

Mfaktc – Primenet Specification
First Draft
May 4, 2010, 02:00 GMT
Eric Christenson

This preliminary specification indicates how the current release of mfaktc (currently approaching 0.17) will be modified to automate the interaction of mfaktc with primenet.

The current interaction of mfaktc with primenet is that a visit to www.mersenne.org, manual assignments, is required for both fetching work (that is, to get good data to create worktodo.txt) and for reporting results (that is, for reporting results.txt). Mfaktc is controlled from the mfaktc.ini file, which has keys that control mfaktc's behavior, including the names of the work file and the results file.

While menu-oriented editing of mfaktc.ini might be desirable and convenient, without so much as stopping mfaktc, it is beyond the scope of this specification.

First, mfaktc needs to be able to operate as it does now, with only manual primenet interaction as described above. Therefore, as with Prime95, a key "UsePrimenet" will be added to mfaktc.ini. If 0, the communications system is disabled. If 1, it is enabled.

Second, the communications system needs to be able to write to the console to indicate state, as with mfaktc. I propose that a line-by-line lock be instituted for the console, where either the communications system can write a line to the console, which it promises to complete quickly, or the computation system can write a line to the console, but not both at the same time.

Third, it is important that both the communications and the computation system not attempt to write the work file, worktodo.txt, at the same time. I again propose a lock, which will block the other system if editing is in progress. I have no issues with locking here, because access should be relatively infrequent.

Fourth, both the communications and computation systems need to edit the results.txt file. A lock is the preferred mechanism for this, again, access should be relatively infrequent.

These conditions allow for a largely independent and relatively simple communications thread whose job it is to periodically connect with the internet in order to get work and report results, and keep a console user informed about status.

A couple of issues come up with this:

At least on my computer, mfaktc generates many, many lines to the console for each work unit, and, if the workunit is small, these can be too frequent to effectively scroll the console upward. One possibility is to throttle the messages from the compute system; I have no difficulty with this except that it will almost certainly increase the number of patched files, and there is another solution that might be more effective – whenever a keypress reaches the console, the communications thread prints a short (few lines) status report to the console.

Second, I hold on to my results.txt files, as a guard against something happening to primenet servers, and mark them as to where the last reported result is. I propose that results.txt be handled in a similar fashion, assuming an mfaktc.ini key, "KeepResultsKB" is present, with a nonzero value indicating the maximum kilobytes of untransmitted results to be kept. This will operate as a log file, and as the end

grows, the beginning will be truncated. In addition, a marker is added to the end of results.txt indicating the end of what was reported to primenet, e.g. “[Credit from Primenet, 05/03/11, 21:58:03, 347.257 Ghz-Days]. (Hmm... an alternative to this would be to move the results to CreditedResults.txt, which would need its own naming key)

Third, GPUs threaten to overwhelm the primenet servers with small workunits. I suggest two mfaktc.ini keys to control this:

“UpperLimitIncrease=2”, to add two bits to the TF range from primenet, and
“MinWorkUnitMinutes=10” (at a minimum, which would increase the TF exponent until the work unit has reached 10 minutes of wall clock time, as reported by the OS).

Fourth, I believe the web interface has us reporting TF on a single bit range at a time. In a world in which the computers on both sides have an arbitrary speed relationship, we need to report on the unassigned ranges (the larger ones) before we report on the assigned ranges, that is, in backwards order. Otherwise, primenet could hand out the next bit range (on which the local mfaktc already has a result) as soon as the first assigned bit range completed reporting.

Fifth, discussion on LMH indicated that Primenet would prefer mfaktc work on standard Mersenne exponent ranges, currently in the 50-60M area. No “WorkType” key will be added to allow mfaktc to do LMH automatically.

Who is this Christenson nut, anyway?

Mr Christenson has an BS in EE/CS from Rose-Hulman, 1986, and has been programming heavily ever since. Along the way, two US patents have issued with him as inventor, one being in interferometry (5,106,192) and the other being for circuitry applied to large electromagnets. He currently works for Cableform, Inc, Troy VA, where he programs all of their DC motor controls and large electromagnet controls. He also began, but was unable to complete, the PhD math program at Lehigh University around 1995.