FFI RAPPORT

ARTIFICIAL INTELLIGENCE AND HUMAN BEHAVIOUR IN SIMULATIONS – Final Report from FFI-project 722 “Synthetic Decision Making”

DAHL, Fredrik A

FFI/RAPPORT-2000/04395

FFISYS/722/161.3

Approved
Kjeller 20 October 2000

Bent Erik Bakken
Director of Research

ARTIFICIAL INTELLIGENCE AND HUMAN BEHAVIOUR IN SIMULATIONS – Final Report from FFI-project 722 “Synthetic Decision Making”

DAHL, Fredrik A

FFI/RAPPORT-2000/04395

FORSVARETS FORSKNINGSINSTITUTT
Norwegian Defence Research Establishment
P O Box 25, NO-2027 Kjeller, Norway

FORSVARETS FORSKNINGSINSTITUTT (FFI)
Norwegian Defence Research Establishment

P O BOX 25
N0-2027 KJELLER, NORWAY

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE
(when data entered)

REPORT DOCUMENTATION PAGE

1) PUBL/REPORT NUMBER	2) SECURITY CLASSIFICATION	3) NUMBER OF PAGES
FFI/RAPPORT-2000/04395	UNCLASSIFIED	27
1a) PROJECT REFERENCE	**2a) DECLASSIFICATION/DOWNGRADING SCHEDULE**	
FFIS/722/161.3	-	

4) TITLE

ARTIFICIAL INTELLIGENCE AND HUMAN BEHAVIOUR IN SIMULATIONS –
Final Report from FFI-project 722 “Synthetic Decision Making”

5) NAMES OF AUTHOR(S) IN FULL (surname first)

DAHL, Fredrik Andreas

6) DISTRIBUTION STATEMENT

Approved for public release. Distribution unlimited. (Offentlig tilgjengelig)

7) INDEXING TERMS

IN ENGLISH:	IN NORWEGIAN:
a) Artificial intelligence	a) Kunstig intelligens
b) Simulation	b) Simulering
c) Games	c) Spill
d) Human decision making	d) Menneskelig beslutningsfatning
e) Reinforcement learning	e) Forsterkningslæring

THESAURUS REFERENCE:

8) ABSTRACT

The report describes the results achieved in the FFI-project 722 on synthetic decision making. It is observed how formalised combat simulations fit the game-theoretic framework of two-player zero-sum games. The games of Campaign, Operation Lucid and Operation Opaque are described. These games have been used for experiments with human decision making and artificial intelligence (AI). The human experiments indicate that the game-theoretic concept of randomisation in games of imperfect information fails to explain human decision making. Several AI techniques were utilised in the development of automatic agents playing our three games, including rule-based systems, fuzzy logic, genetic programming, neural nets and constraint satisfaction programming. The most significant contribution from the project was the development of reinforcement learning (including co-evolution) algorithms for games of imperfect information.

9) DATE	AUTHORIZED BY	POSITION
	This page only	
20 October 2000	Bent Erik Bakken	Director of Research

ISBN-82-464-0445-8

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE
(when data entered)

CONTENTS

ARTIFICIAL INTELLIGENCE AND HUMAN BEHAVIOUR IN SIMULATIONS – Final Report from FFI-project 722 "Synthetic Decision Making"

1 INTRODUCTION

Despite the end of the Cold War, there is still considerable interest in modelling of military conflict. With the rapid progress we have witnessed the last few decades in the state of the art of hardware and software, researchers' possibilities for creating accurate and detailed simulation models have increased significantly. However, we believe that researchers' understanding of human decision making has not been able to keep up with this development. Therefore the weak spot of military simulation may have shifted from representing the physical interaction of thousands of weapon systems, to representing human decision making.

For this reason FFI started the project with the purpose of improving our ability to simulate human decision making, for future use in military simulation models. Two different viewpoints were taken for this task. On the one hand, humans are superior to the state of the art of artificial intelligence (AI) in many areas. The project should therefore study different AI techniques, with the goal of applying, improving and combining them to achieve performance closer to that of humans in decision problems that are typical within military simulations. On the other hand, human decision making is far from perfect, and the project should also attempt to improve our understanding of human decision making in synthetic environments. The first of these goals was given the higher priority.

The report is structured as follows. Section 2 describes the approach taken in the project. In Section 3 a brief overview of AI is given. The modelling paradigm of two-player zero-sum games is explained in Section 4, together with performance measures for playing strategies in such games. In Section 5 the specific games studied by the project are described. Section 6 gives an overview of the different automatic agents developed for those games, while Section 7 describes the experiments performed with human learning. In Section 8 the main conclusion from the project are listed.

2 APPROACH

The approach taken was to work with three different games, designed specifically for the project, and use these to study AI techniques and human decision making.

2.1 Goals

The main goals of the project were to:

- Define three simplified models in the form of games, and implement the game rules in separate software modules. It should be easy to plug different decision modules into the rule modules.
- Perform experiments with human decision making on at least one of the games.
- Develop decision modules for each game, using a variety of AI techniques.
- Compare the applicability of different methods to our games.
- Draw conclusions about the usability of different methods on problems of varying kinds and sizes.

2.2 Why we study games

The term “game” may trigger very different associations in laymen, such as computer games in the tradition of Space Invaders, sports games like the Olympics, card games like black jack, board games like chess or dice games like craps. People tend to associate games with fun, rather than serious business like combat simulation. However, we use the term “game” in the strict mathematical sense of a decision process with two or more decision makers. A (mathematical) game is formalised with predefined rules specifying which decisions are allowed at any given point, as well as the players’ utilities of the different outcomes. Note that this concept of games includes *all* mathematical control problems, being games where only one player can make non-trivial decisions. In order to study human or artificial intelligence in a simulated combat situation, it is preferable to specify which decisions can be made, and what the decision makers are trying to accomplish, in a formal way. This is exactly the formalism offered by game theory.

Why then, did we choose to design new games, instead of studying decision making within “sharp” simulation models already in use for the analysis of military operations? This is mostly for practical reasons. Our intention was to try out a wide variety of different AI techniques. “Sharp” simulation models tend to be very complex, which would make it unlikely that we would be able to develop many different AI agents within the limitations of the project. By developing our own games as testing ground, we were able to control their complexities, not only by making them less complex than regular simulation models, but also by making them complex along different complexity dimensions. This enabled us to compare how different AI techniques apply do different kinds of problems.

3 ARTIFICIAL INTELLIGENCE

AI is a vast and quickly developing field, without clear boundaries to other mathematical and computer science disciplines. Nevertheless we now attempt to classify different AI methods. We emphasise that our classification is relative to our area of application and our experience. For a general introduction to AI we refer to (16).

3.1 Agents

A relatively new perspective on AI is that of (intelligent) agents. An agent can be viewed as a self-contained software module that communicates with its environment through some protocol. What makes an agent different from any modular piece of software is mainly one of interpretation. An agent is considered an active subject, which attempts to accomplish some objective by acting on and communicating with its environment.

3.2 Knowledge representation

An established view is that an AI system must contain some element of domain knowledge, represented in a data structure. In the very early days of the field, attempts were made to develop general problem solvers, which used virtually no domain knowledge. This turned out to be almost as hopeless as the late philosopher Descartes' attempt to deduce everything in the world from the fact that he was able to think.

We distinguish between what we call explicit knowledge representation schemes, which can make sense to humans directly, and implicit ones that do not. Rules and facts of expert systems are typical examples of explicit knowledge representation, while the internal weights of a neural net are implicit knowledge. There is a large grey area of knowledge representations that a human can read, although they are not very intuitive, such as the code of a badly structured computer program.

3.3 Knowledge processing

Most AI applications use some kind of processing to deduce new knowledge from the knowledge represented directly. As long as there are several ways of deducing knowledge, the processing can be viewed as a search in some space. In a rule based system, the search is given by the different orders in which one can apply the rules to the set of facts. In a chess-playing program, the search space is given by the different sequences of moves that can be played.

The processing of knowledge typically incorporates meta-knowledge, which directs the search by identifying which pieces of knowledge are likely to be relevant.

3.4 Knowledge acquisition

Given that an AI system must contain domain knowledge, the problem of knowledge acquisition must be addressed.

The "classical" approach to knowledge acquisition is to apply an explicit representation, and identify chunks of knowledge as if-then rules by interviewing an expert. We denote this approach *knowledge intensive*.

Machine learning (ML) is a relatively recent area of AI that offers alternatives to knowledge intensive methods of knowledge acquisition. The most common application of ML relies on a database of some sort, from which the learning algorithm extracts knowledge. This process is

sometimes referred to as *data mining*. In our game-playing application we refer to this approach as *imitation learning*, because our database consists of logs of well-played games, and our learning agents imitate the decisions made there.

Because a game is a closed world, it is possible to design ML agents that learn from game-playing experience. This is referred to as *reinforcement learning*, because behaviour which leads to favourable outcomes are reinforced (applied more often in the future). When it works, reinforcement learning is very attractive, because it enables agents to learn "by themselves".

3.5 Artificial intelligence and games

An overview of AI techniques that are useful for games is given in (9). Historically games have always been important testing-grounds for AI. However, most of the effort has gone into deterministic games of perfect information like chess and checkers (16). For these games massive search has proven more important than advanced knowledge representation. Although it was a great success for AI when Deep Blue defeated Kasparov (13), it was disappointing that this was achieved mainly with brute force calculations. Reinforcement learning has been successful for checkers (17) and backgammon (19). These applications also benefited from search (added after training).

For games of imperfect information (see the next section) there has been done relatively little work in the AI community (12). The survey on game-playing ML applications found in (10) indicates that the bias towards games of perfect information is even greater within the ML-community.

4 TWO-PLAYER ZERO-SUM GAMES

We will now give a brief explanation of some key terms of game theory, and explain how combat simulation models relate to two-player zero-sum games. We will also give performance measures for evaluating agents playing two-player zero-sum games.

4.1 Games and combat simulation

A *game*, in the mathematical sense, is a decision problem with two or more decision makers, called *players*, who all attempt to maximise their outcome. A combat situation, real or simulated, fits this formalism well. In most combat simulation models there are two opposing parties, with no common interest. In game-theoretic terms this means that most combat models can be viewed as *two-player zero-sum games*. "Zero-sum" means that one side's victory is equivalent to the other side's defeat, so that they have no incentive to co-operate.

The formalism of two-player zero-sum games is a flexible mathematical framework. It can represent sequences of predefined events or decisions made by the players, or uncertainty, in the form of random events. Two-player zero-sum games can also represent strategic uncertainty, by the use of imperfect information. This means that the players may have different information about the state of the game, or that a player's decision is not exposed to

the opponent until a later stage of the game. In games with imperfect information, bluff and deception may be important, to keep the opponent uninformed about the true state of the game.

By the famous minimax theorem (15), any (finite) two-player zero-sum game has a numeric value. For both players there exist behavioural patterns, called minimax solution strategies, which guarantee their expected outcomes to be at least as good as the value. If both sides apply minimax solution strategies, the game is in equilibrium, as neither side can improve his expected outcome by changing his strategy. For a classical treatment of game theory, we refer to (15).

4.2 Performance measures

One of the main goals for the project has been to develop computer programs that play our games well. Although game theory gives the strong solution concept of minimax play, and also defines a game's value, it does not provide evaluation measures for sub-optimal players.

We have therefore developed a set of evaluation criteria (10). In doing this, we realised that there exists no completely consistent evaluation measure. Ideally we would prefer a performance measure which evaluates an agent's playing strength, such that a weaker agent loses against a stronger agent, at least in the long run. However, such measures do not in general exist, because agents may beat each other in circle. Instead we have focused on developing measures that conform to game theory, in the sense that agents applying minimax solutions receive the maximum score.

Our strictest evaluation criterion is denoted by Geq (equity against globally optimising opponent). The Geq of an agent is defined as its average outcome against the opponent that is the most effective against it (the best response strategy, in game-theoretic terms). The maximum Geq score is equal to the game's value, and an agent receives this if and only it applies a minimax strategy.

A milder evaluation measure is denoted Peq (equity against perfect opponent). The Peq of an agent gives the agent's average outcome against an opponent that applies a given minimax strategy. The maximum Peq score is also equal to the game's value. Maximum Peq performance is necessary, but not sufficient, for minimax play.

5 GAMES STUDIED BY THE PROJECT

In this chapter we give a brief introduction to the games that have been defined and studied in the project. A more thorough explanation of game rules and mathematical properties can be found in (6). All three games are two-player and zero-sum.

5.1 Campaign

Campaign is a simplified air campaign model, where the two sides allocate their air combat resources between the three roles of defence, profit and attack. The rules are deterministic, so

that if the players act deterministically, the course of the game is determined. The allocations are performed in a sequence of consecutive stages, and at each stage the players choose their allocations simultaneously. This simultaneity implies that the game features imperfect information. In games with imperfect information, optimal play (in the minimax sense) may require randomised actions from the players, and this is the case with Campaign. The goal of the game is a combination of taking profit, destroying enemy resources and preserving friendly ones.

Before each stage both players are informed about the current game state and the opponent's previous action. This property implies that there exist minimax strategies that only depend on the current game state, disregarding the sequence of actions that produced it. This in turn makes it possible to decompose Campaign into a sequence of related matrix games. In (6) we explain how this property enables us to calculate a minimax solution, using a computer.

The class of two-player zero-sum games with sequences of simultaneous decisions and perfect information prior to each stage has been studied by game theorists, and is denoted Markov games (14). Our strategy of decomposing the game to a collection of interrelated matrix games is standard for solving such games.

It should be noted that we developed Campaign as a Markov game only to be able to calculate its solution, which gave us a point of reference for our AI-based agents. Ideally we would have defined Campaign without the Markov property, to make it more challenging, and therefore we are more interested in AI methods that do not utilise the Markov property.

5.2 Operation Lucid

Operation Lucid is a simplified land-combat model. The combat area is modelled as a 5x5 rectangular grid with some edges removed. Each of the two armies is represented as a collection of units, which move along edges. Blue's goal is to break through Red's defence, and Red's goal is to prevent this. The game models randomness in combat outcomes, randomness in movement, the effect of force concentration, the advantage of defensive posture and supply lines, in a simple way.

In Operation Lucid there is perfect information, which means that both sides have the same information about the game state, at all times. Therefore optimal play does not require random actions. However, the state space of the game is extremely large, and well above the computational limits of present-day computers. In fact, even the calculation of the set of legal actions for a player is infeasible for some game states. This is because a player is allowed to move all units independently of each other.

5.3 Operation Opaque

Operation Opaque is equivalent to Operation Lucid, except for the information perfection. In Operation Opaque a player only receives information about nodes where he has units present, and neighbouring nodes. Therefore, the game features imperfect information, and optimal

strategies most likely include random actions (although we are not able to prove it, because the game is too complex to solve). Note that Operation Opaque is not a Markov game, as actions taken by a player may not be revealed until a much later stage of the game.

5.4 Problem dimensions

A purpose in defining our games has been to span as many dimensions of problem complexity in combat simulation as possible, and relate the usefulness of different AI methods to these dimensions. We consider the dimensions of *information complexity*, *environment randomness* and *strategic uncertainty*, to be explained below. We do not give formal definitions of these concepts, but their meaning should hopefully be clear.

5.4.1 Information complexity

By information complexity we mean the amount of information that an agent must consider, including the game rules, combined with the number of actions that a player can choose from (the game's branching factor). Campaign has a low information complexity, as the game state is specified by only four parameters, the rules are simple, and there are no more than 21 legal actions. Operation Lucid clearly has far more complex information, as the location of all units may affect the optimal strategy, and the number of legal actions can be extremely high. Note that in Campaign and Operation Lucid the states visited earlier in a game do not contribute to the information complexity, because the current game state contains all relevant information.

It is less obvious how Operation Opaque relates to Operation Lucid. In the early stage of a game a player will have little information about the location of the opponent's units, and therefore low information complexity. Later in the game, however, a player should consider not only the present information about the location of the opponent, but also observations made earlier, as they may give a clue about the current location of the opponent's units. Summed over the course of a game, the latter problem of fusing old and new information appears to be more important than the reduction in early-game information, making Operation Opaque more complex information-wise than Operation Lucid.

5.4.2 Environment randomness

A complexity dimension that is an aspect of uncertainty is that of environment randomness. By this we mean the presence of random events that neither player can predict or affect. While Campaign has no such uncertainty, both versions of Operation have, making them more complex with respect to environment randomness. The environmental randomness of Operation Lucid and Opaque are identical.

5.4.3 Strategic uncertainty

Strategic uncertainty is another complexity dimension related to information, referring to uncertainty about the opponent's behaviour. Campaign has some level of strategic uncertainty, as the main problem in this game is to predict which action the opponent is taking at a given stage. Yet, we evaluate the strategic uncertainty only as medium level, as the Markov game property implies that there is uncertainty only about the current action taken by the opponent.

In Operation Lucid there is no strategic uncertainty, at least in theory, as there is perfect information: A player does not have to commit to a plan beforehand, and each action is revealed to the opponent immediately. In practice a player (human or automatic) may have decided on a given plan early on, and stick to it even if it is not optimal, in which case there is some element of strategic uncertainty in guessing the opponent's plan. Against a perfectly rational opponent, however, there is no point in guessing his plan, because he is free to change it whenever that is profitable.

Operation Opaque, on the other hand, has a high degree of strategic uncertainty. If Blue chooses to attack in one area of the board, he will be committed to this choice for some time, and Red may not be aware of Blue's choices until several turns later. Because an action may not be exposed until several turns later, the strategic uncertainty of Operation Opaque concerns not only the current, but also the previous, actions taken by the opponent. We therefore consider Operation Opaque as more complex than Campaign with respect to strategic uncertainty.

5.4.4 Comparison

We summarise our evaluations of the complexities for our three games in Table 5.1. The labels "Low", "Medium" and "High" are relative quantities interpreted so as to distinguish between the games.

	Campaign	Operation L	Operation O
Information complexity	Low	Medium	High
Randomness	No	High	High
Strategic uncertainty	Medium	Low	High

Table 5.1 Comparison of the complexity dimensions of games.

The complexity of Operation Opaque dominates those of the two other games. Note that although Operation Lucid scores higher than Campaign in most dimensions, Campaign has the higher strategic complexity.

6 AUTOMATIC AGENTS

In this chapter we give a brief overview of the methods used in our development of automatic agents for Campaign and Operation.

6.1 Campaign

Several different AI techniques have been applied to construct automatic agents for Campaign. We give only the main points, and refer to (5) for details.

6.1.1 General agent design

The dominant complexity dimension of Campaign is that of strategic uncertainty. To handle this kind of uncertainty in a rational way, game theory demands that an agent must behave randomly, in order to be unpredictable. The general agent design shared by (virtually) all of our Campaign agents relies on evaluation of candidate actions. For a given game state the agent evaluates the playability of each legal action, and then draws a random action using these playability evaluations as probability weights. The core element of this design is a function that takes a game state and a candidate action as input, and produces a probability weight as output.

6.1.2 Machine learning paradigms

Although it was not planned beforehand, all the agents implemented utilised some kind of machine learning method. (Note, however, that we use this term in a somewhat weak sense.) We have applied the learning paradigms of optimisation, imitation and reinforcement learning.

By optimisation we mean methods that take our performance measures Geq and Peq as feedback, and tune the agent's probability-weight function directly. These methods are applicable only because Campaign has a low complexity, and serve mainly to establish upper limits of the performance of the different agent designs.

With imitation learning we use a set of perfectly played games, sampled from the calculated solution, and tune our agents to imitate the actions seen in these. In this form, imitation learning requires access to the game solution, but the paradigm would also be applicable if one has access to samples of high-quality games played by humans. For an application of this paradigm to the Asian game of go, see (4).

Reinforcement learning may be the most interesting approach to agent training, as it relies only on game outcomes as feedback. Reinforcement learning therefore has the potential to discover playing strategies on its own, by experimenting with the game. The most celebrated reinforcement learning application is Tesauro's backgammon program TD-Gammon (19), which has reached a very high playing level, challenging the best human players. TD-learning is not applicable to Campaign due to the strategic uncertainty, but the project has succeeded in developing different reinforcement learning methods that work for games with imperfect information (3, 7 and 11).

Under a learning paradigm, an update mechanism must be used to adjust an agent's probability-weight function according to feedback, be it derived from the performance measure, from imitation or from game outcome. We have applied update mechanisms from evolutionary algorithms and gradient search.

6.1.3 Specific agent designs

The agents were implemented using the techniques of rule-based systems, fuzzy logic, genetic programming and neural nets. It is beyond the scope of this report to explain the algorithmic details, so we will only give a brief explanation of the different methods.

A rule-based system consists of a collection of facts and a collection of if-then rules, together with a mechanism for deriving new facts by applying rules to the current set of facts. Rule-based systems may include certainty factors associated with facts and rules, but there appears to be no universally sound way of handling them. In the 1980's rule-based systems (with extensions) were often referred to as expert systems, and this was close to synonymous with AI at that time. Expert systems did not quite live up to researchers' expectations, and is currently somewhat "out of fashion", although the technique can be useful for many different tasks. In our rule-based agent the initial collection of facts is a game state and a suggested action. Our rule base represents a common sense description of good play, and the inference engine attempts to deduce from them that the action is playable. Certainty factors are used in a pragmatic way, and machine learning techniques are used to tune them. Our rule-based agent was developed using the framework ExperTalk (2).

In a fuzzy logic system there is also a set of if-then rules, but uncertainty is handled in a more systematic way. Our fuzzy logic agent consists of two fuzzy rule bases: a planning base and a course-of-action base. For a given game state the planning base is used to derive a suggested "average action". Then the playability of a legal action is computed, using the course-of-action base, by calculating how compatible the action is with the average allocation.

In genetic programming, evolutionary algorithms are applied to parse trees, which are graphical representations of calculation procedures. The nodes of the tree represent input values or calculation of values, which may be logical (true/false) or numerical. The edges combine values in operations such as logical ones (and, or, if-then, =, <, >) or numerical ones (+, -, *, /). When a parse tree evaluates a given combination of game state and suggested action, the attributes of the state and action are inserted in input nodes of the tree. Then the values of all nodes are computed, and the value of the tree's root is taken as output. Because neither the shape, nor size, nor content of the parse tree is predefined, only evolutionary algorithms are applicable to them.

Neural nets can be viewed as synthetic brain cells, neurons, trained to solve some task, but we prefer to see them as flexible non-linear functions. The "knowledge" of a neural net lies in the weights associated to connections between neurons. Our simplest neural net agent conforms directly to our general agent design, taking the state and suggested action as input, and giving probability-weight as output.

A different neural net design has also been tested, where the net only takes a game state as input. This net estimates the expected outcome for a given starting state of the game. The state-evaluating function represented by the net must be combined with a matrix-game algorithm to

produce action probabilities. This design exploits the Markov property of Campaign, and therefore cannot be applied to non-Markov games.

6.1.4 Results

The rule-based approach was by far the easiest one to implement, and the performance of this agent was relatively good. The advantage of having a system that can be explained to a human in a simple way is important. The fuzzy logic agent required more work than the rule-based agent, while not producing stronger results. However, one should not draw too firm conclusions from a single test game. Like the rule-based agent, it has the advantage of being easy to comprehend by a human. The parse tree agent using genetic programming also achieved relatively low scores, relative to the amount of work invested in them. The neural net agents scored relatively well, in particular the one that utilised Campaign's Markov property.

6.2 Operation Lucid

In this section, the work done on Operation Lucid is described. Again, we give only the main points, and refer to (18) for details.

6.2.1 Variance reduction

To evaluate the relative merits of agents playing Operation Lucid we use Monte Carlo (MC) simulations. This essentially means that we run a sequence of independent pseudo-random simulations of the game, and take the average output as an estimate of the expected outcome. A problem with this method is that the random error is inversely proportional to the square root of the number of simulation runs, which means that one must increase the number of simulations by a factor 100 to reduce the error by a factor 10. We are therefore interested in methods that can reduce the variance of the estimate.

We have implemented a variance reduction technique known as control variates, which is based on the idea of modifying the actual outcome of a game by subtracting an estimate of Blue's "accumulated luck". This estimator – a control variate – is constructed in a way that guarantees it to have expected value 0, which ensures that the modified output is an unbiased estimate of the actual outcome. The effect of the variance reduction depends on the agents that play the game, but in most of our cases the method reduces the variance by a factor approximately equal to 4.0.

6.2.2 General design

Because Operation Lucid features perfect information, perfect play does not require random actions. It may still be a good idea to introduce some randomness in an agent, if it should play against an opponent that has the opportunity to seek out its weaknesses. Nevertheless our general design is deterministic, and randomisation is added to it in an ad hoc manner.

The most obvious design of an agent playing a game with perfect information is that of a state evaluator, which is a function taking the game state as input, and giving expected outcome as output. Such an evaluator might then be combined with lookahead, such as alpha-beta or

expectimax search, which investigates the consequences of different sequences of actions for the players. This is essentially how commercial software for chess and backgammon works. However, this design is not feasible in itself for Operation Lucid, because the branching factor of the game tree is far too high.

We therefore apply a hybrid design, in which one module is given the task of identifying a reasonably small set of candidate actions, and another module chooses between these, possibly using a state evaluator function.

6.2.3 Construction of reasonable candidate actions

For the task of constructing sets of reasonable candidate actions, the technique of constraint satisfaction programming (CSP) was used. In a CSP system, the problem is formulated by a set of numeric variables and a set of constraints. The constraints represent the knowledge of the system, in much the same way as the rules of a rule-based system do. The inference process of the CSP system consists of a search for value assignments for the variables, such that all the constraints are satisfied.

In our initial design we implemented what we believed to be general rules for reasonable play, with constrictions like “do not spread your units over more than *n* nodes” and “do not move forward along more than *m* lines”. To make these rules work properly, we added some meta-reasoning outside of the CSP module to find appropriate values of parameters like *n* and *m*. In some cases the meta-reasoning would include a trial and error process of testing different parameter values, until the set of candidates was of reasonable size.

6.2.4 Directed search for candidate actions

The approach of formulating general constraints describing reasonable play worked quite well, together with action-selecting methods. Under this design, most of the agent’s knowledge is implemented in the module that selects the action. From experience we realised that it might be advantageous to implement more knowledge into the candidate selection process, so that the set of candidate actions could be constructed with more specific intentions than reasonable play.

We therefore developed an “area language” (ALA), with a syntax that enables a separate module to describe the desired set of candidate actions to the CSP module in more detail. The ALA syntax allows a set of area constraints, where each area constraint specifies a set of nodes and a number. The interpretation of a constraint $\left(\{n_1, n_2, n_3, ...\}, m\right)$ is that there should be at least a total of *m* units inside the nodes $\left(n_1, n_2, n_3, ...\right)$.

6.2.5 Neural net agent

In our neural net agent, a neural net is used for evaluating game states. The net takes an encoding of the game state as input, and produces an output interpreted as the expected outcome of the game. The neural net was trained by the reinforcement learning algorithm of TD-learning. In the learning phase the CSP module was used with the general constraints set for reasonable play.

Although the neural net agent was quite successful, there were aspects of the game that it was unable to learn, such as the utility of maintaining unbroken supply lines. After the training process we therefore implemented a separate module with the purpose of guiding the agent towards better play. This module communicates with the CSP module using ALA, and successfully rectified some of the flaws in the agent.

6.2.6 Fuzzy logic agent

A fuzzy logic agent was developed using the same general design as for Campaign. A fuzzy rule base, called the planning base, is used to derive an approximate description of a desired game state in ALA-syntax. The CSP module then generates a set of candidate actions from the ALA-constraints. A fuzzy course-of-action rule base, which evaluates the candidates, is used to choose an action.

Within the fuzzy rule bases there is a set of adjustable parameters, and optimisation techniques were used to search the space of parameter value assignments. In this optimisation process a given benchmark opponent was used. Despite the use of variance reduction, the optimisation was very computationally expensive, due to the randomness of the game outcomes.

6.2.7 Ad hoc agent designs

Because Operation Lucid is too complex to be solved on a present-day computer, we were not able to calculate the true performance values of our agents directly. We therefore implemented a set of benchmark agents to be used as reference opponents. These agents are mostly very simple, with strategies like “always move forwards”. They work by constructing an action directly, instead of having the CSP module generating a set of candidate actions to choose from, so they do not follow our general agent design.

Some of these agents were more advanced, and utilised rule-based techniques. The rules would typically state tactical advice like “Do not move into a node where the opponent has more units than you”. These agents also included graph calculations to identify likely attacking routes for Blue.

6.2.8 Results

The ad hoc agents using graph calculations and rule based systems were rather successful compared to the limited effort invested in them. However, they relatively quickly reached a point where it was difficult to improve them further.

The neural net-based agent was successful, in particular after the ALA-based correction module was added. This combination of a CSP module utilising human domain knowledge, and a neural net-based black box module trained from the agent’s own experience, is strong.

The fuzzy logic agent was less effective, according to our tests against benchmark opponents, but has the advantage of a transparent design with knowledge that can be interpreted by a human.

6.3 Operation Opaque

Several of the agents developed for Operation Lucid were modified for handling the imperfect information of Operation Opaque.

6.3.1 Randomisation

Because Operation Opaque features imperfect information, agents should apply random actions to keep the opponent uninformed about the true game state. This was mainly achieved by adding some randomising strategy outside of the Operation Lucid-agents' reasoning. For the Blue side this approach appeared to work quite well: A Blue agent that chooses an attacking line randomly, and sticks to it until the endgame, utilises Red's lack of information rather successfully. A similar Red strategy of randomly choosing an area for defence is too risky, as the loss when Blue attacks in a different area outweighs the gain when he does not.

6.3.2 Intelligence module

To improve our Red agents' ability to defend, we saw the need for a module handling intelligence (in the military sense of the word). Although it seemed less crucial, Blue would also benefit from intelligence, particularly in the endgame. The intelligence module should draw conclusions about the location of the opponent's units from the observations made earlier in the game.

An intelligence module was defined, although not fully implemented, using CSP. The module would take input in the ALA-syntax. In this context the interpretation of an ALA-constraint $\left(\{n_1, n_2, n_3, ...\}, m\right)$ is as a query about whether the opponent may have at least *m* units in the area $\{n_1, n_2, n_3\}$. The history of observations in the game is included as constraints on the locations of the opponent's units.

6.4 Discussion

A general observation is that more specialised methods appear to work better. The most extreme example of this is the neural net design that utilises the Markov property of Campaign, which significantly outperformed the other agents. In the other end of the spectrum, the genetic programming Campaign agent learned rather slowly, and did not reach a very high performance relatively to the CPU time spent for training. This can be explained by the fact that evolutionary algorithms are a class of blind search algorithms that utilise no differentiability or other problem structure, and can therefore be applied to any problem.

A very clear result from the work with Campaign that it is far easier to achieve a high Peq than Geq performance. It is therefore far simpler to perform well against an opponent that plays the game's solution, than it is against an opponent that exploits your weaknesses.

Ideally we would like to discuss the dimensions of representation, processing and acquisition of knowledge separately, and relate them to our different game complexity dimensions. This can not easily be done, because a given representation restricts the possible processing and acquisition strategies available. For the rest of this section we focus on the knowledge acquisition dimension, discussing reinforcement learning, imitation learning and knowledge intensive methods.

However, we can make an interesting observation regarding knowledge processing. The reader may have observed that none of our agents apply so-called lookahead strategies like alpha-beta search. This may be a surprise, because chess has been the most popular arena for AI game-applications historically, and chess programs tend to rely almost entirely on lookahead search. For Campaign and Operation Opaque the explanation to this is that they feature imperfect information, which makes lookahead search meaningless. Lookahead search is in theory applicable to Operation Lucid, but the branching factor of this game is so enormous that searching even a single move (ply) beyond the immediate move is out of the question. Our experience therefore supports the view that although alpha-beta search is very successful for chess, it is not very relevant for games that are closer to representing problems of the “real world”.

6.4.1 Reinforcement learning

A game is a closed world with formalised rules, which makes it possible for an agent to learn from experience. This learning paradigm is called reinforcement learning. Note that we include evolutionary algorithms as reinforcement learning as long as they learn only from game outcomes. Reinforcement learning is an attractive paradigm for training agents in two-player zero-sum games, because it enables agents to develop strategies without the use of an external source of knowledge. In isolation this paradigm scales badly with problem size, in the sense that it quickly fails when the information complexity increases. With larger problems like Operation Lucid, one should therefore combine reinforcement learning with complexity-reducing techniques, such as CSP.

Reinforcement learning scales very favourably with environment randomness. In fact, randomness in the game rules is often advantageous, as it forces the agent to explore a larger portion of the game’s state space, making it more likely that good strategies will be discovered.

Prior to the synthetic decision making project, one would say that reinforcement learning scales extremely badly with strategic uncertainty, as there were virtually no known reinforcement learning algorithms for games with imperfect information. However, algorithms developed by the project have changed this.

6.4.2 Imitation learning

The general trend with Campaign is that reinforcement learning worked better than imitation learning. This may sound surprising, as imitating the behaviour of a perfect teacher appears promising. We believe that this can be explained by the fact that reinforcement learning uses

game outcomes as feedback, while imitation does not. If the imitating agent is unable to reproduce the behaviour of the teacher completely, it will have to make compromises. Logs of expert play do not contain information about the relative importance of the decisions made, and therefore the imitating agent is likely to make bad compromises. One can say that imitation learning is syntax without semantics, because it does not incorporate the consequence or meaning of an action.

However, experience with the game of go (4) indicates that imitation learning can produce modules that are useful for identifying playable candidate moves, as a filter for other more computationally expensive methods.

Imitation learning is likely to scale relatively badly with information complexity. In the go application, this was controlled by feeding the learning module only partial information about the game state. This was reasonable because the module was only intended for producing partial conclusions. Imitation learning only learns to identify correlation between game state information and possible actions. It is therefore not directly affected by the semantic complexity dimensions of environment randomness and strategic uncertainty.

6.4.3 Knowledge intensive methods

By knowledge intensive methods we mean methods that formally represent human domain knowledge. In cases where one has access to human domain knowledge, rule-based systems appear to be a cost-effective way of implementing agents of a reasonable quality. Fuzzy rule-based systems are closely related, and treat uncertainty in a more systematic way. The variables and constraints of our CSP module is also a way of encoding human domain knowledge, which has proven very useful.

From our experience with Campaign and Operation Lucid, knowledge intensive methods appear to scale relatively favourably with information complexity. The same is likely to be true for the other complexity dimensions, unless the game is so complex that high quality human domain knowledge cannot be acquired.

7 HUMAN EXPERIMENTS

Experiments with human decision making were performed with Campaign. The game was played with only three stages, as opposed to five, which was used for the automatic agents. The experiments are thoroughly documented in (1).

The purpose of the experiments was to test the correspondence between the human concept forming, decision making and learning on one side, and the predictions of game theory on the other. Experiments with human decision making have been published earlier (8), some of which support the hypothesis that human decision making tends towards the predictions of the theory, even in games that require randomisation. However, these studies were performed on very simple games, and we intended to test if these conclusions remain valid for games as

complex as Campaign. Our ambition of sampling the conceptual development of the subjects through playing experience also separates our study from previously published ones.

Our hypotheses were that after playing a sequence of Campaign games the subjects would:

- move towards applying strategies predicted by game theory
- develop elementary concepts of game theory

These general hypotheses were broken down to more concrete and testable ones.

7.1 Experimental design

The subjects were recruited on a voluntary basis, and money prizes were used to create extra incentives for the players. The subjects were paired randomly one-on-one, and played two sequences of 25 games each. Before the first session, the subjects filled in questionnaires that were designed to measure their game-theoretic concepts. After the second session, the subjects filled in identical copies of these forms.

The games played by the subjects were logged, and for each subject, mathematical tests of the playing strategies were performed. We thereby compared the correspondence with game theory in the first and second 25-game sequence for each subject.

7.2 Findings

The overall findings of the experiments were negative. The subjects did not tend towards thinking about the games in the terms used by game theory. This was particularly true for the concept of randomisation. In game theory, randomisation is a purely defensive measure taken to ensure that the opponent will not be able to figure out the player's strategy. To be in line with game theory, one should therefore act randomly if, and only if, the opponent is considered "smarter" than oneself. The questionnaires were carefully designed to measure this. The subjects apparently moved in the opposite direction, evaluating random actions as relatively more useful against a less smart opponent after the game-playing experience. The absolute values were also largely negative, as the subjects on average evaluated random actions as almost equally useful against more and less capable opponents. This surprised us, because there really is very little reason to behave randomly against a weaker opponent. Our conclusion must be that the concept of randomisation as a defensive measure against a clever opponent is unnatural for humans. If our pre-stated hypothesis had been that there would be "negative learning", the data would have supported this conclusion with a significance level of 0.05. However, our pre-stated hypothesis was that of positive learning, so this significance level should not be taken literally.

In terms of actual randomisation the subjects also showed negative development on average, from the first to the second sequence of 25 games. This was measured by sampling the players' distributions of opening moves, which tended towards being more vulnerable (predictable) in the last 25-game sequence. These results were not statistically significant.

Positive learning was found only in the subjects' ability to find playable moves. This means that although the subjects drifted towards more vulnerable strategy profiles, which is contrary to game theory, the building blocks of their strategies improved. In games of perfect information, like chess, finding playable moves is sufficient for playing according to game theory, because no randomisation is then required. We therefore conclude that it is the presence of imperfect information, which implies the need for random actions, which is the problem for the subjects.

8 CONCLUSION

On the whole, the project has been a success: Simplified combat models in the form of two-player zero-sum games have been defined and implemented in an object-oriented way. Network software for connecting human and automatic player agents has been developed. Human decision making has been studied with experiments, establishing limitations of game theory's ability to explain human behaviour. Several techniques in artificial intelligence and machine learning have been applied successfully in the development of automatic players, and the project has succeeded in developing new algorithms.

Our main conclusions concerning simulations, games, artificial intelligence and human decision making are the following:

- Two-player zero-sum games are a natural paradigm for modelling combat
- Game theory does not succeed in describing human decision making, nor learning, in complex games of imperfect information
- When human knowledge is available, knowledge intensive methods like rule-based systems, fuzzy logic systems or constraint satisfaction programming systems will often be cost-effective for developing automatic agents of reasonable quality
- It is much harder for an agent to succeed against an opponent that exploits the agent's weaknesses, than to succeed against an opponent applying the game-theoretic solution
- A game is a closed world, which makes reinforcement learning possible, and it appears successful for two-player zero-sum games
- With algorithms developed by the project, reinforcement learning (including co-evolution) is made possible also for games of imperfect information
- Neural nets work well with reinforcement learning, unless the information complexity (branching factor) is too high
- With high information complexity, reinforcement learning should be combined with a complexity-reducing technique, such as constraint satisfaction programming
- Evolutionary algorithms appear preferable mostly with agent representations that do not allow gradient search
- In general, algorithms that utilise specific properties of the given problem, tend to work better than those that do not

References

(1) Bakken B T, Dahl F A (2000): AN EMPIRICAL STUDY OF DECISION MAKING AND LEARNING IN A COMPLEX TWO-PERSON ZERO-SUM GAME WITH IMPERFECT INFORMATION, FFI/NOTAT-2000/03919

(2) Bergli J (1998): EXPERTALK FFI-VERSJON 1.0 - Teknisk dokumentasjon og brukerveiledning (in Norwegian), FFI/NOTAT-98/04152

(3) Dahl F A (1999): THE LAGGING ANCHOR ALGORITHM - Learning games with imperfect information by self play, FFI/NOTAT-99/01852 (submitted for possible publication in the journal *Machine Learning*)

(4) Dahl F A (1999): Honte, a Go-Playing Program Using Neural Nets, International Conference on Machine Learning 1999,Workshop on Machine Learning in Game Playing

(5) Dahl F A (2000): MACHINE LEARNING IN CAMPAIGN - Comparing machine learning techniques used for a two-player zero-sum game with imperfect information, FFI/RAPPORT-2000/04400

(6) Dahl F A, Halck O M (1998): THREE GAMES DESIGNED FOR THE STUDY OF HUMAN AND AUTOMATED DECISION MAKING – Definitions and properties of the games Campaign, Operation Lucid and Operation Opaque, FFI/RAPPORT-98/02799

(7) Dahl F A, Halck O M (2000): MINIMAX TD-LEARNING WITH NEURAL NETS IN A MARKOV GAME - A paper presented at the ECML-2000 conference, FFI/NOTAT-2000/01126

(8) Erev I, & Roth A E (1998): Predicting How People Play Games: Reinforcement Learning in Experimental Games with Unique, Mixed Strategy Equilibria, *The American economic review*, Volume 88, pp 848-881.

(9) Hagenson N (1997): ARTIFICIAL INTELLIGENCE TECHNIQUES WITH POTENTIAL FOR USE IN TWO PERSON ZERO SUM GAMES, FFI/RAPPORT-97/05291

(10) Halck O M, Dahl F A (1999): ON CLASSIFICATION OF GAMES AND EVALUATION OF PLAYERS - WITH SOME SWEEPING GENERALIZATIONS ABOUT THE LITERATURE - A paper presented at the ICML-99 Workshop on Machine Learning in Game Playing, FFI/NOTAT-99/04875

(11) Halck O M, Dahl F A (2000): ASYMMETRIC CO-EVOLUTION FOR IMPERFECT-INFORMATION ZERO-SUM GAMES - A paper presented at the ECML-2000 conference, FFI/NOTAT-2000/01124

(12) Koller D, & Pfeffer A (1997): Representation and solutions for game-theoretic problems, *Artificial Intelligence*, *94* (1), July 1997, 167-215.

(13) Korf R E (1997): Does Deep Blue Use Artificial Intelligence?, *ICCA Journal*, Vol 20, No 4, 1997

(14) Littman M L (1994): Markov Games as a Framework for Multi-Agent Reinforcement Learning, *Proceedings of the 11th International Conference on Machine Learning*, 157-163. Morgan Kaufmann, New Brunswick.

(15) Luce R D, & Raiffa H (1957): *Games and Decisions*, Wiley, New York.

(16) Russell S, Norvig P (1995): Artificial Intelligence: A Modern Approach, Prentice Hall, Upper Saddle River, New Jersey

(17) Samuel A L (1959): Some studies in machine learning using the game of checkers, IBM J Res. Develop. 210-229.

(18) Sendstad O J, Halck O M, Dahl F A, Braathen S (2000): DECISION MAKING IN SIMPLIFIED LAND COMBAT MODELS – On design and implementation of software modules playing the games of Operation Lucid and Operation Opaque, FFI/RAPPORT-2000/04403

(19) Tesauro G J (1992): Practical issues in temporal difference learning, *Machine Learning 8*, 257-277.

DISTRIBUTION LIST

FFISYS **Dato:** 20 oktober 2000

RAPPORTTYPE (KRYSS AV)	RAPPORT NR.	REFERANSE	RAPPORTENS DATO
X RAPP ☐ NOTAT ☐ RR	2000/04395	FFIS/722/161.3	20 oktober 2000
RAPPORTENS BESKYTTELSESGRAD		ANTALL EKS UTSTEDT	ANTALL SIDER
Unclassified		40	27
RAPPORTENS TITTEL		FORFATTER(E)	
ARTIFICIAL INTELLIGENCE AND HUMAN BEHAVIOUR IN SIMULATIONS – Final Report from FFI-project 722 "Synthetic Decision Making"		DAHL, Fredrik A	
FORDELING GODKJENT AV FORSKNINGSSJEF:		FORDELING GODKJENT AV AVDELINGSSJEF:	

EKSTERN FORDELING

ANTALL	EKS NR	TIL
1		Tekn dir R Fjellheim Computas AS Pb 444 1301 Sandvika
1		F aman J Frihagen Krigsskolen Pb 42 Linderud 0517 Oslo
1		Prof O Hallingstad UNIK 2027 Kjeller
1		Prof H R Jervell Inst for lingvistiske fag Universitetet i Oslo P A Munchs hus Pb 1102 Blindern 0317 Oslo
1		Prof T Lensberg Inst for samfunnsøkonomi Norges handelshøgskole Helleveien 30 5035 Bergen
1		KK J K Nyhus Forsvarets stabsskole Oslo mil/Akershus 0015 Oslo
		www.ffi.no

INTERN FORDELING

ANTALL	EKS NR	TIL
14		FFI-Bibl
1		Adm direktør/stabssjef
1		FFIE
4		FFISYS
1		FFIBM
1		R H Solstrand, FFISYS
1		B E Bakken, FFISYS
1		J E Torp, FFISYS
1		B T Bakken, FFISYS
1		S Braathen, FFISYS
5		F A Dahl, FFISYS
1		O M Halck, FFISYS
1		O J Sendstad, FFISYS
1		K A Veum, FFIE
		FFI-veven

FFI-K1 Retningslinjer for fordeling og forsendelse er gitt i Oraklet, Bind I, Bestemmelser om publikasjoner for Forsvarets forskningsinstitutt, pkt 2 og 5. Benytt ny side om nødvendig.