Rows 7 and 8 of Table III contain the results of the tests when using the GPU. On Row 7, the entire DNA sequences were sent to the GPU whereas on Row 8, sequences larger than 1 million bps were broken down to blocks of that size. It should be noted that Rows 7 and 8 yield the same results up to the size of 1 million bps. After that point, when the sequence is not broken down, the chosen GPU card has insufficient parallel resources and the performance diminishes. This means that, after 1 million bps, the memory requirements for every GFOR iteration become too great, thus hindering the parallel processing power of the GPU. For that reason, breaking larger sequences into blocks of 1 million bps yields the best results. This is illustrated in Fig. 7 which shows the speedup of the two GPU implementations when compared to the fastest CPU implementation. At 1 million bps, both implementations provide a 22% speed improvement compared to the CPU. When the sequence is broken down into blocks, the relative performance compared to the CPU remains constant. However, when the sequences are processed as a whole, the performance drops to the point that it would be slower than CPU processing.

DISCUSSION

The tests performed in the previous section show that the choice of algorithm and of implementation plays an important role in the feasibility of using gene prediction in MATLAB. Using a straightforward goertzel.m approach with a single thread would not have been possible within a reasonable time frame. Even with eight threads enabled, the extrapolated processing time for 15 million bps would be close to 53 minutes. When GoertzelMEX and a custom Goertzel algorithm were used, the processing time was reduced to close to a minute and a half. Finally, with the GPU, the processing time could be reduced to 67s. These results show that acceleration via a CUDA-enabled graphics card yields the best performance when the sequence to be analyzed is large enough to justify the overhead and small enough so as to not deplete available resources. For a relatively simple yet efficient implementation, a custom Goertzel algorithm using MATLAB’s parallelism can provide results that are only about 22% slower than with a GPU.

While the absolute difference in processing time may seem inconsequential, it should be noted that frequency analysis is often combined with other techniques to make gene prediction more robust. In addition, to improve on the reliability of the approach, it is sometimes relevant to perform frequency analysis for different window sizes. In those cases, the benefits of using a GPU can be justified. Otherwise, a well-designed MATLAB algorithm running on CPU would provide satisfactory results.

CONCLUSION

In this paper, we presented a number of ways of implementing gene prediction using MATLAB. The different implementations were described and evaluated to test for processing time. In order to validate their usefulness in gene prediction, a representative size of 15 million bps was used. In addition, each implementation was also evaluated for shorter DNA sequences to help analyze how the processing time evolves with different sequence lengths. Results show that, with common desktop computers, it is possible to perform gene prediction on sequences of 15 million bps in a short period of time. Using MATLAB’s FFT, an eight-core parallel implementation was able to complete the operation in less than five minutes whereas a GPU-accelerated version did it in just over one minute. The fact that these results can be achieved within the MATLAB environment without calling upon custom software means that researchers already familiar with MATLAB can readily use this technique without requiring additional IT resources. This can help make gene prediction tools more accessible to geneticists and can help speed the discovery of new genes.

ACKNOWLEDGMENT

This work was supported in part by NSERC. The authors would also like to thank RESMiQ, CMC Microsystems and AccelerEyes for access to software and other resources. Finally, we would also like to thank M. J.-D. Otis for helpful discussions and suggestions.

REFERENCES

