Generating an Attribute Space for analyzing Balance in Single Unit RTS Game Combat

Shaun Bangay and Owen Makin
School of Information Technology
Deakin University, Melbourne, Australia

***Abstract*—Achieving balance in a large real-time strategy (RTS) game requires consideration of all contributing factors. We propose an attribute space representation as a common framework for reasoning about balance in the combat scenarios found in these games. Attribute space search is used to identify balanced configurations and game features such as special abilities and terrain act as modifiers to a basis attribute space. The approach is evaluated through the development of an attribute space representation for single unit conflict. An efficient and accurate predictive model is developed corresponding to a simulation of a common RTS combat representation employing the typical attributes of range, speed, health and damage. This attribute space is used to relate the properties of each attribute to the resulting game balance and to identify new balance points when individual attributes are enhanced.**

***Index Terms*—balance, attribute space, RTS games, formal models.**

I. Introduction

Real-time strategy (RTS) games make extensive use of combat between units on opposing sides. Combat outcomes are influenced by the number of units involved, the relative strengths of heterogeneous units involved, strategic use of short term special abilities, the effects of obstacles and terrain and the artificial intelligence that guides each unit. We specifically exclude the role of the player in this categorization as we want to quantify and control the *balance* in an RTS game. A balanced game would have the property that no side has any inherent advantage and that the outcome of the game is purely dependent player actions rather than any implicit advantage through the configuration of units available.

A balanced game is one that has been tuned for fairness and entertainment. RTS games in particular require extensive balance tweaks throughout development and frequently update via patches after analysis of accumulated play statistics. The ability to quantify and manipulate balance in RTS games would benefit the game industry by reducing development time and providing a systematic approach to achieving game balance. It would also improve replayability by allowing unit configuration to be customized while maintaining balance.

Achieving balance is complicated by the range of parameters that need to be considered. Factors that affect RTS game balance include unit attributes, units numbers, unit control modifiers, terrain factors and special abilities. Our goal is to provide a consistent framework in which reasoning about balance can take place. In particular we introduce the concept of an attribute space augmented with a balance metric. The factors that affect balance are categorized as either dimensions of this attribute space or as modifiers to it. We demonstrate the generation of an attribute space for a core RTS scenario and explore how attribute space reasoning can offer insight into the effect of attribute changes. This work is an early step in applying the attribute space reasoning framework to the problem of balance in an RTS context.

A. Attribute Space

This section defines the elements used to create the attribute space and relates these elements to popular RTS games.

Definition 1. Attribute. Each unit is completely specified by providing values of n attributes from a set $A = \{A_k : k \in \{1, 2, \dots n\}\}$. The value of attribute A_k is denoted a_k, thus a unit is defined as the tuple $(a_1, \dots, a_n)$. An attribute is any parameter which can differ between individual units *and* that will affect the outcome of a conflict.

Attributes thus represent the complete unit state with respect to combat. In RTS games such as Starcraft **[20]** explicit unit attributes include ground-attack damage, air-attack damage, attack range, sight range, seed, transport slots, hit points, armor, cool down rate, resource consumption rate and supply rate. In this game all units are subject to the same control routine although we could envisage attributes such as aggressiveness which would change the behaviour of individual units. Games that regularly release updates to improve the balance of their original design do this by modifying individual unit attributes.

Definition 2. Conflict. A conflict C consists of the attribute values of units from t teams. Let m_j be the number of units belonging to team j . $C = \{a_{i,j,k} : i \in \{1, 2, \dots, m_j\}, j \in \{1, 2, \dots, t\}, k \in \{1, 2, \dots n\}\}$, where $a_{i,j,k}$ represents the value of attribute k of the ith unit of team j.

In practical terms a conflict represents the automated (game controlled) battle between several units in an RTS setting after the players have moved them into position. We also approximate the conflict as a self-contained system with no further units entering or leaving and victory going to the last team with surviving units. Solving the balance problem for a complete RTS game, taking all player actions into account, would involve analyzing all reachable conflict outcomes.

The conflict outcome indicates the victorious team. While this could simply be the winner index, we use a more general representation to indicate the extent of balance in the conflict.

Definition 3. Outcome (of a conflict). The outcome r_C of conflict C is the tuple $(b_1, \ldots, b_t) \in \mathbb{R}^t$ where $b_i$is a signed measure of the extent of success enjoyed by team i. A *balanced outcome* is the tuple $(0, 0, \ldots, 0)$.

Balancing a game then becomes a problem of ensuring a *balanced outcome* for a set of possible *conflicts* in the game.

The nature of the measure used varies with the context of the conflict. Deterministic conflict can have its outcome reported as a tuple of binary values, with only one non-zero value corresponding to the winning team. Games with stochastic aspects might use an aggregated probabilistic or fuzzy measure of victory. A metric based on accumulated final health of surviving units indicates not only the winner, but the extent to which the outcome was in doubt.

Definition 4. Balance Augmented Attribute Space (shortened to attribute space). The balance attribute space is a function $a : C \to \mathbb{R}^t$ which provides an outcome for any given conflict.

These definitions make it possible to examine the balance problem in a systematic fashion using a common foundation. Problems relevant to balance include:

1) Let a_{1-1} be the attribute space for conflict between 2 teams with a single unit in each team. How well can the attribute space a_{n-m} for 2 teams with n and m copies of these units (i.e. homogeneous teams) be approximated by modifying a_{1-1}? This would offer insights into the effects of larger swarms of units. Further extensions include incorporating teams of heterogeneous units as found in pioneering RTS games such as Starcraft **[20]**, Age of Empires **[22]** and Warcraft III **[21]**.
2) Given an attribute space a, the effect of deploying special abilities would be represented as changing the position of the current conflict in attribute space. A series of special abilities being deployed by all sides would be represented as a trajectory in attribute space allowing the implications for balance to be quantified. Multiplayer online battle arena (MOBA) games such as League of Legends often involve single unit conflict (from a_{1-1}) with the option to use temporary special abilities.
3) Some game operations do not require additional attributes but instead modify, or constrain, existing attributes. Given an attribute space a, effects of terrain effectively limit speed or cap a range attribute. These would be represented by deforming or pruning the attribute space and changing the accessible outcomes.

All of these problems require an initial attribute space. In this paper we investigate the process of generating the attribute space a_{1-1} for a common RTS game combat setting. Direct simulation to compute individual conflict outcomes is not likely to be efficient when performing any exhaustive search of these spaces. We develop an abstract model of the attribute space a_{1-1}intended to provide an efficient and accurate model that can be used to address problems relevant to balance. We further use the resulting attribute space to achieve insight into the possible outcomes for different conflict settings.

B. Outline

Previous work [3] developed a model to predict conflict outcomes for a common RTS combat scenario involving combat between two teams each with a single unit. A simple game simulation is used test the unit conflict outcomes. A model is used to predict the outcome of the simulation; for the purpose of making balance adjustments before actual game runtime.This scenario utilizes the attribute space $A = \{range, speed, damage, health\}$ and results in an 8-dimensional attribute space which we denote as a_{1-1}. We refine the model for this scenario in section III by approximating the space-time trajectory of the unit in the combat simulation. Section IV compares the performance of this new combat modeling approach to that of the previous model, showing the advantage of this modeling strategy. The balance measure used for each team is the final value of the health attribute. We demonstrate the use of the resulting attribute space a_{1-1} to relate the conflict outcomes to attribute space projections and thus to individual attributes. Even this simple model provides insights into the attribute value relationships and the balanced outcomes for the combat.

II. Literature Review

The balance problem has been addressed in many areas, including those related to RTS games. The understanding of the balance problem differs according to the area of application. We discuss alternative definitions of balance that have been used in different contexts involving competitive agents.

A. Definitions of Balance

There are a range of interpretations of the concept of balance, although all involve a consideration of fairness. Specific criteria for identifying balance include: mix of skill and randomness, complexity of player strategies available, capabilities of individual units and actions, predictability, variety of winning states and fairness of starting conditions [11].

While not tied directly to balance, dynamic difficulty adjustment (DDA) is re-balancing games during game play. For single player games this involves adjusting challenges for individual players or teams, against an environmental [9] or AI challenge [2]. In contrast, handicapping aims to disadvantage a skilled opponent by compensating for a player's skill. Balance instead considers fair starting conditions.

B. Modeling Competitive Systems

Balance issues are relevant to fields such as game development but also combat modelling, economics and biodiversity.

Strategy recognition in RTS games is a search problem to identify more efficient ways of winning for the sake of stronger AI in competition. Strategies mirror the technology tree which describes the types of structures built and what units can be spawned [1]. The technology tree provides finite states that represent the sequence of player choices. The type of units being spawned and structures built model opponent behaviour [19]. Strategy development employs machine learning [1], case recognition mechanisms [14] and simulation

environments such as the Warcraft II clone Wargus [1] and the open source RTS engine Spring [19].

Multi-agent systems use multiple agents to solve computational problems rather than a singular AI. An example uses ant-agents which distribute files evenly over a network [17]. Multi-agent systems use both cooperative [18] and competitive agents [5] for applications such as combat modelling.

Combat modelling focuses on combat analysis for the military. The purpose varies from accurately representing agent behaviour to searching for an efficient victory. These lead to rules, such as a 3-1 advantage for attaining victory [6]. Combat models include predator-prey scenarios which represent stages of stalk, attack, and subdue in each agent [7].

Balance of power theory describes politically systems where neighbouring states or countries react to one another when an imbalance of power occurs. States adjust their power preemptively to avoid neighbouring states becoming dominant. Increasing power is not always militaristic and can also be in the form of economic growth. Otherwise another option is to form an alliance with the powerful neighbouring state [23].

This represents a competitive system where balance considerations are present, but in this case individual agents within the system maintain their own level of balance. In this regard balance of power theory describes balanced behaviour.

Economics also shares a connection to game balance involving systems of agents in competition. The desired balance provides system stability in order to remain beneficial for participants. Distributive justice embodies fair wealth distribution within economic systems which affects system health [12].

Game theory describes competition between rational beings that distrust one another. It represents a mathematical model for predicting player choices based on the comparative loss and payoff for any given option in a competition and is common for describing dominant strategies in balance definitions [16].

Biodiversity relates to balance particularly when a species is threatened. An extinction debt is incurred by a forcing event, such as a new species migrating and lowering the population. In this regard the extinction of the species can be analyzed by comparing the extinction debt to the immigration credit [10].

C. Mechanisms for Achieving Balance

DDA adjusts the difficulty for players. For example environmental resources can be used to adjust the difficulty by spawning extra ammunition or health to assist the player [9]. Another approach is to focus on AI strategies and learn methods of countering human opponents in an attempt to win. This includes altering an AI opponent's moves in a fighting game to better counter a human opponent [2]. If a player were to repeat the same option then the game would learn a counter to stop this moveand negate its effectiveness. Similar methods are also used in a capture the flag style game [15].

1) Dynamic Balance: Evolutionary methods are common for dynamic balance as they adapt to complex competitive scenarios. This has use in probable path prediction of game simulations [4] and evolving better agents to defeat one another [4], [13]. Evolving agents is a common method of determining balance fitness, as the evolved agents search competitive options for the best. If it is possible for an agent to evolve into a dominant instance, then the game is unbalanced [13]. It is common to test balance in environments of two agents in competition [11], [4], [13].

Another approach uses genetic algorithms to reshuffle premade rules within board games [8]. Evaluation works by recording the win to loss ratio of a sample. If the ratio is even then the game is considered balanced.

Alternatively, another method is to check the impact of game options by restricting play [11]. This is used to analyse different balance questions, including game predictability, reactions to game states, power of combinable options, avoiding game states and fairness. When an option is removed a designer can assess the impact. If the removal makes no significant difference, then this indicates that the option doesn't have a strong influence in the game. The game simulation is based on a simple RPG scenario, where agents face one another in statistical turn based combat.

III. Method

The temporal model of single unit combat [3] identifies 3 phases of combat behaviour and identifies a number of possible configurations of unit attributes during each phase. By estimating the time required in each phase it is able to estimate the outcome of the game. It produces predictions that provide a reasonable match with the results of an actual simulation of the game. The model itself uses analytical expressions involving the unit attribute thus providing insight into the role of each attribute during each phase. The drawback is that the number of configurations per phase is relatively large, complicating analysis due to the many possible paths through the set of configurations. It also makes simplifying assumptions that are difficult to overcome using this modeling approach.

We introduce an alternative modeling approach that retains the 3-phase model but considers the space-time trajectory followed by the individual units. This removes the need for certain simplifications (we can consider the moving reference frames of each unit) and potentially scales to more complex settings. The details of this model are presented below, after a brief description of the nature of the combat scenario used.

A. Combat Scenario

Two enemy units engage one another in adversarial combat within a simple RTS-inspired setting. Units are comprised of the four attributes representative of those used in RTS games: Range r: minimum distance to enemy before unit shoots, Speed s: the maximum speed for the unit, Health h: the number of damage points the unit can sustain before dying and ending the game and Damage d: the number of damage points inflicted to an enemy on each successful bullet impact.

Bullet fire rate and bullet speed are included as constant attributes set to constant values of 1 bullet/s and 20 squares/s unless otherwise specified. The goal is the development of an attribute space $a_{1-1} : \{a_{1,j,k} : j \in \{1,2\}, k \in \{1,2,3,4\}\} \rightarrow \mathbb{R}^2$ that, given the two attribute tuples for the pair of units, can determine the success for each of the units. In this case we use the final health value of each unit (h_1, h_2) as the measure

when	if $s_1 \neq s_2$	if $s_1 = s_2$
$m \leq r_1$ and$m > r_2$	P2A, P2C	-
$m > r_1$ and $m \leq r_2$	P2B	P2D-
$m \leq r_1$ and $m \leq r_2$	P2B, P2C	P2D
$n \leq r_2$	P3B	P3C
$n > r_2$	P3A	P3D

Table I: Path options for ϕ_2 and ϕ_3.

of success. Since the health of one unit will always be 0, we compress this result to a single value $h_1 - h_2$ which is positive if team 1 wins, negative if team 2 wins and 0 if balanced.

Units will attempt to stay at a distance between $r-1$ and r and will attack or retreat accordingly. If the distance between units is less than this ideal range, the unit will retreat backwards, but is able to fire if it is within range. If the distance between units is above this range, then the unit will advance toward its opponent. If the unit is within the ideal range, they will match the speed of the opponent (if speed attribute is equal or higher) and continue to fire.

Each unit has a health value initialized from the health attribute. The damage attribute of each bullet is a constant value removed from the opponent's health for each bullet that strikes. Each unit will follow their attack and retreat strategy until at least one unit has exhausted its health.

B. 3-Phase Model

The original temporal model of game combat identified that behaviour between units is divided into 3 phases:

- ϕ_1: Both units are out of range because the initial separation is higher than the individual range attributes of either unit. During this phase both units must move closer to one another. No bullets can be fired during this phase. In games where one or both units start within range this phase has zero duration.
- ϕ_2: At least one unit is within range and can fire. This phase ends when the steady state of ϕ_3 is achieved. Phase separation between units is denoted by m.
- ϕ_3: A steady state scenario is achieved when the separation between units is the range of the faster one. During this phase the faster unit can exclusively determine the separation and the control method used maintains separation at its range threshold. Phase separation between units is denoted by n.

Unit trajectories are bounded in complexity thanks to the three phase model. In each phase, a unit will move in a particular direction at a fixed speed. Behaviour is limited to a small number of cases as listed in Table I. Modeling the movement through space allows events in the game to be traced and a range of properties predicted. In particular, constraints such as collision precision due to the implementation platform can be easily incorporated into this representation. This model does not directly provide analytical solutions but still reveals explicit relationships between attribute values and game outcomes. The computational complexity for predicting the winner is constant allowing efficient search for points where balance exists.

C. Spatial Model

The spatial model tracks the trajectories of the two units. The 3-phase representation identifies periods during which each unit moves with constant speed (for specific values see [3]). Thus the space-time trajectory of each unit, $d_i(t)$, can be represented by a number of line segments, tagged with the unit's firing ability while on that segment. The intersection points between the bullet path launched from one unit's trajectory to the other unit's can be efficiently determined using line segment intersection tests. The number of segments in any trajectory is always $\leq$4, allowing such tests to be conducted in a constant time, independent of unit attribute values.

The game winner is calculated using the firing time at which unit i would have depleted its opponent's health: $t_{total,i}$. This is translated into the spatial coordinate, $x_{total,i} = d(t_{total,i})$. This final bullet is launched from $(x_{total.i}, t_{total,i})$ and direction determined by the bullet's speed. Let $t_{e,i}$ be the time coordinate of the bullet's trajectory intersection with the other unit trajectory. The unit i with the lowest $t_{e,i}$ is the winner.

This model is easily adapted to other game mechanics. Bullets can inherit the speed of the parent unit by including this in the determination of the bullet's trajectory. A collision radius around the target can be simulated by offsetting the second trajectory by this radius.

Final health values for the two units and the number of bullets in transit at the game end are calculated using intersection tests, by tracing each of the bullets exchanged. The calculation complexity depends on the number of bullets, but is bounded for bounded values of health and damage.

D. Model Evaluation

We validate the spatial model by comparing its predictions with the simulated game outcomes. Several data sets are chosen to comprehensively test different combinations of attributes but also to represent typical scenarios when the units are employed in an RTS setting. Environmental parameters are fixed for each data set. Unless otherwise specified, units start 30 squares apart and are able to fire at a rate of 1 bullet/s. Bullet speed defaults to 20 squares/s. We also include results for the temporal model [3] for comparison purposes.

- Set A: Attribute values for both units are sampled randomly from a region of attribute space. A total of 54810 cases are present. The bullet lifetime is set to 0.5 s beyond the time required to traverse the unit range.
- Set B: Unit 2 is given fixed attributes of $(d = 5, r = 25, s = 5, h = 30)$. The values of the attributes for unit 1 are sampled at regular intervals around this point: $d, s \in \{1, 3, 5, 7, 9, 11\}$, $r \in \{1, 11, 21, 31, 41\}$ and $h \in \{1, 6, 11, 16, 21, 26, 31, 36, 41, 46, 51, 56\}$. This set has 2160 cases. Bullet lifetime is infinite.
- Set D: Attribute values are generated randomly with health and damage constrained so that at least 2 shots are required to kill an opponent. Thus chance opening shots do not dominate results. The bullet speed is

	Set A	Set B	Set D
Accuracy	88.1%	98.8%	94.0%
Time RMSE	3.1 s	0.7s	0.4s
Bullet Count RMSE	-	-	-
Final Health RMSE	-	-	-

Table II: Results for temporal model [3]

	Set A	Set B	Set D
Accuracy	97.1%	99.4%	98.3%
Time RMSE	2.0 s	0.2s	0.2s
Bullet Count RMSE	1.7 bullets	0.3 bullets	0.2 bullets
Final Health RMSE	5.1 health	0.9 health	7.0 health

Table III: Results for spatial model

increased to 60 squares/s. This set has 10000 cases. Bullet lifetime is infinite.

Outcomes measured are: the winner of the game, the total game time, the final health values and the number of bullets fired. We compute an accuracy score as the percentage of games where the predicted winner matches the winner value from the simulation. For time of game, health and bullet count we calculate a root-mean-square error (RMSE) score. For parameter p, $RMSE_p = \sqrt{\frac{\sum_{i=1}^{n}(p_i - p_{pred,i})^2}{n}}$ where p_i is the value reported from the ith simulation run and $p_{pred,i}$ is the value predicted by the model.

IV. Results

The accuracy of the model predictions when compared to simulation output is shown in Tables II and III. The temporal model only predicts winner and time for game. It also mispredicts due to assuming a constant bullet transit time and ignores the relative speed of units and bullets changing.

The benefits of the spatial approach are reflected in the higher accuracy and in the increased number of game parameters that are readily computed. The disadvantage is loss of a closed form solution. Differences between prediction and simulation are more often due to practical issues with the implementation. Mismatches result from:

- Unit position rounding, evidenced by variation when rerunning the same simulation at different orientations.
- Proximity thresholds used in collision testing. This is particularly noticeable when units are close together.
- Bullets having a finite lifespan and expiring before they strike. Sets B and D use an infinite bullet lifetime to avoid this, after this issue was uncovered.

RMSE error values are typically higher for set A than for the others. Mispredictions usually involve extreme attribute values where the game ends quickly if one unit wins but be drawn out if the alternative outcome is assumed. These outliers significantly affect the average error. For example, the RMSE in time of game and health for the spatial model halves if these 3% of cases are removed, while the bullet count RMSE reduces to 20% of the original value. A count error of less than 1 represents a desired goal, as bullets are a discrete quantity.

A. Model Analysis

The model selects specific behaviour in each of the three phases based on the relative values of the attributes. The game outcome depends on the alternative selected in each phase, with the choice of alternative at each phase representing the path way chosen for each conflict. The number of potential game paths is bounded by the product of the number of alternatives at each level. However some alternatives are mutually exclusive, reducing the path amount in practice. The spatial model path ways on data set A are shown in Figure 1.

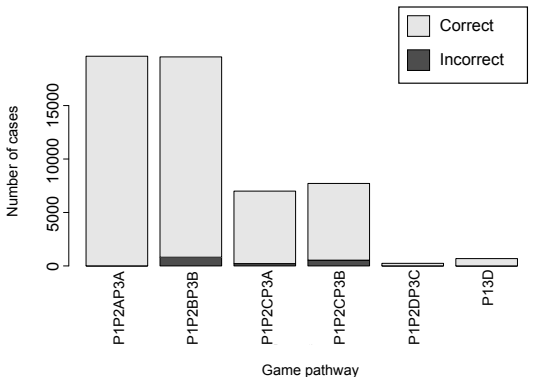


Figure 1: Distribution of path ways used by the spatial model.

	P1P2AP3A	P1P2BP3B	P1P2CP3A	P1P2CP3B	P1P2DP3C	P1P3C	P1P3D
Unit 2 wins	9	17327	232	6483	11	1	3
Draw	1	137	12	77	3	4	0
Unit 1 wins	19606	2089	6754	1156	209	14	682

Table IV: Victor outcomes for spatial model path ways.

Compared to the temporal model fewer paths are considered because it is able to more precisely determine the game outcomes in each case. The outcomes are still strongly biased towards one of the units in each case, with Table IV showing the possible outcome frequency produced by simulating the game using attributes from data set A. Path P1P2AP3A in particular is when unit 1 dominates in range and speed attributes and almost inevitably achieves victory. This region of attribute space can be effectively eliminated in the search for balanced games. Other cases show that outcomes in favour of either unit are possible, indicating that balance between sets of attributes should be possible.

B. Time of Game Predictions

The distributions of differences between simulated game duration and the predicted value are shown in Figure 2 for the two models. While the histogram for the temporal model appears to indicate systematic overestimation, the time predictions have a mean of close to zero (0.01 s). This is due to a significant tail and several widely spaced outliers (several on the order of 60 s). The spatial model tends to underestimate the game duration on average although by far smaller amounts.

C. Bullet Speed Sensitivity

Our analysis permits evaluation of the sensitivity of outcomes with respect to specific parameter values. We illustrate

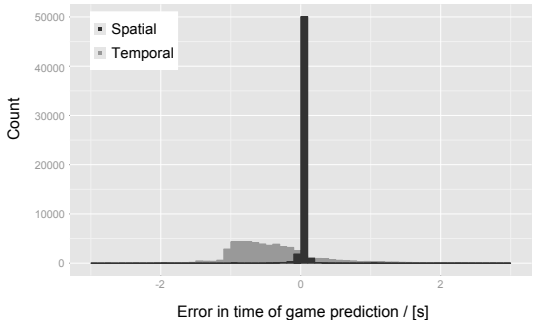


Figure 2: Time error histograms.

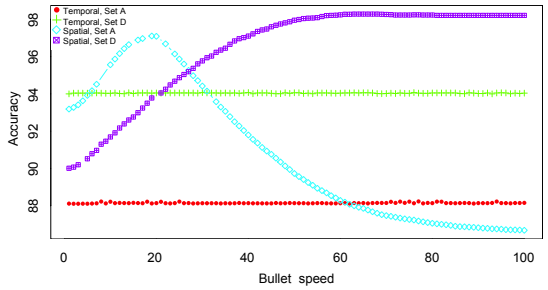


Figure 3: Effect of changing bullet speed parameter in the models.

this with respect to the accuracy of predicting the winner outcome. The predictive models are run with values of bullet speed that differs from the value actually used in the simulation. Accuracy values for doing this with data sets A and D are shown in Figure 3. Accuracy peaks where the bullet speed used in the model matches that used in the simulation, as happens for the spatial model with data set A. Where the bullet speed is greater than the maximum speed of the individual units (data set D), variation is only observed when the bullet speed is underestimated. The temporal model exhibits no dependence on the bullet speed parameter. In this model bullet speed is only used to provide a constant transit time factor, which is largely canceled out when determining the winner.

V. Attribute Space Analysis

The availability of an attribute space representation of game combat allows us to gain insight into how attribute relationships affect game outcomes. The spatial model is a more efficient method than direct simulation for predicting game outcomes at given points in attribute space. In this section we show examples of how our model is applied to yield insight into the behaviour of this common RTS sub-game.

The efficient predictive model allows rapid exploration of the attribute space. We present some views representing projections of the attribute space that reveal interesting properties of game balance in this two unit setting. Unit 2 is set to a mid range set of attributes ($d = 9, r = s = h = 26$) while unit 1 is given similar base attributes ($d = 10, r = 26, s = 31, h = 26$). The slight differences in attribute values avoid the less interesting cases where attributes are duplicated. We take 2D slices through the attribute space around this point by varying two of the attributes at a time for unit 1 ($d\epsilon\,[1, 17]$, $r, s, h\epsilon\,[1, 50]$). Bullet speed is set to 60 squares/s.

We visualize 2 quantities as shown in Figure 4. :

- a normalized measure of balance $b = \frac{h_{1,final} - h_{2,final}}{h_{1,init} + h_{2,init}}$ - the difference in health at the end of the game predicted by the spatial model and normalized relative to the total initial health (shown using a colour scale), and
- the path way through the 3 phase model (shown using the symbol shape).

Many examples of non-intuitive behaviour are evident. Figure 4a shows that range is a deciding factor (particularly with the slight speed advantage). An increase in damage can compensate when the advantage is not large. The anticipated advantage of speed is inverted (Figures 4b and 4f) where an increase in speed actually favours the other unit. This is due to the moving reference frame in phase 3; at low speeds the unit is moving away from the other increasing the distance bullets have to travel, in the other it is advancing and running into bullets. This difference invalidates the speed advantage for the attribute combination considered. The balance region achieved by trading health against damage is clearly visible in Figure 4c. The lack of any possibility of balance when both speed and range dominate is again clearly shown in Figure 4d although some measure of balance can be achieved in units that dominate in only one of these factors.

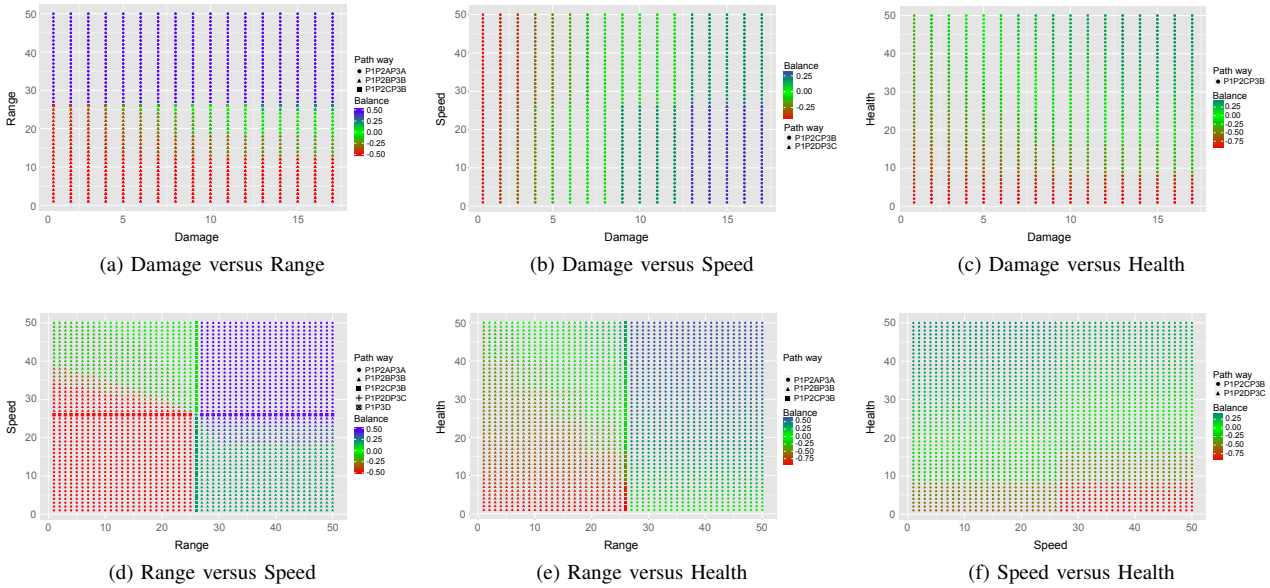
The current balance metric is derived from bounded discrete health values and consequently has limited resolution. As a result, the search space tends to be very flat as evidenced by the regions of constant colour in Figure 4. A gradient descent style search would require a random walk element to ensure that attribute space is thoroughly explored.

A. Automatic Unit Balancing using the Attribute Space

We demonstrate the value of predictive balance models by applying them to the problem of automatic game balancing. Given a particular unit, the goal is to find an equally effective counter unit, ideally with different attributes. The outcome is then dependent on player choice, rather than any inherent bias.

This search is phrased as an optimization process, subject to the minimization of the balance metric: $b\,(u_1, u_2) = \left|\frac{h_{1,final} - h_{2,final}}{h_{1,init} - h_{2,init}}\right|$ where u_1 and u_2 represent the attribute descriptors of the two units, and $h_{i,init}$ and $h_{i,final}$ are the initial and final values of the health attribute respectively provided by the spatial model. We apply a gradient descent process to search attribute space where the gradient is estimated by randomly sampling neighbouring positions in attribute space. We artificially avoid cases where $u_1 = u_2$.

We apply the gradient descent optimization on five scenarios. The base scenario uses standardized values of 20 for each attribute of u_1. In the others one attribute is doubled to 40, to give it a distinct advantage in a single ability. All attributes are constrained to the range $[0, 60]$ for the purpose of this experiment. The resulting position of counter-unit u_2 in attribute space for each of the scenarios is shown in Figure 5. The red marker shows the value of u_1 in each scenario. Greyed out empty squares represent low quality balance points where $b > 0.1$, while coloured circles have $b \leq 0.1$.



(a) Damage versus Range (b) Damage versus Speed (c) Damage versus Health

(d) Range versus Speed (e) Range versus Health (f) Speed versus Health

Figure 4: Views of balance in attribute space.

The distribution of values of u_2 is indicative of implicit constraints imposed by the game mechanics in choosing the nemesis to u_1. Clustering indicates tendencies towards common attribute values. For example, relative to u_1 the counter units for the base scenario (Figure 5a) have higher speed but lower range chosen from a constrained region. Health and damage values are also generally higher but less tightly constrained. In terms of the other scenarios:

- Increased damage is countered by increased health, with other attributes largely unaffected (Figure 5b).
- An increase in health is countered by increase in range and health, but *decrease* in damage and speed (Figure 5c).
- An increase in range and speed increases health but other attribute values are widely distributed (Figures 5d and 5e). Points with good balance are dispersed indicating where this diversity is through preference rather than failure of optimization. Vertical clustering with respect to the health axis is ascribed to the discrete nature of the conflict in that game outcomes change suddenly when health is close to $20k$; the damage inflicted by k bullets.

We expected clear combinations of attributes that would consistently counter one another. Symmetry is not a requirement, for example an increase in health shows a clear decrease in speed, but an increase in speed does not show a corresponding decrease in health..

VI. Conclusions

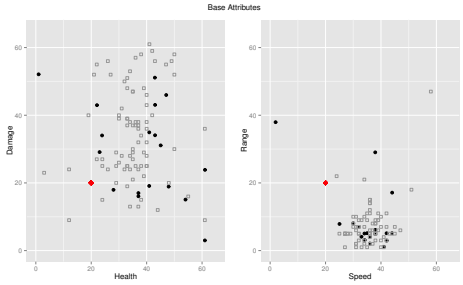
The spatial model of game combat tracks trajectories explicitly and accurately, manages moving reference frames and can be extended to include other game-play elements. It is better suited to numerical solution, although the complexity of predicting the victor is bounded. The values computed by the model are relevant to producing measures of balance.

The model implicitly encodes the relationships between attributes that drive combat simulation. Counter attributes in other units can be found by searching attribute space directed by the balance metric. The model is applied directly to predicting and measuring balance in combat between units with different attribute values. Discrepancies can reveal issues with the game implementation and uncover assumptions in the model, identifying potential exploits in the game; modes of operation that may unintentionally provide opportunities for cheating, or at the least, potential for loss of game balance. Predictive models that efficiently compute balance implications of attribute changes are ingredients of handicapping systems. They support dynamic game balancing and adaptation to game settings to remain interesting and challenging.

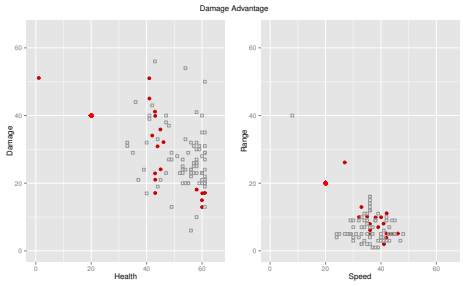
We introduced the concept of attribute space defined in terms of attributes, conflicts and outcomes and argued that this provides a framework supporting the analysis of balance in the context of RTS games. Game operations such as deploying special abilities and introducing terrain constraints can be seen as trajectories and constraints applied to an attribute space representation. We also show that this problem is common to many other areas involving adversarial agent interaction.

We demonstrate the generation of an attribute space representation for the core component of an RTS combat scenario, a_{1-1}: one-on-one conflict between two units of differing capabilities. Compared to the previous model, the spatial model is superior in its accuracy and produces a simpler model of the events that take place in each phase of combat.

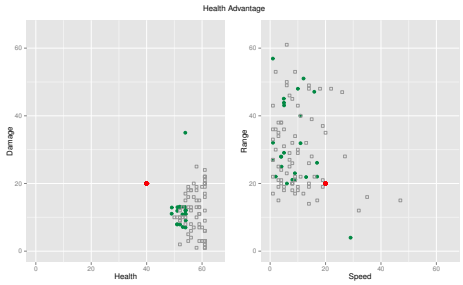
The efficient attribute space generation allows this model to identify unit behaviours that lead toward dominating attribute configurations. Algorithms that explore attribute space provide an effective method of balance analysis for our game simulation. These results support an attribute space representation for examining more complex balance settings (involving multiple units, effects of special abilities and terrain constraints) in RTS games, or achieving dynamic game balancing. They also have relevance to competitive scenarios in domains as diverse as



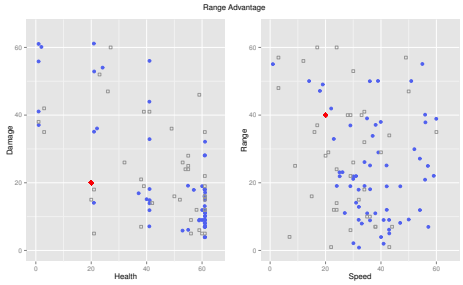
(a) Base



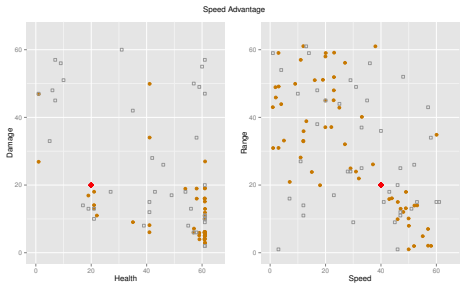
(b) Damage advantage



(c) Health advantage



(d) Range advantage



(e) Speed advantage

Figure 5: Attributes for counter-units for each scenario.

multi-agent systems, biodiversity and economics.

REFERENCES

[1] D. Aha, M. Molineaux, and M. Ponsen. Learning to win: Case-based plan selection in a real-time strategy game. In *6th International Conference on Case-Based Reasoning*, 2005.

[2] G. Andrade, G. Ramalho, H. Santana, and V. Corruble. Extending reinforcement learning to provide dynamic game balancing. In *Nineteenth International Joint Conference on Artificial Intelligence*, pages 7–12, Edinburgh, Scotland, 2005.

[3] S. Bangay and O. Makin. Modelling attribute dependencies in single unit game combat settings. In *5th International IEEE Consumer Electronics Society Games Innovation Conference*, 2013.

[4] H. Chen, Y. Mori, and I. Matsuba. Solving the balance problem of on-line role-playing games using evolutionary algorithms. *Journal of Software Engineering and Applications*, 5(8):574–582, 2012.

[5] I. Cil and M. Mala. A multi-agent architecture for modelling and simulation of small military unit combat in asymmetric warfare. *Expert Systems with Applications*, 37(2):1331–1343, 2010.

[6] PK Davis. Aggregation, disaggregation, and the 3:1 rule in ground combat, 1995.

[7] Lance Finney, Jade Vinson, and Derek Zaba. A three-phase model for predator-prey analysis. *The UMAP Journal*, 18(3):277–292, 1997.

[8] V. Hom and J. Marks. Automatic design of balanced board games. In *The Third AAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2007.

[9] R. Hunicke and V. Chapman. Ai for dynamic difficulty adjustment in games. In *Challenges in Game Artificial Intelligence AAAI Workshop*, pages 91–96, 2004.

[10] ST Jackson and DF Sax. Balancing biodiversity in a changing environment: extinction debt, immigration credit and species turnover. *Trends in Ecology and Evolution*, 25(3):153–160, 2010.

[11] A. Jaffe, A. Miller, E. Anderson, Y-E Liu, A. Karlin, and Z. Popovic. Evaluating competitive game balance with restricted play. In *The Eigth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, pages 26–31, 2012.

[12] T. A. Klein. Assessing distributive justice in marketing: A benefit-cost approach. *Journal of Macromarketing*, 28(1):33–43, 2008.

[13] R. Leigh, J. Schonfeld, and S. J. Louis. Using coevolution to understand and validate game balance in continuous games. In *Proceedings of the 10th annual conference on Genetic and evolutionary computation*, pages 1563–1570, 2008.

[14] S. Louis and C. Miles. Playing to learn: Case-injected genetic algorithms for learning to play computer games. *IEEE Transactions on Evolutionary Computation*, 9(6):669–681, 2005.

[15] J. Ludwig and A. Farley. A learning infrastructure for improving agent performance and game balance. In *Annual Artificial Intelligence and Interactive Digital Entertainment Conference*, Stanford University, California, 2007. AAAI.

[16] M. Mcguire and OC Jenkins. *Creating Games Mechanics, Content, and Technology*, chapter Balance, pages 185–214. 2009.

[17] A. Montresor, H. Meling, and O. Babaglu. Messor: Load-balancing through a swarm of autonomous agents. *Agents and Peer-toPeer Computing*, pages 125–137, 2003.

[18] J.H Obit, D. Landa-Silva, D. Ouelhadj, T.K Vun, and R. Alfred. Designing a multi-agent approach system for distributed course timetabling. In *Hybrid Intelligent Systems (HIS), 2011 11th International Conference on*, pages 103–108, 2011.

[19] F. Schadd, S. Bakkes, and P. Spronck. Opponent modeling in real-time strategy games. In *GAMEON'2007*, pages 61–69, 2007.

[20] Unknown. *Goliath – StarCraft and StarCraft II Wiki*. Available: starcraft.wikia.com/wiki/Goliath, Retrieved 2013-01-13, 2011.

[21] Unknown. *Knight (Warcraft III) – WoWWiki – Your guide to the World of Warcraft*. Available: www.wowwiki.com/Knight%28WarcraftIII%29?action=history, Retrieved 2013-01-13, 2011.

[22] Unknown. *Clubman – The Age of Empires Series Wiki – Age of Empires Wiki, Age of Mythology Wiki, Age of Empires III and more*. Available: ageofempires.wikia.com/wiki/Clubman, Retrieved 2013-01-13, 2012.

[23] WC Wohlforth, R. Little, SJ Kaufman, D. Kang, CA Jones, VTB Hui, A. Echstein, D. Deudney, and WL Brenner. Testing balance-of-power theory in world history. *European Journal of International Relations*, 13(2):155–185, 2007.