Rotation BiS Solver Algorithm Documentation

Leonhard Euler @ Gilgamesh Discord : iampythagoras

September 2023

1 Introduction

In this document you will find the documentation about the algorithm and code used in the rotational BiS solver that is currently available in the **ffxivcalc** library in Python.

In short, the algorithm is mainly a simple brute force approach to the issue. It however uses a **top-down** approach to the melding problem, which makes it faster (as will be shown later).

This document will also discuss the results of using this solver compared to the Balance's recommended gear sets.

Some required notions will be explained before diving into the functioning of the algorithm.

2 Oversaturating gear pieces

As is general knowledge, a gear piece can receive a certain number of **materias** as meld which affects the final stats of the gear piece. A usual gear piece can receive 2 materias and at the current time each materia can give a maximum of 36 of a given sub stat. Furthermore, any stat of a gear piece cannot exceed a certain fixed value once materias are added. This fixed limit is equal to the maximum value of a sub stat on a gear piece before any materia is added. This limits the possible number of materias that can be added on a gear piece. Note that if a gear piece is being oversaturated it will be denoted as being **illegal** in difference of being **legal** if it is not oversaturated. Another way to see it is that a legal gear piece can exist in game, an illegal gear piece cannot.

The action of **oversaturating** a gear piece is the action of illegaly adding more materias (under certain restrictions) that could usually be added in the game. Oversaturation of gear pieces is of course not possible in game, but within the constraints of **ffxivcalc** it is possible to do so.

For example, consider a weapon that has 2 meld slots. Melding two **critical hit** materias to it is completely **legal** and within the scope of the game :

Figure 1: legal melding on weapon piece

This melding is legal since there is only 2 materias on the gear piece, and the final value of all sub stat does not exceed the value of crit on the weapon.

However, any meld set that would result in **more than the materias limit** or in **any of the final stat exceeding the sub stat limit on the gear piece** would be considered **oversaturating** the gear piece.

Consider these examples :

Figure 2: Illegal melding due to exceeding materias limit

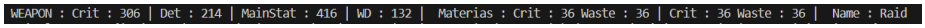

Figure 3: Illegal melding due to exceeding sub stat limit

Note that in **figure 3**, the crit sub stat is still being flagged as **wasted**. In fact, within the restrictions of the solver it has been decided to not add sub stat beyond the limit of the gear piece even if it is being oversaturated. The reasons will be explained later.

2.1 Oversaturation within the restrictions of the solver

Oversaturation happens multiple time during the executing of the solver, it however has some restrictions that were not detailed above. When the algorithm resorts to oversaturating a gear piece, it will add **materias (up to the materia limit of the gear piece) of every type (present in the materia space) or until the sub stat of the gear piece reaches the sub stat limit**. Consider the same weapon as before for an example :

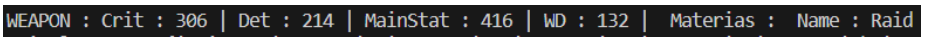

Figure 4: Before being oversaturated

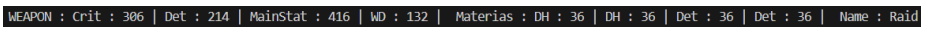

Figure 5: After being oversaturated

The gear piece after oversaturation has received 2 **direct hit** materias and 2 **determination** materias. This is because the value for both sub stat had not reached the sub stat limit, so the code added 2 of both materia type. Note that here the materia space was **critical hit, direct hit and determination**.

Furthermore, it is possible to reapply oversaturation on the gear set obtained in **figure 5**. This will give the following result :

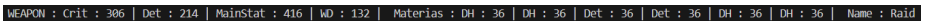

Figure 6: After oversaturating gear piece in figure 5 again

The gear piece this time did not receive any determination materias since if it had it would have exceeded the sub stat limit for this gear piece.

This process could be reapplied as many times as wanted, but it would at some point reach an equilibrium which will happen when the sub stat limit is reached for all types in the materia space.

This is all the required knowledge to continue further. The reasoning behind oversaturation is discussed at the end of this text if such is an interest to the reader. The rest will assume general game knowledge and general algorithm knowledge. Just remember that when the algorithm will use oversaturation, it is using the definition of **2.1** just above and not the general definition of oversaturation.

3 Algorithm steps

The first thing to discuss is the different inputs to give the algorithm for it to do its job. It requires the following :

- A **Fight** to optimize a gear set on.
- A **gear set space**. Which is defined as all gear piece to look through.
- A **Materia space** which is defined as all types of materia the algorithm will use to optimize a gear set. This materia space cannot include Speed types.
- A **Food space** which is defined as all food sorts to look through when optimizing a gear set
- A **minimum** and **maximum** speed value for the final gear set.
- A number of time to apply oversaturation **pre gear looking** and **post gear looking**.
- A boolean value indicating if the algorithm swaps direct hit and determination materias before or after swapping for Speed materias.

Note that when using the solver in the **ffxivcalc** library there will be more inputs than described above. However, the other required inputs are coding related rather than logic related.

Here is a list of operations happening during the execution of the solver.

1. The solver will look for all possible GCD timer of the player to optimize a gear set on. It will then render **prebakedsimulation** for every different GCD timer. There is no need to understand what a prebakedsimulation is, simply see it as a way to simulate the DPS faster and more efficiently.
2. The solver will **oversaturate** every gear piece in the gear space up to the value of **pre gear looking oversaturation**.
3. The solver will now sample gear sets from the gear space until the whole space has been searched. Note that the following steps are also repeated for every possible gear set.
4. The solver will try all food from the food space on the gear set. The following steps are repeated for all food.
5. The solver will begin the **materia optimizing** part of the algorithm. The gear set will be oversaturated once again up to the number of time equal to the value of **post gear looking oversaturation**.
6. At this point every gear piece of the gear set is **illegal**. This step consists of **removing the materia from the gear set that results in the smallest DPS reduction**. This is repeated until the whole gear set is considered **legal**.

7. This step could vary depending on the value of **swapDHDetBeforeSpeed**. If **swapDHDetBeforeSpeed** is **false**, the solver will now attempt to swap **determination** and **direct hit** materias to see if a DPS gain is possible. The reasoning of this step will be discussed later. If **swapDHDetBeforeSpeed** is **true**, this step happens after the next step.

8. This step consists of swapping the materias of the gear set by speed materias (by swapping the materia resulting in the smallest DPS loss by a speed materia) until the gear set cannot receive any more speed materias or until the gear set has a speed value higher than the **maximum** speed value.

These are all the main steps. Most of them will be repeated until the whole search space has been looked through. Throughout this execution there are also some places where the speed of the gear set is checked in order to eliminate gear sets that do not meet the speed requirements. The highest DPS gear set is returned at the end.

4 Discussion

4.1 Over saturation discussion

The idea behind over saturation was to put all gear piece on a more equal pedestal. Over saturating a gear piece results in a gear piece that is closer to its possible potential once melds have been added.

For example, a gear piece with no critical hit will most likely see itself receive critical hit meld. By over saturating the gear before solving for optimal melding we can already get a closer approximation to its maximum potential.

This is the whole reasoning behind over saturating. I have no actual proof that this indeed results in a better final gear set, but from experimentation it does appear to work well.

The idea of having post gear looking and pre gear looking values was used before looking through all gear sets and optimizing the melding of everything. The approach of solving the best materias for every gear sets will most likely only work for expected damage comparison (unless a faster way to compute percentile damage is found). So I kept these values in since they would be useful if we were looking for a certain percentile's BiS.

After much experimentation, a total sum of **over saturation** of 2 seems best when looking for lower Speed gear sets, and a total sum of 1 seems best when looking for high speed gear sets. This is however not a rule and testing different values regardless of the speed is always good practice.

4.2 Swapping DH and Det discussion

Over saturating and then removing materias until the gear set is legal means that the highest resulting materias will stay. Meaning critical hit will usually be maxed out and direct hit and determination will be added in certain amount. The added step of seeing if swapping one by the other results in higher DPS has proven to increase DPS in most cases.

Furthermore, doing this step before or after swapping materias with speed materias sometime affects the resulting gear set. It might increase or decrease the DPS. So again it is good practice to try both values of this parameter.

4.3 Top down approach discussion

As was said in the introduction, this algorithm is a simple brute force approach. However, instead of looking for optimal materias by building from the ground up, we start at the top (oversaturation) and go down until the gear set is legal.

It is not super hard to see the advantage. Most gear sets have up to 22 materias slots available. Assuming there is at least 3 possible materia type per slot this means a total of 3^{22} total materia arrangement. This isn't a space that can be searched with brute force methods. We could instead look for up to n melds and take the most optimal of these n melds and then repeat until the total meld has been achieved. This would reduce the search space to 3^n for

every iterations. This does make it possible to look for a good enough materia set, but is clearly not optimal and takes exponential time as n increases. This method is currently already implemented in the current version of ffxivcalc, but it is not recommend to use because of its clear issues.

Doing the top down approach greatly reduces the number of materias to process until the gear set is legal. Note that the algorithm currently removes one at a time, but it would be possible to remove multiple at a time and remove the set of n materias that together result in the smallest DPS loss.

4.4 Optimality discussion

I have no proof that the algorithm will result in the most optimal gear set. From experimentation it does appear to be quite optimal, but I cannot yet prove if it is or not. I have suspicion to believe that the algorithm will not be optimal given certain parameters input, but can be given certain parameters. This is the reason I recommend to trial multiple values for the input.