Assistance in Decision Making: Decision Help and Decision Analysis

Joel Brynielsson[*]
Department of Numerical Analysis
and Computer Science
Royal Institute of Technology
SE-100 44 Stockholm, Sweden
joel@nada.kth.se

Rego Granlund
Department of Computer
and Information Science
Linköping University
SE-581 83 Linköping, Sweden
reggr@ida.liu.se

Abstract

Decision help and decision analysis are tools that will be of importance in decision support systems. They are likely to exist both in real operational systems and in simulation based systems used for training. In command and control an example of a decision task is to command units in a geographical environment. We have taken a closer look at this type of decision task, using a simulated microworld as a research tool. In particular we are interested in the case when the decision maker already has selected a course of action, but wants to have critique and suggestions of improvements. To analyze a decision where the "main direction" is already known in this way we denote *decision analysis*, as opposed to *decision help*.

An algorithm for decision analysis in maps has been developed. This algorithm is based on rules. An implementation in Java with classes suited for reuse has been developed. The implementation has been tested in a microworld system, C^3FIRE, suited for practice in forest fire fighting.

1 Introduction

Decision making in emergency- and military organizations is classified as *distributed decision making*[9] which means that the decision making is distributed among the actors in the organization. These ideas influence the ongoing research

[*]NUTEK program for Complex Technical Systems

in command and control[1, 11, 13, 16, 18, 20], and also seem to be the goal for products currently under development[19].

Making a decision can be seen as the task of finding a certain process that can be used to control some other process/processes in the system. The goal of the decision maker is to analyze the state of the system and try to see what existing processes he or she can use and control to fulfill his goal. In [6, 7, 10] decision making in a complex system is characterized by the following properties:

- A series of decisions are necessary.
- The decisions are dependent on each other.
- The state of the decision problem changes, both autonomously and as a consequence of the decision makers actions.
- The decisions have to be made in real time.

These four properties form a subset within the distributed decision making, classified as *dynamic decision making*.

2 Decision making

2.1 Decision theory

There exist many theories for decision making. In classic decision making the *decision tree* is often used. This gives a programmed decision rule, that can be obtained from knowledge of plausible situations and their right responses, or by machine learning from a repository of previous cases. In this paper another common theory for decision making using *utility matrices* and probability estimates is described[5, 14, 17]. The utility matrix partly solves the problem of the decision tree.

2.1.1 Introduction

We apply a Bayesian approach, although it is not undisputed. It is however supported by numerous of successful applications and experiments[2]. Roughly speaking, the Bayesians, or *subjectivists*, wish to introduce intuitive judgments and feelings directly into the formal analysis of a decision problem. The non-Bayesians, or *objectivists*[17], feel that these subjective aspects are best left out of the formal analysis and should be used only, if at all, to bridge the gap between the real world and the objective results one obtains using a formal model.

The analysis of a decision problem under uncertainty requires the following steps[17]:

1. list the viable options available to you for gathering information, for experimentation, and for action,
2. list the events that may possibly occur,
3. arrange in chronological order the information you may acquire and the choices you may make as time goes on,
4. decide how well you like the consequences that result from the various courses of action open to you,
5. judge what the chances are that any particular uncertain event will occur.

After taking these steps, meaning that your problem is systematically described and your preferences and judgments are recorded, we can start applying a strategy aiming towards a solution.

2.1.2 Theory

Consider every possible world, ω_i, and let every such world occur with possibility p_i. We have a number of possible actions, H_j, that we can perform. Further, let $M_{i,j}$ be the utility of performing action H_j in the world ω_i. If we perform action H_j we get the following expected utility for action H_j:

$$E(M_j) = \sum_{i=1}^{n} p_i M_{i,j}$$

We want to choose an action, H_j, so that we achieve as big expected utility as possible. Therefore we choose H_j so as to maximize the sum above.

In many cases the possible worlds and actions are described as continuous. In such cases the mathematics gets more complicated. The principles are the same, though.

2.1.3 An example

A military base observes radar echos to assess a possible missile threat. Hence there are two possible worlds, ω_1 and ω_2:

- ω_1 means that a real threat exists,
- ω_2 means that there is no threat.

We can perform two possible actions, H_1 and H_2, meaning the following:

- H_1 means that we do not do anything at all,
- H_2 means that we fire.

In this way we achieve a 2×2-matrix with four different M-values, where for example $M_{2,2}$ defines the utility of shooting when no real threat exists and $M_{1,2}$ defines the utility of shooting when there is a real threat. If the decision support system is implemented in a clever way $M_{2,2}$ should then contain a low value (low utility) and $M_{1,2}$ a high value (large utility).

The utility of firing if the observed echo comes from a real threat is of course big. If you are really being attacked the radar plots are likely to advocate world ω_1 with larger and larger possibility (i.e., p_1 increases and p_2 decreases). After a while the expected utility for action H_2 will win and we fire.

For the interested reader it can be interesting to know that this example has a connection to reality. According to [14] a NATO-base on Greenland had problems with birds in the 1960:s. These birds flew higher than thought possible when developing the system. This resulted in increasing possibility for a real threat and one came very close to firing (exactly how close is secret).

2.2 Decision help

We define *decision help* as a tool that helps the commander look at the whole picture, in order to suggest a course of action. We think of decision help implemented as a tool connected to a computer system where one needs to make decisions.

2.3 Decision analysis

We define *decision analysis* as a tool that gives the commander feedback and suggestions of improvements regarding a specific course of action. We think of decision analysis implemented as a tool connected to a computer system where one needs to analyze decisions.

Suppose that a decision maker has understood the situation and knows what to do (i.e., situation awareness). The decision maker is now interested in analyzing his decision to get feedback regarding risks he ought to pay attention to and suggestions for improvements regarding his decision. He is not interested in finding out what to do, when looking at the whole picture, since he has already selected a course of action. He is interested in help with the decision he has actually decided to perform.

2.4 Decision help versus decision analysis

In research performed at the Swedish National Defence College[3, 4, 16] the issue of decision help versus decision analysis has been analyzed. Decision help and decision analysis are both tools that should be included in a computer support for a commanders decision making. It has, however, been pointed out that the decision help must not be given a status of great importance. It is important that a decision maker takes part in the decision process and takes his own decisions, regardless what a computer tells him. Moreover, humans are good at finding general patterns, but are not good at analyzing details. With this in mind the decision analysis has been considered as a more important tool than the decision help.

3 A way to perform decision analysis

3.1 Limitations

Decision analysis can be performed in many ways. In this work a limitation has been that the analysis is supposed to be performed on decisions concerning "moves" in a geographical area (i.e., a map). The specifications on this geographical area are best kept as general as possible.

The geographical area can be represented as a large matrix where each matrix element corresponds to a particular spot on the geographical area. Each element then needs to keep track of the following information:

1. a set of environmental parameters characterizing this spot,
2. the prevailing state for this spot.

For a decision support system, where the domain is forest fire fighting, the above mentioned information could be the following:

1. normal tree, young pine tree (fast burning), birch tree (slow burning), and houses,

2. not burning, burning, put out, and burned out.

By requiring that our map is represented in this way we can use our system for decision analysis in a broad range of applications. By changing the attributes of the matrix elements we can define cities, mountains or an archipelago. By changing the possible states we can define all types of possible scenarios that we are interested in, for example fire situations, nuclear disasters and war scenarios.

In a large system we want to have more details and we obtain that by increasing the number of possible geography types and the number of possible states. We can not keep everything we make and reuse it, though. Every system is designed for a special purpose and uses particular types of states for this purpose. Also different scales regarding geography resolution is used.

Additionally, a system consist of moving units. In a decision support system designed for forest fire fighting you need fire fighting units that can move around. Since these moving units do not belong to a particular spot on the map, it is better to keep track of them separately. Besides, such moving units have their own characteristics and they are therefore best seen as a set of "objects" (easily modeled in an object oriented environment). One way to describe the characteristics of a particular moving unit is to let it belong to one particular state, out of a finite number of different states. We believe that this way of describing the characteristics of a moving unit is sufficient for most of the systems one wants to develop.

A way of referring to these units must be possible. Therefore the units, as well as the geography, must be defined in a general way.

3.2 Solution

The idea we have developed is to design a system based on rules that performs the decision analysis on a given move on the map. As input data the analysis application requires a map (represented as a matrix on the form discussed above) and the coordinates for the move that is going to be analyzed. The map is matched towards a set of rules (actually a set of sets of rules, see section 3.3.3). As output the system delivers the results of the analysis. The result consists of a number of text strings with suggestions of improvements, criticism etc. regarding the proposed move. Exactly what type of answers you will get depends on how you define your rules.

To keep the system as general as possible, the rules and the map definitions are kept in text files that can be easily adjusted. Besides that, the rules are defined using the terminology developed when constructing the map- and unit definitions.

The rules focus on small matrices around the path that the decision maker has decided he wants to move along. If a small matrix exists that fulfills the requirements of a rule, this rule is a matching one.

3.3 Rules

The amount of possible input data for an analysis application grows rapidly when the playing field (the map matrix) increases in size. Due to all different types of vegetation, states and the fact that the amount of matrix elements grows faster when the matrix is big, the amount of different possible inputs soon becomes impossible to handle. Thus, we can conclude that the trivial approach, meaning that we make a rule for each map, is out of the question. Other techniques must be considered.

3.3.1 One rule

A rule consists of a *pattern* and an *answer*. When the pattern in a certain rule matches the input data the answer in this particular rule applies. Our idea is to look at the start- and/or the stop coordinates of the move to be analyzed. The pattern in a rule defines a small matrix around either the start- or the stop point (or both), see figure 1. This small matrix contains information about what states, vegetations and moving units that are allowed for each spot in the matrix. Besides this, the small matrix also contains information on where in the small matrix the start- and/or stop point should be located. This idea reduces the amount of rules needed significantly.

When the matching is to be made the start- and/or stop point is used to efficiently indicate where in the map it shall search for the small matrix. The matching is performed by comparing the information in the small matrix with the corresponding information in the map. As soon as an item that does not fit is encountered, the matching stops with an error. On the other hand, if all items in the small matrix are walked through and match the corresponding items in the map this rule is a matching rule.

The idea of only looking at the start- and/or stop point can clearly be discussed. However, in this project we found this idea to be the best one and wanted to test the results it gives. We do not in any way claim that this is the only way or the

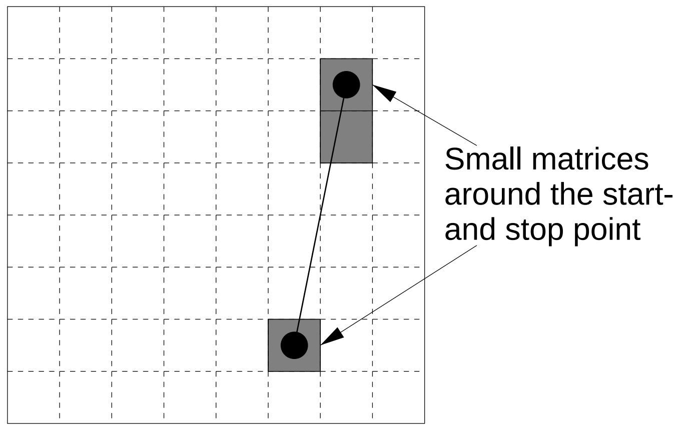

Figure 1: The rules focus on small matrices around the start- and/or stop point of a given move. Information in the big matrix must agree with the information in these small matrices in order for a rule to apply.

best way an analysis should be committed. We do think that it is an interesting approach worth trying, though.

3.3.2 Rotation

For now we have not been able to come up with one single reason to make the system different in any of the four cardinal points. Therefore we suggest that the system is being implemented so that each rule is applied in all four different directions. This is in theory easily applied by rotating the small matrix 90° three times after the first matching, applying new matches between each rotation.

It would be easy to include an option in the rule that indicates however this rule should be rotated or not. As mentioned we have not found a reason for this, though.

3.3.3 Many rules

Some of the rules you might think of when writing your rule base depend on each other while other rules do not depend on each other. An example of rules depending on each other in the case of a system developed for fire fighting is when you want to develop the following idea:

> If the commander proposes that a fire unit shall leave a spot with a house, then give a warning (it is more important to protect houses than to protect ordinary forest). However, if another fire unit is located on the same spot as the one that is proposed to leave, no warning should be given (the warning is then useless).

A nice way to implement this when “thinking rules” is to use the following two rules where you require that only the first matching one is taken:

1. If a fire unit is leaving a spot with a house and at least one more fire unit is present, do not suggest anything at all.
2. If a fire unit is leaving a spot with a house then give a warning.

Typically you want to be able to instruct the system that a certain rule should only be used if another rule has not been giving a correct matching. However, there are rules that still should be considered independently of the fact that other rules have already given a correct matching. For example, look at the following rule and compare it to the examples above:

> Look at the stop point of the proposed move. If there is fire there, then give positive feedback to the commander saying “Your target position is on fire. This seems like a good move.”.

We consider this problem by proposing that the rules are divided into subgroups where only one rule in each subgroup can apply. In each such subgroup the rules are listed in priority. When the analysis is performed each group with rules is traversed until a matching rule is found or the group is empty. After that the analysis moves on to the next rule group.

3.4 Generality

An important aspect of the idea discussed above is that it will work well for analyzing any type of map. The concepts are general and are not limited to maps within a specific system, but will work with any system where you use a map for representation of a certain world or situation.

The typical situation is that you have a map representing the world in the currently known way. It does not matter if information on the map is incorrect because what you want to do is to analyze the situation with a map representing the world in the way you *think* it looks like. If you later on get information about changes in the world that the map represents you analyze again given this new map data.

Primarily we think of a system that already uses maps where you can integrate the ideas of decision analysis directly. Such systems typically update the information on the map with respect to the information that the system receives. In such systems the decision analysis can be performed in an easy way by just indicating the position you are aiming for to receive the analysis.

4 Application on a dynamic problem

4.1 Microworld simulations

Computer simulations of real world phenomena gains importance in all sciences. As computer power increases the real world simulations get more detailed. We will study schematic simulated worlds called *microworlds*[8]. In such microworlds it is possible to provide the players with the latest technology available today as well as the technology of tomorrow.

Many examples of realistic microworlds can be seen in the war games available for personal computers on the commercial market. The games available today are not suited for research purposes[15], mainly because you are not allowed to register information.

4.2 The C^{3}FIRE microworld

In our research we have implemented a decision support system in the C^{3}FIRE microworld[12]. C^{3}FIRE is a command, control and communication experimental simulation environment. The primary purpose of the system is to allow researchers to experiment with different strategies for studies and training of personnel. The domain, which is forest fire fighting, is of subsidiary interest and has been chosen simply to demonstrate the principles.

The system generates a task environment in which team members cooperate to extinguish a forest fire. The simulation includes the forest fire, houses, different kinds of vegetation, computer-simulated agents such as reconnaissance persons, and fire-fighting units. The participants in the simulation are parts of a fire-fighting organization and take the roles of staff members or fire-fighting unit leaders, see figure 2.

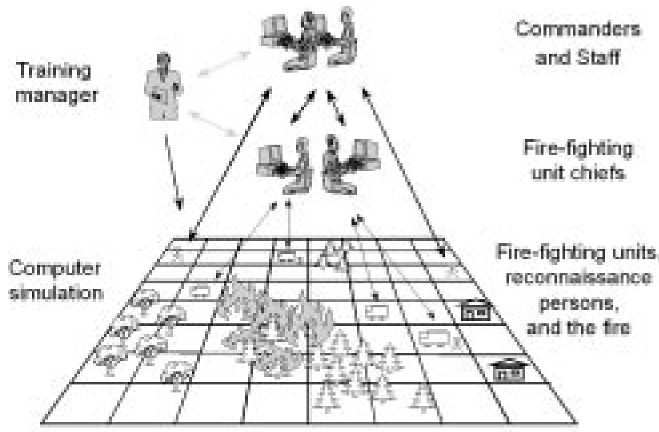

Figure 2: The C^{3}FIRE microworld.

4.3 C^{3}FIRE integration

Since the C^{3}FIRE is implemented in Java it felt natural to use Java for the implementation of the decision analysis. Java is an object oriented programming language. This suits us very well since there are many natural objects in a microworld system that are easy to identify.

The implementation has, however, been made completely separated from the C^{3}FIRE. The goal has been to construct a system as general as possible. Therefore we made a system which uses a world totally defined in easily adjustable text files. Also, abstraction and documentation of the provided interface have been important issues. The development has been made in three phases:

1. Implementation of a class representing the decision analysis. The goal has been to make a good abstraction so that the system can be easily integrated in any system implemented in Java.

2. Construction of a tool, DAT, that parses log-files from C^{3}FIRE, creates a map given the data from these log-files and, creates an instance of the decision analysis class and then uses this instance to perform an analysis on the generated map. This tool gave us the opportunity to test our ideas in an easy way.

3. Integration in C^{3}FIRE, see figure 3.

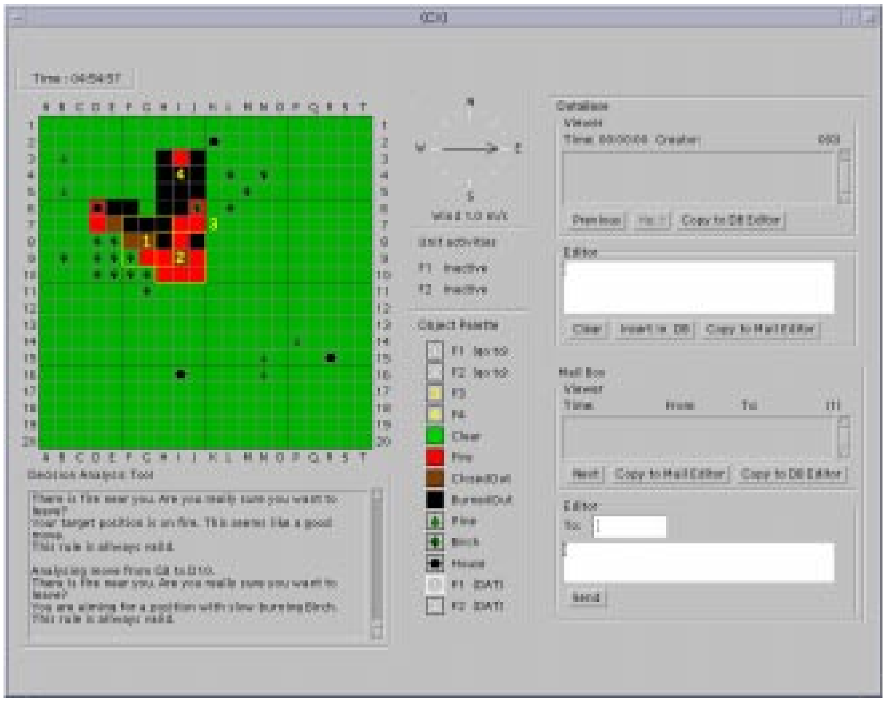

Figure 3: C³FIRE with new analysis-capabilities implemented.

By performing the work in this way we got feedback in phase three on how well the work in phase one was actually performed. The integration in C³FIRE went very smooth so the above described way of performing the work now seems to have been a successful idea.

5 Conclusions

We have defined two very helpful tools that should be available when implementing a system for command and control, *decision help* and *decision analysis*.

The decision help should be implemented so that the commander does not tend to use it too much. It is important that the commander takes part in the decision process to fully understand the whole situation picture and its potentials.

The decision analysis has shown to be a very useful tool and research ought to be continued in this area. Exactly how the decision analysis should be implemented is still to be explained, but it seems obvious that a tool for decision analysis imposes no drawbacks on a system for command and control.

6 Future work

The area of decision analysis should be investigated further and tested in other applications. Different sets of rules should be implemented and investigated. Questions that need to have answers are:

- Is it possible to create a generic algorithm for decision analysis?
- Are you only interested in critique, or is it also interesting to get positive feedback? Should the critique be given higher priority than the positive feedback?
- Different users may have different requirements regarding the given analysis. One user might wants to have positive feedback while another user prefers not to have it. One user might wants to have detailed information while another user prefers to see the whole picture. Should the system include different rule-bases that the commander can choose between?
- How many rules are required when implementing a tool for decision analysis in reality? Can we make a system that performs good enough using a limited amount of rules?
- How do you eliminate information overflow when many rules apply at the same time?
- Scalability is an important issue. How do we implement applications so that they can take care of maps covering the whole world as well as maps covering a single neighborhood?

Acknowledgments

The significantly ideas presented here have been influenced by intensive discussions within the IQ Ledning working groups and in the Friday sessions of the Swedish National Defence College. We particularly want to mention Berndt Brehmer, the Swedish National Defence College, Stefan Arnborg, Henrik Artman and Klas Wallenius, the Royal Institute of Technology, and Lars Eriksson, Ledab.

References

[1] David S. Alberts, John J. Garstka, and Frederick P. Stein. *Network Centric Warfare: Developing and Leveraging Information Superiority*. CCRP Publication Series, 2 edition, 1999.

[2] Stefan Arnborg, Joel Brynielsson, Henrik Artman, and Klas Wallenius. Information Awareness in Command and Control: Precision, Quality, Utility. In *Proceedings Third International Conference on Information Fusion (FUSION 2000)*, pages ThB1/25–32, Paris, July 2000.

[3] Henrik Artman and Mats Persson. Old practices - New technology: Observations of how established practices meet new technology. In Claes Sundin and Henrik Friman, editors, *ROLF 2010 – The Way Ahead and The First Step*, pages 126–143, 2000.

[4] Henrik Artman and Yvonne Wærn. Distributed Cognition in an Emergency Co-ordination Center. *Cognition, Technology & Work*, 1:237–246, 1999.

[5] James O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer-Verlag, New York, second edition, 1985.

[6] Berndt Brehmer. Dynamic Decision Making: Human Control of Complex Systems. *Acta Psychologica*, 81:211–241, 1992.

[7] Berndt Brehmer and Robert Allard. Dynamic Decision Making: The Effects of Task Complexity and Feedback Delay. In J. Rasmussen, B. Brehmer, and J. Leplat, editors, *Distributed Decision Making: Cognitive Models for Cooperative Work*, pages 171–184. John Wiley & Sons Ltd, 1991.

[8] Berndt Brehmer and Dietrich Dörner. Experiments With Computer-Simulated Microworlds: Escaping Both the Narrow Straits of the Laboratory and the Deep Blue Sea of the Field Study. In *Computers in Human Behavior*, volume 9, pages 171–184, 1993.

[9] Berndt Brehmer and Peter Svenmarck. Distributed Decision Making in Dynamic Environments: Time Scales and Architectures of Decision Making. In J. P. Caverni, M. Barhillel, F. H. Barron, and H. Jungermann, editors, *Contributions to Decision Making – I*. Elsevier Science B.V., 1995.

[10] Ward Edwards. Dynamic decision theory and probabilistic information processing. *Human Factors*, 4:59–73, 1962.

[11] Per-Olof Fjällström, Göran Neider, Mats Persson, Tore Risch, and Per Svensson. Architecture Principles for Information Superiority in Future Command and Control Systems (in Swedish). Technical Report FOA-R--00-01435-505--SE, Defence Research Establishment (Sweden), 2000.

[12] Rego Granlund. Web-Based Micro-World Simulation For Emergency Management Training. In *Proceedings International Conference on Web-based Modelling and Simulation*. Society for Computer Simulation International, San Fransisco, CA, 1999.

[13] HKV. RMA – A new foundation for defense forces development (in Swedish). Technical Report HKV 09 100:63046, Swedish National Defence, March 1999.

[14] Edwin T. Jaynes. *Probability Theory: The logic of Science*. Preprint: Washington University, 1996.

[15] Jan Kuylenstierna, Joacim Rydmark, and Tonie Fåhraeus. A commanders need for information (in Swedish). Technical report, Swedish National Defence College, October 1999.

[16] Mats Persson. Future Command and Control Systems: operational, organisational, and functional prerequisites. In *Proceedings Systemics and Informatics Research Seminar on AIH – Applied Informatics for Improvement of Human Life (SIMS'01)*, Östersund, January 2001.

[17] Howard Raiffa. *Decision Analysis: Introductory Lectures on Choices under Uncertainty*. Addison–Wesley, Reading, MA, 1968.

[18] Claes Sundin and Henrik Friman, editors. *ROLF 2010 – A Mobile Joint Command and Control Concept*. Swedish National Defence College, 1998.

[19] Klas Wallenius. Use of Modern Information Technology: WASP – A Common View of the Situation. In *Technet Europe 2000. 21st AFCEA Europe Symposium and Exposition*, Prague, October 2000.

[20] Rickard Westberg. Decision support for naval command and control systems. Master's thesis, Department of Computer and Systems Sciences, Royal Institute of Technology, Stockholm, Sweden, April 2001.