Performance Analysis of go-bitcoin

Purpose

The purpose of this report is to analyze the effects of the number of miners and available processing threads on increasingly difficult puzzles.

Background

go-bitcoin is a reconstruction of two primary aspects of bitcoin: its miners, to provide proof-of-work for bitcoin transactions by solving a hash “puzzle,” and the immutable logger, which stores the transactions using a block chain.

A hash puzzle consists of transaction data and a target hash value. Miners search for a number which, when inserted into the transaction data, provides a hash that satisfies the target. More specifically, bitcoin utilizes a double-sha256 hash: when the transaction is hashed twice using 256, its resulting hash value must be less than or equal to the target.

The target should be a 256-bit integer, to match the size of a sha256 hash. However, to simplify reporting and analysis, the difficulty is stored as a 8-bit integer, with a max value of 255. This is later converted into a 256-bit integer by computing $2^{\text{difficulty}}$.

For the remainder of this analysis, “difficulty” will refer to this 8-bit number between 0 and 255. Because the hash must be less than or equal to the target, a higher number signifies a lesser difficulty. The axes for puzzle difficulty have been flipped to reflect this, so that the lower values which signify a greater difficulty extend to the right.

The logger is less relevant to this performance analysis. It is a block chain, such that each “block,” which stores the transaction information, points to the previous block via its hash value. The logger maintains the longest chain and sends completed puzzles to the miners, to use for the next puzzle.

Puzzle Difficulty

The puzzle difficulty increases by a factor of 2 with each decrease of the difficulty number. For example, with a difficulty of 250, the hash must be below 2^{250}. A properly written hash function should give a practically random output, and sha256 has a 256-bit output. Thus, there are 2^{256} possible outputs. Finding a correct solution to the puzzle will then have a probability of $2^{250}/2^{256}$ = $1/2^{6}$ = 1/64. Increasing the difficulty to 230 reduces the probability to $1/2^{26}$, or approximately 0.0000001%. It would take on average 67 mil. attempts to find a solution to a puzzle with difficulty 230, compared to 64 with a difficulty of 250.

For all the performance benchmarks, we recorded the time needed to construct a 10-block chain. We chose to solve 10 puzzles instead of a single one, to reduce any variation between runs.

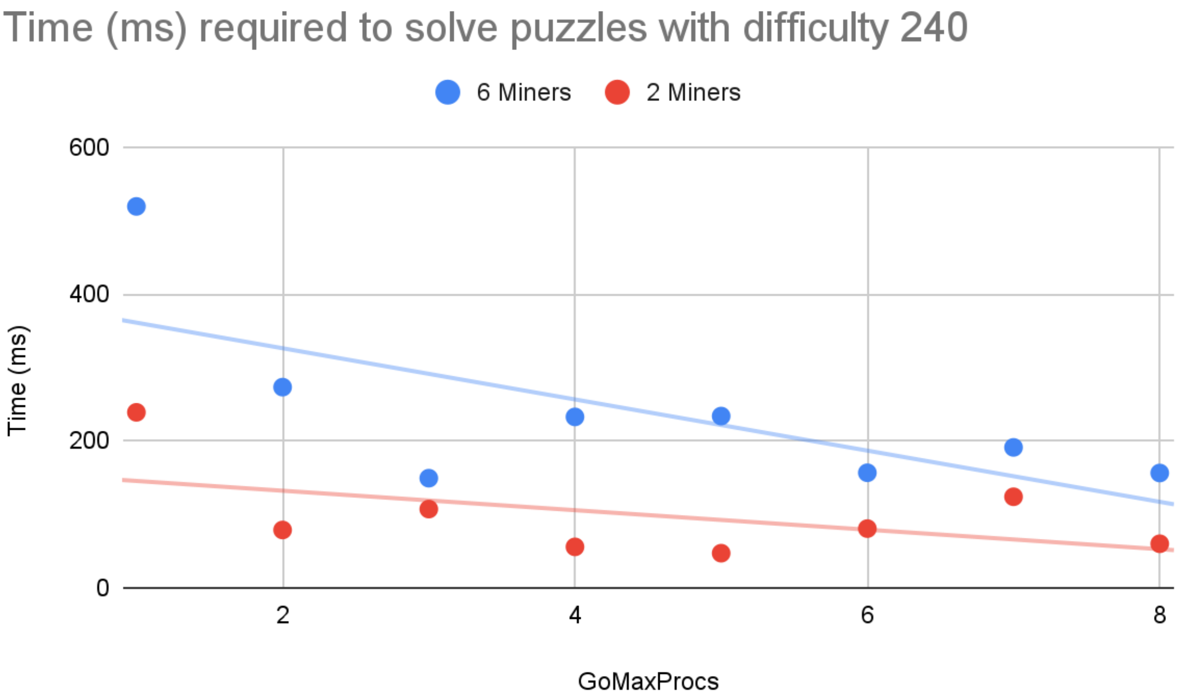
System Requirements

Because the test utilizes up to eight cores, to exactly replicate the test, one must have at least 8 CPU cores available. Additionally, the inherent difficulty of replication is because of the high stress put on processing these hash “puzzles” of increasing difficulty.

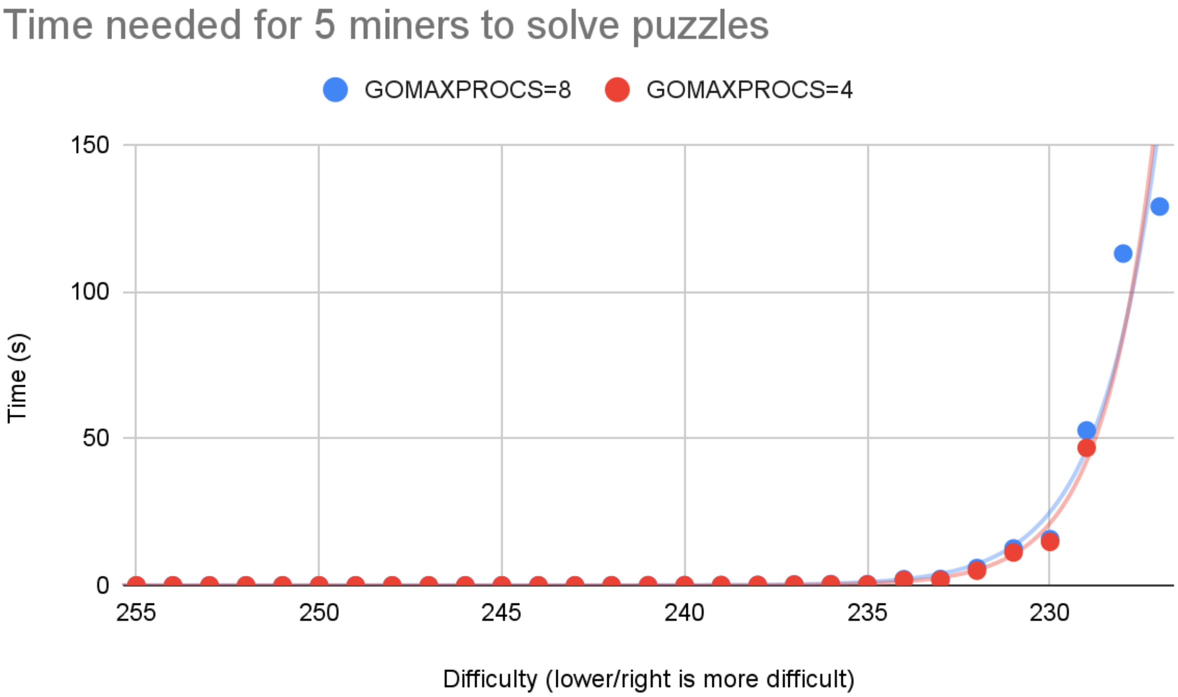
For reference, the processor used for this experiment was an 8-threaded Intel i5-8250U. The “stress test” on the mentioned processor pushed the puzzle difficulty to roughly ~230 (as shown in Figure 3). Afterward, the program ended with a “time out,” in which the program abandons a puzzle if it takes more than 5 minutes.

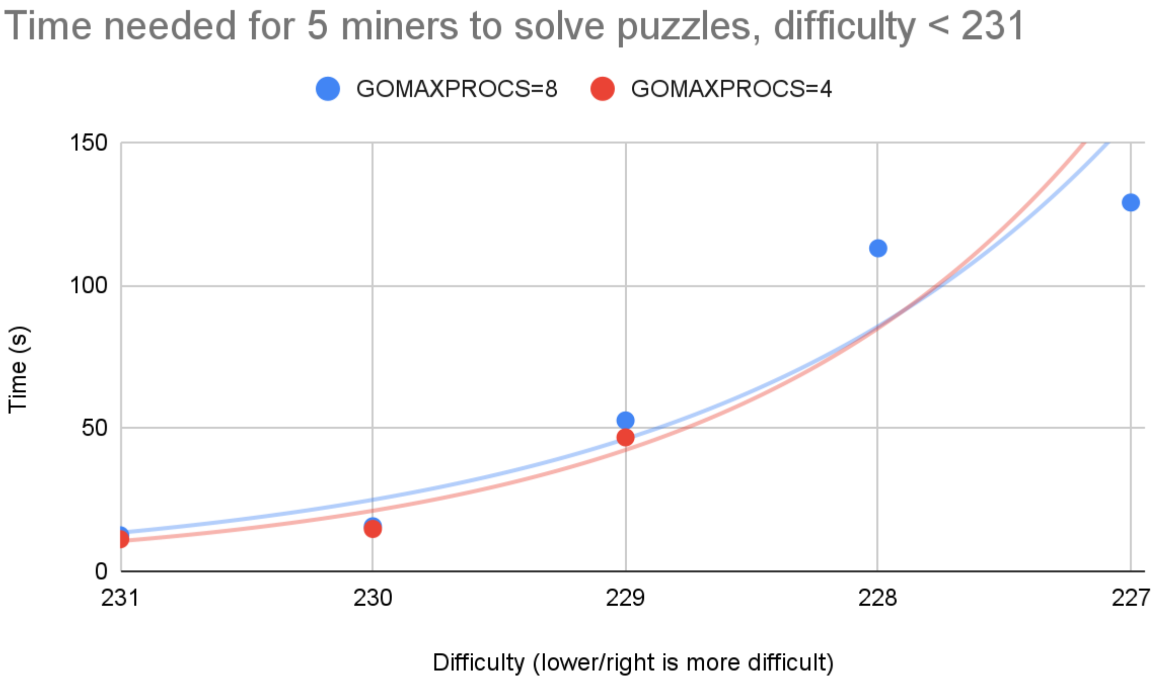
Performance vs. GOMAXPROCS

Increasing the number of GOMAXPROCS had a slight impact on performance, with both 6 miners and 2 miners. This is to be expected because increasing the number of available processes allows for greater parallelism and reduces the fighting for resources between goroutines.



However, when we increase the difficulty of the puzzles, GOMAXPROCS seems to have a lesser effect on the time needed to solve a puzzle.

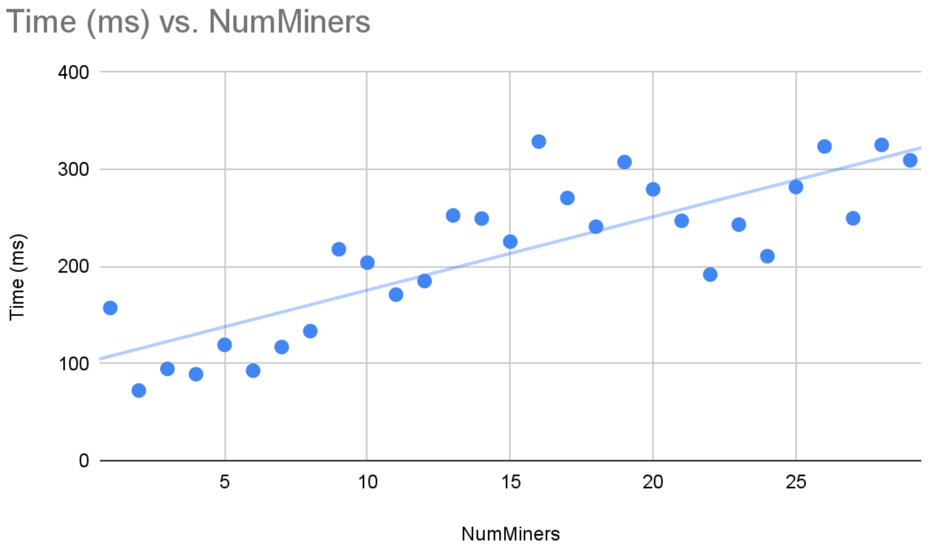




In this experiment, we kept the number of miners fixed, and increased the difficulty for GOMAXPROCS values of 4 and 8. While GOMAXPROCS=8 has a slight edge for the most difficult puzzles (see the second, zoomed in view), we were not able to find a significant distance. This could be due to the limitations of a 5-minute timeout, and we may be able to see a greater difference with more difficult puzzles.

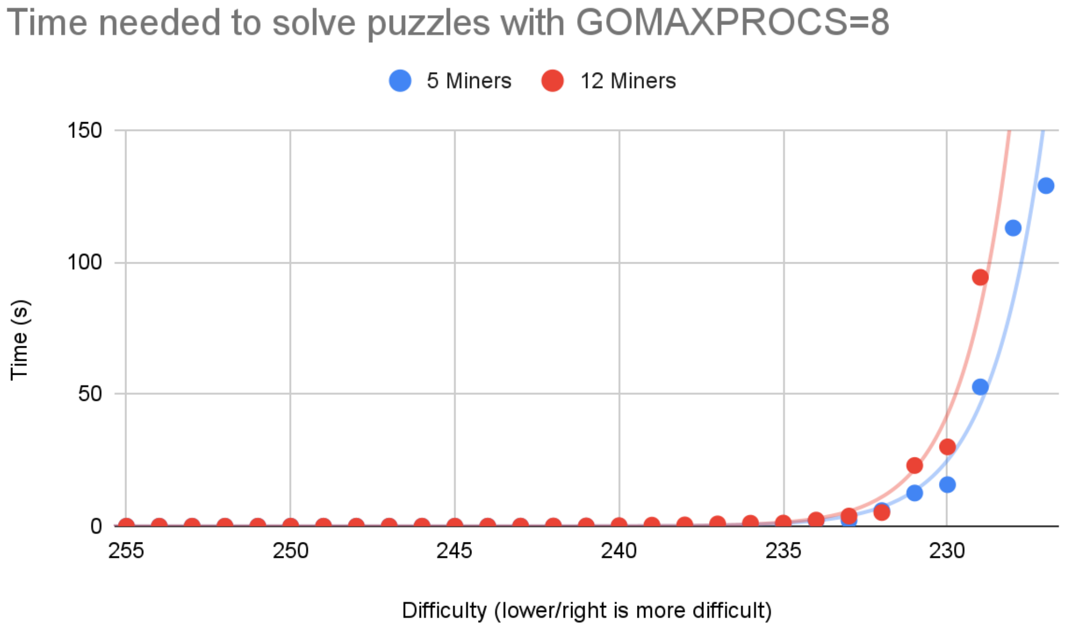
Performance vs. Number of Miners

In analyzing the performance of go-bitcoin, we also varied the number of miners working on a problem. When solving 10 blocks of difficulty 240 and GOMAXPROCS set to 8, there is actually a *positive* linear correlation between the number of miners and the time needed to complete the puzzles: increasing the number of miners made the program slower.



While against our initial predictions, we realized that this made sense. The decrease in performance is likely due to the miners having to share threads in the system. Once there are more goroutines than available processes, go must switch between goroutines, giving each one a little bit of processing time before switching to the next. The overcrowding of threads and the switches between goroutines is likely slower than simply permitting one goroutine to continue running on one thread. There is also no performance benefit to doing this: by switching between goroutines, they are not actually running in parallel, and there are at most 8 (i.e. the number of threads in the system) miners running at one time.

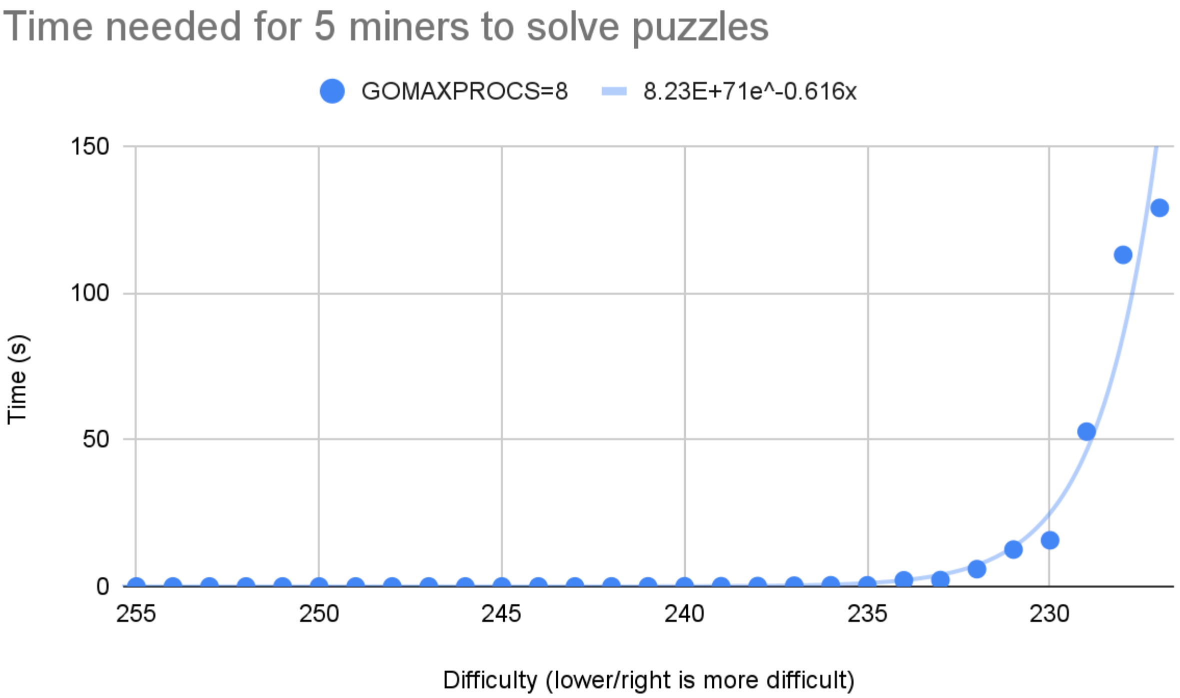
This was seen when we varied puzzle difficulty as well:



Increasing the number of miners from 5, which could each utilize an individual thread on the processor, to 12, which required sharing threads, provided a noticeable decrease in performance and increase in time needed to solve the puzzles.

Performance vs. Puzzle Difficulty

As we have seen earlier, increasing the puzzle difficulty increases the time needed to solve the puzzles. We expect there to be an exponential relationship between the difficulty and the time needed to solve a puzzle, because a decrease of one in the difficulty number halves the target value and doubles the actual difficulty. This exponential relationship was seen in the experiment, with an approximate trend of $e^{-0.616x} = 2^{-0.889x}$. This is slightly faster than the expected 2^{-x}, but not by a significant amount. We suspect this is because the effect of the extra overhead in sending a puzzle solution to the logger, and the logger's checking and adding it to the chain, is reduced with more difficult puzzles.


Time needed for 5 miners to solve puzzles
● GOMAXPROCS=8
8.23E+71e^-0.616x
Time (s)
Difficulty (lower/right is more difficult)