RMM's Solution Concept and the Equilibrium Point Solution.

José M. Vidal and Edmund H. Durfee
Department of Electrical Engineering and Computer Science
University of Michigan, Ann Arbor, MI 48109-2122.
{jmvidal,durfee}@umich.edu

Abstract

Research in distributed AI has dealt with the interactions of agents, both cooperative and self-interested. The Recursive Modeling Method (RMM) is one method used for modeling rational self-interested agents. It assumes that knowledge is nested to a finite depth. An expansion of RMM, using a sigmoid function, was proposed with the hope that the solution concept of the new RMM would approximate the Nash EP in cases where RMM_S knowledge approximated the common knowledge that is assumed by game theory. In this paper, we present a mathematical analysis of RMM with the sigmoid function and prove that it indeed tries to converge to the Nash EP. However, we also show how and why it fails to do so for most cases. Using this analysis, we argue for abandoning the sigmoid function as an implicit representation of uncertainty about the depth of knowledge, in favor of an explicit representation of the uncertainty. We also suggest other avenues of research that might give us other more efficient solution concepts which would also take into consideration the cost of computation and the expected gains.

1 Introduction

Research in distributed AI has dealt with the interactions of agents, especially rational agents. In order to make a good decision, an agent must be able to anticipate the actions of others, which can be accomplished using an internal model of the other agents. Typically, an agent can assume that the other agents possess some degree of rationality. If so, the agent should also model the other agents modeling him, and so on. This situation leads to a recursive nesting of models, which is the basic architecture of the Recursive Modeling Method (RMM), see [4].

The recursive nesting can be infinite in some situations, as would happen if common knowledge was assumed. Common knowledge is the case where all agents know that all agents know ...that all agents know a fact p, where the phrase "all agents know" is repeated an infinite number of times. Game theory has studied such cases extensively and has an agreed upon fixed point solution, the Nash Equilibrium Point (EP), for them. Recently, some game theorists have questioned the common knowledge assumption on the grounds that such an infinite nesting would be impossible to achieve in reality[5]. There have also been some computational tractability concerns which have lead to other solution concepts [1] which approximate the EP solution but only assume E^k knowledge[1].

RMM has employed several different solution concepts through its history. Originally it would simply recurse down until it reached a known finite level and then propagate a solution back up. This method leads to a correct solution only when the agent knows that it is exhausting all the possible levels of knowledge. If the final level is unknown or we are uncertain about it, the next version of RMM would handle this by recursing down until it found some convergence or a cycle.

[1] E^k knowledge is recursive knowledge but only up to some finite k level of nesting

This new method could conceivably recurse indefinitely using the same models at deeper levels, as would happen if there was common knowledge. Durfee, Lee and Gmytrasiewicz [3] set about understanding the relationship between RMM and Nash EP for this particular case and discovered some differences. In an attempt to reconcile them, they considered a new solution concept, RMM with the use of a sigmoid function which we will refer to as RMM_S. The sigmoid method would at times approximate the Nash EP, but suffered from stability problems and the authors gave some thoughts as to why this was happening .

In this paper, we give a more complete analysis of the behavior of RMM_S and answer questions about whether it can be expected to always converge to the EP. We prove that it is trying to converge on a Nash EP, after defining what we mean by this. We also address its stability problems and explain why these problems arise, with the conclusion that it would be impossible to reliably use this technique for all games. The paper starts by reviewing the aforementioned solution concepts. Section 4 presents a differentiation method, used in [3], and proves that it sometimes finds a Nash EP solution for some games. We will later prove, in Section 6, that this is the same solution that RMM_S attempts to converge to when it converges on a mixed strategy solution. In this section we will also define what it means to attempt to converge on a solution. We then use this result to prove that RMM_S always tries to converge to a Nash EP. Note, however, that we said that RMM_S *tries* to converge to a Nash EP. In fact, Section 5 will show, using nonlinear system theory, why the sigmoid method actually fails to converge to the Nash EP. We also provide a better method for understanding the behavior of RMM_S. Our analysis is then expanded to non-symmetric games for which we show that it is impossible for RMM_S to converge on a mixed strategy Nash EP.

With this improved understanding of RMM_S's solution concept, we then return to the broader question of recursive modeling in arbitrary domains. In its current and latest form, RMM explicitly represents the probabilities for deeper nesting of knowledge and overcomes the limitations of the previous versions of RMM. However, it is still sometimes time consuming or not practical to use all the finite knowledge that is available. We end this paper with a look toward ways of expanding RMM so that it can take into account computation and time costs versus possible payoffs, which would facilitate the maximization of expected payoffs. We also look into the question of whether it is wise, in certain circumstances, to assume common knowledge, and into the question of metareasoning.

2 The Recursive Modeling Method

In our discussion, we will be concentrating on a subset of the RMM theory. Specifically, we will assume two agents of interest both of whom have accurate knowledge of each other's payoff matrices, to some certain finite level. We also assume that this level is sufficiently large. For these cases, RMM builds a recursive hierarchy where each level is a payoff matrix that represents the payoff the player expects.

For instance, if player A has payoff matrix M and player B has matrix N, player A will use its matrix M to determine which move it should make. It determines its move by calculating which of its options will give it the highest payoff. But, to do this, it needs to know what player B is playing. It will then assume that player B will also try to maximize its payoff and so it will model player B using its payoff matrix N. Player A will model player B, recursively, modeling A with its matrix M, and so on. This hierarchy will continue to expand until we run out of knowledge. (Note that if we had real common knowledge the levels will be extended to infinite depth.) The final level will then be filled with the "zero knowledge" solution, where we simply represent our lack of knowledge as the player playing each of its options with the same probability. Once we reach the final level, we can propagate the values back up the hierarchy and determine which move is the best one for

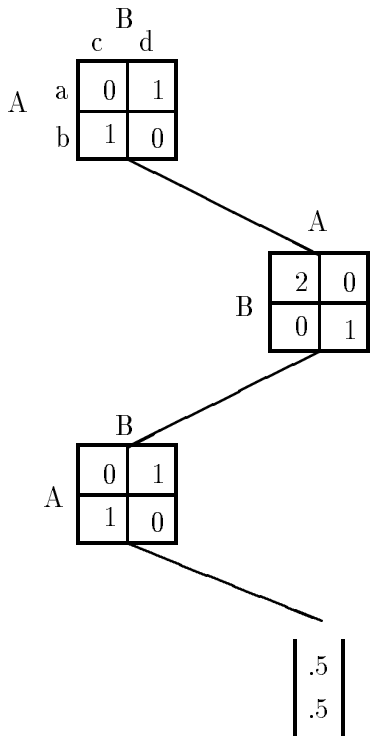

Figure 1: This is the hierarchy of a simple RMM game to four levels for two players A and B. The bottom most element is the "zero knowledge" solution. After propagating values to the top the solution we get tells us that the player A should pick choice b.

us to take. Propagation is done by, at each level, assuming that the strategy from the level below is the one the other player will play and, with it and the game matrix, calculating what the best strategy for this player is. An example can be seen in Figure 1.

The full version of RMM allows the game matrices to change for each level. That is, if my payoff matrix is M, it is still possible that I think that my opponent thinks that my payoff matrix is something other than M. In fact, every payoff matrix in the hierarchy could be different. We are not concerned as much with these cases because these situations do not lead to Nash equilibriums. In fact, if this knowledge really represents the true state of the world then it should be clear that RMM does come up with the best strategy (since it considers all the knowledge available and tries to maximize payoffs at each stage). However, we are concerned about the simpler case where both agents are presented with a standard game-theoretic payoff matrix, of which they have complete or almost complete common knowledge. For this case, game theory predicts an EP solution, which is not the solution that the original RMM necessarily arrives at.

3 The different solution concepts.

3.1 Simple RMM solution

In its original specification, RMM's solution would sometimes oscillate as we increased the number of levels used. It would, for example, return one answer if we applied the algorithm to an odd

number of levels, and another answer if we applied it to an even number of levels. The ambiguity was resolved, in some instances, by averaging out the different answers. The resulting solution proved to be different than the EP solution.

3.2 Nash EP solution

The Nash EP solution states that, given an EP vector of strategies, no individual player, regarding the others as committed to their choices, can improve his payoff by deviating from its strategy. There is no standard way of finding the Nash EP for all games, although different methods and tricks can be used in certain situations.

3.3 RMM solution using the sigmoid function.

It was hypothesized that the reason RMM failed to reach the EP solution was because it committed to one conjecture or another too early (i.e. it was overeager). The problem was fixed by incorporating a sigmoid function whose role is to reduce the importance of the solutions close to the leaves of the tree. Details on this function can be found in [3]. We will call this version RMM_S. The deeper levels of the hierarchy were multiplied by a sigmoid function with a very small exponent k. A small exponent has the effect of transforming the old strategies into new ones with a reduced difference in the probabilities for choosing any one option. At the extreme, for $k = 0$ all the options receive the same probability. The higher levels of the hierarchy got bigger values of k, up to infinity. With a value near infinity the sigmoid functions transforms the strategy such that the one option with the higher probability than all the others will now have probability equal to 1, while all the other options will have probability equal to 0. The function for the probability of option i, given its expected payoff relative to the payoffs of all the options J, is:

$$p_i = \text{Payoff}(i)^k / \sum_{j \in J} \text{Payoff}(j)^k$$

Implementation of the sigmoid function leads to some peculiar behavior where the algorithm seemed to get closer and closer to the mixed EP solution only to then diverge away from it and go back to the pure strategy solutions (1,0) and (0,1). Such behavior always appeared no matter what rate of increase we chose for the exponent of the sigmoid function. Our goal for this paper is to explain why this behavior occurs and what it means.

4 How to find an EP solution

A Nash EP solution is very well defined; however, there is no definitive method for finding it. The differentiation method used to find the EP solution in [3] is not guaranteed to always work. It might not find all the EPs. However, this method is fairly powerful and, as we shall see later, the solution it returns corresponds to the one RMM_S sometimes tries to converge to. It will, therefore, be useful for us to exactly define the solution that the differentiation method derives. The following theorem can be proven using some algebraic manipulations. Its proof can be found in Appendix A.1.

Theorem 1 *The solution strategy arrived at by using the differentiation method is X such that: given that our opponent's payoff matrix is M, $M \cdot X = N$, where N is the vector matrix that has all its values equal to $1/N$ (given that M is an $N \times N$ matrix).*

Given a specific game matrix

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

we can use Theorem 1 to conclude the following:(Appendix A.2)

Corollary 1 *For two players with two choices, the solution arrived at by using the differentiation method is x:*

$$x = \frac{b-d}{c+b-d-a} \tag{1}$$

Where x is the strategy for one of the players and the matrix is the payoff matrix for the other player.

In other words, the differentiation method returns a strategy such that, if we play it, our opponent will get the same payoffs no matter what he plays. This information is all we need in order to come up with an equation for the solution in the 2×2 case. Please note that the solution arrived at by the differentiation method does not take into account the player's payoff matrix, only his opponent's. Since we know that our opponent's strategy is such that no matter what we play we get the same payoff we can prove the following:(Appendix A.3)

Theorem 2 *For two players, the strategy x arrived at by the differentiation method (if any) is always a weak Nash equilibrium point.*

A weak Nash EP is one were we are not doing strictly better with our strategy but we are doing as well as with any other strategy. It should be clear that the differentiation method might fail to arrive at a legal answer (i.e. it might set the probabilities to be more than 1 or less than 0). For such cases, the theorem does not hold. Also note that the converse of Theorem 2 is not true so we can conclude that, while the differentiation method, when it works, will return an EP, we can not rely on it to find any or all of the EPs. [2]

5 Understanding the behavior of RMM$_S$

It is easier to understand the behavior of RMM$_S$ by looking at a picture of a symmetric game where both players have the same payoff matrix, with only two possible choices for each player. Since there are only two possible moves, we can assume that one of the moves is selected with probability x and the other one with probability $1-x$. We represent the probability x on the x-axis. The y-axis represents the normalized expected payoff to the other player, also known as the sigmoid function. We then draw a different payoff curve for each value of the exponent. We also draw the line $y = x$. Since both opponents have the same payoffs, and therefore the same sigmoid function, we can map the behavior of the RMM$_S$ algorithm in much the same way as we would trace the behavior of a non-linear system [7]. We start with the value $x = 1/2$, our initial guess, and trace the progression of the solution, as seen in Figure 2.

As can be seen, as the exponent k increases the sigmoid curve gets steeper around $x = 1/3$, the EP solution. If we trace the RMM$_S$ solution we see how it approaches, at the beginning, the EP solution. However, when it gets too close, it starts diverging back the the pure strategy solutions.

[2] As a side note, we might mention that all the Nash EPs, of a 2×2 game, can be found by plotting both sigmoid functions for $k = \infty$. One of them is plotted in the standard way, and for the other one we switch the x and y axes. These functions will cross at either one or three points. The intersection points will correspond to the EPs.

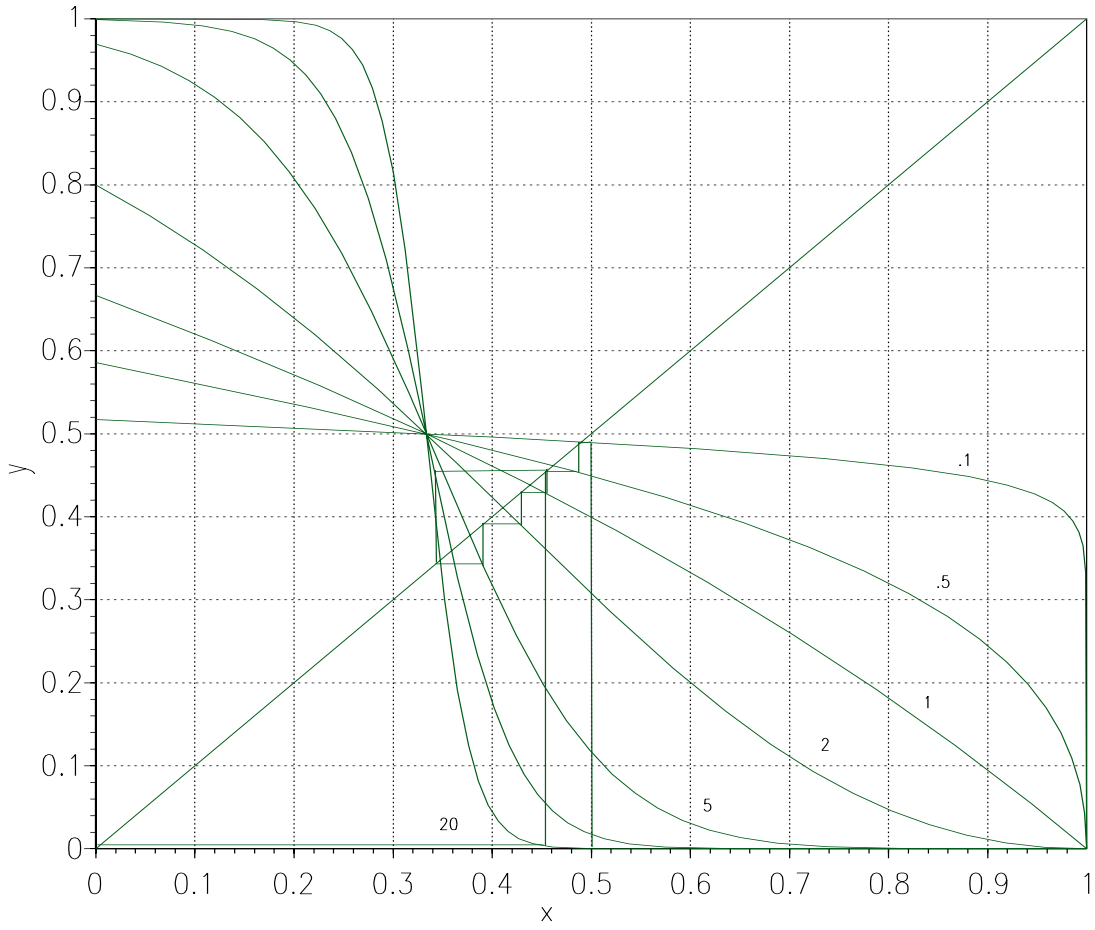

Figure 2: Here we see the progression of the solution for the RMM_S algorithm. The numbers on the curves are the exponents (k) of their respective curve. The lower levels of the RMM_S hierarchy have the smaller values of k, which keeps increasing as we go up in the hierarchy. The solution starts at .5 (going up to meet the sigmoid, then left to meet $y = x$, and so on) and works its way towards the pure strategy solutions.

Figure 2 gives us a good intuition as to why the EP solution, for this example, is not a stable solution in the RMM_S model. We can see that the sigmoid function gets steeper and steeper around $x = 1/3$. We can also see that at each step of the algorithm, the solution line goes horizontally to meet the curve $y = x$ and vertically (down) to meet the sigmoid curve. It should be clear that the moment the solution falls below $y = 1/3$ it will immediately spinout to the pure solutions. It should also be clear that this can easily happen if we increase the exponent too fast because then the sigmoid curve will be so steep that, when the solution tries to meet the curve it will fall below $1/3$. This is the reason why RMM_S needs to be careful when increasing the exponent. Unfortunately, even being careful will not always work.

The reason the solution might still diverge is that the point of intersection might not be an attractor. Nonlinear systems theory tells us that if the magnitude of the slope of the curve at the point of intersection with $y = x$ is greater than 1, then the point is not an attractor. Figure 3 shows a plot of the slope at the intersection point as a function of x. Experimentally, we have determined that, for this case, the point of intersection stops being an attractor when k gets bigger than 1.6976,

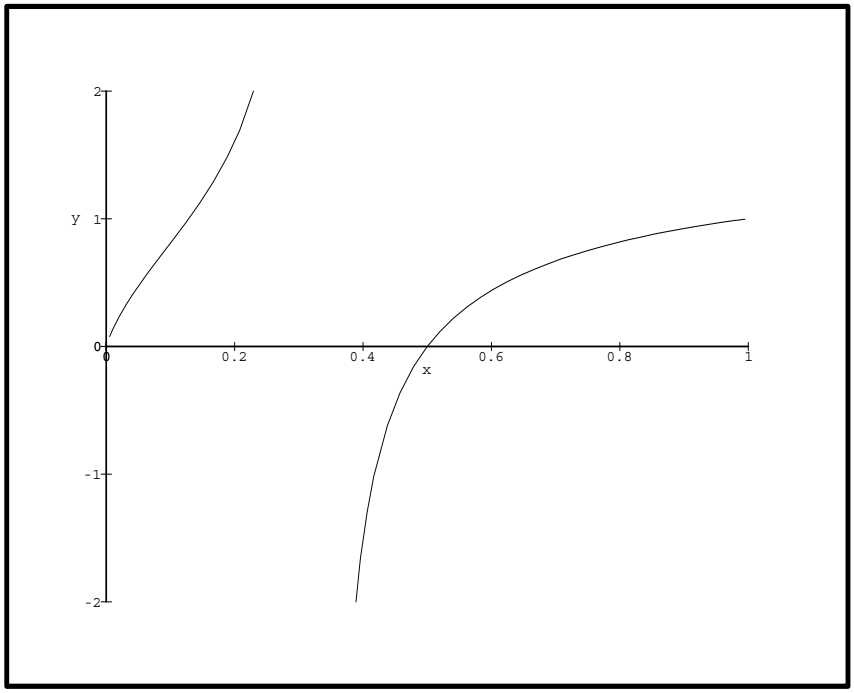

Figure 3: Plot of the slope at the intersection point x, for the same game as in the previous figure. Note that the point will stop becoming an attractor when the magnitude of the function is greater than one, in this case when $x = .4175$. Also, slopes greater than zero correspond to negative values of k so we will ignore them.

with $x = .4175$. We have derived an equation, for the general 2×2 case, that gives us the slope at the point of intersection for arbitrary values of x. With it, we can determine the x value for which the magnitude of the slope becomes greater than 1. Once we have the x value, we can determine it's corresponding k. These equations are quite complex, so we have included them in Appendix A.4. Unfortunately, we have been unable to solve for k and find an equation for the value after which the point of intersection will not be stable. This task seems especially daunting due to the complexity of the equations. If we could find it, such an equation would tell us exactly after which point RMM_S should start being especially careful. Fortunately, with the equations given, we can easily find k using simple numerical methods.

Looking at these numbers, we notice that the point $x = .4175$ at which the RMM_S solution becomes unstable is far removed from its desired solution point of $x = .3333$. The RMM_S algorithm would have to be responsible for keeping the solution from diverging as the sigmoid curve travels through this area (i.e. from $k = 1.6976$ to $k = \infty$). Especially since, as k gets bigger the intersection point becomes more unstable. For $k = \infty$ it is infinitely unstable.

These reasons seem to indicate that there is no easy way of modifying the RMM_S algorithm

to always converge on the intersection point. If we try to slowly increase the exponent so that the solution will not go down too far, we will find that the solution will still diverge because we are no longer near an attractor. It is true that the solution might seem to get closer to the desired EP solution even after passing the last attractor point, but this would simply be due to the fact that we are changing the exponent. If at any moment we were to stop increasing the exponent, the solution would diverge since the point is not an attractor. On the other hand, to keep increasing the exponent in a such a way that the solution always stays close to the intersection point would mean that we have prior knowledge of the value of the EP solution we are trying to reach.

Keeping the "best" solution (i.e. the one with the higher expected payoff) at each step will result in a smoother approach since we will be ignoring the divergences that appear because we are not near an attractor. But it will lead to a situation where we either stop moving or we jump towards the pure solutions once one of the player's sigmoid curves make it go below the EP.

Please note that although the previous reasoning dealt only with our one example it also applies to all game matrices with a mixed strategy Nash EP. The reason is that the sigmoid curve for them will have basically the same shape as the one we show. The only possible differences are that it might have a different crossover point or it might have the values to the left of the crossover equal to zero and the ones to the right equal to one. None of these differences is significant given our reasoning.

6 RMM$_S$ desired solution concept

We established in the previous section that RMM$_S$ cannot reliably converge on the intersection solution (i.e. on the intersection point, as explained in Section 5). In this section, we will provide a method for arriving at the intersection solution and prove that it is a Nash EP. Our reasoning will prove that RMM$_S$ is trying to converge towards a Nash EP. That is, the intersection point presented in the previous section (i.e. the one RMM$_S$ was *trying* to converge to) is a Nash EP. Furthermore, we show that in a 2×2 symmetric game RMM$_S$ will always try to converge to the weak mixed Nash EP, if there is any. If none exists then it will converge to the pure Nash EP.[3]

For the general 2 by 2 symmetric game matrix, we can mathematically determine the solution that RMM$_S$ will try to converge to. We start with the following game matrix:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

The sigmoid function for this matrix is $f(x)$, where x is the probability that the player will choose to play his first option, while playing the second option with probability $1 - x$.

$$f(x) = \frac{(ax + (1-x)b)^k}{(ax + (1-x)b)^k + (cx + (1-x)d)^k} \tag{2}$$

To find the point where this function intersects the line $y = x$ (i.e. the solution RMM$_S$ is trying to converge to) we set $x = f(x)$, and after some manipulations we get the following equation:

$$x^{1/k}(x(c-d)+d) = (1-x)^{1/k}(x(a-b)+b) \tag{3}$$

The solution that RMM$_S$ tries to approximate is the x as $k \to \infty$ such that Equation 3 is true. We notice that as $k \to \infty$ the terms $x^{1/k}$ and $(1-x)^{1/k}$ drop off to 1, assuming that $0 < x < 1$.

[3]Game theory tell us that, for this case, we will have either one pure Nash EP, or one mixed EP and two pure EPs.

We can then solve for x, the solution point.

$$x = \frac{b-d}{c+b-d-a} \tag{4}$$

Please note that $0 < x < 1$ must be true. The only other two possibilities RMM_S allows are for $x = 0$ or $x = 1$. So, if we use the previous equation for x and it returns a value that is not legal, we can still find the value returned by RMM_S by assuming that it is going to return either 0 or 1 and doing the calculations to confirm or deny this. In other words, if we assume that $x = 0$ then it must be true that

$$x = \lim_{k\to\infty} f(0) = \lim_{k\to\infty} \frac{b^k}{b^k + d^k} = 0 \tag{5}$$

$$b < d \tag{6}$$

So if $b < d$ then RMM_S will return $x = 0$. Similarly it will return $x = 1$ if the following is true:

$$\lim_{k\to\infty} f(1) = \lim_{k\to\infty} \frac{a^k}{a^k + c^k} = 1 \tag{7}$$

$$c < a \tag{8}$$

If both of these are true then we have the case where, $b < d$ and $c < a$ but if we look at our original equation for x we can see that if this were the case then the value of x would have been legal (i.e. $x = (b-d)/(c+b-d-a)$ and $0 < x < 1$), so we are back to our original case. The same logic applies if $b > d$ and $a > c$. The only possible case left over is if both $b = d$ and $c = a$; then $x = 1$ if $a > b$, $x = 0$ if $a < b$ and $x = 1/2$ if $a = b$.

Cases where $b < d$ and $c < a$ are ones where all the values on the diagonal are good but the agents cannot decide which one to pick. For example, in the matrix

$$\begin{pmatrix} 5 & 1 \\ 1 & 3 \end{pmatrix}$$

RMM_S would try to converge towards $x = 1/3$. This solution is a Nash EP but it has a payoff for both players of $7/3$. This payoff is much lower than the payoff of 5 they would get if they had played the "common sense" strategy of $x = 1$ (which is also a Nash EP). The reason this solution was picked is because RMM_S gives preference to any mixed strategy EP, no matter what the payoff is.

Another example would be the following matrix:

$$\begin{pmatrix} 5 & 2 \\ 1 & 1 \end{pmatrix}$$

Here we can calculate the value of x to be

$$x = \frac{2-1}{1+2-1-5} = -\frac{1}{3} \tag{9}$$

so the value of x is illegal, but since $1 < 5$ this implies that $x = 1$. So both players choose to play their first option for an expected payoff of 5.

We should note that this analysis proves that, for symmetric games, RMM_S *tries* to converge to a Nash EP. The analysis was done for the 2×2 case since it is the one we are more concerned with. However, it is our belief that the same manipulations can be applied to bigger matrices and

will provide us with the same results. Notice that equation 4 is identical to equation 1. This tells us that, when RMM_S is trying to converge to a mixed strategy (i.e. $0 < x < 1$), it is actually trying to converge to the same solution as the differentiation method and we proved earlier that this solution is always a weak Nash EP (if it exists). If RMM_S instead is trying to return a pure solution then we know that it is a Nash EP because it is the strategy that both players prefer. That is, the strategy RMM_S will try to return in these cases will be the value of $f(x) : 0 \leq x \leq 1$ when $k = \infty$. That means that it is the strategy that maximizes our payoffs no matter what our opponent plays. This strategy is also a Nash equilibrium since it implies the definition of a Nash EP: given that our opponent played some $0 \leq z \leq 1$ we can do no better than to play this strategy.

To summarize, our analysis showed that in cases where a mixed weak Nash EP exists, RMM_S *will try* to converge to it. If no mixed EP exists, then RMM_S *will* converge to the pure Nash EP. The mixed weak Nash EP solution might not be pareto-optimal and might therefore seem "wrong" to us, but it is a Nash EP solution. Such a solution can be justified by considering the agents as "risk averse". Such agents try to choose a strategy that minimizes their possible losses no matter what the other agent does. The strategy that satisfies these constraints best is the weak EP strategy. In this section, we also provided a direct mathematical method for finding the EP solution that RMM_S is *trying* to converge to.

6.1 Non-symmetric games

Non-symmetric games can be represented by using two curves, one for each payoff matrix. The solution will then alternate between each curve. In such cases, we will see an initial (for small k) unpredictable movement of the solution value as it shifts back and forth between the changing curves. But as k increases, these curves take on a definite shape over the interval $0 \leq x \leq 1$. Effectively, the final curves ($k = \infty$) can have one of four shapes over this interval:

$$f(x) = 0 \tag{10}$$

$$f(x) = 1 \tag{11}$$

$$f(x) = \begin{cases} 0 & \text{if } x < c \text{ for some } 0 < c < 1 \\ 1/2 & \text{if } x = c \\ 1 & \text{if } x > c \text{ for some } 0 < c < 1 \end{cases} \tag{12}$$

$$f(x) = \begin{cases} 1 & \text{if } x < c \text{ for some } 0 < c < 1 \\ 1/2 & \text{if } x = c \\ 0 & \text{if } x > c \text{ for some } 0 < c < 1 \end{cases} \tag{13}$$

The values RMM_S is trying to converge to are those points where the two curves cross $y = x$. If we examine the 16 possible combinations (i.e. 4×4) of final curves, we see that there are only four types of behavior the final solution can show. It can be stable at either 0 or 1, it can cycle between these two, or it can cycle repeating each number twice (e.g. 0,0,1,1,0,0...). We should point out that all these behaviors were previously observed when doing experiments with RMM_S. Our reasoning provides an explanation for why RMM_S always settled on one of these behaviors. We show some of the possible combinations in Figure 4. As can be seen, for some combinations, the final behavior might depend on the initial guess. For example, the initial guess might determine if the solution stabilizes at 0 or at 1. Notice, however, that none of the possible final behaviors allows for any other solution besides 0 or 1.[4] We can then see that it is impossible for RMM_S to converge

[4] Actually, this is not completely true, if both functions have the same $c = 1/2$ then this point will be a trembling hand equilibrium. The payoff matrix would also have to be symmetric. However, since this is not a stable equilibrium we will ignore it.

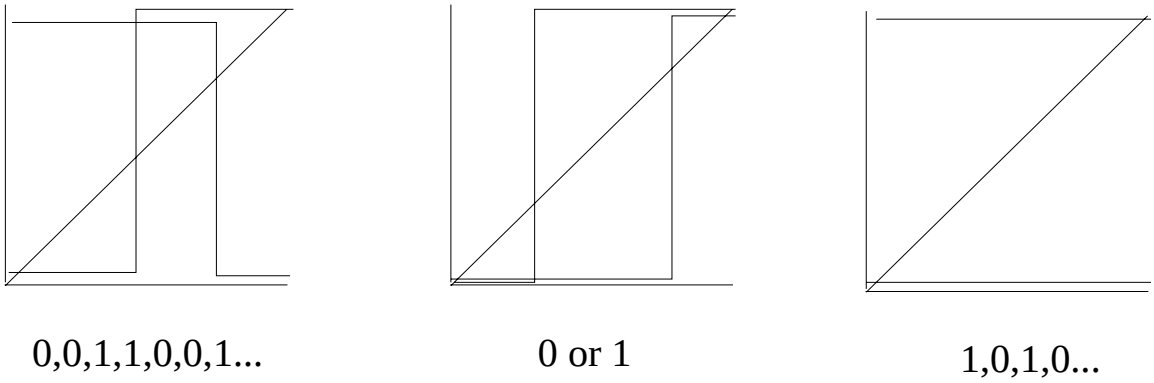

Figure 4: We show some of the final configurations that might be arrived at by the curves in a non-symmetric game. In each case the solution will go up to one curve then horizontally to the line $y = x$ then vertically to the other curve and back. We can see how this behavior of the solution will lead only to the specified behaviors. In the second example, the final behavior will depend on the initial guess.

to a mixed solution when it is modeling two players with different payoffs.

Furthermore, since the shapes of the curves directly map to the existence and type of Nash EPs, we can predict, given the number of Nash EP that exist, what the final behavior of RMM$_S$ will be. We can do this because we know that if there is one mixed Nash EP then the curves will have a step shape but one will go from 0 to 1 while the other one will go from 1 to 0 (i.e. the leftmost case on Figure 4). If there are two pure and one mixed EP we have the middle case of Figure 4, and if there is one one pure EP this means that both curves are either zero or one. Now that we know this it is easy to deduce the following:

- If there is only one mixed EP, the RMM$_S$ solution will have each agent alternatively playing 0 and 1 (i.e. the 0,0,1,1 cycle).
- If there are 2 pure and 1 mixed EP, the RMM$_S$ solution will, depending on initial conditions, converge on one of the pure solutions.
- If only one pure EP exists, then the RMM$_S$ solution will return this solution.

7 Summary

In the RMM$_S$, paper it was argued that we should not let $k = \infty$ all the time (i.e. the original RMM formulation) because then RMM$_S$ would be overeager and converge to a solution too quickly. By letting k increase slowly, it was hoped that RMM$_S$ would converge on the mixed Nash EP, instead of cycling among the pure equilibriums. As we have seen in the graphs of the sigmoid equations when $k \to \infty$, the point RMM$_S$ is trying to converge to is $0 < x < 1$ where the equation $f(x)$ (i.e. the payoff function with $k = \infty$) falls from 0 to 1 or from 1 to 0. If point x does not lie in the interval $0 < x < 1$, then RMM$_S$ returns the value that $f(x)$ had on the 0 to 1 interval.[5] We showed that this value, the value RMM$_S$ is trying to return, is a Nash EP. We also determined that if more

[5]Please remember that when $k = \infty$ then function $f(x)$ is 0 or 1 for all values except for the crossover point, where it has a value of 1/2

than one EP exists then the one returned will be a weak mixed Nash EP. The weak Nash EP might have lower expected payoff than the other pure Nash EPs.

We have also shown that RMM_S will find it very hard to converge on the EP solution for symmetric games because the intersection point will stop being an attractor for rather small exponent values and the solution will not be reached until the exponent equals infinity.

For non-symmetric games, we showed that it will be impossible for RMM_S to converge on the mixed strategy solution because, when the exponent reaches infinity (or approaches it) the only possible behaviors for the solution are to either cycle or stay at one of the pure solutions.

Finally, we found that the same method used to characterize the solution that RMM_S is trying to reach can be used to determine it. That is, we can use the equations given before to find the solution that RMM_S is trying to converge to (for the 2×2 case), which eliminates the need to use RMM_S for this task.

7.1 Further work

This work makes clear the limitations of RMM_S and argues for the abandonment of the sigmoid function as a way to reconcile the solution concept of RMM with the Nash EP solution. We recognize that the solution that RMM arrives at is correct when we have a finite amount of knowledge, all of which is exploited by RMM. In these cases, the depth of our knowledge, and therefore the depth of the hierarchy, is finite and completely understood. RMM can traverse this hierarchy and retrieve the best strategy to play. However, when there is common knowledge present or when there is not enough time to traverse the whole hierarchy, RMM will return a value that might not coincide with, or even approximate, the Nash EP solution. The reason is that RMM in its original form, has absolute certainty about which is the last level in the hierarchy. The sigmoid function was an attempt at reducing this certainty but, as we have shown, it is not a stable solution.

The newest version of RMM [4] incorporates some of the lessons learned from this work. It does not have a sigmoid function and, instead, it chooses to explicitly represent the uncertainty about the depth of knowledge. This is accomplished with the use of probabilities. The resulting model does not require the iterative deepening of RMM levels or the detection of convergence or cycles. It derives a single solution using the explicit model of all knowledge and uncertainties.

This new model adds more information to the RMM hierarchy which makes it even more costly and time consuming to compute a solution. The complexity is especially troublesome for cases, like the ones we dealt with in this paper, where a cheap, game theoretic solution is a valid solution as long as we are willing to assume common knowledge. It is arguably impossible to achieve common knowledge[5]. However, we propose that there are cases where it might be of higher utility to assume common knowledge and arrive at an answer quickly and cheaply using some simple EP method, rather than use RMM. The question we are left with is: When is it appropriate to assume common knowledge? How many levels of nesting are enough for us to justify the assumption of an infinite level of nesting?

These questions can only be answered if we also look at the expected payoffs versus the cost of computation. This issue also arises when we try to implement RMM for a real-time application. The fact that RMM must always exhaustively search through all its knowledge base for a particular situation is quite restrictive. Further complications arise by the fact that in some situations, the payoffs will change as a function of time, so that the time spent thinking is valuable. What we really want is for RMM to return the best answer it can get given a limited amount of time and processing power. We are currently using some of the work done on limited rationality [6] to expand RMM so that it will provide this functionality. The method involves the use of expectation functions which calculate the value of expanding a node in the hierarchy. A node will only be expanded if its value

is sufficiently large (i.e. we have something to gain). Using this technique, we hope to develop an algorithm for expanding the RMM hierarchy in such a way that there will always be an answer readily available (the one we currently believe to be the best). Nodes in the hierarchy that do not contribute to the answer are not expanded. We stop expanding the hierarchy when there is no more to be gained. That is, either the solution will not change, or the cost of expanding a node is greater than any payoff benefits.

References

[1] Ken Binmore. Modelling rational players, part 1. *Economics and Philosophy*, 3:179–214, 1987.

[2] Ken Binmore. Modelling rational players, part 2. *Economics and Philosophy*, 4:9–55, 1988.

[3] Edmund H. Durfee, Jaeho Lee, and Piotr J. Gmytrasiewicz. Overeager reciprocal rationality and mixed strategy equilibria. In *Proceedings of the eleventh National Conference on Artificial Intelligence*, 1993.

[4] Piotr J. Gmytrasiewicz, Edmund H. Durfee, and David K. Wehe. A decision-theoretic approach to coordinating multiagent interactions. In *Proceedings of the twelfth international joint conference on artificial intelligence*, 1991.

[5] Joseph Y. Halpern and Yoram Moses. Knowledge and common knowledge in a distributed environment. In *Proceedings of the 3rd ACM Conference on Principles of Distributed Computing*, 1984.

[6] Stuart Russell and Eric Wefald. *Do The Right Thing.* The MIT Press, Cambridge, Massachusetts, 1991.

[7] Robert Savit. When random is not random: An introduction to chaos in market prices. *Journal of Futures Markets*, 1988.

[8] Martin Shubik. *Game theory in the social sciences.* The MIT Press, Cambridge, Massachusetts, 1985.

A Appendix

A.1 Proof of theorem 1

We let our opponent's payoff matrix be M such that:

$$M = \begin{pmatrix} e_{1,1} & \cdots & e_{1,n} \\ \vdots & & \vdots \\ e_{m,1} & \cdots & e_{m,n} \end{pmatrix}$$

We also let p_i be the probabilities that player A chooses move i, similarly we let q_i be the probabilities that player B chooses move i, like so:

$$\begin{array}{c} \\ p_1 \\ \vdots \\ p_m \end{array} \begin{array}{c} \begin{array}{ccc} q_1 & \cdots & q_n \end{array} \\ \begin{pmatrix} e_{1,1} & \cdots & e_{1,n} \\ \vdots & & \vdots \\ e_{m,1} & \cdots & e_{m,n} \end{pmatrix} \end{array}$$

We can now calculate the payoff P to player A:

$$P = \begin{array}{ccc} p_1 q_1 e_{1,1} + & \cdots & + p_1 q_n e_{1,n} + \\ \vdots & & \vdots \\ + p_m q_1 e_{m,1} + & \cdots & + p_m q_n e_{m,n} \end{array} \tag{14}$$

The differentiation method now tells us to take the derivative of P with respect to each p_i and set all of them equal to zero in order to find the desired solution vector q_i. We do this for some valid p_i and notice that since

$$p_m = 1 - p_1 - p_2 - \cdots - p_{m-1} \tag{15}$$

we can substitute for p_m in our payoff function P. We then take the derivatives with respect to p_i, for $i < m$, and set them to zero.

$$\frac{\partial P}{\partial p_i} = q_1 e_{i,1} + \cdots + q_n e_{i,n} - q_1 e_{m,1} - \cdots - q_n e_{m,n} = 0 \tag{16}$$

Which implies that the differentiation method finds an answer vector q such that for all $i < m$ the following will be true:

$$q_1 e_{i,1} + \cdots + q_n e_{i,n} = q_1 e_{m,1} + \cdots + q_n e_{m,n} \tag{17}$$

Now, given that player B is playing q we can calculate the expected payoffs for player A as follows:

$$\begin{pmatrix} e_{1,1} & \cdots & e_{1,n} \\ \vdots & & \vdots \\ e_{m,1} & \cdots & e_{m,n} \end{pmatrix} \cdot \begin{pmatrix} q_1 \\ \vdots \\ q_n \end{pmatrix} = \begin{pmatrix} q_1 e_{1,1} + \cdots + q_n e_{1,n} \\ \vdots \\ q_1 e_{m,1} + \cdots + q_n e_{m,n} \end{pmatrix} \tag{18}$$

If we now look back at Equation 17 we can see that all the terms on the payoff vector will be the same. Therefore, vector q is always chosen so that the payoffs for the other player will be the same no matter which play he chooses.

A.2 Proof of Corollary 1

Corollary 1 can be derived by following the formulas in the proof of Theorem 1 but using the specific 2×2 matrix instead of the general $m \times n$ matrix we used.

A.3 Proof of Theorem 2

By looking at Equation 18 we can see that the strategy chosen by one of the players is such that the other players receives the *same* payoffs no matter which move he chooses. Since this is true for both players we have a situation where, if any one player decides to play something else, different from the EP solution, his payoffs will remain the same and will not decrease. If we now look at the definition of a weak Nash EP (i.e. where the players can deviate from the solution without reducing their payoffs) we will see that the definition applies to this case and the solution is indeed a weak Nash EP.

A.4 Equations for the sigmoid

The following is the sigmoid function for the 2×2 case:

$$\frac{(ax + (1-x)\,b)^k}{(ax + (1-x)\,b)^k + (cx + (1-x)\,d)^k} \tag{19}$$

The following is the slope of the sigmoid function, that is, the derivative of f with respect to x.

$$\frac{\frac{(ax+(1-x)b)^k k(a-b)}{(ax+(1-x)b)\left((ax+(1-x)b)^k+(cx+(1-x)d)^k\right)} - (ax + (1-x)\,b)^k}{\left(\frac{(ax+(1-x)b)^k k(a-b)}{ax+(1-x)b} + \frac{(cx+(1-x)d)^k k(c-d)}{cx+(1-x)d}\right)\left((ax + (1-x)\,b)^k + (cx + (1-x)\,d)^k\right)^{-2}} \tag{20}$$

For a given x the following function returns the value of k for which the sigmoid function at x will equal x.

$$\frac{\ln(1-x) - \ln(x)}{\ln(cx + (1-x)\,d) - \ln(ax + (1-x)\,b)} \tag{21}$$

Finally the value of the slope at the intersection point, as a function of x, can be found by replacing Equation 21 for k in Equation 20.