* subsystems sharing one resource - the fuel. The three subsystems sharing the fuel are: the X- dynamics, the Y-dynamics, and the refrigerator dynamics.

The amount of the resource spent from some time t_1 until t_2 by all three subsystems is

$$F(t_1,t_2) = \int_{t_1}^{t_2} (f_x(t) + f_y(t) + f_r(t))dt$$

To find the optima (61.2) terminal time t^* is initially fixed as a constant and the optimization problems are solved individually for all three subsystems comprising the boat+refrigerator system. Below are the solutions of individual subsystem's optimization problems *for a fixed* value of t^*. To simplify the matters, an index identifying a particular subsystem is omitted in the derivation, since the solutions for all three subsystems are similar. The aggregation of the individual subsystem's solutions is done exactly as described in Chapter 5.

The only difference between this example and the general case treated in Chapter 5 is that in the boat example the resource variable (the fuel) is *the amount of resource remaining* in the system, while the general case deals with *the amount of resource spent.* Due to this difference the resource goal in the boat example corresponds to the *right* bound of the active range, while the in the general case it is the *left* bound.

As it was noted in the definition of the attainability set (see Chapter 1) the perturbations were considered constant for the purpose of the DRT calculations. In the context of our example it means that $C_x(x,y) = C_x'$ and $C_y(x,y) = C_y'$. Under these assumptions the boat's dynamics in X- and Y- directions, as well as the refrigerator's dynamics could be written in *the same* form^2:

$$\begin{cases} \dot{x}_1(t) = \alpha x_2(t) + \beta \\ \dot{x}_2(t) = m x_2(t) + n f(t) + k \end{cases} \tag{61.5}$$

where f(t) is a control variable satisfying conditions (61.6) (arising from (61.4)) that define a set of all admissible control functions:

$$0 \leq f(t) \leq U. \tag{61.6}$$

U in (61.6) is the maximum possible value of the control, and k in (61.5) is a constant approximating the disturbance in the cases of X- and Y- dynamics and some constant in the case of the refrigerator dynamics.

Function $F'(t)$: $F'(t) = \int_{t_0}^{t} f(\tau)d\tau$ gives the total amount of the resource spent by a single subsystem (61.5) from time t_0 until t,

In the boat example the attainability set is a set in the 4D space of the variables x, y, F (fuel), and M (chemical) (61.2). Therefore the attainability of each of the subsystems (61.5) (which comprise the boat+refrigerator system) is defined in terms of x_1. As the result, the following three functions (61.5) are needed for the computations of the DRT bounds of the boat+refrigerator system, (see Chapter 5):

$$\begin{cases} G_1^{min}(L_F,R_F,t^*,t_0,x_1(t_0),x_2(t_0)) = \min\limits_{0 \le f(t) \le U} x_1(t^*) \\ \qquad \text{subject to: } L_F \le F'(t^*) \le R_F \end{cases} \tag{61.7}$$

$$\begin{cases} G_1^{max}(L_F,R_F,t^*,t_0,x_1(t_0),x_2(t_0)) = \max\limits_{0 \le f(t) \le U} x_1(t^*) \\ \qquad \text{subject to: } L_F \le F'(t^*) \le R_F \end{cases} \tag{61.8}$$

2. For the X- dynamics: x_1=x, $x_2=\dot{x}$, $f=f_x$, the Y- dynamics: x_1=y, $x_2=\dot{y}$, $f=f_y$, and for the freezer: x_1=M, x_2=T, $f=f_r$.

$$\begin{cases} G_F^{min}(L_1, R_1, t^*, t_0, x_1(t_0), x_2(t_0)) = \min\limits_{0 \le f(t) \le U} F'(t^*) \\ \qquad\qquad \text{subject to: } L_1 \le x_1(t^*) \le R_1 \end{cases} \tag{61.9}$$

The above optima must be computed for an arbitrary terminal time time t^* and for arbitrary initial conditions (61.10).

$$x_1(t_0), \quad x_2(t_0) \tag{61.10}$$

at time t_0[3]. Without a loss of generality (for this particular example) we can take $t_0 = 0$.

Equations (61.5) can be easily integrated for any time t provided some initial conditions $x_1(0)$ and $x_2(0)$ at time $t_0 = 0$:

$$\begin{cases} x_1(t) = A_1 + B_1 t + C_1 e^{mt} + D_1 \int\limits_0^t f(\tau)(e^{m(t-\tau)} - 1)d\tau \\ x_2(t) = A_2 + C_2 e^{mt} + D_2 \int\limits_0^t f(\tau) e^{m(t-\tau)} d\tau \end{cases} \tag{61.11}$$

where

$$A_1 = x_1(0) - \frac{\alpha}{m} x_2(0) - \frac{\alpha k}{m^2}, \quad B_1 = \beta - \frac{\alpha k}{m}, \tag{61.12}$$

3. The conditions under which the optimal control problems (61.7) - (61.9) are *consistent* will be established later.

$$C_1 = \frac{\alpha}{m}(x_2(0) + \frac{k}{m}),\ \ D_1 = \frac{\alpha n}{m};$$

$$A_2 = -\frac{k}{m},\ \ C_2 = x_2(0) + \frac{k}{m},\ \ D_2 = n$$

First we note that $L_F = 0$ since it is a minimum expenditure on the resource (see Chapter 5). An optimum in (61.7) - (61.9) could be obtained as a solution of the corresponding optimum control problem. However it follows from (61.11), that these optima can be found as the solutions of the following *variational* problems (61.13) - (61.15):

$$\begin{cases} G_1^{\min} = \min\limits_{0 \le f(t) \le U} \{A_1 + B_1 t^* + C_1 e^{mt^*} + D_1 \int\limits_0^{t^*} f(\tau)(e^{m(t^*-\tau)} - 1)d\tau\} \\ \text{subject to: } 0 \le \int\limits_0^{t^*} f(\tau)d\tau \le R_F \end{cases} \tag{61.13}$$

$$\begin{cases} G_1^{\max} = \max\limits_{0 \le f(t) \le U} \{A_1 + B_1 t^* + C_1 e^{mt^*} + D_1 \int\limits_0^{t^*} f(\tau)(e^{m(t^*-\tau)} - 1)d\tau\} \\ \text{subject to: } 0 \le \int\limits_0^{t^*} f(\tau)d\tau \le R_F \end{cases} \tag{61.14}$$

$$\begin{cases} G_F^{\min} = \min\limits_{0 \le f(t) \le U} \int\limits_0^{t^*} f(\tau)d\tau \\ \text{subject to: } L_1 \le A_1 + B_1 t^* + C_1 e^{mt^*} + D_1 \int\limits_0^{t^*} f(\tau)(e^{m(t^*-\tau)} - 1)d\tau \le R_1 \end{cases} \tag{61.15}$$

One of the above problems, namely (61.13), has a straightforward solution:

$$G_1^{min} = A_1 + B_1 t^* + C_1 e^{mt^*} \qquad (61.16)$$

and the control function f_{opt} delivering this minimum is:

$$f_{opt}(\tau) \equiv 0 \text{ for all } \tau\text{: } 0 \leq \tau \leq t^*$$

This solution follows from the fact that for all subsystems described by equations (6.1) and (6.4), both coefficients m and D_1 in the corresponding solutions (61.11) are negative. Indeed, for the boat dynamics equations (6.1): $m = -\frac{k_b}{m_b} < 0$ and $D_1 = -\frac{1 \times k_b}{m_b} < 0$. For the freezer dynamics (6.4): $m = -\frac{h}{C_v}$ and $D_1 = -\frac{\phi\lambda}{h} < 0$. Solution (61.16) follows from $D_1 < 0$, $f(\tau) \geq 0$, and from the fact that the exponential expression in the integral (61.13), $e^{m(t^*-\tau)} - 1 < 0$ for all τ such that $0 \leq \tau \leq t^*$.

Solution (61.16) was to be expected. It simply means that the boat would drift the farthest distance to the left by the time t^* when no fuel is used to propel it in the X- direction. Similarly, the most amount of the chemical would be absorbed (and the least amount would be left) by the time t^*, if no fuel is used in the freezer from the start at time 0 up until the end at time t^*.

Next we are going to solve (61.8), or equivalently, (61.14). It follows from (61.14) that determining G_1^{max} is equivalent to solving the following variational problem:

$$\begin{cases} \Gamma_1^{\max} = \max\limits_{0 \le f(t) \le U} D_1 \int\limits_0^{t^*} f(\tau)(e^{m(t^*-\tau)} - 1)d\tau \\ \text{subject to: } 0 \le \int\limits_0^{t^*} f(\tau)d\tau \le R_F \end{cases} \tag{61.17}$$

The solution of (61.8) can be then obtained from (61.18):

$$G_1^{\max} = \Gamma_1^{\max} + A_1 + B_1 t^* + C_1 e^{mt^*} \tag{61.18}$$

To solve (61.17) let's introduce a slack variable w:

$$w = \int\limits_0^{t^*} f(\tau)d\tau \tag{61.19}$$

From the conditions of the maximization problem (61.17) we have:

$$0 \le w \le R_F \tag{61.20}$$

We are going to use a type of a Lagrange multiplier technique (see e.g.[8][5][43]). Let's introduce a Lagrange multiplier λ so that the variational problem (61.17) could be re-written as (61.21):

$$\Gamma_1^{\max} = \max\limits_{0 \le f(t) \le U} \{D_1 \int\limits_0^{t^*} f(\tau)(e^{m(t^*-\tau)} - 1)d\tau + \lambda(w - \int\limits_0^{t^*} f(\tau)d\tau)\} \tag{61.21}$$

subject to (61.19) and (61.20).

Next we use identities (61.22) and (61.23):

$$\int\limits_a^b f(x)g(x)dx = \int\limits_a^b f^+(x)g(x)dx + \int\limits_a^b f^-(x)g(x)dx \tag{61.22}$$

$$\text{where } f^+(x) = \begin{cases} 0, & \text{for x such that } f(x) \leq 0 \\ f(x), & \text{for x such that } f(x) \geq 0 \end{cases}$$

$$\text{and } f^-(x) = \begin{cases} f(x), & \text{for x such that } f(x) \leq 0 \\ 0, & \text{for x such that } f(x) \geq 0 \end{cases}$$

$$\max_{g(x):\ g_{\min} \leq g(x) \leq g_{\max}} \int_a^b f(x)g(x)dx = \int_a^b f^+(x)g_{\max}dx + \int_a^b f^-(x)g_{\min}dx \tag{61.23}$$

Applying (61.22) to (61.21) we get:

$$\Gamma_1^{\max} = \max_{f_{[0,t^*]}} \{\int_0^{t^*} f(\tau)[D_1(e^{m(t^*-\tau)} - 1) - \lambda]^+ d\tau \tag{61.24}$$

$$+ \int_0^{t^*} f(\tau)[D_1(e^{m(t^*-\tau)} - 1) - \lambda]^- d\tau$$

$$+ \lambda^+ w + \lambda^- w \}$$

and using (61.23) together with (61.6) we get from (61.24):

$$\Gamma_1^{\max} = \int_0^{t^*} U[D_1(e^{m(t^*-\tau)} - 1) - \lambda]^+ d\tau + \lambda^+ w + \lambda^- w \tag{61.25}$$

From (61.23) we get the expression for the optimum function $f_{opt}(\tau, \lambda)$ in (61.24) for any given value of λ:

$$f_{opt}(\tau, \lambda) = \begin{cases} 0, & \text{when } D_1(e^{m(t^*-\tau)} - 1) - \lambda < 0 \\ U, & \text{when } D_1(e^{m(t^*-\tau)} - 1) - \lambda \geq 0 \end{cases} \tag{61.26}$$

The estimate for the upper bound of $\Gamma_1^{\max}$ is obtained applying (61.20) to (61.25):

$$\Gamma_1^{\max} \leq \int_0^{t^*} U[D_1(e^{m(t^*-\tau)} - 1) + \lambda]^+ d\tau + \lambda^+ R_F \tag{61.27}$$

We note that the right-hand side of (61.27) depends only on λ and no longer is a function of $f(\tau)$ and w. We denote this function $\Psi(\lambda)$:

$$\Psi(\lambda) = \int_0^{t^*} U[D_1(e^{m(t^*-\tau)} - 1) + \lambda]^+ d\tau + \lambda^+ R_F \tag{61.28}$$

so that (61.27) could be re-written as:

$$\Gamma_1^{\max} \leq \Psi(\lambda) \tag{61.29}$$

Recalling that $\Gamma_1^{\max}$ is independent on λ (see the original problem (61.17)) we can minimize $\Psi(\lambda)$ and use the minimum of the dual function $\Psi(\lambda)$ as an upper bound of $\Gamma_1^{\max}$:

$$\Gamma_1^{\max} \leq \min_{\lambda \in R^1} \Psi(\lambda) \tag{61.30}$$

The original optimization problem is a *concave* one[4] and it can be shown [5][52] (and it also can be verified by a direct substitution of the final results λ_{opt} and f_{opt} in (61.31)) that a stronger statement (61.31) holds:

$$\Gamma_1^{\max} = \min_{\lambda \in R^1} \Psi(\lambda) \tag{61.31}$$

4. The concavity of the variational problem (61.21) follows from the linearity of constraints (61.19) - (61.20) and the *concavity* of function $D_1(e^{m(t^*-\tau)} - 1)$ in (61.17).

Function $\Psi(\lambda)$ can be integrated in the closed form. The resulting expression for $\Psi(\lambda)$ is:

$\Psi(\lambda) =$

for $\lambda \leq 0$: (the *first branch*)

$$U(-\frac{D_1}{m} + \frac{D_1}{m}e^{mt^*} - D_1 t^* - \lambda t^*) \tag{61.32}$$

for $0 \leq \lambda \leq D_1(e^{mt^*} - 1)$: (the *second branch*)

$$U(-\frac{D_1}{m} + \frac{D_1}{m}e^{mt^*} - D_1 t^* - \lambda(t^*+\frac{1}{m}) + \frac{D_1}{m}\ln(1+\frac{\lambda}{D_1}) + \frac{\lambda}{m}\ln(1+\frac{\lambda}{D_1})) + \lambda R_F \tag{61.33}$$

for $\lambda \geq D_1(e^{mt^*} - 1)$: (the *third branch*)

$$\lambda R_F \tag{61.34}$$

From (61.32) - (61.34), $\Psi(\lambda)$ can be minimized as a function of λ. Since $t^* \geq 0$, the first branch (61.32) reaches its minimum at $\lambda = 0$. Similarly, the third branch (61.34) reaches its minimum at $\lambda = D_1(e^{mt^*} - 1)$. Thus function $\Psi(\lambda)$ reaches its minimum somewhere on the interval $[0, D_1(e^{mt^*} - 1)]$, i.e. within the second branch (61.33):

$$\min_{\lambda \in R^1} \Psi(\lambda) = \min_{\lambda \in [0, D_1(e^{mt^*} - 1)]} \Psi(\lambda)$$

The derivative in respect to λ of the expression of $\Psi(\lambda)$ for the second branch (61.33) is:

$$\frac{d\Psi(\lambda)}{d\lambda} = R_F - Ut^* + \frac{U}{m}\ln(1+\frac{\lambda}{D_1}) \tag{61.35}$$

Since $\lambda \geq 0$ for the second branch and $D_1 < 0$ and $m < 0$, the logarithmic term in (61.35) is always greater than or equal to zero. Therefore if $R_F - Ut^* \geq 0$ then $\Psi(\lambda)$ reaches its minimum at $\lambda = 0$. On the other hand, if $R_F - Ut^* < 0$ then $\Psi(\lambda)$ reaches its minimum at the zero of $\frac{d\Psi}{d\lambda}$. It is easy to verify that the zero of $\frac{d\Psi}{d\lambda}$ belongs to the second branch (61.33). At the conclusion:

$$\lambda_{opt} = \arg \min_{\lambda \in \mathbf{R}^1} \Psi(\lambda) = \begin{cases} 0, & \text{if } R_F - Ut^* \geq 0 \\ D_1(e^{m(t^* - \frac{R_F}{U})} - 1), & \text{if } R_F - Ut^* < 0 \end{cases} \tag{61.36}$$

Substituting λ_{opt} from (61.36) into (61.26) we get the following solution (61.37)-(61.39) of the variational problem (61.21)[5]:

$\Gamma_1^{max} =$

if $R_F \geq Ut^*$:

5. Substitution of the results (λ_{opt} and f_{opt}) in (61.36) and (61.37)-(61.39) shows that the equality (61.31) actually holds. This is a direct way to verify that (61.37)-(61.39) is indeed the solution of (61.17).

$$U\left(-\frac{D_1}{m} + \frac{D_1}{m}e^{mt^*} - D_1 t^*\right) \qquad (61.37)$$

if $R_F < Ut^*$:

$$U\left(-\frac{D_1}{m} + \frac{D_1}{m}e^{mt^*} - D_1 t^*\right) + U\left(D_1 t^* - \frac{D_1}{m}e^{m(t^* - \frac{R_F}{U})} + \frac{D_1}{m}\right) - R_F D_1 \qquad (61.38)$$

and the optimal control function $f(\tau)$ is:

$$f_{opt}(\tau) = \begin{cases} U, & \text{for } 0 \leq \tau \leq t' \\ 0, & \text{for } t' = \tau \leq t^* \end{cases} \qquad (61.39)$$

where the switching time t' is determined from (61.40):

$$t' = \begin{cases} t^*, & \text{if } R_F > Ut^* \\ \frac{R_F}{U}, & \text{if } R_F \leq Ut^* \end{cases} \qquad (61.40)$$

The last result, (61.40), is intuitively obvious. It means that the optimum control regime is to spend the entire resource at the maximum rate for as long as the resource would last, beginning from time 0. If there is an ample amount of the resource, it should be spent at the same highest rate all the time beginning at 0 and ending at t^*. It is also clear that there is a solution of (61.17) for *any* combination of the initial conditions (61.10) and the final time t^*.

Finally, the expression for X_1^{max} is obtained from (61.18).

Next we solve variational problem (61.9), or equivalently, (61.15). Proceeding similarly to the solution of (61.8), we introduce a slack variable w:

$$w = A_1 + B_1 t^* + C_1 e^{mt^*} + D_1 \int_0^{t^*} f(\tau)(e^{m(t^*-\tau)} - 1)d\tau \tag{61.41}$$

which satisfies (61.42):

$$L_1 \leq w \leq R_1 \tag{61.42}$$

Introducing a Lagrange multiplier λ and denoting

$$E(t^*) = A_1 + B_1 t^* + C_1 e^{mt^*},$$

we can re-write problem (61.15) in the following equivalent form (61.43):

$$G_F^{\min} = \min_{0 \leq f(t) \leq U} \{\int_0^{t^*} f(\tau)d\tau - \lambda E(t^*) - \lambda D_1 \int_0^{t^*} f(\tau)(e^{m(t^*-\tau)} - 1)d\tau + \lambda w\} \tag{61.43}$$

subject to (61.41) and (61.42).

Using identities (61.22) and (61.44) (that is similar to (61.23))

$$\min_{g(x):\ g_{\min} \leq g(x) \leq g_{\max}} \int_a^b f(x)g(x)dx = \int_a^b f^+(x)g_{\min}dx + \int_a^b f^-(x)g_{\max}dx \tag{61.44}$$

and applying them to (61.43), we get (61.45):

$$G_F^{\min} \geq \Psi(\lambda) \tag{61.45}$$

where

$$\Psi(\lambda) = \int_0^{t^*} U[1 - \lambda D_1(e^{m(t^*-\tau)} - 1)]^- d\tau - \lambda E(t^*) + \lambda^+ L_1 + \lambda^- R_1 \tag{61.46}$$

and the optimum control $f_{opt}(\tau)$ for any given λ being:

$$f_{opt}(\tau,\lambda) = \begin{cases} 0, & \text{when } [1 - \lambda D_1(e^{m(t^*-\tau)} - 1)] \geq 0 \\ U, & \text{when } [1 - \lambda D_1(e^{m(t^*-\tau)} - 1)] < 0 \end{cases} \tag{61.47}$$

As it was argued in the solution of (61.14), G_F^{min} is independent of λ and therefore (61.48) follows from (61.45):

$$G_F^{min} \geq \max_{\lambda \in R^1} \Psi(\lambda) \tag{61.48}$$

From the convexity of problem (61.15) (since $D_1 < 0$), it follows that (61.48) can be re-written as an equality (61.49)[6]:

$$G_F^{min} = \max_{\lambda \in R^1} \Psi(\lambda) \tag{61.49}$$

Function $\Psi(\lambda)$ (61.46) can be calculated in the closed form:

$\Psi(\lambda) =$

for $\lambda \leq 0$: (the *first branch*)

$$\lambda(R_1 - E(t^*)) \tag{61.50}$$

for $0 \leq \lambda \leq \dfrac{1}{D_1(e^{mt^*} - 1)}$: (the *second branch*)

6. As before, in the case of (61.14), this fact can be verified by direct substitution of the solution $f_{opt}(\tau)$ of the direct problem (61.15) and the solution λ_{opt} of the dual problem $\max\Psi(\lambda)$ into (61.49).

$$\lambda(L_1 - E(t^*)) \tag{61.51}$$

for $\lambda \geq \frac{1}{D_1(e^{mt^*} - 1)}$: (the *third branch*)

$$\lambda\Theta + U(t^* + \frac{1}{m}) - \frac{U}{m}(1 + \frac{1}{\lambda D_1})\ln(1 + \frac{1}{\lambda D_1}) \tag{61.52}$$

where

$$\Theta = L_1 - E(t^*) + \frac{UD_1}{m}(1 + mt^* - e^{mt^*}) \tag{61.53}$$

It can be observed from the expression for first branch (61.50) that if $R_1 - E(t^*) < 0$ then $\max\limits_{\lambda \in R^1} \Psi(\lambda) = \infty$ and therefore (61.15) has no finite solution. Physically this corresponds to the case when the *upper* bound R_1 on $x_1(t^*)$ is set so *low* that the boat cannot get below it *even spending no resource* , i.e. drifting with the ocean current (it is clear from the expression for x_1 in (61.11)).

Thereby it must be requested for the existence of a finite solution of (61.15) that.

$$R_1 - E(t^*) \geq 0 \tag{61.54}$$

From here we can conclude that the maximum of $\Psi(\lambda)$ is achieved somewhere at $\lambda \geq 0$ (since the first branch (61.50) is a linear function with a positive slope).

If on the other hand the following condition (61.55) also holds:

$$L_1 - E(t^*) \leq 0 \tag{61.55}$$

then the maximum *of the second branch* (61.51) is attained at $\lambda = 0$. Since $\Psi(\lambda)$ is a *concave* function of λ, $\lambda = 0$ is is also a *global* maximum of $\Psi(\lambda)$ (because of the signs of the slopes of the first (61.50) and second (61.51) branches). This minimum translates into a trivial optimum solution (61.56) of (61.15):

$$f_{opt}(\tau) \equiv 0 \text{ for all } \tau\text{: } 0 \leq \tau \leq t^* \tag{61.56}$$

Physically (see (61.11)) this solution means the obvious: if L_1 and R_1 are such that $L_1 \leq E(t^*) \leq R_1$, then x_1 gets within these bounds without spending any resource. In this case $G_F^{min} = 0$.

The general case therefore is when (61.55) does not hold. So we consider the following case:

$$L_1 - E(t^*) > 0 \tag{61.57}$$

Since in this case the second branch (61.51) is a linear function with a positive slope, it is clear that the maximum of $\Psi(\lambda)$ is be reached somewhere within the third branch of (61.52). Without further mentioning we hereupon will refer to the third branch of (61.55) simply as $\Psi(\lambda)$.

Calculating $\frac{d\Psi(\lambda)}{d\lambda}$ from (61.50) - (61.52) we get:

$$\frac{d\Psi(\lambda)}{d\lambda} = \Theta - \frac{UD_1}{m}(\ln(1+\frac{1}{\lambda D_1}) - \frac{1}{\lambda D_1}) \tag{61.58}$$

Introducing a new variable μ and substituting

$$\mu = \frac{1}{\lambda D_1}$$

into (61.58) we get the condition for the maximum of $\Psi(\lambda)$ as:

$$\Theta = \frac{UD_1}{m}(\ln(1+\mu) - \mu) \tag{61.59}$$

This equation is to be solved in respect to μ (or λ) as the unknown. The range of possible roots is determined by the condition on λ for the third branch (61.52) and from $D_1 < 0$:

$$e^{mt^*} - 1 \leq \mu < 0 \tag{61.60}$$

The right-hand side of (61.59) is a monotonically increasing function of μ that approaches its supremum when $\mu \to 0$, and reaches its minimum at $\mu = e^{mt^*} - 1$. So, the range of values this function can take when μ varies within (61.60) is:

$$\frac{UD_1}{m}(mt^* - e^{mt^*} + 1) \leq \frac{UD_1}{m}(\ln(1+\mu) - \mu) < 0 \tag{61.61}$$

From (61.61) it can be concluded that (61.59) has a solution when Θ is within the same limits:

$$\frac{UD_1}{m}(mt^* - e^{mt^*} + 1) \leq \Theta < 0 \tag{61.62}$$

It follows from (61.57) and (61.53) that for the case under consideration Θ already satisfies (61.63):

$$\frac{UD_1}{m}(mt^* - e^{mt^*} + 1) < \Theta \tag{61.63}$$

So the only requirement for the existence of a root of $d\frac{\Psi}{d\lambda}$ or the existence of a solution of (61.59) is

$$\Theta < 0 \tag{61.64}$$

In fact, $\Theta = 0$ is also acceptable as a limiting case with a solution $\lambda_{opt} \rightarrow \infty$. Condition (61.64) has a simple physical meaning that can be concluded from the expressions (61.37) and (61.18). Condition (61.64) effectively specifies an upper limit on the lower level L_1 of x_1 so that this level *can* be reached at all, i.e. if the resource is used at the maximum rate U for the entire duration of time from 0 until t^*.

Conditions (61.64) together with (61.54) comprise *the necessary and sufficient* condition for the existence of a finite minimum in (61.15). These conditions are, in fact, the conditions on the possible values of t^* given initial conditions (61.10). The smallest t^* that satisfies these conditions is at the same time the solution of a *minimum time* optimal control problem for system (61.5):

$$\min_{0 \leq f(t) \leq U} t^*$$

subject to: $L_1 \leq x_1(t^*) \leq R_1$ and the initial conditions (61.10).

λ_{opt} is obtained from (61.59). The switching time t' for the control function $f_{opt}(\tau)$ is obtained from (61.47) as the solution of equation (61.65):

$$1 - \lambda_{opt} D_1 (e^{m(t^* - t')} - 1)] = 0 \tag{61.65}$$

The optimum control function $f_{opt}(\tau)$ is determined similarly to (61.39), with the switching time t' for the entire problem (including the case described by (61.55):

$$t' = \begin{cases} 0, & \text{if } L_1 \leq E(t^*) \\ t^* - \frac{1}{m}\ln(1+\frac{1}{\lambda_{opt} D_1}), & \text{if } L_1 > E(t^*) \end{cases} \tag{61.66}$$

Lastly, the minimum G_F^{min} is: determined from (61.50) - (61.52) using the optimum value λ_{opt}:

$$G_F^{min} = \Psi(\lambda_{opt})$$

Proceeding along the lines of Chapter 5, the next and the final step is making t^* a variable in (61.7) - (61.9) and to optimizing over it. This step had to be done numerically. In the experiment the optimization over t^* was implemented using a modified Brent method with the use of the first derivatives[47]. The union of the solutions of the necessary and sufficient conditions (61.64) and (61.54) for all three subsystems determined the domain of t^* for these optimizations.

A similar solution was implemented for the case when the DM chose to optimize the expenditure of the chemical instead of the fuel.

The same algorithm was implemented in the Automatic control mode. A target (x_g, y_g, M_g, F_g) specified by the DM was translated into a range

$$\begin{aligned} L_1^A &= x_g + 0.01, & R_1^A &= x_g \\ L_2^A &= y_g + 0.01, & R_2^A &= y_g \\ L_3^A &= F_g, & R_3^A &= F(t_0) \\ L_4^A &= M_g, & R_4^A &= M(t_0) \end{aligned} \qquad (61.67)$$

and the solution F_x, f_y, f_r minimizing fuel expenditures was realized for range (61.67). If the solution of this minimization problem did not exist, the control algorithm switched to a target pursuit control (see Appendix E). The same took place if the active range (61.1) turned out to be infeasible.

Appendix E

The Boat Example: Target Pursuit Control

When the DM, using the Automatic or Decision control mode, specified an unattainable goal, the Autopilot executed the *Target pursuit* strategy. Below are the details of its implementation for the boat example.

Consider sampled instances of time when control values of f_x, f_y, f_r were updated: $t_1,\ t_2, \ldots, t_n, \ldots$ Let's denote an (i+1)th sampling period Δt_{i+1}: $\Delta t_{i+1} = t_{i+1} - t_i$, and the *average* sampling period[7] Δt: $\Delta t = \overline{\Delta t_i}$.

Taking the system state at time t_i as the initial condition we can estimate the state at time t_{i+1} from the integrated equations of motion (61.11) - (61.12):

$$\begin{cases} x(t_{i+1}) = x(t_i) + A_x + B_x \Delta t_{i+1} + C_x e^{m\Delta t_{i+1}} + D_x \int_{t^i}^{t^{i+1}} f_x(\tau)(e^{m(\Delta t_{i+1}-\tau)} - 1)d\tau \\ y(t_{i+1}) = y(t_i) + A_y + B_y \Delta t_{i+1} + C_y e^{m\Delta t_{i+1}} + D_y \int_{t^i}^{t^{i+1}} f_y(\tau)(e^{m(\Delta t_{i+1}-\tau)} - 1)d\tau \end{cases} \tag{62.68}$$

During the interval between two consecutive sampling times t_i and t_{i+1} the controls remain constant:

7. As it was noted before, the sampling times were not synchronized by the clock. The occurred within a loop in the program at the instances determined by the operating system.

$$f_x(\tau) = f_{xi}, \quad f_y(\tau) = f_{yi}$$
$$\text{for } t_i \leq \tau < t_{i+1} \tag{62.69}$$

Substituting (61.6) into (62.68) we get expressions for *approximate* values of $x(t_{i+1})$ and $y(t_{i+1})$. These values are approximate for two reasons: firstly, they are sampled estimates based on the zero-order hold approximation of the signals; secondly, equation (62.68) do not use the correct values of the ocean current, instead they use the constant approximation (6.9) thereof. In addition to this the sampling time t_{i+1} is not known in advance (at time t_i).

The controller used in the experiments is based on the estimates of the system state at some *future time* t'_{i+1}: $t'_{i+1} = t_i + 2\Delta t$. The multiplier 2 in the last expression was used to improve the stability of the controller.

Substituting $2\Delta t$ instead of Δt_{i+1} and (61.6) in (62.68) we get the predicted values of $x(t'_{i+1})$ and $y(t'_{i+1})$ as *linear* functions of the controls at time t_i:

$$\begin{cases} x(t'_{i+1}) = x(t_i) + A'_x + B'_x f_{xi} \\ y(t'_{i+1}) = y(t_i) + A'_y + B'_y f_{yi} \end{cases} \tag{62.70}$$

where the coefficients in (62.70) are:

$$A'_x = a_x + 2b_x\Delta t + c_x e_b, \quad B'_x = d_x(\frac{e_b - 1.0}{m} - 2\Delta t)$$
$$A'_y = a_y + 2b_y\Delta t + c_y e_b, \quad B'_y = d_y(\frac{e_b - 1.0}{m} - 2\Delta t)$$

and

$$a_x = -\frac{\dot{x}(t_i)}{m} - \frac{\mu C'_x}{m_b m^2},\quad b_x = -\frac{\mu C'_x}{m_b m},\quad c_x = \frac{\dot{x}(t_i) + \frac{\mu C'_x}{m_b m}}{m},\quad d_x = \frac{\nu_x}{m_b m}$$

$$a_y = -\frac{\dot{y}(t_i)}{m} - \frac{\mu C'_y}{m_b m^2},\quad b_y = -\frac{\mu C'_y}{m_b m},\quad c_y = \frac{\dot{y}(t_i) + \frac{\mu C'_y}{m_b m}}{m},\quad d_y = \frac{\nu_y}{m_b m}$$

and

$$e_b = e^{2m\Delta t},\quad m = -\frac{k_b}{m_b}$$

and (C'_x, C'_y) is the approximation of the ocean current, and all other constants are defined in 6.1.

The predicted vector of the boat's displacement $\begin{pmatrix}\Delta x\\ \Delta y\end{pmatrix}$ from time t_i to time t'_{i+1} is therefore (from (62.70)):

$$\begin{aligned}\Delta x &= A'_x + B'_x f_{xi}\\ \Delta y &= A'_y + B'_y f_{yi}\end{aligned} \tag{62.71}$$

Such controls f_{xi} and f_{yi} were selected that *minimized the angle* between the predicted vector of displacement $\begin{pmatrix}\Delta x\\ \Delta y\end{pmatrix}$ and vector $\begin{pmatrix}x_g - x(t_i)\\ y_g - y(t_i)\end{pmatrix}$ of the *desired direction towards the target* (x_g, y_g). The angle minimization problem almost always had a non-unique solution. Out of all its solutions such controls f_{xi} and f_{yi} were selected that minimized the distance between the predicted boat's position at t'_{i+1}, $(x(t'_{i+1}), y(t'_{i+1}))$ and the target (x_g, y_g).

It is important to note that this controller does not bring the boat to the goal in the minimum time. A minimum time controller would select the controls f_{xi} and f_{yi} only on the basis of minimizing the distance between the target (x_g,y_g) and the predicted boat's position $(x(t'_{i+1}),y(t'_{i+1}))$. The latter controller however would generate a counter intuitive zigzag trajectory.

The DM knew that when the goal was attainable, the boat would swing away from the goal and then come back to it. It was acceptable since the DM knew that the boat would eventually reach the goal. In the case of an unattainable goal, however, any trajectory that did not direct the boat towards the target was misleading. Preliminary experiments with the controller based solely on the minimization of the distance from the target showed that that controller's behavior was confusing to the subjects.

In the final implementation of the system the controller that aligned the expected direction of the boat with the target and only then minimized the distance from the target was used.

After the boat thrusters control values f_{xi} and f_{yi} have been selected according to the just described algorithm, the freezer thruster control f_{ri} was selected to minimize the difference between the predicted *Fuel and Chemical times* at time t'_{i+1}. This was done to balance the expenditure of the fuel and chemical so that the boat would not run out of one of them while there was an ample supply of the other.

The *Fuel Time* at some instance t was the time for how long the fuel would last if the fuel flows in the thrusters and the freezing unit continued at the same rate as it did at t. This time was calculated according to (62.72) (see (6.1) for notation):

$$T_{fuel}(t) = \frac{F(t)}{f_x(t) + f_y(t) + f_r(t)} \tag{62.72}$$

The *Chemical Time* at t was the time for how long the chemical would last if it continued to be absorbed at the same rate as it was at time t. The Chemical time was calculated according to (62.73) (see (6.4) for notation):

$$T_{chem}(t) = -\frac{M(t)}{-\phi T(t) + \psi} \tag{62.73}$$

The bang-bang control law for the freezer thruster was:

$$f_r(t) = \begin{cases} U_r, & \text{if } T_{chem}(t) < T_{fuel}(t) \\ 0, & \text{if } T_{chem}(t) \geq T_{fuel}(t) \end{cases}$$

Appendix F

Input Data for the Experimental Tasks

The data for the experimental tasks were taken from 5 system dynamics data bases, 10 ocean currents data bases and 10 data bases of unexpected failures.

The system dynamics databases contained coefficients for the equations of the boat and refrigerator dynamics (6.1) and (6.4) taken in the following form:

$$\begin{cases} \ddot{x} = m_b\dot{x} + \nu_x C_x(x,y) + f_x \\ \ddot{y} = m_b\dot{y} + \nu_y C_y(x,y) + f_y \end{cases}$$

and

$$\begin{cases} \dfrac{dM(t)}{dt} = -\alpha T + \beta \\ \dfrac{dT(t)}{dt} = m_r T + \nu_r - \lambda_r f_r \end{cases}$$

The variables were measured:

Time -	in seconds (real time).
Distance -	in miles. The map was re-scaled every time it was expanded or shrunk in the map window.
Velocity -	in miles/sec.
Fuel -	in volumetric Unit. The screen gauge gave measurements in Units.
Fuel Flow -	volumetric Unit/sec. The maximum on a fuel control dial

corresponded to 1 Unit/sec of the fuel flow.

Chemical - volumetric Unit. The screen gauge gave measurements in Units.

The dynamics was simulated in real time. The values of the dynamics coefficients are summarized in the table below:

Data	*Boat*			*Refrigerator*				
Base	μ_b	ν_x	ν_y	α	β	m_r	λ_r	ν_r
1	-0.5	3	5	0.18	2.6	-2.5	125	175
2	-0.3	5.5	8	0.18	2.6	-2.5	125	175
3	-0.3	3.5	9	0.6	11	-0.5	25	35
4	-0.3	8	8	0.6	11	-0.5	25	35
5	-0.4	8	5	0.6	11	-0.5	25	35

The coefficient of the refrigerator dynamics were selected to get a range of temperatures from $T_{min} = 20°$ up to $T_{max} = 70°$. The initial conditions x(0), y(0), T(0), M(0) (the initial supply of the chemical), F(0) (the initial supply of the fuel), and the coordinates of the island point with 0 score were:

Boat		*Refrigerator*		*Fuel*	*Islands*	
x(0)	y(0)	T(0)	M(0)	F(0)	$x_{Isl}(0)$	$y_{Isl}(0)$
220	90	32	160	160	250	90

The maximum values of the controls were 1:

$$0 \leq f_x \leq 1, \quad 0 \leq f_y \leq 1, \quad 0 \leq f_r \leq 1$$

The data for the ten basic tasks were formed using the 5 dynamics data bases twice - once with the island angles equal to 0.35 (tasks 1-5) and the second time with the angles equal to 2.15 (tasks 6-10).

All the data for target approaching tasks (1-10) were the same as the data for the line crossing tasks (11-20). The score achieved when crossing the island line in a line crossing task was the distance of the crossing point from the origin (zero score) of the island line. The target for a target approaching task was located on the island line at the distance:

Task	1	2	3	4	5
600	600	400	1800	600	600

The ocean current flow lines were always parallel lines. The intensity of the current in a point in the ocean was a linear function of the point's distance from the islands[8]. The intensity reached its maximum at the island line and the minimum at some specified distance D^{min}. The intensity of the current varied from 1.0 (the maximum) down to I^{min}:

$$C_x(x,y) = In(x,y)\, C_x^{max}$$
$$C_y(x,y) = In(x,y)\, C_y^{max}$$

where In(x,y) is a linear function of the distance of (x,y) from the island line, reaching its maximum when the distance is 0 and reaching its minimum, I^{min}, when the distance is D^{min}.

The following ocean current data were used for the basic tasks:

Task	1	2	3	4	5	6	7	8	9	10
C_x^{max}	-1	-4	-2	-6	-6	-2	-3	-2	-7	-7
C_y^{max}	-3	-6	-8	-5	-4	-1	-7	-5	-5	-3
D^{min}	50	300	300	200	300	50	1000	150	20	1000
I^{min}	0.9	0.8	0.83	0.9	0.85	0.9	0.9	0.85	0.98	0.6

8. The subjects were told that the ocean current was not a simple linear function but a *noisy* function reaching its maximum at the island line and its minimum at the distance D^{min} from the islands. The subjects could see D^{min} and the direction of the current flow lines.

The *unplanned failures* consisted of a loss of 20% of the initial supplies of the fuel and the chemical. The losses happened in 1 to 3 bursts at random moments of time during the first 35 seconds of the process.

Appendix G

Individual Subjects Performance

The following are the raw experimental results. They are collected into two tables: one table for the *Target Approaching* tasks (1-10) and another for the *Line Crossing* tasks (11-20). For the tasks of the first group *lower* scores signify better performance. Alternatively, ***higher*** scores reflect better performance for the tasks of the second group.

Only the scores achieved by the subjects are presented here. Additional data accumulated during the experiments (such as the process times, boat trajectories, etc.) have not been used in the assessment of the experimental results.

The input data for each task is in Appendix F. As a reminder, the island line angle was less than $\frac{\pi}{2}$ (more difficult tasks) for tasks 1-5 and 11- 15, and greater than $\frac{\pi}{2}$ (easier tasks) for tasks 6-10 and 16-20.

Task	Subject	Manual		Automatic		Decision	
		Trial 1	*Trial 2*	*Trial 1*	*Trial 2*	*Trial 1*	*Trial 2*
1	1	462.23	410.56	410.16	410.49	406.95	409.21
	2	440.83	436.14	429.71	420.50	417.01	412.80
	3	414.97	431.00	420.26	409.85	415.31	413.94
	4	407.13	435.22	408.78	408.91	409.10	408.05
2	1	390.14	367.89	353.00	356.56	359.39	347.65
	2	381.99	400.78	387.05	364.19	335.17	340.84
	3	466.02	400.68	383.14	406.73	367.72	354.22
	4	392.75	356.54	392.33	361.98	343.69	341.16
3	1	275.31	263.29	239.85	244.71	239.42	246.51
	2	321.86	254.40	275.89	263.78	238.28	238.39
	3	248.73	250.96	249.87	281.98	250.61	241.73
	4	270.07	251.71	251.16	284.15	246.73	241.88
4	1	1547.27	1489.96	1423.09	1435.71	1425.85	1412.24
	2	1554.60	1531.97	1518.13	1717.32	1423.73	1408.69
	3	1543.46	1530.63	1566.79	1418.11	1409.69	1432.55
	4	1422.11	1413.68	1457.60	1473.44	1410.79	1413.36
5	1	537.60	440.72	400.81	397.94	400.24	396.69
	2	423.90	412.47	427.94	436.03	462.71	399.99
	3	522.88	409.95	465.94	458.76	392.92	398.41
	4	395.90	414.47	403.07	414.88	405.97	397.24
6	1	391.97	134.32	160.87	210.42	109.17	596.97
	2	273.61	278.98	193.45	131.11	72.80	115.68
	3	76.80	68.12	100.88	111.47	107.51	73.03
	4	80.23	131.36	193.91	139.92	101.35	78.19
7	1	444.35	378.98	594.94	370.11	361.17	364.25
	2	385.07	406.33	437.27	450.57	328.07	350.36
	3	404.66	395.26	491.80	418.41	356.34	340.02
	4	373.17	316.97	489.48	423.36	342.41	350.13
8	1	249.60	291.59	31.60	30.84	45.40	32.51
	2	379.18	371.67	141.83	164.71	19.10	16.33
	3	68.68	61.34	108.84	474.70	83.77	67.65
	4	28.03	44.07	165.79	267.33	90.24	86.75
9	1	1530.95	1452.07	960.43	946.03	1262.65	1248.66
	2	1107.70	1145.24	1182.84	1164.13	960.34	952.18
	3	985.84	1083.16	1597.87	1136.47	940.33	954.29
	4	944.17	923.94	1297.86	1726.77	933.44	923.37
10	1	345.74	294.63	328.06	319.78	186.64	201.45
	2	357.27	190.17	253.55	227.79	187.60	204.07
	3	196.19	189.41	595.47	251.79	200.81	184.65
	4	191.11	168.84	262.40	220.62	186.53	201.47

Table G.1. Experimental Results: *Target Approaching* tasks.

Task	Subject	Manual		Automatic		Decision	
		Trial 1	*Trial 2*	*Trial 1*	*Trial 2*	*Trial 1*	*Trial 2*
11	1	105.75	0.00	156.23	167.68	151.86	124.60
	2	52.62	171.25	0.00	151.85	0.00	183.08
	3	170.41	155.30	155.29	155.09	181.45	0.00
	4	172.70	173.64	0.00	180.22	184.69	183.44
12	1	128.20	150.32	206.23	230.85	208.61	204.97
	2	0.00	0.00	216.24	221.64	243.80	248.56
	3	198.75	207.84	0.00	205.45	244.82	248.01
	4	224.09	226.98	213.93	203.53	0.00	241.23
13	1	0.00	141.15	0.00	133.23	0.00	150.73
	2	100.26	98.06	0.00	0.00	148.84	150.43
	3	126.66	138.45	126.20	0.27	146.92	0.00
	4	131.96	135.76	131.21	152.63	0.00	148.74
14	1	0.00	251.91	0.00	319.35	189.49	250.29
	2	0.00	294.66	0.00	327.18	336.60	344.20
	3	276.48	290.96	0.00	0.00	347.25	270.90
	4	320.95	332.67	280.56	319.22	353.72	0.00
15	1	0.00	0.00	0.00	81.92	175.23	190.22
	2	0.00	60.79	0.00	0.00	201.07	0.00
	3	0.00	188.85	112.90	43.96	0.00	196.79
	4	193.35	185.58	182.19	195.91	198.62	0.00
16	1	298.04	319.91	399.38	417.36	347.33	447.75
	2	371.09	328.57	443.84	447.93	333.67	337.31
	3	73.36	0.00	474.99	35.74	459.22	482.55
	4	387.05	456.97	485.49	467.92	290.22	449.36
17	1	48.30	117.88	170.06	192.13	152.83	71.79
	2	0.00	152.25	115.60	166.15	192.91	199.97
	3	57.82	189.70	111.50	98.84	182.11	175.36
	4	193.19	185.06	3.80	105.58	164.96	180.37
18	1	557.64	564.98	645.14	74.21	520.04	74.22
	2	291.16	161.10	74.20	11.61	74.21	663.12
	3	14.71	489.88	551.68	30.55	653.62	671.58
	4	652.64	638.16	13.83	593.72	647.93	74.21
19	1	194.22	295.51	11.95	236.14	455.11	476.64
	2	541.04	453.51	389.43	525.06	543.23	560.55
	3	547.16	534.99	427.93	532.61	11.95	565.63
	4	566.30	506.21	512.62	506.91	567.82	553.35
20	1	64.17	130.50	275.78	23.32	337.83	342.67
	2	351.00	10.29	10.29	306.57	368.06	370.20
	3	332.83	345.91	286.10	324.91	363.82	360.05
	4	342.84	335.47	10.29	306.76	352.30	356.03

Table G.2. Experimental Results: Line Crossing tasks.

References

1. Alvrod, C.H., (1983). "The Pros and Cons of Goal Programming: A Reply", *Computers and Operation Research,* Vol.10, No.1, pp. 61-62.

2. Arrow K.J., and Raynaud H., (1986). **Social Choice and Multicriterion Decision-Making**, MIT Press, Cambridge.

3. Bard, J.F. (1986). "A Multiobjective Methodology for Selecting Subsystem Automation Options", *Management Science,* Vol. 32, No. 12, pp. 1628-1641.

4. Benayoun, R., De Montgolfier, J., Terngny, J., and Larichev, O. (1971). "Linear Programming with Multiple Objective Functions: STEP-Method (STEM)", *Mathematical Programming,* No. 1, pp. 366-375.

5. Bertsekas, D.P. (1982). **Constrained Optimization and Lagrange Multiplier Methods,** Academic Press.

6. Bischoff, E.E. (1984). "A Posteriory Trade-off Analysis in Reference Point Approaches", in *Interactive Decision Analysis - Laxenburg, Austria 1983.* (M.Grauer and A.P.Wierzbicki eds.), Springer-Verlag, pp. 139-145.

7. Bischoff, E.E. (1986). "Multi-objective Decision Analysis - the Right Objective?", in *Large-Scale Modelling and Interactive Decision Analysis - Eisenach, GDR 1985.* (G. Fandel, M.Grauer, A.Kurzhanski and A.P.Wierzbicki eds.), Springer-Verlag, pp. 155-160.

8. Bryson, A.E., and HO Y.-C., (1969). **Applied Optimal Control,** Ginn and Co.

9. Chankong, V., and Haimes, Y.Y., (1983). **Multi-objective Decision Making: Theory and Methodology,** North-Holland.

10. Charny, L., Hornsby, M.E., and Sheridan, T.B., (1987). "An Interactive Multi-Objective Decision-Aiding System for Tactical Mission Planning", in *Proceedings of the Human Factors Society 31st Annual Meeting,* pp. 432-436. October 1987, New York, N.Y.

11. Charny, L., and Sheridan, T.B. (1986). Satisficing Decision-Making in Supervisory Control, MIT Man-Machine Systems Laboratory report.

12. Chernoff, H. (1973). "Using Faces to Represent Points in k-Dimensional Space Graphically", *J. Amer. Statist. Assoc.,* Vol. 68, pp. 361-368.

13. Christofides, N., (1975). **Graph Theory, an Algorithmic Approach,** Academic Press.

14. Cohon, J.L., (1978). **Multiobjective Programming and Planning,** Academic Press.

15. Crama, Y., and Hansen, P. (1983)." An Introduction to the ELECTRE Research Programme", in *Essays and Surveys on Multiple Criteria Decision Making - Mons, Belgium 1982.* (P.Hansen ed.), Springer-Verlag, pp. 31-42.

16. Eschenauer H.A., (1986). "Multicriteria Optimization. Procedures in Application on Structural Mechanics Sysytems", in *Recent Advances and Historical Development of Vector Optimization. Proceedings of the International Conference on Vector Optimization, Darmstadt, West Germany, 1986,* pp. 345-376.

17. Edgeworth, F.Y., (1881). **Mathematical Psychics,** C. Kegan Paul & Co.

18. Even, S., (1979). **Graph Algorithms,** Computer Science Press, Inc.

19. Fichefet, J., (1985). "Data Structures and Complexity of Algorithms for Discrete MCDM Methods", in *Multiple Criteria Decision Methods and Applications* (G. Fandel, J. Spronk eds.), Springer-Verlag, pp. 197-226.

20. Friedman, M., (1962). **Price Theory: A Provisional Text**, Aldine, Chicago.

21. Fujimoto, H., (1986). "Computer Aided Interactive Multiobjective Satisfaction and its Application to Design Problems", in *Toward Interactive and Intelligent Decision Support Systems. Proceedings of the Seventh International Conference on Multiple Criteria Decision Making, Kyoto, Japan, 1986* (Sawaragi, K., Inoue, K., and Nakayama, H. eds.), Springer-Verlag, pp. 323-332.

22. Grauer, M., Lewandowsky, A., and Wierzbicki, A. (1984). "DIDASS - Theory, Implementation and Experiences", in *Interactive Decision Analysis - Laxenburg, Austria 1983.* (M.Grauer and A.P.Wierzbicki eds.), Springer-Verlag, pp. 22-30.

23. Hsieh, W.N. and Kleitman, D.J., (1973). "Normalized Matching in Direct Products of PArtial Orders", *Studies in Applied MAthematics,* Vol. L11, No. 3, pp. 285-289, Massachusetts Institute of Technology.

24. Habenicht, W. (1983). "Quad Trees, a Datastructure for Discrete Vector Optimization problems", in *Essays and Surveys on Multiple Criteria Decision Making - Mons, Belgium 1982.* (P.Hansen ed.), Springer-Verlag, pp.

136-145.

25. Haimes, Y.Y., and Li, D. (1988). "Hierarchical Multiobjective Analysis for Large-scale Systems: Review and Current Status", in *Automatica*, Vol. 24, No. 1, pp. 53-69.

26. Hansen, P. (1980). "Bicriterion Path Problems", in *Multiple Criteria Decision Making Theory and Application - Hagen / Konigswinter, West Germany 1979.* (G. Fandel and T. Gal eds.), Springer-Verlag, pp. 109-127.

27. Hemming, T., (1986). "Guide Lines for Testing Interactive Multicriterion Methods by Simulation", in *Toward Interactive and Intelligent Decision Support Systems. Proceedings of the Seventh International Conference on Multiple Criteria Decision Making, Kyoto, Japan, 1986* (Sawaragi, K., Inoue, K., and Nakayama, H. eds.), Springer-Verlag, pp. 250-259.

28. Hobbs, B.F. (1986). "What We Can Learn from Experiments In Multiobjective Decision Analysis", *IEEE Transactions on Systems, Man, and Cybernetics,* Vol. SMC-16, No.3.

29. Hwang C., and Yoon K., (1981). **Multiple Attribute Decision Making. Methods and Applications. A State-of-the-Art Survey,** Springer-Verlag.

30. Ignizio, J., (1977). "A Review of Goal Programming: A Tool for Multiobjective Analysis", *Journal of the Operational Research Society,* 29, pp. 1109-1119.

31. Ignizio, J., (1983). "Generalized Goal Programming", *Computers & Operational Research,* Vol. 10, No. 4, pp. 277-289.

32. Isermann, H., (1984), "An Analysis of the Decision Behavior of Individual Decision Makers in the Course of A computer-Assisted Interactive Decision Process", in *Decision Making with Multiple Objectives - Cleveland, Ohio 1984.* (Y.Y. Haimes and V. Chankong eds.), Springer-Verlag, pp. 236-249.

33. Karp, R.M., (1972). "Reducibility Among Combinatorial Problems", in *Complexity of Computer Computations,* (R.E. Miller and J.W. Thatcher eds), Plenum Press.

34. Keeney, R.L. and Raiffa H. (1976). **Decisions with Multiple Objectives: Preferences and Value Tradeoffs,** John Wiley & Sons, Inc.

35. Korhonen, P. (1986). "Solving Discrete Multiple Criteria Problems by Using Visual Interaction", in *Large-Scale Modelling and Interactive Decision Analysis - Eisenach, GDR 1985.* (G. Fandel, M.Grauer, A.Kurzhanski and

A.P.Wierzbicki eds.), Springer-Verlag, pp. 176-185.

36. Korhonen, P. (1986). "On Using Computer Graphics for Solving MCDM-Problems", Solving Discrete Multiple Criteria Problems by Using Visual Interaction", in *Toward Interactive and Intelligent Decision Support Systems. Proceedings of the Seventh International Conference on Multiple Criteria Decision Making, Kyoto, Japan, 1986* (Sawaragi, K., Inoue, K., and Nakayama, H. eds.), Springer-Verlag, pp. 154-162.

37. Larichev O.I., and Nikiforov A.D. (1986). "Analytical Survey of Procedures for Solving Multicriteria Mathematical Programming Problems", in *Toward Interactive and Intelligent Decision Support Systems. Proceedings of the Seventh International Conference on Multiple Criteria Decision Making, Kyoto, Japan, 1986* (Sawaragi, K., Inoue, K., and Nakayama, H. eds.), Springer-Verlag, pp. 95-104.

38. Lee, E.B., and Markus L., (1967). **Foundations of Optimal Control Theory,** John Wiley & Sons, Inc.

39. Lieberman, E.R., (1988). "Multi-Objective Programming in the USSR", PhD Thesis, John Hopkins University.

40. Liu, C.L., (1968), **Introduction to Combinatorial Mathematics,** McGraw-Hill.

41. Lockett, G., Hetherington, B., and Yallup, P., (1985). "Subjective Estimation and Its Use in MCDM", in *Decision Making with Multiple Objectives - Cleveland, Ohio 1984.* (Y.Y. Haimes and V. Chankong eds.), Springer-Verlag, pp. 358-374.

42. Mendel, M.B., (1989). "Development of Bayesian Parametric Theory with Applications to Control", PhD Thesis, Department of Mechanical Engineering, MIT.

43. Minoux, M., (1986), **Mathematical Programming. Theory and Algorithms,** John Wiley & Sons, Inc.

44. Michalowski, W., (1984). "An Experiment with Zionts - Wallenius and Steuer Interactive Programming Methods", in *Decision Making with Multiple Objectives - Cleveland, Ohio 1984.* (Y.Y. Haimes and V. Chankong eds.), Springer-Verlag, pp. 424-429.

45. Nakayama, H., Takeguchi, T., and Sano, M. (1983). "Interactive Graphics for Portfolio Selection", in *Essays and Surveys on Multiple Criteria Decision Making - Mons, Belgium 1982.* (P.Hansen ed.), Springer-Verlag, pp. 280-289.

46. Osyczka, A., (1984). **Multicriterion Optimization in Engineering**, Ellis Horwood Ltd., Halsted Press.

47. Press, W.H., Flannery B.P., Teukolsky S.A., Vetterling W.T. (1988). **Numerical Recipes in C,** Cambridge University Press.

48. Radford A.D. and Gero J.S., (1988). **Design by Optimization in Architecture, Building, and Construction**, Van Nostrand Reinhold Co.

49. Ramesh, R., Karwan., M.H., Zionts, S. (1986). "An Empirical Assessment and Insights on Two Multicriteria Integer Programming Algorithms", in *Toward Interactive and Intelligent Decision Support Systems. Proceedings of the Seventh International Conference on Multiple Criteria Decision Making, Kyoto, Japan, 1986* (Sawaragi, K., Inoue, K., and Nakayama, H. eds.), Springer-Verlag, pp. 182-195.

50. Rasmussen, J., (1983). "Skills, rules, and knowledge; signals, signs and symbols; and other distinctions in human performance", *IEEE Transactions on Systems, Man, and Cybernetics,* SMC-13, pp. 257-266.

51. Ratschek H., Rokne, J., (1988). **New Computer Methods for Global Optimization,** Ellis Horwood Ltd.

52. Rockafellar, R.T. (1970). **Convex Analysis,** Princeton University Press.

53. Roseborough, J.B., (1988). "Aiding Human Operators with State Estimates", PhD Thesis, Department of Mechanical Engineering, MIT.

54. Salukvadze, M.E., (1979). **Vector-valued Optimization Problems in Control Theory**, Academic Press.

55. Savage, L.J., (1972). **The Foundations of Statistics**, Dover Publications.

56. Serafini P., (1986). "Some Considerations about Computational Complexity for Multi Objective Combinatorial Problems", in *Recent Advances and Historical Development of Vector Optimization. Proceedings of the International Conference on Vector Optimization, Darmstadt, West Germany, 1986,* pp. 222-232.

57. Serrano, D., (1987). "Constraint Management in Conceptual Design", PhD Thesis, Department of Mechanical Engineering, MIT.

58. Sheridan, T.B., (1976). "Toward a General Model of Supervisory Control", in Sheridan, T.B., Johannsen, G., (Eds.) *Monitoring Behavior and Supervisory Control,* Plenum Press, N.Y.

59. Sheridan, T.B., (1983). "Supervisory Control of Remote Manipulators, Vehicles and Dynamic Processes: Experiments in Command and Display Aiding," MIT Man-Machine Systems Laboratory Technical Report.

60. Silverman, J., Steuer, R.E., and Whisman, A.W. (1985). "Computer Graphics at the Multicriterion Computer/User Interface", in *Decision Making with Multiple Objectives - Cleveland, Ohio 1984*. (Y.Y. Haimes and V. Chankong eds.), Springer-Verlag, pp. 201-213.

61. Simon, H.A. (1969). "A Behavioral Model of Rational Choice", *Quarterly Journal of Economics*, 69: 99-118.

62. Skulimowsky, A.M., (1986). "An Interactive Modification of the Decision Set to Attain a Target Point in Vector Optimization Problems", in *Toward Interactive and Intelligent Decision Support Systems. Proceedings of the Seventh International Conference on Multiple Criteria Decision Making, Kyoto, 1986* (Sawaragi, K., Inoue, K., and Nakayama, H. eds.), Springer-Verlag, pp. 142-153.

63. Sobol, I.M. and Statnikov, R.B. (1981). **Optimal Parameters Selection in Multi-Criteria Problems,** (in Russian: *Vybor Optimalnykh Parametrov v Zadachakh so Mnogimi Kriterjiami*), Nauka.

64. Stadler, W., (1984). "Applications of Multicriterion Optimization in Engineering and the Sciences", in *MCDM: Past Decade and Future Trends,* (Milan Zeleny, ed.), JAI Press.

65. Stadler, W., (1984). "Multicriteria Optimization in Mechanics (A Survey)", *Applied Mechanics Reviews,* Vol. 37, No. 3, pp. 277-286.

66. Stadler, W., (1986). "Initiators of Multicriteria Optimization", in *Recent Advances and Historical Development of Vector Optimization. Proceedings of the International Conference on Vector Optimization, Darmstadt, West Germany, 1986,* pp. 3-25.

67. Stewart, T.J., (1984). "Inferring Prefernces in Multiple Criteria Decision Analysis Using a Logistic Regression Model", *Management Science,* Vol. 30, No.9.

68. Szidarovszky, F., Gershon, M.E., and Duckstein L., (1986). **Techniques for Multiobjective Decision Making in System Management,** Elsevier.

69. Tremblay J.P., and Mahonar R., (1975). **Discrete Mathematical Structures with Applications to Computer Science,** McGraw-Hill.

70. Wierzbicki, A.P. (1982). "A Mathematical Basis for Satisficing Decision Making", *Mathematical Modelling,* Vol.3, pp. 391-405.

71. Zeleny, M., (1977). "Adaptive Displacement of Preferences in Decision Making",*TIMS Studies in Management Sciences,* 6, pp. 147-157.

72. Zeleny, M., (1981). "The Pros and Cons of Goal Programming", *Computers and Operation Research,* Vol.8, No.4, pp. 357-359.

73. Zeleny, M., (1982). **Multiple Criteria Decision Making,** McGraw Hill.

74. Zionts, S, and Wallenius, J. (1976). "An Interactive Programming Method for Solving the Multiple Criteria Problem", *Management Science,* Vol.22, No.6, pp. 652-663.