How I managed to find the seed from the famous Minecraft video by J!NX

Script (not really methodology, sorry about that) written by pablion (pabli0n).
Website: https://pablion.net/

Attempting (many times) to find the seed from the Siege on Castle Steve video created and published by **J!NX** in late 2011, which, as of today (2024-19-01), has 62 million views on YouTube.

This project was, well... quite a rollercoaster. Some time ago, around June 2022, I thought it would be interesting to delve into the field of seed finding while uncovering something not widely accessible to others. As you might suspect, this wasn't my first encounter with the topic. I had previously discovered a few other worlds through dungeons, also in an older version of Minecraft (one of them was less popular but still quite noteworthy). I considered it a significant achievement (at least for me) worth sharing with the world and the **Minecraft@Home** community.

Note! I'm not a mathematician but rather a passionate programmer who doesn't work in the industry due to health issues and insufficient age! For this reason, some of the approaches may be described non-technically or unclearly. If you spot any errors, please let me know.

In the initial approach to the problem, I had the idea, as with most previous projects by others, to use trees and tallgrass. However, this approach turned out to be too time-consuming and inefficient, considering I had to work only during the day and was hardware-limited (no cryptocurrency mining rig at home...).

Later, I tried to change my approach a bit and search for coordinates using clouds. However, I quickly realized that the clouds visible in the video were not consistent (despite having similar positions in the game), meaning I couldn't determine when a particular scene was recorded or how long the game was open.

So, I set aside this project for a while (about a year) to come up with a somewhat unusual idea in 2023, namely searching for a seed that generates a lake, which was probably the most visible feature of the entire population stage in this video. I wasn't sure about the positions of trees or whether they were planted or generated. As mentioned earlier, one detail I had overlooked was the lava lake. I started to examine the Minecraft code fragments responsible for generating lava lakes more closely.
I noticed that additional random sequences were generated for lava, visible as randomly placed stone blocks, which later allowed me to filter the results with GPU filtering.

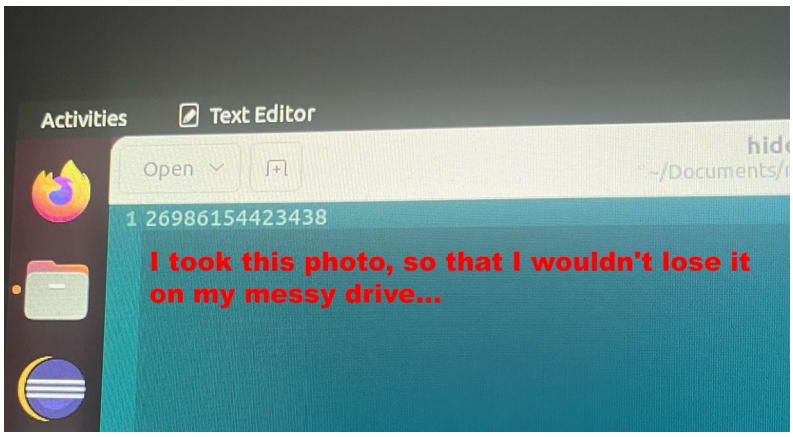
Recently, I managed to optimize and improve the code so that I could do it in this century (ha, very funny!)

As you probably guessed, to do this, I had to spend some time searching the entire seed space (**2^48**). Therefore, I divided the work into smaller parts, which I called stages (output files were named **stage71-%%number-of-inner-steps%%.txt**). So I could do each stage in about 5 hours.

The first (and more important) constraint was that the lava lake visible in the video was generated on the surface, so you could already see that the Y-level of this structure was >= 64 (the terrain height at this point was 69). From my observation, some lava lakes generated on the surface have a Y-level higher than 64, even more than 76. This is because there is a line in the Minecraft code that, when the lake is too high, lowers the Y value until it hits non-air block. Therefore, I decided to apply a condition in my algorithm that looked like this "*Y >= 70 && Y <= 120*".

The second equally important issue is that I limited the precision from double to float, allowing me to check the positions of stones in the lake a bit faster. (**It doesn't seem to make much of a difference when it comes to the resulting blocks**).

It turned out that the seed generating this lake was in the final sub-step of checking the GPU filter. The seed was: **26986154423438**



In the last stage, I used code from the project: https://github.com/hube12/DungeonCracker. Testing the lava lake seed with just one change (I reduced the number of regressions from 1000 to 2 since the lava lake is one of the first features to generate in a chunk, which also sped up the entire process and turned out to be the case after all). I found about 50 candidates (checking in-game if brown mushrooms and birch trees would be near the lava). **The 45th world seed was the long-awaited one**, as I didn't check further, considering it simply didn't make sense.

Whole process (without research) took about 5 days.

The source code I used: https://github.com/pabli0n/siege-castle-kernel

I utilized the computational power of **2x RTX 2060**, without which this process would have been much longer or even impossible for me.
Part of the approach seems to be empty, as things truly began to move forward only in late November 2023. For most of the time, I wondered about the purpose of various algorithms and what they are applied to.