# Relevant Information as a Formalised Approach to Evaluate Game Mechanics

Christoph Salge
Department of Computer Science
University of Hertfordshire
Hatfield AL10 9AB
United Kingdom
Email: c.salge@herts.ac.uk

Tobias Mahlmann
IT University of Copenhagen
Center for Computer Games Research
2300 Copenhagen
Denmark
E-mail: tmah@itu.dk

***Abstract*—We present a new approach to use adaptive AI in the game design process to aid evaluating of game mechanics. During production, this is a crucial task to improve the player satisfaction with a game title. The problem with automated game evaluation via AI is the measurement of values that indicate the quality of the game mechanics. We apply the Information Theory based concept of "Relevant Information" to this problem and argue that there is a relation between enjoyment related game-play properties and Relevant Information. We also demonstrate, with a simple game implementation, how an adaptive AI can be used to approximate the Relevant Information, and how those measurable numerical values related to certain game design flaws.**

***Index Terms*—AI and Games, Information Theory, Game Development, Strategic Games**

## I. Introduction

### A. Motivation

The development of a current, state of the art, computer games has become a large financial enterprise, involving many people and serious investment of money. This necessitates the use of project management techniques, such as risk analysis and quality control for all the components of a game. But while certain aspects, such as graphics quality or code stability, are somewhat straight forward to test, the evaluation of the actual "game mechanics" of a computer game often poses a problem.

Not only is it difficult to actually define what makes good game mechanics, but the common way of testing those, the use of several human play testers, is cost intensive, slow, and mingles the quality assessment of the game mechanics with the quality assessment of several other aspects of the game. Furthermore, play testing requires the game to have passed through most of its initial development stages beforehand.

### B. Related Work

An alternative is the use of an adaptive AI that plays and evaluates the game mechanics. Already, there are several adaptive AI approaches, such as genetic algorithms [1], particle swarm optimization [2], reinforcement learning [3] and neural network s[4], [5], all aimed to create high performing AIs. But the problem for AI in game design, as Yannakakis [6] points out, is not to create a good AI, but one that is enjoyable to play against, and one that can be used to improve the game itself.

AI aided game balancing [7], [8] is a good step towards automated game improvement. A more general approach would be to have an AI play the game then evaluate how much "fun the AI had". One approach here [9] is the application of the theory of *Artificial Curiosity*. A more empirical way to approach this goal would be to model an AI after actual neurological and physiological data, to simulate the emotions of a real player [10]. Another approach would be to observe an adaptive AI as it plays the game and analyse the resulting strategies in regard to exploits and other game play flaws [11]. But again, this currently requires a human analysis of the strategies to evaluate how much fun they would be for a real human.

### C. Overview

In this paper we apply the Information Theory [12] based concept of "Relevant Information" [13], [14] to this problem. First, we will give a short introduction to Information Theory, explain "Mutual Information" and how to apply it to games to calculate their Relevant Information. Then we argue that Relevant Information is a numerical value that corresponds to enjoyment related game-play properties. We will then demonstrate, for a simple, turn based, strategy game, how Relevant Information can be approximated by gathering statistical data from an adaptive, genetic algorithm driven, neural network. This data is then used through several scenarios to detect game play flaws, and adapt the game mechanics.

## II. Information Theory

Information Theory [12], [15] can be used to study the properties of random variables. If a random variable $X$ can assume the states $x$, and $P(X = x)$ is the probability for $X$ to assume the specific state $x$, we can define a measure $H(X)$ called entropy as:

$$H(X) = -\sum_{x} P(X = x) \cdot log(P(X = x))$$

This is often described as the uncertainty about the outcome of $X$, or the average expected surprise, or else the information

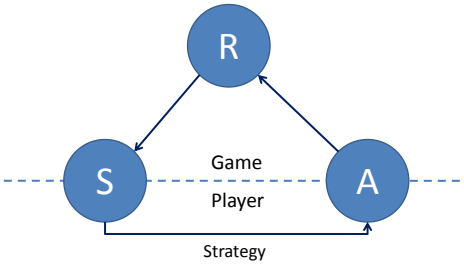


Fig. 1. A causal Bayesian network depicting the abstract model of player game interaction.

gained if one was to observe the state of $X$, without having prior knowledge about $X$. The entropy has a number of important properties. Among other, the *a priori* uncertainty (i.e. entropy) is larger if the outcomes are more evenly distributed than if the outcomes are more concentrated on a particular value in other words, concentrated values are easier to predict than if uniformly spread ones.

Consider two jointly distributed random variables, $X$ and $Y$; then we can calculate the conditional entropy of $X$ given a particular outcome $Y = y$ as:

$$H(X|Y=y) - \sum_x P(X=x|Y=y) \cdot log(P(X=x|Y=y))$$

This can also be generalized to the entropy of $X$, given $Y$ in general, averaged over all possible outcomes of $Y$:

$$\begin{aligned} H(X|Y) &= -\sum_y P(Y=y) \sum_x (P(X=x|Y=y) \\ &\quad \cdot log(P(X=x|Y=y)) \end{aligned}$$

This is the entropy of $X$ that remains if $Y$ is known. Consider here $H(X)$ and $H(X|Y)$, the entropy of $X$ before and after we learn the state of $Y$. Thus, their difference is the amount of information we can learn about $X$ by knowing $Y$. Subtracting one from the other, we get a value called *mutual information*:

$$I(X;Y) = H(X) - H(X|Y)$$

The mutual information is symmetrical [15] and measures, in bits, the amount of information one random variable contains about another (and vice versa, by symmetry). This quantity will be of central importance in the following argument.

### A. Game Player Interaction Model

In order to apply Information Theory to a specific game we model the interaction between the game and the player with a "Causal Bayesian Network" [16], as seen in Fig. 1. Since we are dealing with a computer game it should be no problem to assign discrete values to the parameters that describe the current state of the game. The actual causal Bayesian network now consists of compound random variables. $S$ and $A$ represent the user interface. $A$ corresponds to the action the player chooses, its state space are the available actions for the player. $S$ corresponds to the sensor input of the player. The state s of S is the state of the game world as shown to the player. $R$ represents the rest of the parameters that are needed to describe the current state of the game. The arrows denote the causal relations between those nodes as defined in [16]. This indicates that everything that influences a state of a variable either has a arrow pointing to that variable, or does so through another variable. The game thereby if fully defined by $A, S, R$, the causal relations from $A$ to $R$ and from $R$ to $S$. In addition we assume that there is a function that assigns a performance value (like winning or losing, or points) to $R$. The player, or the agent playing the game, determines the conditional probability $P(A|S)$, which we will call a strategy. This probability determines what actions are chosen depending on the sensor input of the player.

For our arguments we assume that the agent can see all of the world, and therefore $R$ equals $S$. This also makes agent memory, which would record information about the game state that cannot be observed every round, obsolete.

### B. Relevant Information

Relevant Information is a formalism based on mutual information. It measures how much information an agent needs to obtain from the environment to determine its optimal strategy. We will adapt its definition for the previously introduced game player interaction model. Are more general definition can be found in [13].

Given that the game has a utility function that determines the performance of the player in regard to the state of the game $R$ then there exists one, or several strategies $P(A|S)$, that result in the highest expected utility. We will call those strategies the optimal strategies.

Relevant Information is defined as the (minimal) amount of information the agent needs to acquire to be able to realise one of its optimal strategies. For a given strategy the amount of information that is needed about R to chose the state of $A$ can be measured by the mutual information $I(A;R)$, or $I(A;S)$ if we assume that the sensor state and the state of the environment are identical. The Relevant Information for the system is therefore defined as the mutual information of the optimal strategy with the lowest mutual information. See Fig. 2 for some examples. If $S$ and $R$ are not identical then we can still treat S as a subset of all the world information $R$, and then ask ourselves what the Relevant Information in that subset $S$ is. By extension, it is also possible to calculate the amount of information needed to obtain a certain payoff level, by calculating the minimum Relevant Information of all strategies that reach at least that level of expected payoff [14]. Since the strategy with the minimal mutual information is chosen from a set of strategies that gets smaller as performance increases it follows that an increase in the payoff level increases the Relevant Information or keeps it at least the same. So the function that relates Relevant Information and performance is monotonously increasing.

World 1

| (Pay Off) | State 1 | State 2 | State 3 | State 4 |
|---|---|---|---|---|
| Action 1 | 1 | 0 | 0 | 0 |
| Action 2 | 0 | 1 | 0 | 0 |
| Action 3 | 0 | 0 | 1 | 0 |
| Action 4 | 0 | 0 | 0 | 1 |

World 2

| (Pay Off) | State 1 | State 2 | State 3 | State 4 |
|---|---|---|---|---|
| Action 1 | 0 | 0 | 0 | 0 |
| Action 2 | 0 | 0 | 0 | 0 |
| Action 3 | 0 | 0 | 0 | 0 |
| Action 4 | 0 | 0 | 0 | 0 |

World 3

| (Pay Off) | State 1 | State 2 | State 3 | State 4 |
|---|---|---|---|---|
| Action 1 | 0 | 0 | 2 | 1 |
| Action 2 | 0 | 0 | 1 | 2 |
| Action 3 | 2 | 1 | 0 | 0 |
| Action 4 | 1 | 2 | 0 | 0 |

World 4

| (Pay Off) | State 1 | State 2 | State 3 | State 4 |
|---|---|---|---|---|
| Action 1 | 0 | 0 | 1 | 1 |
| Action 2 | 0 | 0 | 1 | 1 |
| Action 3 | 1 | 1 | 0 | 0 |
| Action 4 | 1 | 1 | 0 | 0 |

Fig. 2. This figure depicts the relation between action and utility, given a certain world state. In world 1 the Relevant Information can be computed as two bits. The optimal strategy chooses a different action for every state of the world; therefore it has to obtain two bits of information to determine what state the world is in. World 2 is an example of a scenario where there is no Relevant Information. The actions of the agent do not matter, and it needs to know nothing to make the best choice. World 4 has a Relevant Information of only 1 bit. One optimal strategy here would still be to choose a different action for every state, but there is also another strategy that just relies on finding out if the world is in the first two or last two states. Since we look for the minimum mutual information across all strategies, the agent only needs to obtain one bit. World 3 demonstrates that a suboptimal pay-off level can be reached with less Relevant Information. Using the same strategy as in World 4, the agents obtains an average payoff of 1.5, but if the agents would obtain two bits of information he could achieve an average pay-off of two

It should also be clear, that this function is defined for a given game mechanic and does not actually depend on the strategy chosen by the player, but on the strategy that has the lowest mutual information. Therefore, Relevant Information is a property of the game mechanic itself.

## III. Relevant Information and Player Satisfaction

In this chapter we argue how Relevant Information corresponds with game mechanical properties that foster or hinder enjoyment. Since it is questionable if fun can be ensured by some mathematical formalism, we are mainly looking at factors that hinder fun in games. Those factors are mainly taken from literature, such as [17], [18], or determined by commons sense. Some of them might be debatable, but this is beyond the scope of this paper, as is a psychological or sociological evaluation of those factors and their relation to game play fun.

What we want to demonstrate instead, is the relation of some measurable numerical values with some common game play properties, most of which should be avoided. The first measurement point we are looking at is the actual Relevant Information, the minimal mutual information of the set of optimal strategies.

### A. Inferior Choices

On possible design flaw is to offer the player an action that is never a good action to play in any given situation, so this option is never played by an optimal strategy. Since the Mutual

World 1

| (Pay Off) | State 1 | State 2 | State 3 | State 4 |
|---|---|---|---|---|
| Action 1 | 1 | 0 | 0 | 0 |
| Action 2 | 1 | 0 | 0 | 0 |
| Action 3 | 1 | 0 | 0 | 0 |
| Action 4 | 1 | 0 | 0 | 0 |

World 2

| (Pay Off) | State 1 | State 2 | State 3 | State 4 |
|---|---|---|---|---|
| Action 1 | 0 | 0 | 0 | 0 |
| Action 2 | 0 | 0 | 0 | 0 |
| Action 3 | 0 | 0 | 0 | 0 |
| Action 4 | 0 | 0 | 0 | 0 |

Fig. 3. Two utility matrices, depicting the payoff for a specific action and world state combination. In both worlds the actions of the player have not impact on the utility received. In world one, the payoff depends only on the world state, in the second world nothing has any influence on the payoff.

information between $R$ and $A$ is calculated as either

$$I(A;R) = H(A) - H(A|R) = H(R) - H(R|A)$$

it is bound by $H(A)$ and $H(R)$, because the conditional entropy is always positive. If $A$, the actions of the players, would be of size $n$, meaning that there are $n$ options, then the maximal entropy in case all actions are chosen with the same frequency is:

$$max(H(A)) = -\sum_a P(A=a) \cdot log(P(A=a)) = log(\frac{1}{n})$$

If we now eliminate one option for $A$, the maximum entropy is $log(\frac{1}{n-1})$, which is smaller. So, for every inferior choice the maximum Relevant Information is more limited. This should lead to a diminishing of the maximum Relevant Information.

### B. Irrelevant Actions

A different fault is to design a game mechanic where the agent's effort has no impact on the outcome of the game. Apart from the question if this is a game at all then, we assume that this is not desirable. Also it is doubtful that such a scenario is designed by a human designer, it is possible in a complex game world that such a pathological case sneaks in, and if we evolve game mechanics, it might be possible that the computer creates a game like we see in World 1 or 2 in Fig. 3. World 2 has describes the payoff of a scenario where neither the agent's action nor the states of the environment matter. All strategies have the same payoff, and therefore, the Relevant Information is 0, because the strategy that plays random is also optimal. World 1 has a payoff that depends only on the world state, so random again is a viable strategy, and the Relevant Information for all those cases is 0.

To tell those two cases apart, we can now also look at the performance of the AIs. There is another measurement point that can be easily identified, the one of the strategy that plays random. There is no mutual information in that strategy, so the Relevant Information for that level is 0. We can now look at the performance of the random strategy and note two things. First, how well does it actually perform. If the performance level of the random strategy is too high, it could be argued that there is no player effort involved in solving the game, since simply pressing buttons at random will already yield very good results. The other thing to look at is how much the optimal strategy is better than the random strategy. If they are close,

World 1

| (Pay Off) | State 1 | State 2 | State 3 | State 4 |
|---|---|---|---|---|
| Action 1 | 1 | 1 | 1 | 1 |
| Action 2 | 0 | 0 | 0 | 0 |
| Action 3 | 0 | 0 | 0 | 0 |
| Action 4 | 0 | 0 | 0 | 0 |

Fig. 4. A utility matrix depicting the payoff for a specific action and world state combination. In this world the player has an impact on the payoff, but the state of the world does not. Therefore, the player needs no information about the world state to make a decision.

or identical, than we can conclude, that we are dealing with a non empowered agent, a scenario where the agent's choices do not matter. If there is actually a difference, then this is not the case.

### C. Dominant Strategies

Related to the inferior choices problem is another flaw, the existence of a dominant strategy. In those scenarios an agent will always choose the same actions, regardless of its sensor input, because no matter what the environment does, the action is always optimal. Such a case is also undesirable, because once the player finds this strategy he is forced to play it continuously. In the case seen in Fig. 4 the player would always play action 1, the amount of information one would need to acquire is 0, so the Relevant Information is also 0, given that the states of the environment are equally spread out.

This would allow the player to reach a higher performance level than the random strategy, without actually using any game information, so the function that relates Relevant Information and performance would stay 0 for higher performance levels.

### D. Optimal Case

To achieve a maximum amount of Relevant Information it is therefore necessary to design a game that :

1) Uses all possible options, in similar frequency
2) The decision of the player have an impact on the world
3) The optimal decision depend on the different states of the environment

It seems, that it should be desirable to have a high degree of Relevant Information for the best strategy. Furthermore, the performance for the random strategy should be low, and the increase in performance should lead to an increase in Relevant Information, so the player does not just learn a strategy template he then plays regardless of the actual game world.

### E. Partitioning of the World Space

Additionally, all of the considerations discussed in the last part can also be applied to a part of the world. Instead of calculating the mutual information between $A$ and $S$, we can observe a subset of $S$, called $S^*$, and calculate $I(A; S^*)$. This makes it possible to answer the question, of whether a certain part ($S^*$) of the world contains information relevant to the player. If this is not the case the display of that part of the game world to the player might be unnecessary or confusing. Alternatively, it might be possible that $S^*$ is supposed to have an impact on the players decision, in which case one could pinpoint where the game mechanics have to be altered to make $S^*$ relevant.

Furthermore, for any part of $R$, called $R^*$ we can use mutual information to determine if that part of the game world is having an actual impact on the game's outcome; meaning that there is mutual information between $R^*$ and a random variable encoding if a game is lost or won. If there is no mutual information then it follows that there is no statistical dependence. If there is also no Relevant Information in $R^*$ it becomes questionable how that part of the world relates with the game mechanics at all.

Considering the cases where there is a relation between $R^*$ and the game's outcome we can again identify several cases. If $R^*$ is not perceivable by the player, than we are introducing a hidden random factor into the game. If $R^*$ is actually visible to the player, but yields no Relevant Information, than an optimal strategy does not need to take $R^*$ into account. This means, $R^*$ is a factor in the game world that we can do nothing about. If this is desirable or not is a design question (it could be information about how close the player is to winning the game). But an information theoretic analysis allows us to identify those factors, nonetheless.

## IV. Technical Implementation

To test our hypothesis we implemented a simple game, where the player has to take control of an army, and decide whom to attack. We will demonstrate how AIs, that are adapted to the game via genetic algorithm, are used to approximate the actual relation between Relevant Information and performance, and how this information can then be used to improve the game mechanics.

### A. Game Mechanics

Our test-bed is a small game inspired by the turn-based tactical battles of the *Heroes of Might & Magic* series. Two players are facing each other, each player starts with three stacks of creatures each containing three creatures. The goal of each player is to kill the other player's stacks by attacking them with their own stacks.

All creatures start with the same attributes for attack damage and hitpoints. Additionally we removed the spatial component so stacks can attack each stack of the enemy, regardless of position. Every stack gets to act once per round, the order is random. If a stack attacks another, the damage dealt is calculated by multiplying the hitpoint of all remaining creatures in the stack and their attack damage. There is a random element in the attack damage, so each creature has a certain damage where the actual damage is chosen from at random. The damage is then subtracted from the hitpoints of the first enemy creature. If the hitpoints of a creature reach zero the number of creatures in the stack is decreased, and the

remaining damage is subtracted from the next creature. If the number of creatures in a stack reaches zero, the stack dies and is removed/ignored until the game ends. A special rule is the concept of retaliation. If a stack has the ability to retaliate it can attack back after it has been attacked.

The AI input is determined as follows: We uses two bits to encode the stack's topmost creature's health and two bits for the actual number of creatures in a stack. An additional bit was used to determine if the stack has the ability to retaliate in the current round. So overall we used five bits to encode a stack's current state. Two players with three stacks each make six stacks in the game which makes the signature of each game state an array of thirty bits.

### B. Approximation via Genetic Algorithm

Calculating the actual Relevant Information for each performance level would make it necessary to look at all possible strategies. But this approach becomes quickly unfeasible, once the complexity of a game grows. Our alternative suggestion is to use a genetic algorithm to select a subsection of all strategies, those adapted to be of high performance and low mutual information. We then record the mutual information and performance of those strategies and use those to approximate the actual Relevant Information.

Note that, since the computation of mutual information requires the joint probability of both variables, it is not sufficient to only look at a strategy $P(A|S)$ to compute $I(A;S)$; it is also necessary to get data about the distribution of $P(S)$.

Our AI is a neural network that uses the sensor states as an input. Each input node receives one bit of the sensor data as a float value of either 1.0 or 0. The output nodes are associated with the different actions the player can take. The genome of our genetic algorithm is the collected internal weights of the neural network. We first create 20 genomes with random values, and then we create the associated neural networks that play the game for 1000 games against an opponent that picks random actions.

As a next step, for each strategy, we measure both the performance, as the fraction of games won, and the mutual information for the recorded joint distribution of sensor states and chosen actions for those games. Note that each game consists of several pairs of sensor inputs and taken actions.

We then evaluate the genomes with a fitness function that favours high performance and low mutual information, weighted with a variable weighting factor lambda. Both values, performance and mutual information are normed to values between 1.0 and 0. For the performance we divide the number of won games by the number of played games. For the mutual information we divide the results by the maximum entropy of the actions, in our case 2. The mutual information is then subtracted from 1.0, since we want to minimise it. We usually use the values of 0, 0.25, 0.5, 0.75 and 1.0 for lambda, where lambda = 0.0 means that only mutual information matters, and lambda = 1.0 means that only performance is taken into account. The most fit genomes are procreated and we repeat

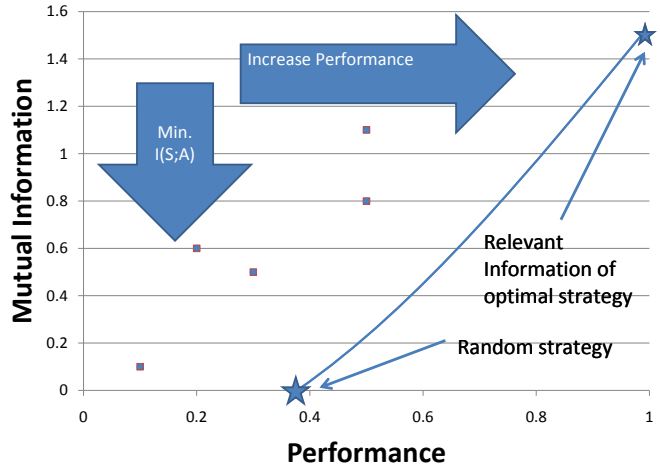


Fig. 5. A diagram of the graph relating performance and Relevant Information. The line indicates the actual Relevant Information function, all data point have to be either on or above that function. The fitness function of the genetic algorithm prefers AIs that are closer to the function, thereby trying to approximate the actual function.

this process until we exhaust a set amount of generations, usually 200.

We measure the relation between performance and mutual information for all genomes in all generations and the result is a graph as seen in Fig. 5. Every data point in the graph is a strategy, the values indicate its performance and its mutual information. The actual graph for the Relevant Information would be a line that all data point are either on or above, since it is possible for a strategy to have higher mutual information, but not lower. The two factors in our fitness function are used to evolve the strategies towards higher performance, and lower mutual information, thereby moving the resulting strategies closer to the actual graph. Note that the graph is not an average of the strategies we are looking it, but a lower bound. Therefore, it is possible to combine the results of several evolutionary runs and combine them all into the same graph. This can only improve the approximation. Also, since the mutual information is a function defined by the game mechanics, it is possible to vary lambda, and evolve strategies that are more optimized towards performance or mutual information reduction, and still combine them in the same graph. Indeed, our experience suggests that this is advised to get a good selection of strategies.

### C. Technical Problems

*1) Deterministic Strategy:* One problem in approximating the actual Relevant Information of a game is the use of deterministic strategies. The neural network usually picks one action based on its inputs, and normally it would always choose the same action for the same input. This automatically limits the strategies $P(A|S)$ to those where $H(A|S) = 0$ , since the action is determined by the sensor inputs. This leads

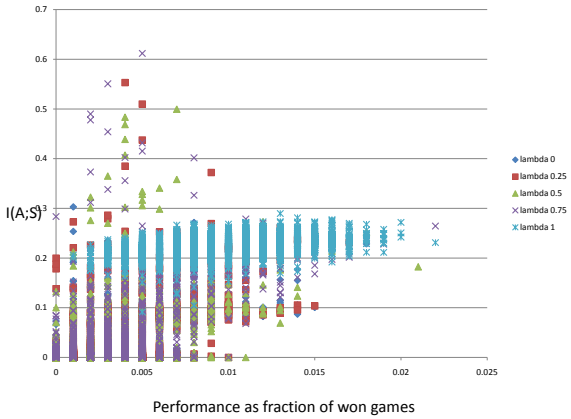


Fig. 6. A graph showing the relation between performance and mutual information based on the statistics for strategies in case 1. Note that the scale for performance only reaches from 0.0 to 0.025, otherwise all datapoints appear in a line above 0.

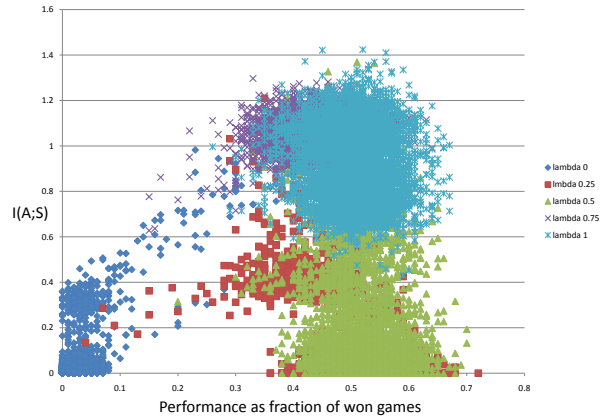


Fig. 7. A graph showing the relation between performance and mutual information based on the statistics for strategies in case 2.

to the mutual information being calculated as

$$I(A;R) = H(A) - H(A|R) = H(A) - 0$$

Since we are looking for the strategy with the least amount of mutual information, limiting us to deterministic strategies seems to hinder a good approximation. Strategies that take a random decision in those circumstances where it does not matter are not included, and therefore the overall mutual information is pushed upwards.

One solution is to modify the way the neural network chooses the actions. Instead of picking the actions whose nodes got the highest values, we now associate the values of the end nodes with the probability for that action to be picked. This allows the neural network to realise random strategies; strategies that should be favoured if they have the same performance, but lower mutual information.

## V. Scenarios

### A. Case 1, no player effect

In our initial scenario both sides had the same creatures and there was no ability to retaliate. The player had the option to attack the stacks in position one, two or three, and if he would try to attack a stack that was dead, the game would redirect his attack to the next stack alive. The damage a stack dealt was calculated in regard to the remaining combined hitpoints of the stack. Several evolutions of our adaptive AI yielded the results seen in Fig. 6.

Two things can be observed here. Firstly, there is no real difference in performance levels between the different strategies, they all seem to be very close to zero. So it seems that the players actions have no real impact on the outcome of the game. The small variation in performance values is likely due to the random element in damage calculation that allows the player to win in rare cases. The graphs performance scale is actually only reaching from 0.0 to 0.025, otherwise, if the scale would go up to 1.0, the variations would be nearly invisible on the graph.

Note that the performance is near zero, so the player nearly never wins. This is slightly surprising, since we first thought that the player should have an fair chance against the opponent AI, but then we realised with closer examination, that the opponent always seems to go first.

The second thing we observe is that the Relevant Information for all the performance levels up to ca. 0.011 is zero, since there is always at least one strategy that does not use any information. Also, for the other strategies that go above that value the increase in Relevant Information is quite low.

Comparing that to our earlier theories we seem to be dealing with a case where the players actions are irrelevant, and a closer look at our current game mechanics supports this analysis. All actions are attacks, deal similar damage, always hit a valuable target, and even reduce the opponents ability to deal damage in a similar way. So no matter what the player decides, the actions has a similar payoff.

### B. Case 2, dominant strategy

In our second case, we modify the game mechanics so the player has an impact on the game. We introduce the retaliate mechanics, and now each stack can retaliate once per game round. We also introduce the option for a stack to wait and do nothing for one round. Now the player has an impact on the game. A good strategy will have to learn that waiting is a bad choice, since it never does anything, and it should also aim to attack and enemy stack that has already retaliated.

Looking at Fig. 7 we can see, that there are several strategies with better performances than random, and we see several strategies that are able to win the game more than 60 percent of the time, even though the game still lets the opponent start first. But even for those relatively high performance levels the amount of Relevant Information seems to be zero, since there

are strategies that reach that performance, but have no mutual information.

A closer look at the actual decision of the AIs shows that one good strategy is to always attack stack 1. This avoids using wait, it focusses all attacks on the same target to avoid retaliation, and if the first stack is dead, the attack will be forwarded to the next stack. This, arguably, seems to even be an optimal strategy. On the downside, this also seems to be what we earlier identified as a dominant strategy. A strategy, that given the current game mechanics, always seems to be optimal, no matter what the actual state of the world is. So there is no need for the player to actually look at the game world to make a decision. This works well with our predictions, since the graph of the mutual information would indicate this as well. As discussed, we get strategies that improve well above the performance level of the random strategies but still keep a mutual information of zero.

This graph also shows how the different weights in the fitness function push the AIs along different paths in the two dimensional projection (to mutual information and performance) of the solution space. The adaptation towards minimal mutual information (lambda = 0.0) moves quickly towards the random strategies and then ends up in a cluster around zero performance and zero mutual information. The strategies that maximise performance (lambda =1.0) dont move towards the lower mutual information, but their cluster pushes to the right to explore strategies with higher performance. Finally, the strategies that balance both constraints (lambda = 0.5) develop good strategies that also use no mutual information.

### C. Case 3, positive Relevant Information

We further modify the game so it is necessary for a good strategy to acquire information about the game world. Now retaliate is stronger, and it will only be activated if a stack has waited in the last turn. Since the AI chooses strategies at random this should lead to some opponents randomly being able to retaliate. A good strategy should avoid those stacks. Furthermore, we also stop the forwarding of attack orders. So, if an AI now attacks a stack that is dead its attack will have no effect. Thus, the information of whether a stack has remaining creatures should become relevant.

Looking at the graph in Fig. 8 we can see that our game play modifications have lead to a measurable change in the Relevant Information. It seems possible for the AI to actual develop good strategies, some of them win in more than 70 percent of the cases, but for all the strategies that go beyond a performance of 10 percent there seems to be at least a certain amount of information those strategies need to pick up. Also, the better the strategies get the more information they seem to use. This indicates, that a higher performance level also needs a better analysis of the different factors of the game world. All in all, this graph does not indicate any of the flaws we discussed earlier.

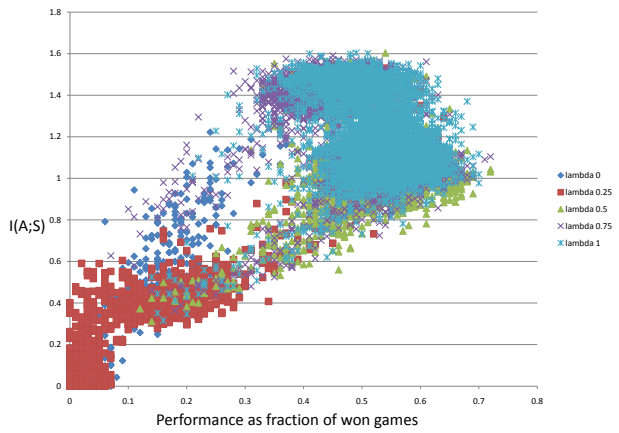


Fig. 8. A graph showing the relation between performance and mutual information based on the statistics for strategies in case 3.

## VI. Discussion

The analysis of the different cases seems to support our initial theory about how to relate measurements from information theory with enjoyment related game parameters. If the properties discussed, such as inferior choices and dominant strategies, are actually undesirable for a game or not, is a discussion that is out of the scope of this work, but we think that having a implementable solution to detect them might be useful, nonetheless.

We also demonstrated a technique to approximate Relevant Information. Here we should note that there exist better ways to determine the actual Relevant Information, as discussed in [14], but we believe those are more complicated in application, and require detailed knowledge of the system they are studying, while the genetic algorithm approach can treat the studied game as a black box.

All in all, it seems that this approach is something that can be implemented with the techniques described, and can be used, if for nothing else, as an additional mean to evaluate the quality of game rules, especially those of strategic decision making games. The analysis of Relevant Information could of course also be extended to different kinds of games, since it would also make sense to check if a player actually needs to react to the environment in an ego shooter, but the modelling of such games in terms of random variables would require some additional consideration. Turning the input and output of the game into discrete state is usually not difficult, since everything is implemented on a computer, and ultimately made of bits and bytes, but the enormous increase in the state space can make the calculation of the mutual information unfeasible.

## VII. Future Work

The discussed results are only a first step towards a fully automated evaluation of game mechanics. A closer look at how the Relevant Information is distributed inside the game world could yield additional information about what aspects of the

game should be exposed to the player. Also, there are other information theoretic measures, such as "Empowerment", that could be used to take a context independent look at games. Empowerment seems to be particularly interesting here, because it basically measures how much impact the actions of a player have on the world he can perceive.

Once more robust evaluation mechanics are found an interesting next step would be to automate both, the design and the evaluation processes of a game. Given that one has a formalised language that expresses game mechanics, it would be possible to make a game itself the subject of a genetic algorithm. The genomes would represent different game mechanics, which would then be evaluated in turn by an adaptive AI, itself driven by an genetic algorithm.

Another, more practical, approach would be to apply a clustering technique to the different Relevant Information graphs to predict what kind of game play flaw a certain game mechanic has. As we discussed in this paper, different flaws e.g. inferior choices or dominant strategies generate different patterns in these diagrams.

## REFERENCES


[1] C. Louis, S.J.and Miles, "Playing to learn: case-injected genetic algorithms for learning to play computer games," *IEEE Transactions on Evolutionary Computation*, 2005.

[2] P. Huo, S. C.-K. Shiu, H. Wang, and B. Niu, "Application and comparison of particle swarm optimization and genetic algorithm in strategy defense game," in *Fifth International Conference on Natural Computation*. IEEE, 2009, pp. 387–392.

[3] S. Wender and I. Watson, "Using reinforcement learning for city site selection in the turn-based strategy game civilization iv," in *CIG'08: IEEE Symposium on Computational Intelligence and Games.*, 2008.

[4] P. Huo, S. C.-K. Shiu, H. Wang, and B. Niu, "A neural-evolutionary model for case-based planning in real time strategy games," in *Next-Generation Applied Intelligence*. Springer Berlin / Heidelberg, 2009, pp. 291–300.

[5] ——, "Case indexing using pso and ann in real time strategy games," in *Pattern Recognition and Machine Intelligence*. Springer Berlin / Heidelberg, 2009, pp. 106–115.

[6] G. N. Yannakakis and J. Hallam, "Capturing player enjoyment in computer games," in *Advanced Intelligent Paradigms in Computer Games*, 2007, pp. 175–201.

[7] G. Andrade, G. Ramalho, H. Santana, and V. Corruble, "Automatic computer game balancing: a reinforcement learning approach," in *AAMAS*, F. Dignum, V. Dignum, S. Koenig, S. Kraus, M. P. Singh, and M. Wooldridge, Eds. ACM, 2005, pp. 1111–1112.

[8] R. Leigh, J. Schonfeld, and S. J. Louis, "Using coevolution to understand and validate game balance in continuous games," in *GECCO '08: Proceedings of the 10th annual conference on Genetic and evolutionary computation*. New York, NY, USA: ACM, 2008, pp. 1563–1570.

[9] J. Togelius and J. Schmidhuber, "An experiment in automatic game design," in *Proceedings of IEEE Computational Intelligence and Games 2008*, 2008, pp. 111–118.

[10] G. N. Yannakakis and J. Hallam, "Towards Capturing and Enhancing Entertainment in Computer Games," in *Proceedings of the Hellenic Conference on Artificial Intelligence*, 2006, pp. 432–442.

[11] C. Salge, C. Lipski, T. Mahlmann, and B. Mathiak, "Using genetically optimized artificial intelligence to improve gameplaying fun for strategical games," in *Sandbox '08: Proceedings of the 2008 ACM SIGGRAPH symposium on Video games*. New York, NY, USA: ACM, 2008, pp. 7–14.

[12] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, July 1948.

[13] D. Polani, T. Martinetz, and J. T. Kim, "An information-theoretic approach for the quantification of relevance," in *ECAL '01: Proceedings of the 6th European Conference on Advances in Artificial Life*. London, UK: Springer-Verlag, 2001, pp. 704–713.

[14] D. Polani, C. L. Nehaniv, T. Martinetz, and J. T. Kim, "Relevant information in optimized persistence vs. progeny strategies," in *Artificial Life X : Proceedings of the Tenth International Conference on the Simulation and Synthesis of Living Systems*. The MIT Press (Bradford Books), August 2006, pp. 337–343.

[15] T. M. Cover and J. A. Thomas, *Elements of information theory*. John Wiley and Sons, Inc., 1991.

[16] J. Pearl, *Causality*. Causality, by Judea Pearl, pp.˜400.˜ISBN 0521773628.˜Cambridge, UK: Cambridge University Press, March 2000., March 2000.

[17] R. Koster, *A theory of fun for game design*. Paraglyph press, 2005.

[18] J. Juul, "The game, the player, the world: looking for a heart of gameness," in *DIGRA Conf.*, 2003.