Compromise Strategies for Action Selection: Supplementary Material

Frederick L. Crabbe

U.S. Naval Academy, Computer Science Department, 572M Holloway Rd, Stop 9F, Annapolis, MD 21402 U.S.A. (crabbe@usna.edu)

A Dynamic Programming Algorithm Specification

The dynamic programming technique used in the paper overlays a grid of points on top of the problem space and calculates the maximum expected utility of each location given optimal future actions. This is done recursively starting at the target locations and moving outward until values for all points are calculated, and then repeating the process until the values converge. Using the equations given in the main paper, the expected utility of each grid point in the environment can be calculated provided the $EU(A_\theta|t_a, t_b, \lambda')$ function can be accurately determined and θ can be found. As the EUs are only known at grid points, the local EU function must be interpolated from the EU values of the surrounding points in order to determine the EU of a movement in an arbitrary direction. Using the interpolated surfaces for the local values of $EU(A_\theta|t_a, t_b, \lambda')$, the value of θ can be determined by searching for the angle that maximises the function described by equation (3). Once the expected utility is determined for a grid point, its value is then used to calculate the expected utility of its neighbouring grid-points. This process is repeated until values are determined for all the grid points. Because the estimated utility value can change for a point when the values of its neighbours change, the values of all the points are repeatedly re-estimated until the values stabilise. The pseudocode for the algorithm is given in figure 4. Implementation details follow.

Open: list of grid-points to be updated.
Closed table of updated points.
N is the point currently being updated.
V_N is the current EU estimate at N.
repeat
  Open $\leftarrow$ enqueue target locations
  Closed $\leftarrow \emptyset$
  repeat
    $N \leftarrow$ dequeue from *Open*
    when $N \notin$ *Closed*
      $EU(A_\theta|t_a, t_b, \lambda') \leftarrow$ interpolated EU function from neighbours
      $V_N \leftarrow \max_\theta p^2 EU(A_\theta|t_a, t_b, \lambda') + p(1-p)EU(A_\theta|t_a, \lambda') + p(1-p)EU(A_\theta|t_b, \lambda')$
      Open $\leftarrow$ enqueue neighbours of N
      Closed $\leftarrow$ add N
  until *Open* $= \emptyset$
until convergence

Figure 4: The dynamic programming algorithm for estimating the expected utility at all the grid points in the environment. Description is found in the text.

A.1 Interpolation

The algorithm for approximating the optimal strategy is straightforward, but contains details that must be carefully attended. The most important of these are the interpolation of the expected utility function and finding the maximum of equations (3) or (6).

With respect to the interpolation, the primary question is what class of function to use as the interpolating function. Equations (3) and (6) are nonlinear, and thus might be inaccurately modelled using a linear function. Interpolation using a cubic or higher order function would require a large number

of sample points and greatly increase the complexity. As will be seen, even the quadratic case can be complex. The next two sections will consider and compare the costs and benefits of quadratic and linear interpolation.

A.1.1 Quadratic Interpolation

Assuming a six term quadratic:

$$z = Ax^2 + By^2 + Cxy + Dx + Ey + F \qquad (9)$$

six points must be selected to determine the coefficients. Figure 5 depicts this scenario for estimating the EU function surrounding the point $(0, 0)$ in the first quadrant; the other quadrants are reflections of this (in the algorithm, treating each point under consideration as the origin of the system and transforming the targets and dangers to this new coordinate frame simplifies both coding and explanation). From the origin, the agent can move to any point on the circle one unit from the origin, thus it is most important for the interpolated function to be accurate along that circle. The six points that are closest to the circle change depending upon the portion of the circle in question. In the angle range $(0; \frac{\pi}{6}]$, the six closest points are $\{(1, 0), (1, 1), (2, 0), (1, -1), (0, 1), (2, 1)\}$. In the range $(\frac{\pi}{6}; \frac{\pi}{4}]$, the closest points are: $\{(1, 0), (1, 1), (2, 0), (1, 2), (0, 1), (2, 1)\}$. In the range $(\frac{\pi}{4}; \frac{\pi}{3}]$, the closest points are $\{(1, 0), (1, 1), (0, 2), (1, 2), (0, 1), (2, 1)\}$, and in the range $(\frac{\pi}{3}; \frac{\pi}{2}]$, the closest points are $\{(1, 0), (1, 1), (0, 2), (1, 2), (0, 1), (-11)\}$. At the angles $\frac{\pi}{6}, \frac{\pi}{4}$, and $\frac{\pi}{3}$, the circle is equi-distant between the two points not in the neighbouring regions, so points can be chosen arbitrarily.

Thus, to use quadratic interpolation, the function must be interpolated sixteen times around the circle, the maximum must be found of equations (3) or (6) in each of the regions, and then the maximum of these maxima must be found. Due to the computational expense of finding the maxima (see below), this can be a costly overall process, encouraging the consideration of using the linear interpolation for faster results.

A.1.2 Linear Interpolation

When examining the closest three points for linear interpolation, the circle is partitioned into three segments: $(0; \frac{\pi}{8}]$, $(\frac{\pi}{8}; \frac{3\pi}{8}]$, and $(\frac{3\pi}{8}; \frac{\pi}{2}]$. In the first and third range, the area inside the triangle made by those three points, ($\{(1, 0), (1, 1), (2, 0)\}$ in the case of first region), does not contain the portion of the circle in question, making the use of those points an extrapolation rather than interpolation. Thus, it is preferable (i.e. more accurate) to use the same 3 points, $\{(1, 0), (1, 1), (0, 1)\}$, for the entire quadrant.

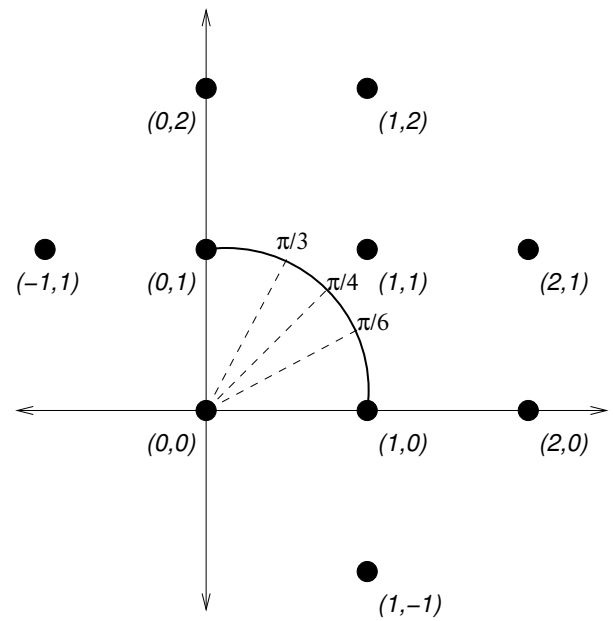

Figure 5: The points used for interpolation.

This has the added benefit of simplifying the interpolation and maximisation process greatly.

If considering the use of linear interpolation, it is helpful to know its relative accuracy compared to the quadratic interpolation. While it is difficult to determine the "correct" values of the function in the general two target case (that is the function the interpolation is being used to find), one can can compare the interpolated values to a calculated value in a single target case. Assume that a single target is located at location $(10, 10)$, with a utility of 1 and p is 0.99, and the EU at location $(0, 0)$ must be determined. The true EU values of the surrounding points can be calculated using equation (4), and then can be used for interpolation. In this case, the true EU of a point on the circle can also be calculated, such as at angle $\frac{\pi}{4}$, using the same equation. The true EU at $\frac{\pi}{4}$ is 0.87626836. Using quadratic interpolation, the estimated value of $\frac{\pi}{4}$ is 0.87626817. Using the linear interpolation, the estimated value at that same point is 0.87617810. Here the quadratic interpolation yields a better value.

Further, the linear interpolation can produce poor maxima in certain situations. In particular, take the example above, but with the target at location at location $(0.2, 10)$. In this case, the calculated value at $(0, 1)$ is the largest of the three points used for interpolation, and therefore that value (and the direction $\frac{\pi}{2}$) will be the maximum of a linear interpolated function in the first quadrant. In this situation, the direction of the true maximum is 1.55079899 radians. The correct EU value is 0.91349888. Quadratic interpolation yields a value of 0.91349886, and linear interpolation yields 0.91348892. This problem can

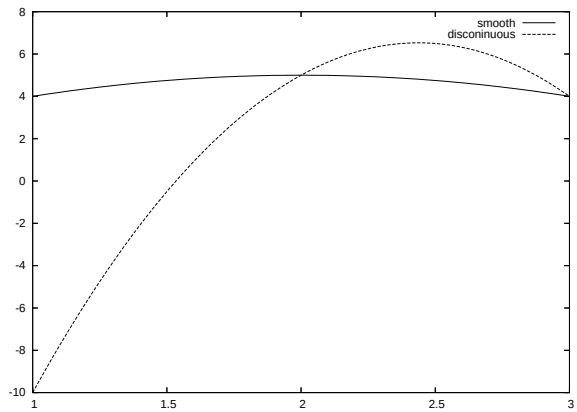

Figure 6: Two interpolated quadratic functions. For each function, the x values used for interpolation are 1,2, and 3. The function describing a smooth space uses y values of 4, 5, and 4, respectively. The function describing the discontinuous space uses y values of -10, 5, 4. Note how the one that crosses the discontinuity has a high crown which often over-estimates the true value in the range (2;3).

be partially addressed by using different interpolation points when this situation occurs. If the three interpolation points are all on one side of the unit circle, and the middle point is the largest, then the linear interpolation can find a maximum at an arbitrary angle away from the largest point. It is only when one of the corner points is the largest of the three that this false can maximum occur. Thus, if the point $(0, 1)$ is the largest of $\{(1, 0), (1, 1), (0, 1)\}$, the used points are shifted to be $\{(-1, 0), (0, 1), (1, 0)$. In the case with the target at $(0.2, 10)$, this new version of linear interpolation yields a value of 0.91349854, and a maximum around 1.551 radians. Experience has shown that linear interpolation of real problems agrees with the quadratic interpolation to three decimal places.

There is also a difficulty with quadratic interpolation, namely it is very sensitive to discontinuities in the EU space. If one of the points used for interpolation has a value that is widely different from the other values, then the interpolated function becomes very steep, resulting in values too large by a great deal (figure 6). This problem is exacerbated by the number of points used, making it harder to avoid crossing such discontinuities. It is impossible to use only those interpolation points on one side of the discontinuity because the location of the discontinuities are unknown. Thus, while the quadratic does tend to be more accurate, it can be inaccurate in certain situations, especially when there are prescriptive goals that can cause discontinuities. Because of this issue and because linear was significantly faster than quadratic, linear interpolation was used in the results reported here.

A.2 Maximisation

Once an equation has been interpolated, it must be combined with the other terms in equation (3) or (6), yielding in the case of equation (3), and assuming λ is $(0, 0)$:

$$\begin{aligned} EU(A_i|t_a, t_b, \lambda) =& p^2(Ax + By + C)+ \\ & p(1-p)G_a p^{\sqrt{(x-t_a^x)^2+(y-t_a^y)^2}}+ \\ & p(1-p)G_b p^{\sqrt{(x-t_b^x)^2+(y-t_b^y)^2}}, \end{aligned} \tag{10}$$

to be maximised along the constraining function

$$0 = x^2 + y^2 - 1.$$

Closed form techniques require taking the derivative of equation (10). It can be parameterised, yielding,

$$\begin{aligned} & EU(A_i|t_a, t_b, \lambda) = \\ & p^2(A\cos(t) + B\sin(t) + C)+ \\ & p(1-p)G_a p^{\sqrt{(\cos(t)-t_a^x)^2+(\sin(t)-t_a^y)^2}}+ \\ & p(1-p)G_b p^{\sqrt{(\cos(t)-t_b^x)^2+(\sin(t)-t_b^y)^2}}, \end{aligned} \tag{11}$$

but the derivative of (11) (shown in in figure 7) is not solvable by algebraic methods. Similarly, the partial derivatives of equation (10) are nearly as complex, thus eliminating the use of Lagrange multipliers (Bertsekas, 1982).

Instead, the maximum must be found using numerical methods. This can be done using any of the standard techniques, but because equation (12) is complex, using it for a numerical technique is computationally expensive. Instead, the maximum of the function can be found using a method that does not take the derivative at all, such as Brent's method (Press et al., 2002). The experiments in the paper use Brent's method to find the maximum of the parameterised version of the EU equation (equation (11)) so that it could operate with just one variable.

The method assumes that it can bracket the maximum and that the function is parabolic within that bracket. This is a reasonable assumption in this case because the ranges being looked at at are small (one must find the maximum of each quadrant separately because EU function was interpolated separately) and smooth.

Brent's Method takes two end points of the region and a point in the middle, interpolates a parabola, and uses the critical point on the parabola as the estimate of the maximum. It uses this new point to replace one of the original three points, interpolates a new parabola, continuing until convergence. Because of this strategy, if the function is concave

$$
\begin{aligned}
&0 = \\
&p^2(B\cos(t) - A\sin(t)) + \\
&\frac{G_a(1-p)p^{1+\sqrt{(-t_a^x+\cos(t))^2+(-t_a^y+\sin(t))^2}}\log(p)(-2(-t_a^x+\cos(t))\sin(t)+2\cos(t)(-t_a^y+\sin(t)))}{2\sqrt{(-t_a^x+\cos(t))^2+(-t_a^y+\sin(t))^2}} + \\
&\frac{G_b(1-p(p^{1+\sqrt{(-t_b^x+\cos(t))^2+(-t_b^y+\sin(t))^2}}\log(p)(-2(-t_b^x+\cos(t))\sin(t)+2\cos(t)(-t_b^y+\sin(t)))}{2\sqrt{(-t_b^x+\cos(t))^2+(-t_b^y+\sin(t))^2}}
\end{aligned}
\tag{12}
$$

Figure 7: The derivative of equation (11)

up, then the method will find the minimum, and if it as concave down it will find the maximum. If the method finds a minimum rather than a maximum, one can return the larger of the two end points as the maximum of that quadrant. Once the maxima of the quadrants are found, the maximum of the function is the largest of those four. This can then be used as the value of $EU(A_\theta|t_a, t_b, \lambda)$ in equation (3) or (6).

B Prescriptive Analysis Experiments

While on average, low-level compromise appears to have little benefit, there may be scenarios where the Optimal or EWF strategies are significantly better than the MEU. This section uses the idea of information gain from information theory to attempt to determine if this is so.

Information gain is the technique used in the decision tree algorithms ID3 and C4.5 (Mitchell, 1997; Quinlan, 1983; Quinlan, 1993). The technique begins with a data set and a classification of the data elements. The data here are the situations generated in the prescriptive experiments, and the classification is whether or not the scenario has a compromise strategy (either EWF or optimal) with an EU better than MEU. The technique then considers a set of n-ary attributes which partition the data set. For each attribute, the technique considers the information gain of applying that attribute to partition the data set. Information gain is the decrease in entropy of the data set, where entropy is:

$$E(S) = \sum_{i \in C} -p_i \log_2 p_i \tag{13}$$

S is a data set, i is a category of the classification, and p_i is the proportion of S categorised as i. Entropy measures the inverse of the purity of a data set with respect to the classification. A set is most pure when all the data have the same classification, and least pure when the data is classifies evenly across all categories.

The information gain of dividing set S based on attribute A is defined as:

$$Gain(S, A) = E(S) - \sum_{v \in V(A)} \frac{|S_v|}{|S|} E(S_v), \tag{14}$$

where v is a value of an attribute, $|S|$ is a size of set S, S_v is the subset of S for which attribute A takes value v, and $V(A)$ is the set of values the attribute might take (e.g. the attribute *colour* might take the values red, green, or blue). The attribute with the largest information gain is the attribute which best partitions the data set, and in our case is the property of a situation which most determines if there is a good compromise candidate.

The data describing a situation is continuous information, but attributes are n-ary classifiers. To cope with this, the attributes partition the values into ranges, such as every 0.05. The attributes used are listed below:

- $\overline{\lambda t_a}, \overline{\lambda t_b}, \overline{t_a t_b}$. These three attributes are the distances between the key locations in the scenario. The distances can be anywhere from 0 to 224, and the attributes break them up into ranges of 5 units.
- G_a, G_b. The goal values from 0 to 100, broken into ranges of 5 units.
- p. The value of p from 0 to 1, broken into ranges of 0.005.
- $R(\overline{\lambda t_a}, \overline{\lambda t_b})$. This encodes the relationship between $\overline{\lambda t_a}$ and $\overline{\lambda t_b}$, while keeping the values between 0 and 1 using the formula $\frac{\overline{\lambda t_a}}{\overline{\lambda t_a}+\overline{\lambda t_b}}$. It is broken into ranges of 0.05.
- $R(\overline{\lambda t_a}, \overline{t_a t_b})$. This encodes the relationship between $\overline{\lambda t_a}$ and $\overline{t_a t_b}$, while keeping the values between 0 and 1 using the formula $\frac{\overline{\lambda t_a}}{\overline{\lambda t_a}+\overline{t_a t_b}}$. It is broken into ranges of 0.05.
- $R(\overline{\lambda t_b}, \overline{t_a t_b})$ This encodes the relationship between $\overline{\lambda t_b}$ and $\overline{t_a t_b}$, while keeping the values between 0 and 1 using the formula $\frac{\overline{\lambda t_b}}{\overline{\lambda t_b}+\overline{t_a t_b}}$. It is broken into ranges of 0.05.

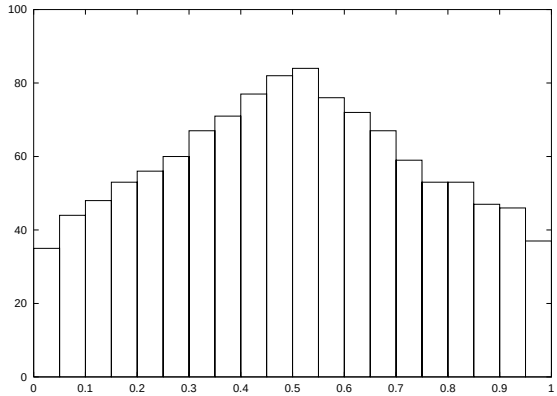

Figure 8: Breakdown of attribute $R_e(G_a, \overline{\lambda t_a}, G_b, \overline{\lambda t_b})$ when comparing optimal to MEU.

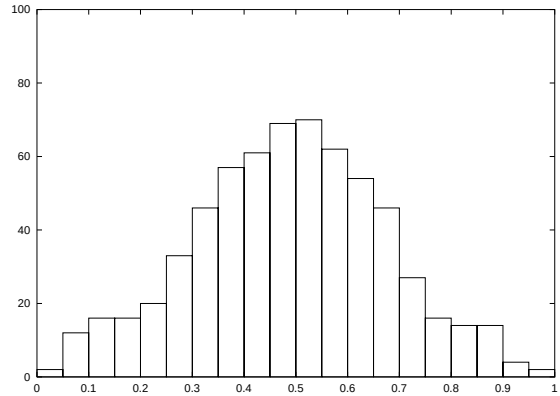

Figure 9: Breakdown of attribute $R(\overline{\lambda t_a}, \overline{\lambda t_b})$ when comparing optimal to MEU.

- $R(G_a, G_b)$. This encodes the relationship between G_a and G_b, while keeping the values between 0 and 1 using the formula $\frac{G_a}{G_a+G_b}$. It is broken into ranges of 0.05.
- $R_l(G_a, \overline{\lambda t_a}, G_b, \overline{\lambda t_b})$ This encodes a *linear* relationship between $G_a, \overline{\lambda t_a}, G_b$, and $\overline{\lambda t_b}$, while keeping the values between 0 and 1 using the formula $\frac{G_a \overline{\lambda t_a}}{G_a \overline{\lambda t_a} + G_b \overline{\lambda t_b}}$. It is broken into ranges of 0.05.
- $R_e(G_a, \overline{\lambda t_a}, G_b, \overline{\lambda t_b})$ This encodes a *exponential* relationship between $G_a, \overline{\lambda t_a}, G_b$, and $\overline{\lambda t_b}$, while keeping the values between 0 and 1 using the formula $\frac{G_a p^{\overline{\lambda t_a}}}{G_a p^{\overline{\lambda t_a}} + G_b p^{\overline{\lambda t_b}}}$. It is broken into ranges of 0.05.

B.1 Optimal vs. MEU

The data is first classified based on whether the performance of the optimal behaviour is a 1% or greater improvement over the performance of MEU for that scenario. Using equation (14), it can be seen that attribute with the largest information gain is $R_e(G_a, \overline{\lambda t_a}, G_b, \overline{\lambda t_b})$ with a gain of 0.07243. Figure 8 shows a breakdown of the data as partitioned by the attribute. The x-axis is the range of values of the attribute, and the y-axis is the percentage of the data that falls in that range and is positively classified (the performance of the optimal behaviour is a 1% or greater improvement over the performance of MEU). The figure shows that the greatest likelihood of good compromise behaviour is when the attribute is close to 0.5. This means that if $G_a p^{\overline{\lambda t_a}}$ is close to $G_b p^{\overline{\lambda t_b}}$ (when the expected utilities of the two targets are roughly equal), it is more likely that optimal behaviour will confer a large advantage.

The attribute with the next largest information gain is $R(\overline{\lambda t_a}, \overline{\lambda t_b})$ with a gain of 0.06917. Figure 9 shows a breakdown of the data as classified by the attribute. Once again, the greatest likelihood of good compromise behaviour is when the attribute is close to 0.5, or in this case when $\overline{\lambda t_a}$ and $\overline{\lambda t_b}$ are roughly equal.

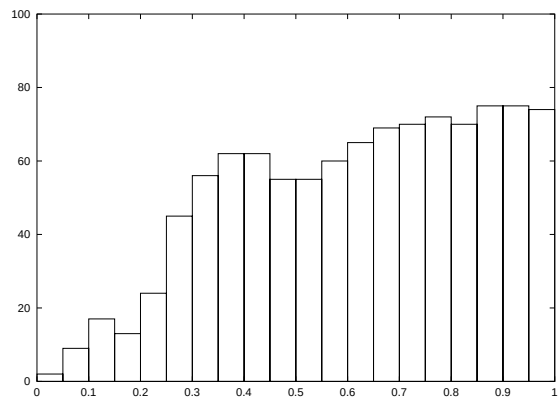

Figure 10: Breakdown of attribute $R(\overline{\lambda t_a}, \overline{t_a t_b})$ when comparing optimal to MEU.

The next two attributes with high gain are $R(\overline{\lambda t_a}, \overline{t_a t_b})$ and $R(\overline{\lambda t_b}, \overline{t_a t_b})$, both with a gain of 0.027. Figure 10 shows a breakdown of the data as classified by $R(\overline{\lambda t_a}, \overline{t_a t_b})$ (the plot of the other attribute is qualitatively identical). While the data in this figure is less distinct than in the previous figures, it does appear to show that there are *few* good compromises when the targets are relatively close to the agent but far away from each other.

B.2 Exponentially Weakening Forces vs. MEU

Although on average EWF performs worse than MEU, it is interesting to analyse the situations in which it performs better. If these situations could be recognised, then EWF could be applied only when they occur. Furthermore, it may lead to insight

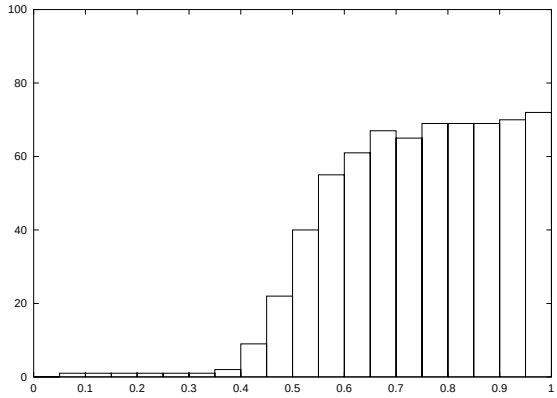

Figure 11: Breakdown of attribute $R(\overline{\lambda t_a}, \overline{t_a t_b})$ when comparing EWF to MEU.

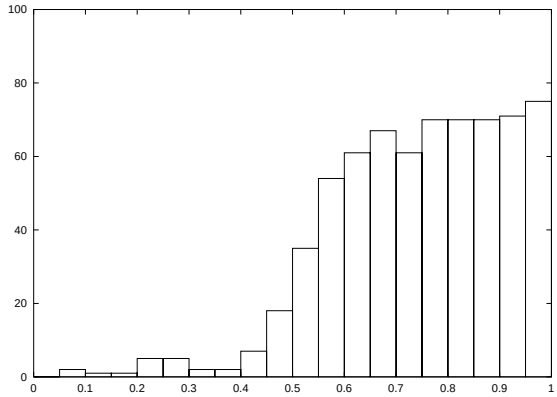

Figure 12: Breakdown of attribute $R(\overline{\lambda t_b}, \overline{t_a t_b})$ when comparing EWF to MEU.

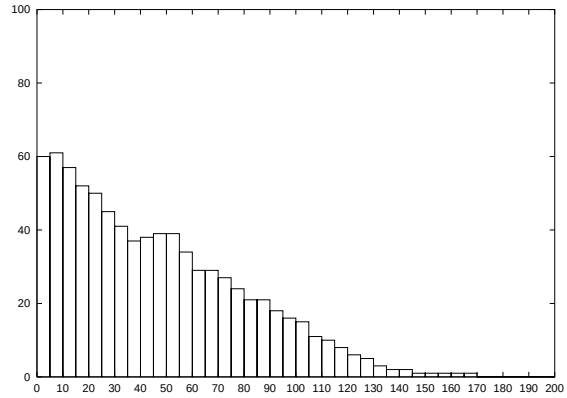

Figure 13: Breakdown of attribute $\overline{t_a t_b}$ when comparing EWF to MEU.

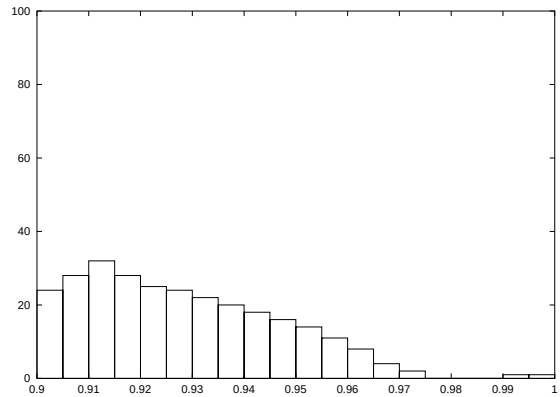

Figure 14: Breakdown of attribute p when comparing EWF to MEU.

on why EWF outperformed the other non-optimal strategies. The data is analysed as in the previous section, and classified based on whether or not EWF out performs MEU. The top two attributes in terms of information gain are $R(\overline{\lambda t_a}, \overline{t_a t_b})$ and $R(\overline{\lambda t_b}, \overline{t_a t_b})$, with gains of 0.1620 and 0.1569 respectively. Figure 11 shows a breakdown of the data as classified by $R(\overline{\lambda t_a}, \overline{t_a t_b})$ and figure 12 shows a breakdown of the data as classified by $R(\overline{\lambda t_b}, \overline{t_a t_b})$.

These graphs strongly suggest that good compromises using EWF occur when the targets are relatively far away from the origin and close to each other. This is backed up by examining the next highest attribute, $\overline{t_a t_b}$ (0.09223) in figure 13. Here, good compromises are more likely when $\overline{t_a t_b}$ is small, that is, the two targets are close together.

The last attribute examined is p, with an information gain of 0.07903. Figure 14 shows its breakdown. It shows that good performance peaks at a p range of 0.91-0.915. This is consistent with previous results that good compromises occur with $p > 0.9$ (Crabbe, 2002). That study only looked at p with a granularity of 0.1, and was thus unable to determine which values of $p > 0.9$ were best.

There is some overlap between the good attributes for the optimal behaviour and for EWF, namely that the targets be relatively close together and far away from the agent. There is no evidence to explain why $R_e(G_a, \overline{\lambda t_a}, G_b, \overline{\lambda t_b})$ does not appear to be a factor for EWF, instead having an information gain of 0.004855.

C Additional Proscriptive Results

When examining the proscriptive scenarios, if p_t is reduced to 0.95, such that the liklihood of the target being in the environment is relatively low (from the position $(60, 30)$ the probability that the target is still there when an agent following the MEU strategy gets there is 0.044), then one might expect that this would cause the agent to move more strongly to the target, even when in the danger zone. Figure 15 shows that this is not the case. Instead, the optimal

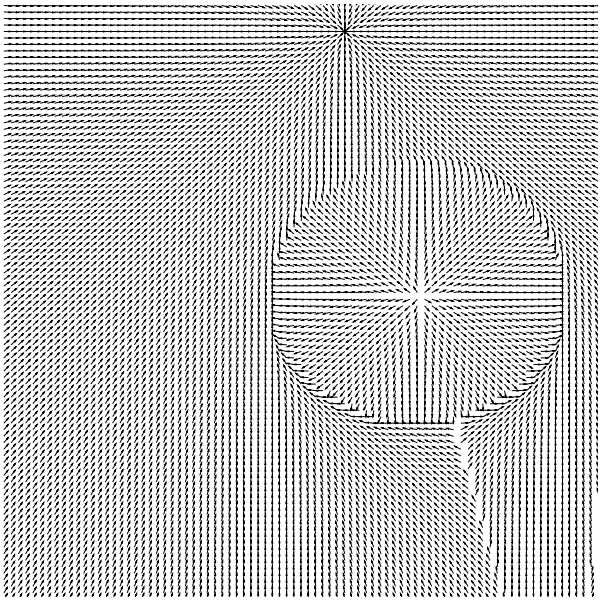

Figure 15: Reducing p_t has no effect on the qualitative behaviour ($p_t = 0.95, p_d = 0.99$, and $p_n(\lambda)$ is Linear A).

strategy continues to be to flee the danger directly.

When using Linear B with high p_t and p_d, the behaviour is identical to figure 2(a). This is surprising because the high liklihood of not being struck was expected to encourage compromise behaviour in the danger zone. Lowering p_t, results in some slight compromise behaviour, seen in figure 16. The figure shows a close up of the top of the danger zone as well as the target. Inside the danger zone there is some slight tilting of the arrows toward the target. When p_t is high but p_d is low (figure 17), this combines with the low probability of a strike to result in obvious compromise behaviour. In this case, when an agent is one unit away from the danger, its overall probability of escaping unscathed is 0.921. When both p_t and p_d are low, the behaviour is similar to figure 17, but more pronounced (figure 18).

With the non-linear $p_n(\lambda)$ functions, compromise action is seen clearly in all cases (figure 19). As predicted there is some more compromise behaviour toward the edges, but there is surprising non-compromise in the centre. This is similar to what is found in cases when $p_n(\lambda)$ is sigmoid, as in figure 3(b).

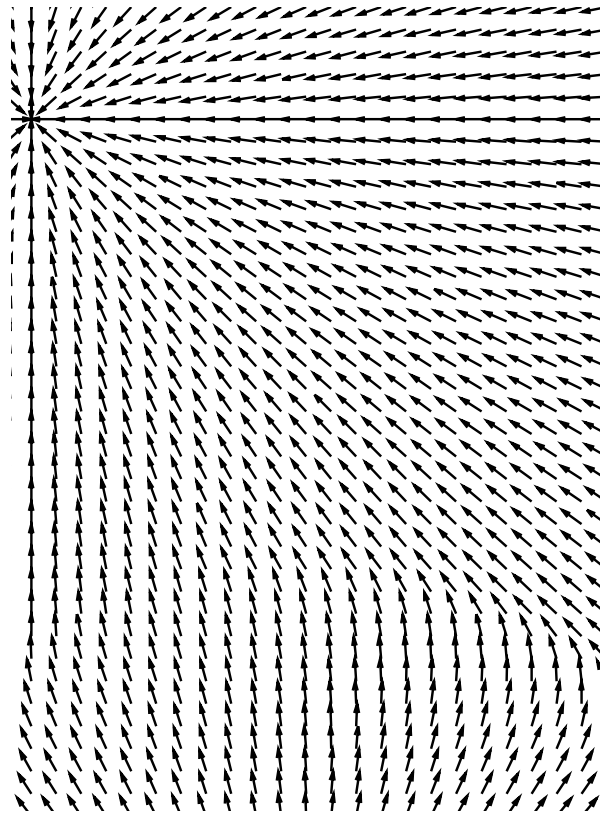

Figure 16: Slight compromise when lowering p_t ($p_t = 0.95, p_d = 0.99$, and $p_n(\lambda)$ is Linear B).

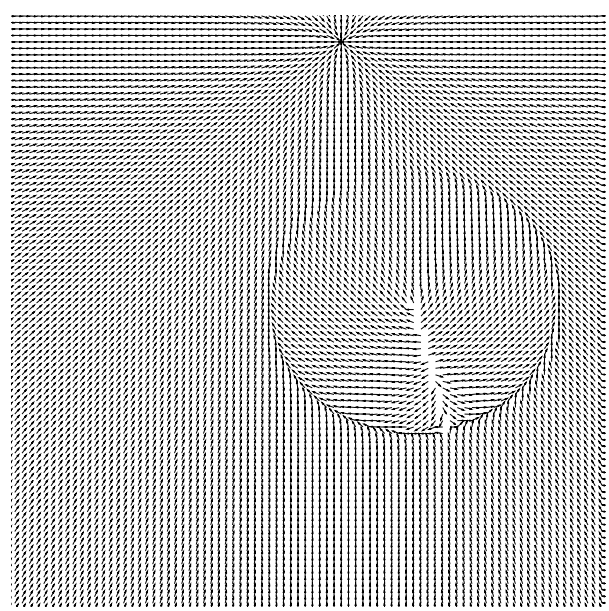

Figure 17: Strong compromise behaviour is seen with low p_d and linear B ($p_t = 0.995, p_d = 0.5$, and $p_n(\lambda)$ is Linear B).

References

Bertsekas, D. P. (1982). *Constrained Optimization and Lagrange Multiplier Methods.* Athena Scientific, Belmont, MA.

Crabbe, F. L. (2002). Compromise candidates in positive goal scenarios. In *From Animals to Animats 7: Proceedings of the Seventh Inter-*

national Conference on Simulation of Adaptive Behavior, pages 105–106.

Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill, Boston.

Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T. (2002). *Numerical Recipes in C++: The Art of Scientific Computing*. Cambridge University Press, Cambridge (UK) and New York, 2nd edition.

Quinlan, J. R. (1983). Learning efficient classification proceedures and their application to chess and games. In Michalski, R., Carbonell, J., and Mitchell, T. M., editors, *Machine Learning: an artificial intelligence approach*. Morgan Kaufmann, San Mateo, CA.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA.

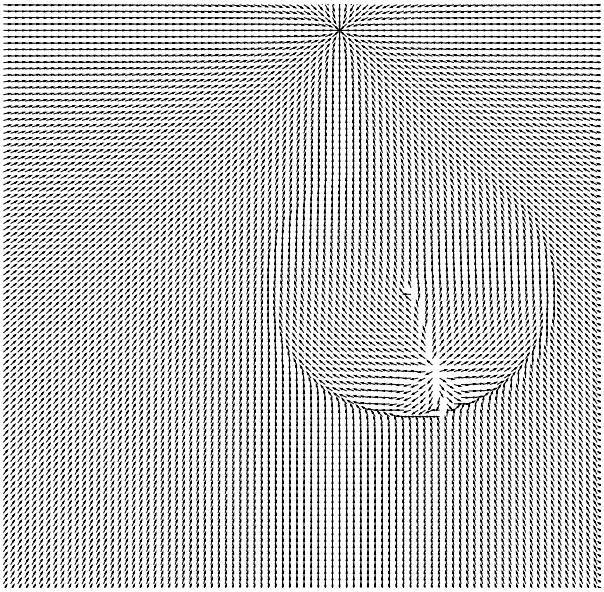

Figure 18: $p_t = 0.95, p_d = 0.5$, and $p_n(\lambda)$ is Linear B.

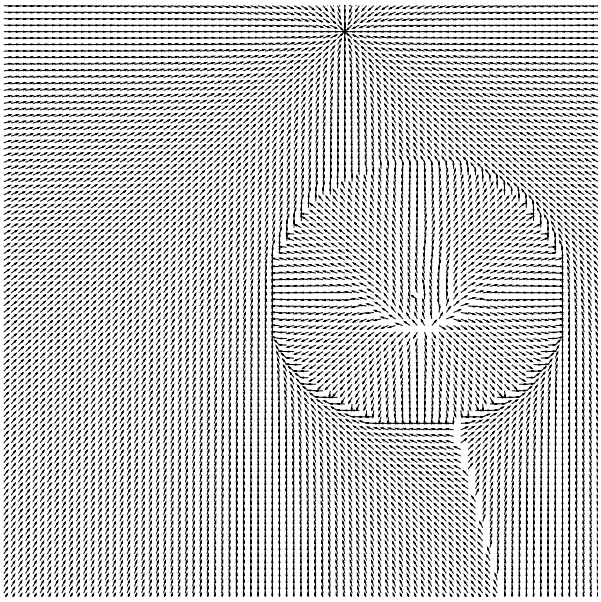

Figure 19: $p_t = 0.995, p_d = 0.99$, and $p_n(\lambda)$ is Quadratic.