MATHEMATICAL GAME MODELING AND OPTIMIZATION

Optimizing Player Decisions in Old School Runescape

Palfore

5th October, 2020

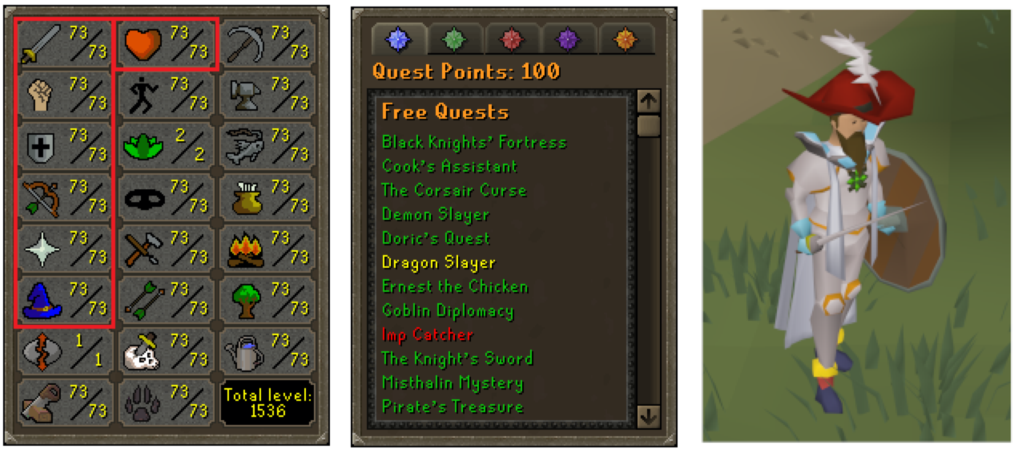
Contents

Introduction

Runescape (RS) is a popular *Massively Multiplayer Online Role-Playing Game* (MMORPG) that was first publicly released on January 4'th, 2001 by the video game developer Jagex Limited. Ranked as the 5'th most popular MMORPG in 2020 by several sources, this game's unique mechanisms and game play make it still successful nearly 20 years after it's incarnation [1, 2, 3]. On the 20th of November 2012, a total overhaul to the game's combat system - an integral part of gameplay - caused a great divide among it's players. As a result, the game bifurcated into two versions: Runescape 3, and *Old School Runescape* (OSRS). The latter was released on February 22, 2013 and reverted to the old mechanics. For several reasons, this version of the game became dominant and serves as the central topic in this text.

In typical role-playing fashion, the majority of game-play centers around fighting monsters and bosses, training skills, completing quests, playing mini-games, and collecting items. This game is played over the course of months or years. In a few years, there will even be some players who have played for *decades*. Over the years, as players gain a more comprehensive understanding of the game, their mentality has generally shifted from one of discovery to one of *efficiency*. Many tools have been created with the goal of improving player efficiency, optimizing game play, and maximizing success in difficult challenges. A non-comprehensive list can be found in the appendix [link].

There are 23 skills that a player can train [4]. A player is rewarded experience for certain actions related to a given skill. For example, cutting an `Oak Tree` yields 37.5 experience per log chopped. 83 Experience is required to go from level 1 to level 2, while reaching the maximum level of 99 requires 13,034,431 total experience [5]. The experience required to level up increases exponentially - hence the drive for efficiency [5]. There are several combat skills that directly influence a player's fighting ability. Quests are often completed for the special items, new training methods, and experience rewards they provide. They have skill requirements and often make use of combat in defeating difficult bosses. And so even in this basic overview, the complexity of the interactions and relations between different actions a player can perform becomes apparent.



Quest Points: 100

Free Quests

Black Knights' Fortress
Cook's Assistant
The Corsair Curse
Demon Slayer
Doric's Quest
Dragon Slayer
Ernest the Chicken
Goblin Diplomacy
Imp Catcher
The Knight's Sword
Misthalin Mystery
Pirate's Treasure

Figure 1: Some relevant interfaces/images that play a central role in game play. The skill panel (left) shows the player's levels in the 23 skills along with their total level (Image slightly modified from Ref. [4]). The combat skills, attack, strength, defence, ranged, prayer, magic, and health (in the middle column), are respectively outlined in red. The quest panel (middle) shows the player's quests that are completed, in progress, and not started (green, yellow, and red, respectively. Image from Ref. [6]). A character that a player would control in a 3D world is shown on the right.

To understand player decisions and optimize them, we will be mathematically modeling the in-game mechanics. A surprising variety of mathematical concepts and techniques will be encountered. Additionally, algorithms derived from computer science are required to solve some of these problems. This serves as an exciting *field* to explore, with a plethora of interesting results and visuals. Some of the details require high level mathematical solutions/descriptions. With the interest of being digestible by a broad audience with varying proficiencies, many arduous solutions are moved to the appendices.

This text accompanies an open-source codebase titled "OSRSmath" that can be found on Github. In addition, there is a video series titled "Optimizing Runescape" that covers some of these solutions. Finally, a Discord Chat Room exists for related discussions.

Part I

Combat

List of In-Game Terms and Associated Values

1. Combat Class: One of [`melee, ranged, magic`]
2. Attack Type: One of [`stab, slash, crush, ranged, magic`]
3. Combat Style: The name of the attack.
    (a) For the `melee` combat class: One of [`punch, kick, chop, hack, smash, block, pound, pummel, slash, lunge, jab, swipe, fend, spike, impale, reap, flick, lash, deflect, bash, focus, scorch`]
    (b) For the `ranged` combat class: One of [`accurate, rapid, longrange, short fuse, medium fuse, long fuse, flare`]
    (c) For the `magic` combat class: One of [`spell, spell (defensive), blaze, accurate, longrange`]. The latter two only apply to `Powered staves`.
4. Attack Style:
    (a) For the `melee` combat class: One of [`accurate, aggressive, defensive, controlled`]
    (b) For the `ranged` combat class: One of [`accurate, rapid, longrange`]
    (c) For the `melee` combat class: One of [`standard, defensive`]
5. Attack Speed: The number of ticks between attacks. Integer between 1 and 15.
6. Attack Interval: The number of second between attacks. Real number between 0.6s and 9s.
7. Attribute: When referring to an opponent/monster, their `attribute` is one of [`Demon, Draconic, Fiery, Kalphite, Leafy, Penance, Shade, Undead, Vampyre, Xerician`]. In addition, we expand this to also include properties like: [`On slayer task, In wilderness`].

Chapter 1

Overview

In this chapter, we will discuss the various factors involved in combat. We will consider combat in two stages. The first considers an autonomous fight in which the player performs no actions once the initial conditions of the fight have been specified. Analyzing this system will allow us to calculate quantities like the expected number of attacks required to defeat an opponent, and the probability of winning a fight. The second considers active player decisions that occur during combat. This will allow us to investigate the effect of performing actions on the aforementioned quantities. *Policies* may be defined to mathematically model a player's decision. As an example, a player may use a healing item any time throughout a fight. To handle this, we can consider a specific policy whereby the player will use a healing item when health is below some threshold. Investigating this threshold will provide some insight.

It is interesting to note that although the descriptions of in-game mechanics likely have no real-world connections (since they are somewhat arbitrarily decided by the game's developers), the mathematics that can be applied to the dynamic variables resulting from these mechanics can actually be applied, and generalized to real-world settings. We will begin with a discussion of the most relevant mechanics, however there is an additional large body of information that can be found on the Official Wiki that provides a greater overview.

In the broadest scope, we define an entity that participates in combat as a *fighter*, $\mathcal{F}$. In general, it is easier to formulate things in terms of an *attacker*, $\mathcal{A}$, and *defender* $\mathcal{D}$, where typically the player is considered the attacker. Thus, $\mathcal{F} \in \{\mathcal{A}, \mathcal{D}\}$. Due to the large number of dependencies, we will need many sub- and super- scripts. For notational convenience, we will occasionally use $(\mathcal{A}|\cdot)$ and $(\mathcal{D}|\cdot)$ to denote that a quantity (represented by $\cdot$) relates to either the attacker or defender, respectively. We will expand on, and define these objects in the following sections.

1.1 Autonomous Mechanics

1.1.1 Combat Skills, Combat Triangle and Attack Styles

Combat is built around the so-called *Combat Triangle* which describes the relation between the three classes of combat in the game [7]. A Melee fighter makes use of close quarters combat, typically wielding swords, daggers, halberds, etc. A Ranged fighter makes use of bow and arrow, crossbows, and thrown objects to deal damage at a distance. Finally, a Mage will make use of staves and magical spells to do damage, also at a distance. The combat triangle refers to the notation that melee users are (generally) weak to magic, which is weak to ranged, which is weak to melee, and is depicted in Fig. 1.1.

Some skills provide benefits to all fighters, while others are specific to the style:

1. Attack, L_a: Increases the accuracy of a melee attacker.
2. Strength L_s: Increases the maximum damage a melee attacker can do in a single attack.
3. Ranged L_r: Increases the accuracy and maximum damage of a ranged attacker.
4. Magic L_m: Most spells have a constant damage (with more powerful spells being unlocked at higher levels), also some scale with magic level. Accuracy however is generally increased with higher magic. In addition, defence against magical attacks is partially determined by the player's magic level.
5. Defence L_d: Decreases the probability that the opponent will have a sucessful attack.
6. Prayer L_p: Acts as a depleteable resource that can boost combat skills.
7. Hitpoints L_h: Increases the amount of damage a player can receive before they lose a fight.

The set of all combat levels is denoted $\{L\}$.

Every weapon has a set of *attack styles* that allow a player to change which combat skill they train. In addition, the attack style may provide a small bonus to combat. Prayer is the only skill that cannot be trained directly through combat. Hitpoints is another exception in that a proportion of the experience awarded to the skill associated with the player's attack style is given to hitpoints.

1.1.2 Equipment Bonuses

Let's begin discussing a fighter's equipment by defining an *item*, $\mathcal{I}$. Equipable items can be worn in one of 11 slots. We let $\mathcal{I}^{\text{slot}}$ represent the item in a given *slot*, where

$$\text{slot} \in \{\text{head, cape, neck, ammo, weapon, torso, shield, legs, hands, feet, ring}\}. \tag{1.1}$$

Each item has some associated equipment bonuses. Most of these are constant, however some bonuses are conditional. The constant bonuses can be represented as a vector:

$$\begin{aligned}\vec{\mathcal{I}}_c^{\text{slot}} = (&A_{\text{stab}}, A_{\text{slash}}, A_{\text{crush}}, A_{\text{magic}}, A_{\text{ranged}}, && (1.2)\\ &D_{\text{stab}}, D_{\text{slash}}, D_{\text{crush}}, D_{\text{magic}}, D_{\text{ranged}}, && (1.3)\\ &S_w, S_r, S_m, P, w, r). && (1.4)\end{aligned}$$

There are many terms to define, so we will explain them here. A, D, and S refers to the attack, defensive, and strength bonuses, respectively. The attack and defence bonuses are associated with the different attack types, while the strength bonuses are associated with the combat class [SEE LIST OF TERMS]. The first three attack and defence bonuses listed are associated with melee combat, the last two are associated with magic, and ranged, respectively. There is a strength bonus associated with each combat class. In the order above we have: melee/warrior, ranged, then magic. The prayer bonuse, P affects how long bonuses from the prayer skill can last without recharging. w is the weight of the item. Finally, if the item is a weapon, r is the attack rate given by $r = 1/s$, where s is the weapon attack speed. If it is not a weapon, $r = 0$. Note that we use the rate since every other bonuses improves fighter ability. This allows us to use a basic comparison operator (at the cost of using real numbers instead of integers).

The total equipment bonuses that a fighter has, $\vec{\mathcal{F}}|E$ is given as the sum over all the slots,

$$\mathcal{F}|\vec{E} = \sum_{\text{slot}\in\{\text{slots}\}} \hat{\mathcal{I}}_c^{\text{slot}}(\mathcal{F}|\mathcal{E}). \tag{1.5}$$

The in-game interface indicating these values is shown in Fig. 1.1. There are a number of conditional effects that may not appear in this interface.

The conditional bonuses can be further divided in to special/attribute, bonuses and equipment set bonuses. Monsters may have a particular weakness due to their so-called attribute. For example, a `Iron dragon` would be `dragonic`, and `dragonbane` weapons would provide an accuracy and damage multiplier. In this sense, we can consider these bonuses to be dependent on information that the item itself does not know, and so we represent these special bonuses as an operator $\hat{\mathcal{I}}$. When acting on a fighter's environment $\mathcal{F}|\mathcal{E}$, these bonuses become concrete:

$$\vec{\mathcal{I}}_s = \hat{\mathcal{I}}_s(\mathcal{F}|\mathcal{E}) \tag{1.6}$$

The total bonuses from all the player's items can be represented as:

$$\vec{\mathcal{I}} = \hat{\mathcal{I}}_s(\mathcal{F}|\mathcal{E}) \cup \sum_{\text{slot } \in \ \{\text{slots}\}} \vec{\mathcal{I}}_c^{\text{slot}}, \tag{1.7}$$

Set effects are also similar except that they are conditional on equipment the player is wearing. For this reason, (and the fact that there are other special cases), we group all these effects into the special bonus operator, $\hat{\mathcal{I}}_s$ from above.

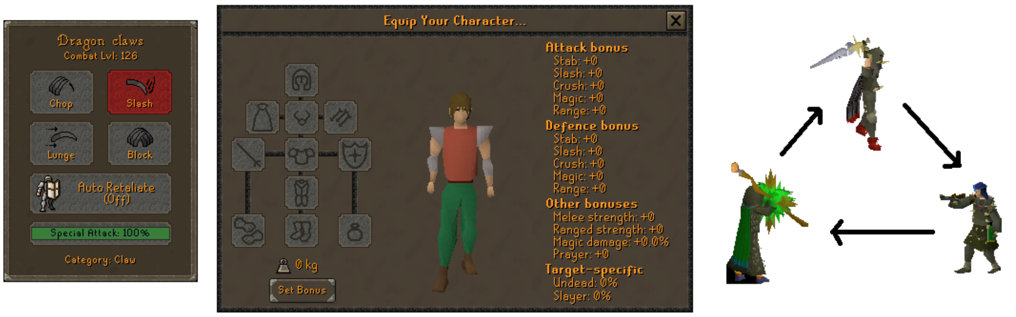
The definition of environment is intentionally vague, as there are a myriad of conditions, essentially limited only by developer imagination and infrastructure. Some of these conditions/dependencies include: attacker & opponent equipment & levels, attack style (which implies combat class), whether a particular `Diary` is completed, the `attribute` of the opponent, and so on. The elements and details of $\vec{\mathcal{I}}$ are also purposefully vague, as there is an additional caveat that makes these a bit trickier to handle both mathematically but more-so programatically. Unlike the constant bonuses, which can be added together, special bonuses are generally multiplicative but also make use of intermediate *flooring*. This makes the special bonus operator non-commutative, since the order does effect the rounding.[1] This means that a vector representing special bonuses would essentially have as many elements as the number of special items! So it is often easier to work on each bonus type with different methods. For this reason, special effects and constant bonuses are treated independent, making the union above is more symbolic rather than practical.

1.1.3 Ticks and Attack Speed

At a fundamental level the entire game operates on a tick-based system. Every 0.6 seconds (called a tick) the game updates. This discretizes the possible game states, and typically means we will be dealing with sums in place of integrals, and recursive equations in place of differential equations.

Once an attacker begins combat with an opponent, the fight continues until either is defeated, or one runs away. The attacks occur at an interval associated with the weapon.

[1] The *specific* ordering of the flooring operations is taken from ref. [bitter-dps calc]. Although, this author is unsure if that ordering is arbitrary, but we assume not. [Reference max hit section?]



Attack bonus
Stab: +0
Slash: +0
Crush: +0
Magic: +0
Range: +0
Defence bonus
Stab: +0
Slash: +0
Crush: +0
Magic: +0
Range: +0
Other bonuses
Melee strength: +0
Ranged strength: +0
Magic damage: +0.0%
Prayer: +0
Target-specific
Undead: 0%
Slayer: 0%

Figure 1.1: The attack styles (left), equipment slots and associated equipment bonuses (middle) along with a depiction of the combat triangle (right). The attack styles for the `Dragon Claws` are Chop, Slash, Lunge, Block and give experience specifically to Attack, Strength, shared, and Defence, respectively. Shared means experience is split equally. In the equipment panel, the player is not wearing any equipment which results in 0 bonuses for all attributes. Starting with the bottom left of the combat triangle, a mage has advantage over the melee equipment typically worn by a melee fighter, a melee warrior has an advantage over the equipment typically worn by a ranged fighter, and ditto for ranged to mage.

Different weapons have different *Attack Speeds*, typically between 3-9 game ticks (1.8s - 5.4s). The attacker's attack speed $\mathcal{A}|s$, is the number of ticks between attacks. On each attack, the player's accuracy will determine the probability of a *successful hit.* On a sucessful hit, a number between 0 and the player's maximum hit will be uniformly sampled as the damage the player does.

A notable consequence of this tick-based system is that a series of precise player actions known as tick-manipulation allows players to perform multiple actions in a single tick, or to take advantage of mechanisms like tick-eating, allowing a player to survive otherwise fatal attacks.

1.1.4 Summary

A fighter has some combat skill levels and will (typically) equip some armour and a weapon. They will select an attack style, which selects the skill they will receive experience in, and which equipment bonuses plays a roll in the accuracy calculation. The problem then reduces to considering an accuracy and maximum hit. Once a fight begins, an attack occurs every couple of ticks. If the attack is successful, a uniform integer between 0 and their max hit is delivered to the opponent, reducing their current hit points. Once a fighter's health reaches 0, the fight is over.

1.2 Agency

1.2.1 Special Attacks

Certain weapons have the ability to use a special attack, typically dealing additional damage, but may also reduce the opponent's levels temporarily.

1.2.2 Temporary Boosts and Healing

Potions provide temporary boosts to skill levels.

1.2.3 Item Switching and Movement

Different items, moving around. Attack delays etc.

Chapter 2

Maximum Hits

These formula are largely given by Ref. [9]. There are very many exceptions, special, and edge cases and so to verify these equations we compare them to a dataset extracted from Ref. [10], a well-established resource.[1] We will not consider special attacks for now.

There are several bonuses that will be used through this chapter: B_{potion}, B_{prayer}, B_{other} are the bonuses due to potions, prayers, and other special considerations.

2.1 Melee

The maximum melee hit is given by:

$$\begin{aligned} m &= \left\lfloor c_0 + c_1 L_s^{\text{eff}} + c_2 S_w + c_3 L_s^{\text{eff}} S_w \right\rfloor && (2.1) \\ L_s^{\text{eff}} &\equiv \lfloor (L_s + B_{\text{potion}}) B_{\text{prayer}} B_{\text{other}} + \mathcal{S} \rfloor && (2.2) \\ \{c_i\} &= \left\{ 1.3, \frac{1}{10}, \frac{1}{80}, \frac{1}{640} \right\}. && (2.3) \end{aligned}$$

For melee,

$$\mathcal{S} = \begin{cases} 3 & \text{if style is } \texttt{aggressive} \\ 1 & \text{if style is } \texttt{controlled} \\ 0 & \text{Otherwise} \end{cases}. \qquad (2.4)$$

A list of B_{potion}, B_{prayer}, and (incomplete) B_{other} can be found in Ref. [11].

2.2 Ranged

The maximum ranged hit is given by:

$$\begin{aligned} m &= \left\lfloor c_0 + c_1 L_r^{\text{eff}} + c_2 S_r + c_3 L_r^{\text{eff}} S_r \right\rfloor && (2.5) \\ L_s^{\text{eff}} &\equiv \lfloor (L_r + B_{\text{potion}}) B_{\text{prayer}} B_{\text{other}} + \mathcal{S} \rfloor && (2.6) \\ \{c_i\} &= \left\{ 1.3, \frac{1}{10}, \frac{1}{80}, \frac{1}{640} \right\}. && (2.7) \end{aligned}$$

[1] This is a damage calculator made in google sheets and was created by Bitterkoekje. To extract a benchmarking dataset from this graphical resource, the python based GUI automation library, PyAutoGUI was used to acquire a random sample.

For ranged,

$$\mathcal{S} = \begin{cases} 3 & \text{if style is } \texttt{accurate} \\ 0 & \text{Otherwise} \end{cases}. \tag{2.8}$$

Note that if the attack style is set to rapid, the weapon attack speed is increased by 1 tick. A list of B_{potion}, B_{prayer}, and (incomplete) B_{other} can be found in Ref. [12].

2.3 Magic

Magic differs slightly, so we need a few additional definitions. First we define m_{spell} as the base max hit of the player's spell/staff. Some of these depend on the player's magic level. A list of these can be found in Ref. [13]. Then there are several special items, listed below as an associated bonus B_{other} and either an additive toggle $\bar{\delta}_{\text{item}}$ which is 1 or 0 based on the accompanying condition or a multiplicative toggle δ_{item} which is either $B^{\text{item}}_{\text{other}}$ or 1 based on the accompanying condition.

1. $B^{\text{chaos}}_{\text{other}} = 3, \bar{\delta}_{\text{chaos}}$ if a `bolt` spell is used along with `Chaos gauntlets`.
2. $B^{\text{tome}}_{\text{other}} = 1.5, \delta_{\text{tome}}$ if a `fire` spell is used along with a `Tome of fire`.
3. $B^{\text{castlewars}}_{\text{other}} = 1.2, \delta_{\text{castlewars}}$ if a `Castle wars bracelet` is worn while attacking a `flag bearer`.
4. $B^{\text{salve}}_{\text{other}} = \textit{varies}, \delta_{\text{salve}}$ if any variant of the `salve amulet` is worn while attacking an `undead`.
5. $B^{\text{slayer}}_{\text{other}} = 1.15, \delta_{\text{slayer}}$ if any variant of the `imbued black mask` is worn while attacking `slayer task monster`.

Then the maximum magic hit is given by:

$$m = \left\lfloor \left\lfloor \left\lfloor \left\lfloor (m_{\text{spell}} + B^{\text{chaos}}_{\text{other}} \bar{\delta}_{\text{chaos}}) * (1 + S_m) \right\rfloor \delta_{\text{salve}} \bar{\delta}_{\text{salve}} + (1 - \bar{\delta}_{\text{salve}}) \delta_{\text{slayer}} \right\rfloor \delta_{\text{tome}} \right\rfloor \delta_{\text{castlewars}} \right\rfloor \tag{2.9}$$

Chapter 3

Accuracy

Chapter 4

Models

4.1 Important Quantities

4.2 Crude

4.3 Averaged Piecewise

4.4 Piecewise

4.5 Markov Chain

Bibliography

[1] 10 most played mmorpgs of 2020. https://bestreamer.com/gaming/most-played-mmorpg-2019/2/. Retrieved October 02, 2020.

[2] Top 6 most popular mmorpgs sorted by population (2020). https://altarofgaming.com/all-mmos-sorted-by-population-2018/. Retrieved October 02, 2020.

[3] Ranking the 15 best mmorpgs of all time. https://www.thegamer.com/best-mmorpgs-ever-wow-runescape/. Retrieved October 02, 2020.

[4] Skills. https://oldschool.runescape.wiki/w/Skills.

[5] Experience. https://oldschool.runescape.wiki/w/Experience.

[6] Quests. https://oldschool.runescape.wiki/w/Quests.

[7] Combat triangle. https://oldschool.runescape.wiki/w/Combat_triangle.

[8] Monster attribute. https://oldschool.runescape.wiki/w/Monster_attribute.

[9] Maximum hit. https://oldschool.runescape.wiki/w/Maximum_hit.

[10] Dps calculator by bitterkoekje. https://docs.google.com/spreadsheets/d/1wzy1VxNWEAAc0FQyDAdpiFggAfn5U6RGPp2CisAHZW8/edit#gid=158500257.

[11] Maximum melee hit. https://oldschool.runescape.wiki/w/Maximum_melee_hit.

[12] Maximum ranged hit. https://oldschool.runescape.wiki/w/Maximum_ranged_hit.

[13] Maximum magic hit. https://oldschool.runescape.wiki/w/Maximum_magic_hit.