BOINC/Gridcoin calculations

I) BOINC computational credit

A BOINC project gives you **credit** for the computations your computers perform for it. BOINC's unit of credit is the **Cobblestone** (1/200 day of CPU time on a reference computer that does 1 GFLOPS based on the Whetstone benchmark).

$$\mathbf{1}\ \boldsymbol{Cobblestone}\ =\ \mathbf{432}\ \boldsymbol{GFLOP}$$

'Whetstone' is a benchmark that is reported as 'Measured floating point speed'. Floats can have fractional parts (like 1.48283 or 3.141592); Whetstone does 8 different groups of tests (repeatedly of course), times how long they took to finish, and produces a number, [ops performed]/[time]. These tests all use floating point math operations of the CPUs being tested. Some of them are simple math (addition, multiplication, division) while others compute trigonometric and exponential functions (sine, cosine, tangent, exponent). Neither of the tests really checks how well/fast a system can access memory, and most BOINC applications access memory a lot.

Eventually, credit **may be adjusted to reflect** network transfer and disk storage as well as computation. The actual computational difficulty needed to run a given work unit is the basis for the number of credits that should be granted. The BOINC system allows for work of any length to be processed and have identical amounts of credit issued to a user. In so doing, BOINC uses benchmarks to measure the speed of a system, combining that figure with the amount of time required for a work unit to be processed. The interface then can "guess" at the amount of credit a user should receive. Since systems have many variables, including the amount of RAM, the processor speed, and specific architectures of different motherboards and CPUs, there can be wide discrepancies in the number of credits that different computers (and projects) judge a user to have earned.

Most projects require a consensus to be reached by having multiple hosts return the same work unit. If they all agree, then the credit is calculated and all hosts receive the same amount regardless of what they asked for. Each project can use their own policy depending on what they see is best for their specific needs. In general, the top and bottom claimed credits are dropped and an average of the remaining is taken. However, certain other projects award a flat amount per work unit returned and validated.

Projects maintain two counts of granted credit:

- **TC**: Total credit = total number of Cobblestones performed and granted.
- **RAC**: Recent average credit = The average number of Cobblestones per day granted recently.

TC: Credits are tracked internally for computers, users, and teams. When a computer processes and returns a work unit, it receives no credit for that action alone. It must first have that work unit validated by the given, project-specific method. Once validated, the computer is granted credit, which can be less than, equal to, or greater than what was requested. This amount is immediately added to the computer, user, and team total. If a work unit is returned past the given deadline (in most cases) or is found to be inaccurate, it is marked as invalid and results in no credit. Users and teams commonly determine world rank by comparing the total number of credits accumulated. This highly favours users and teams that have been around for the longest time. This makes it extremely difficult for new users to rapidly gain ground in the rankings, even if they are running many computers. That said, given the exponential increase in computing power of the average PC, it is relatively easy to surpass inactive BOINC users who have earned all of their points on obsolete machines—even if they were at one time ranked highly. Thus, the highest ranked BOINC users will generally be active crunchers.

RAC: To find the useful amount of work provided by a computer, a special calculation called recent average credit (RAC) is used. This calculation is designed to estimate the number of credits a computer, user, and team will accumulate on an average day. Due to the many variables not taken into account, including the inconsistency of host processing, time it takes to validate work units, discrepancies in benchmarks, and possible project down time, the RAC calculation has proved to be only a guide. RAC was originally meant to help scientists understand the computational power available to them. Today, this calculation allows for increased competition among users and teams by allowing even new users to quickly move up in rank based on RAC, which should theoretically reflect how fast work is being processed.

This average decreases by a factor of two every week according to the following equations.

The Gridcoin approximation

$$\boldsymbol{RAC(new)} = \boldsymbol{RAC(old)} \times \boldsymbol{d(t)} + \boldsymbol{\left(1 - d(t)\right)} \times \boldsymbol{Credit(new)}$$

$$where$$

$$d(t) = \left(\frac{1}{2}\right)^{\frac{d}{604800}}$$

$$t = time\ (in\ seconds)\ since\ last\ calc$$

BOINC's official RAC pseudocode

```
#First calculate the weight value
weight = 2^[ -1 *
days_since_last_update/7days ]
#then determine which function to use
if weight is approx 1 (this occurs as
days_since_last_update approaches 0.0)
new_avg = old_avg * weight +
ln(2)*new_credit/7days #the math here is
a little more complex
else
#this is the main function and will be
used most of the time
new_avg = (old_avg - new_credit_per_day)
* weight + new_credit_per_day
```

---

plot $2^{-\frac{x}{7}}$ $x = 0$ to 100

Plot:

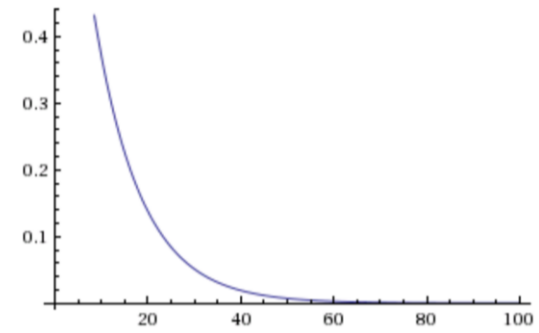

Figure 1: Weight function plot over number of days

If new_credit_per_day == 0, the above simply becomes

$$new\_avg = old\_avg * weight$$

plot $1000 \times 2^{-\frac{x}{7}}$ $x = 0$ to 100

Plot:

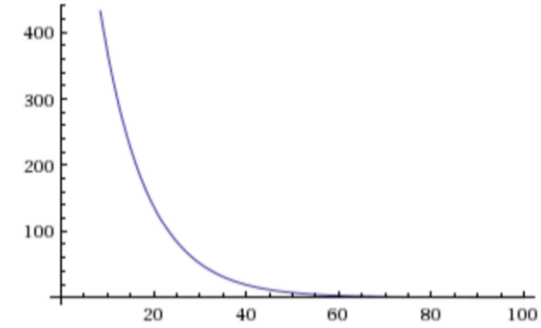

Figure 2: RAC/day @1000 RAC and no new credit

plot $500 \times 2^{-\frac{x}{7}} + 500$ $x = 0$ to 100

Plot:

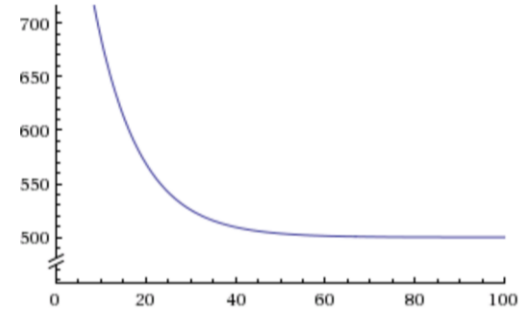

Figure 3: RAC/day @1000RAC and 500 new credit per day

plot $-500 \times 2^{-\frac{x}{7}} + 1500$ $x = 0$ to 100

Plot:

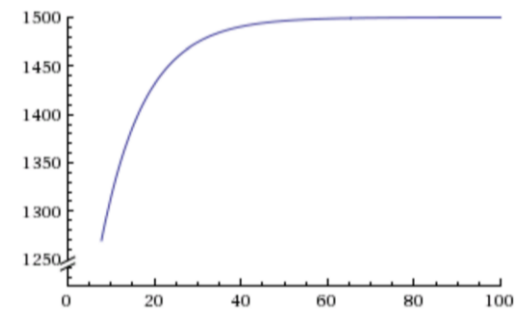

Figure 4: start with 1000RAC and 1500 new credit each day

The average FLOPS (floating point operations per second) achieved by a computer or group of computers can be estimated from its Recent Average Credit (RAC) as follows:

$$GigaFLOPS = RAC/200$$

The credit figures for a particular day may be distorted if a project is catching up or falling behind on validation (the process or granting credit). Thus to get accurate FLOPS estimates you should look at average RAC over a period of a week or so.

Note on credit: *"It's something that the developers are trying to make a system for, that eventually all projects will have to go use, but in the meantime, while it's not there yet, there's no really good way to measure efficiency between projects."*

II) RAC to Gridcoin conversion

$$\boldsymbol{Reward\ (in\ GRC)} = \boldsymbol{research\ age} \times \boldsymbol{magnitude\ unit} \times \boldsymbol{magnitude}$$

$$where$$

$$research\ age = time\ (in\ days)\ since\ last\ PoR\ staked$$

$$magnitude\ unit = adjustment\ factor\ of\ coin\ production\ for\ 1000\ blocks$$

$$magnitude = measure\ of\ work\ done\ by\ computer$$

$$\boldsymbol{magnitude} = \frac{\frac{\boldsymbol{researcherRAC}}{\boldsymbol{teamRAC}}}{\boldsymbol{Number\ Whitelist\ Projects}} \times \mathbf{115\ 000}$$

$$where$$

$$reasearcherRAC = RAC\ of\ the\ gridcoin\ miner/\ BOINC\ researcher\ for\ a\ given\ project$$

$$teamRAC = RAC\ of\ the\ entire$$

III) T. Hollis optimisation solution

1. Select top 10 favourite BOINC projects (CPU & GPU) based on personal interests for BOINC donation.

2. Run one CPU project and one GPU project at a time for 1 month and see how close you get after that month to the teamRAC. Note as a percentage of the team RAC. Also plot Gridcoin magnitude over time to check. Allow delay of up to 1 week for Gridcoin magnitude to scale.

3. Repeat step 2 for all CPU/GPU projects and select the one that gives you maximal researcher/teamRAC.