International Conference on Computational Science, ICCS 2012

Detecting Earthquakes around Salton Sea Following the 2010 M_w7.2 El Mayor-Cucapah Earthquake Using GPU Parallel Computing

Xiaofeng Meng[a,*], Xiao Yu[b], Zhigang Peng[a,*], and Bo Hong[b]

[a]*School of Earth and Atmospheric Sciences, Georgia Institute of Technology, Atlanta, GA30332, USA*
[b]*School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA30332, USA*

Abstract

The recently developed matched filter technique is effective in detecting earthquakes during intensive aftershock or swarm sequences. However, currently our detection code can only process on single-CPU computers, which takes a long time to perform the cross-correlation between continuous seismic data and template events. In this paper, we present a GPU-based computation method to significantly accelerate the detection algorithm. By dividing the procedure into several routines and processing them in parallel, we achieve ~40 times speedup for one Nvidia GPU card compared to sequential CPU code. We apply the paralleled code to search around the Salton Sea geothermal field for missing earthquakes in a 90-day time window around the occurrence time of the 2010 M_w7.2 El Mayor-Cucapah earthquake. We obtain ~70 times more earthquakes than listed in the official Southern California Seismic Network catalog. These newly detected events could be used to help to better understand how earthquakes are triggered in the immediate vicinity of a mainshock rupture.

Keywords: aftershock; earthquake catalog; the matched filter technique; GPU computing

1. Introduction

More than 15 years ago, earthquake data was mostly recorded only when its shaking exceeded certain threshold, known as triggered mode. Thanks to the recent development of cheap and massive data storage devices, earthquake data is now routinely recorded as continuous mode. The resulting data explosion has opened up many new exciting research areas in the field of seismology, such as detection of deep non-volcanic tremor along major plate boundary faults [1, 2] and imaging of subsurface structures based on cross-correlation of continuous ambient noise recordings [3].

Another example of such new area is detecting new earthquakes that are not listed in existing earthquake catalogs. Previously, earthquakes were identified through visual hand-picking and association of seismic phases. However, immediately after a large earthquake or during an earthquake swarm, seismicity rate is extremely high and seismograms from individual earthquakes tend to overlap with each other. In this case, it is extremely difficult to

* Corresponding author. Tel.: +01-404-680-9971; fax: +01-404-894-5638.
E-mail address: xmeng7@gatech.edu.

manually pick and locate all earthquakes, resulting an incomplete earthquake catalog. Recovering those missing events and obtaining a more complete catalog are not only important for understanding physical mechanism of earthquake interaction [4, 5, 6], but also useful for seismic hazard forecasting and mitigation [7, 8].

An effective way to detect missing earthquakes is to use waveforms of previously identified earthquakes as templates and scan through continuous seismic recordings. This is also known as the matched filter technique [9, 10]. A new seismic event is identified when the waveform similarity between a template event and continuous recordings exceeds certain threshold. Based on this technique, 11 times more aftershocks than reported in the official US Geological Survey earthquake catalog have been identified in the first 2 days after a magnitude 6 earthquake along the Parkfield section of the San Andreas Fault in Central California [11].

However, this technique could be very computational intensive. For example, for a standard desktop computer with 2.27 GHz Xeon processor and 64 GB memory, it takes about 1.5 minutes to scan through 1 day of continuous data for one template event recorded at 6 seismic stations with sampling rate of 100/s. For ~2000 template events, it would take ~50 CPU hours (or ~2 days) to scan through 1 day of continuous data on the desktop computer. The computational complexity is therefore a major bottleneck that prevents this technique from being applied at a massive scale – typically involving thousands of template events and years of continuously recorded data.

To reduce computation times, we use GPU computing to accelerate the matched filter technique. GPU computing has recently evolved from a fixed-function graphical device into a highly programmable parallel processor [12, 13, 14, 15] and has been successfully deployed to accelerate a broad range of scientific applications [16-27]. The matched filter technique exhibits regular computation and memory access patterns, and is mostly data parallel, which makes the computation suitable for GPU processing. Our results show that GPU algorithm achieves ~40 times speedup over a single CPU core. By using 2 Nvidia C2070 cards, our GPU-based method achieves the equivalent performance of 20 quad-core processors. Because the computation process exhibits abundant data parallelism, we expect our method to scale to larger GPU systems and thus provide a more cost and energy efficient solution over existing CPU-cluster based method. Using the parallelized version of the code, we have detected many missing earthquakes in the Salton Sea geothermal field in southern California following the 2010 M_w7.2 El Mayor-Cucapah earthquake in Baja California, Mexico. In this paper, we describe how GPU computing is implemented and present initial results from our study.

2. Work region and data set

The 2010 M_w7.2 El Mayor-Cucapah earthquake is the largest earthquake occurred in the vicinity of southern California since the 1992 M_w7.3 Landers earthquake (Fig. 1). Its aftershock zone spans ~120 km between the southern end of the Elsinore fault near the US-Mexico border and the northern tip of the Gulf of California. The mainshock induced significant stress and seismicity rate changes in southern California [28], which is one of the most heavily instrumented areas in the world. Hence, this event provides a great opportunity to study how earthquakes are triggered around a mainshock rupture.

In this study, we focus on the Salton Sea geothermal field mainly because of its dense seismic network, intensive background seismicity and large stress changes following the mainshock. According to the official Southern California Seismic Network (SCSN) catalog, the seismicity rate near Salton Sea increased immediately after the mainshock, but dropped below the pre-shock level within ~20 days and remained low for a few months [29]. However, it is not clear whether such pattern is caused by missing earthquakes after the mainshock or reflect genuine feature of seismicity changes.

To check the existence of missing earthquakes immediately after the mainshock, we apply a 10-40 Hz band-pass filter to the 6 borehole stations (EN network) near Salton Sea (Fig. 1). These stations were installed at 10-40 m depth and recorded seismic data continuously since 12/2008 with a sampling rate of 100/s. We use a 10-40 Hz filter to enhance locally generated signals and reduce potential contaminations of signals from the mainshock and large aftershocks near the mainshock rupture. Fig. 2 shows the 10-40 Hz filtered envelope function at station ELM in the first 1000 s after the mainshock. Many high-frequency spikes are shown in this time period, which are mostly produced by local earthquakes. In comparison, only three events were listed in the SCSN catalog. Hence, it is evident that many earthquakes are missing in the catalog during this period and perhaps at other times.

To recover those missing earthquakes, we apply the matched filter technique to the Salton Sea geothermal field. The continuous data is from 03/01/2010 to 06/01/2010, which is 34 days before and 57 days after the El Mayor-

Cucapah mainshock, respectively. A total of 2088 events from the relocated catalog of [28] are used as templates (Fig. 1).
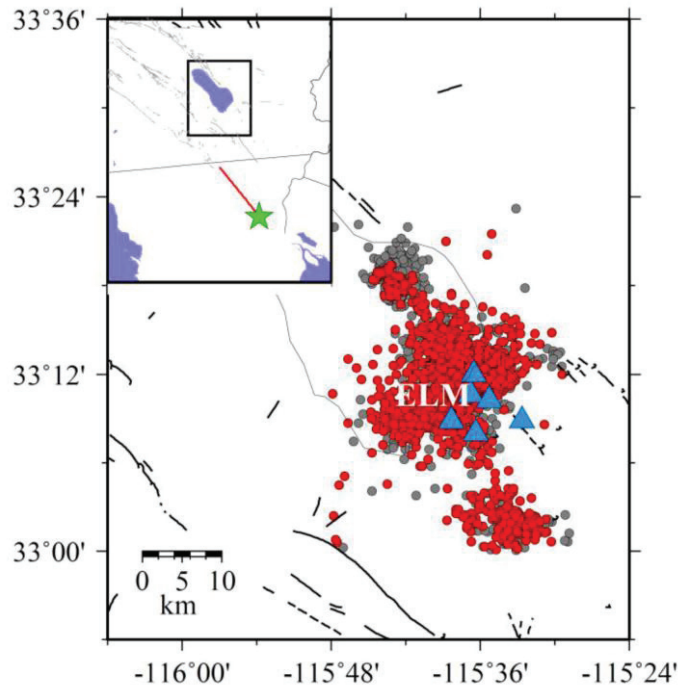

Fig. 1. Map of the Salton Sea geothermal field in southern California. The black lines are active faults. The blue triangles denote 6 borehole stations from EN network. The gray dots are all earthquakes listed in the Southern California Seismic Network (SCSN) catalog. The red dots are 2088 template events we used in this study. The inset is a map of US-Mexico boundary area. The black box corresponds to the study region. The green star is the epicenter of the 2010 M_w7.2 El Mayor-Cucapah earthquake. The red line marks the rupture plane of the mainshock.
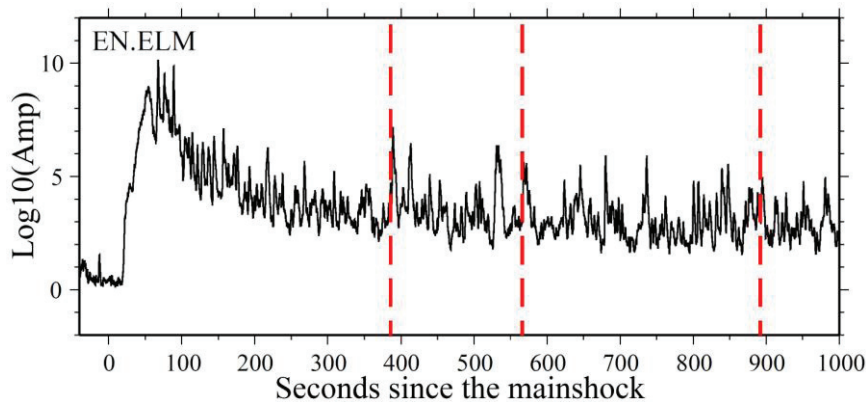

Fig. 2. A 10-40 Hz band-pass filtered envelope function for station EN.ELM showing evidence of locally generated seismic events near the Salton Sea in the first 1000 s following the El Mayor-Cucapah earthquake. The red dashed lines mark the origin time of events around Salton Sea listed in the SCSN catalog.

3. The matched filter technique

The procedure follows previous work [11] and is briefly described as follows. First, we apply a band-pass filter of 10-40 Hz to continuous data and template events to enhance locally generated seismic signal. Each template event

must have at least 12 channels (4 stations) with signal-to-noise ratio larger than 5. The signal and noise amplitude is obtained from a 4-s time window starting from 2 s before S-wave arrival and 6 s before P-wave arrival, respectively.
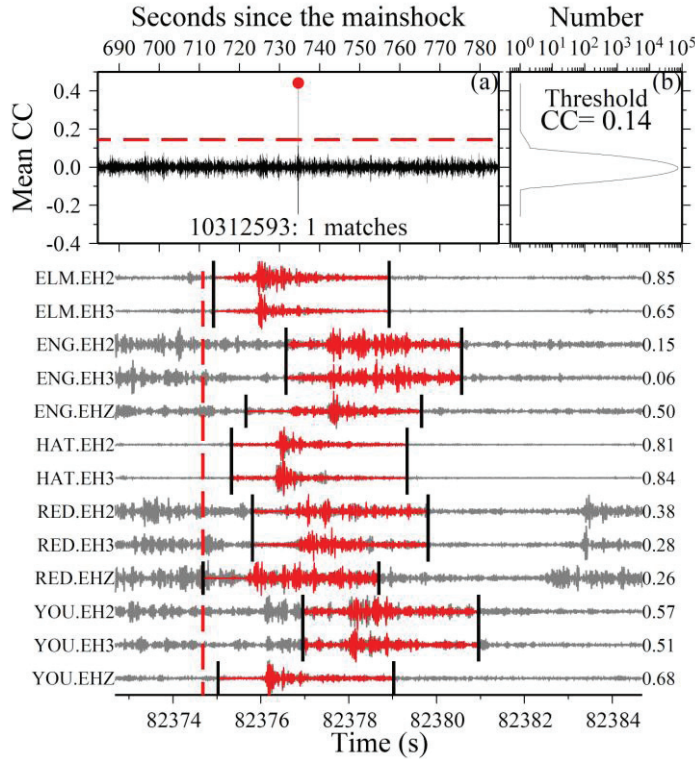

Fig. 3. An example of newly detected event near Salton Sea after the El Mayor-Cucapah mainshock. (a) Mean correlation coefficient trace versus time for the template event 10379125. The red dot corresponds to the detected event at ~12 min after the mainshock; (b) The histogram of the mean CC function; (c) A comparison of the template waveforms (red) and the continuous waveforms (grey) around the origin time of detected event (vertical dashed line). The channel names and the corresponding CC values are labeled on the left and right sides, respectively.

We compute the correlation coefficient (CC) value for all data points within a 4-s time window between the template and continuous waveforms:

$$CC = \frac{\sum_{t_0}^{t_1}((X(t) - \overline{X}) * (Y(t) - \overline{Y}))}{\sqrt{\sum_{t_0}^{t_1}(X(t) - \overline{X})^2 * \sum_{t_0}^{t_1}(Y(t) - \overline{Y})^2}} \tag{1}$$

where CC is correlation coefficient value, t_0 and t_1 is the start and end time of the 4-s time window, respectively; $X(t)$ and $Y(t)$ is the time series for the template and continuous waveforms, respectively.

The 4-s time window is set to be 1 s before and 3 s after *S*-wave arrival time for two horizontal channels, and 1 s before and 3 s after *P*-wave arrival time for the vertical channel. Then, we shift the CC value back to the origin time of the template event by subtracting *S*- or *P*-wave arrival time. Next, we move forward by one data point and repeat the computation for the entire continuous waveform. After we scan through continuous data for all channels and stations, we stack all correlation traces to obtain the mean CC value. We then compute the median absolute deviation (MAD) for each template, and use 9 times the MAD as a detection threshold. For a normal distribution, the probability of exceeding 9 times the MAD is 6.4×10^{-10}, suggesting that it is very unlikely to be a random detection. We finally combine all detections from all templates and assign the location of the detection to that of the template. For multiple detections in each 2-s window, only the detection with the highest correlation coefficient value is kept. The magnitude of the detected event is computed based on the median value of the maximum

amplitude ratios for all channels between the detected and template event. Fig. 3 shows a positive detection on 04 April 2010 at 22:52:56, approximately 734 s after the El Mayor-Cucapah earthquake. This detection has a mean CC value of 0.45, well above the threshold of 0.14.

4. GPU implementation

To improve the computation efficiency of matched filter technique, we explore the parallelism in the problem and use GPU-based systems to achieve significantly acceleration. The matched filter technique exhibits parallelism at multiple levels: there exists multiple templates, each of which needs to be matched with the continuous waveform data; the continuous waveform itself consists of a long sequence of windows of data points; both the templates and the continuous waveform data contain multiple channels of data recorded by multiple stations; further down the parallelism hierarchy, each channel consists of multiple data points, which forms the input for the correlation analysis.

To explore such hierarchy parallelism, we decompose the computation into multiple tasks that computes the correlation of a template and a window of continuous waveform data, where each task is to match one template with a window of continuous waveform of the same length. Our design focus on two algorithmic aspects: (1) task grouping and allocation to the GPU cards, and (2) GPU code optimization for each task.

4.1. Task grouping and allocation

To allocate the tasks onto GPU cards, the allocation strategy needs to focus on three aspects: (1) Each group should not exceed the GPU card memory capacity. (2) The allocation strategy should cover all the tasks and balance among groups. (3) The allocation strategy should minimize the cost of communication and redundant computation.

The first aspect guarantees each GPU card has enough memory to contain the data, including input, output and temporary for the execution. The second aspect ensures that all the tasks are dispatched to a GPU card and the computation time for all GPU cards are relatively equal. For the third aspect, as is illustrated in Fig. 4, the computation of each task requires data preparation and movement. Thus a desirable allocation strategy is expected to minimize those costs. Additionally, tasks shared some of their computations. For example, the mean value and variance of a template is shared when matching this template to multiple windows of continuous waveform. The same sharing applies to the continuous waveform as well. It is thus undesirable to repeat such computation when matching each pair of template and continuous waveform.

In our task allocation strategy, information about task sizes is used to calculate the memory need so as to maximize the number of tasks that a GPU card can process. Because the templates are of the same size, load balance of multiple GPU cards is relatively easy. Allocating the same number of tasks to the GPU cards would ensure the same computation time.

The cost of data movement and shared computation is minimized as follows. Each task of the correlation computation can be indexed by a tuple (i, j) where i represents the template index and j the continuous waveform. The group size of tasks can also be represented by a tuple (p_i, p_j) so that each card will calculate on p_i template and p_j continuous waveforms. The group size of tasks (p_i, p_j) should be a solution for the optimization problem,

$$\min[(k_{copy\_t} + k_{es\_t})p_i + (k_{copy\_c} + k_{es\_c})p_j] \tag{2}$$

where k_{copy_t} and k_{es_t} are the coefficients for consumed time on copying and calculating mean and variance value for template waveform, and similarly k_{copy_c} and k_{es_c} for continuous waveform. Under the constraints,

$$p_i p_j N_p = N_t N_c$$

$$k_{size\_t}p_i + k_{size\_c}p_j \le C$$

$$\tag{3}$$

$$0 \leq p_i \leq N_t$$

$$0 \leq p_j \leq N_c$$

where N_p is the total number of groups; N_t is total number of template waveforms and N_c of continuous waveforms; k_{size_t} is the size of each template data and k_{size_c} is the size of continuous data; C is the GPU memory capacity. The solution to the problem is,

$$p_{i,opt} = \sqrt{\frac{k_{copy\_t} + k_{es\_t}}{k_{copy\_c} + k_{es\_c}} \frac{N_t N_c}{N_p}}$$

$$p_{i,max} = \min(N_t, \frac{C}{k_{size\_t}})$$

$$p_{i,min} = \max(\frac{N_p}{N_t}, \frac{N_t N_c k_{size\_t}}{N_p C}) \qquad (4)$$

$$p_{i,optimize} = \begin{cases} p_{i,max} & \text{if } p_{i,max} \leq p_{i,opt} \\ p_{i,min} & \text{if } p_{i,min} \geq p_{i,opt} \\ p_{i,opt} & \text{if } p_{i,min} \leq p_{i,opt} \leq p_{i,max} \end{cases}$$

$$p_{j,optimize} = \frac{N_t N_c}{N_p p_{i,optimize}}$$

Since the ratio between the data size and the cost of copying time and computation time is approximately fixed, the task allocation strategy can find a solution to minimize the cost based on the data size information collected using Eq. 2.

4.2. GPU computation kernels

For a group of tasks (computes the correlation of a template and a window of continuous waveform data) on one GPU card, fine-grained multiple-threaded kernel routines are designed for the GPUs.

The design of the kernels mostly concern memory coalescing and shared memory usage. To efficiently utilize the GPU resource, threads should access the global memory in a coalesced pattern, i.e. neighboring threads should access continuous memory addresses. Moreover, shared memory access has a much lower latency than global memory and therefore reusable data should be loaded into shared memory to improve the performance. Kernels in our program have different memory access patterns, and thus have different coalescing and sharing strategies. Our design focuses on the computation of correlation routine (the *CalcCorrelation* function) that dominates the execution time. This computation iterates through all the template stations and continuous window pairs to calculate the correlation values. To coalesce the memory access, we reorganize the template station data layout so that neighboring template data in the global memory are of different template station. Using this data layout, neighboring threads working on different correlation pairs can access different template data in a coalesced manner. The kernel threads also have shared memory accesses to continuous data such that it is beneficial to load continuous data into shared memory. Other kernel routines memory access patterns are adjusted and coalesced to accommodate this memory layout.

Another design factor is the configuration of grid size and block size. In CUDA GPU programming model, large number of threads are grouped into blocks and one kernel routine can launch multiple blocks. Different block size and grid size can affect the performance of kernel because of the hardware resource allocation (e.g. number of registers) for each thread. Currently, we configure our kernels with 56 blocks and 1024 threads manually for Nvidia Tesla C2070 card. The execution time of each kernel is monitored at runtime and the program is designed to be extendible to dynamically reconfigure for other generation of GPU cards or application parameters.
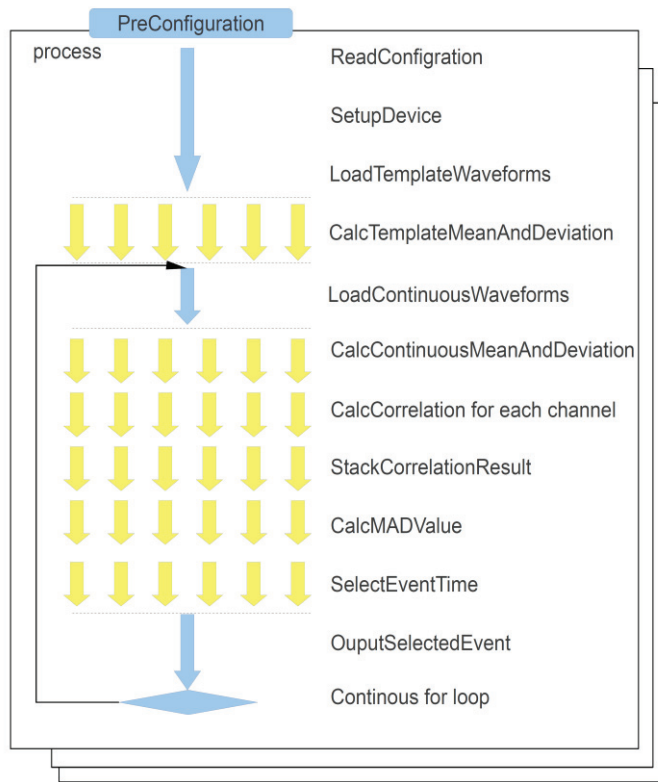
Fig. 4. Program routines and control flow. Blue blocks represent CPU routines, and yellow blocks represent GPU routines. After *Preconfiguration*, CPU processes launch a series of routines in parallel.
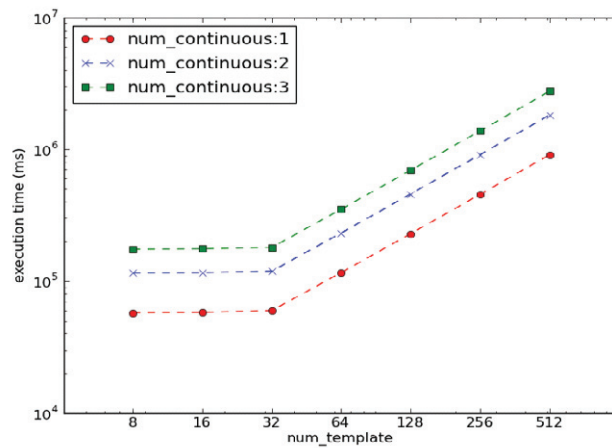

Fig 5. Program scalability. Each line illustrates a different number of continuous waveforms from 1 to 3. X-axis lists different numbers of template waveforms from 8 to 512. Y-axis shows the execution time in ms.
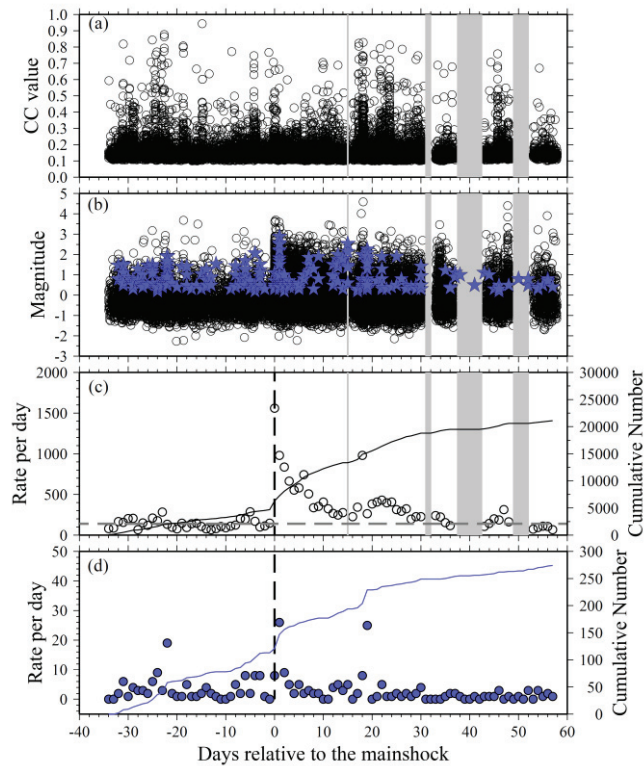
Fig. 6. Summary of newly detected events and the SCSN catalog. (a) The CC values versus the origin times of all detected events. The gray shadow areas mark the gap in the continuous data, which resulted in zero detection in these time periods. (b) The magnitudes versus the origin times of all detected events (black circles) and earthquakes listed in the SCSN catalog (blue stars), respectively. (c) Seismicity rate of all detected events in the Salton Sea geothermal field. The vertical dashed line denotes the origin time of the El Mayor-Cucapah mainshock. The black circles are daily seismicity rate. The horizontal dashed line denotes to the pre-shock level of seismicity rate, which is the median value of all the days before the mainshock. The black curve is the cumulative number of detected events. (d) Seismicity rate of earthquakes listed in the SCSN catalog. Other notations are the same with that of (c).

5. Results

In our study, GPU kernels were running on Nvidia Tesla C2070 cards. The baseline CPU code, as well as the host system for GPU cards, is running on 2.27 GHz Intel Xeon processors. The software tools used are gcc 4.4.1 and nvcc 4.0 release. As shown in Fig. 5, our program scales almost linearly with respect to the number of template and continuous waveforms when number of template waveforms exceeds 32. For smaller number of templates, the execution time remains the same because the program is designed to access different template each GPU warp of thread to allow coalescing memory access. The performance bottleneck, the *CalcCorrelation* for each channel routine, takes around 90% of the execution time such that optimization of the program should be focused mainly on this kernel. Because on average only one memory operation is applied for each global memory access, the GPU kernel is memory bound. Profiling results show that our GPU kernel achieved a memory throughput of around 140 Gbytes/s, which is close to the peak bandwidth of 144 Gbytes/s for Tesla 2070. This demonstrates that our GPU kernel is performing close to the theoretical upper bound of the GPU devices.

Such an accelerated execution speed allows us to complete the entire computation (with ~2000 templates and ~80 days of continuous data recording) with ~48 hours of GPU computation. So far we have detected a total of ~24000 earthquakes, ~70 times more than listed in the SCSN catalog (Fig. 6a-b). Among these events, 4842 earthquakes were before the El Mayor-Cucapah earthquake. We then calculate the seismicity rate changes based on newly detected events and SCSN catalog separately (Fig. 6c-d). Despite the large difference in total number of earthquakes, the daily seismicity rate from the newly detected events shows a similar pattern with the SCSN catalog: a significant increase in the first day after the mainshock, and followed by a rapid decrease in the following days, except on the 18th day after the mainshock, when one burst of seismic activity occurred. The seismicity rate dropped below the pre-shock level at about 50 days after the mainshock.

6. Summary and Future Directions

We have successfully developed a GPU-paralleled version of the matched filter technique, and apply it to search for missing earthquakes at Salton Sea in a 90-day time window around the occurrence time of the 2010 M_w7.2 El Mayor-Cucapah earthquake. We have detected ~70 times more earthquakes than listed in the official SCSN catalog. The seismicity rate changes were similar for both the SCSN catalog and newly detected events, suggesting that in this case, missing earthquakes in the catalog does not appear to change the overall patterns. However, with those newly identified events, we obtained a cleaner short-term increase of seismicity, followed by a decay and possible reduction below the pre-shock level near the end of our analysis period.

Our success suggests that the GPU-paralleled matched filter technique is capable of detecting missing local earthquakes with very efficient speed. We plan to apply the same technique to a much longer time period before and after the El Mayor-Cucapah earthquake to detect more local earthquakes, and use them to study the seismicity rate changes in the Salton Sea geothermal field. We will also compare with the stress changes pattern induced by the mainshock [30] to better understand the physical mechanisms of aftershock triggering.

Our long-term goal is further develop this code and make it available via peer-reviewed publications. We envision that this code will be used by many researchers in observational seismology to detect new seismic events at times/regions that are otherwise impossible for human analysts to handpick all the earthquakes [31, 32]. In addition, this method can be incorporated by seismic data centers to develop automatic earthquake location techniques in near-real time [33, 34].

References

1. K. Obara, Science, 296, 1679–1681 (2002)
2. Z. Peng, Z. and J. Gomberg, Nature Geosci., 3, 599–607 (2010)
3. N. M. Shapiro, M. Campillo, L. Stehly, and M.H. Ritzwoller, Science, 307, 1615-1618 (2005)
4. R. S. Stein, Nature, 402, 605-609 (1999)
5. R. S. Stein, Earthquake conversations, Scientific American, 288, 72-79 (2003)
6. D. P. Hill, and S. G. Prejean, in Treatise on Geophysics, 257–292, ed. G. Schubert, Vol. 4: Earthquake Seismology, ed. H. Kanamori, *Elsevier*, (2007)
7. P. A. Reasenberg, and L. M. Jones, Science, 243, 1173–1176 (1989)
8. M. C. Gerstenberger, S. Wiemer, L. M. Jones, and P. A. Reasenberg, Nature, 435, 328–331 (2005)
9. S. J. Gibbons, and F. Ringdal, Geophys. J. Int., 165, 149–166 (2006)

10. D. R. Shelly, G. C. Beroza, and S. Ide, Nature, 446, 305–307 (2007)

11. Z. Peng, and P. Zhao, Nature Geosci., 2, 877–881 (2009)

12. J. Nickolls, and W. J. Dally, IEEE Micro 30(2): 56-69 (2010)

13. M. Macedonia, Computer 36(10): 106-108 (2003)

14. J. D. Owens, M. Houston, D. Luebke, S. Green, J. E. Stone, and J. C. Phillips, Proceedings of IEEE 96(5): 879-899 (2008)

15. I. Buck, Proc. Int. Symp. Code Generation and Optimization CGO '07: 17 (2007)

16. W. H. Choi, and X. Liu, Proc. IEEE Int Circuits and Systems (ISCAS) Symp: 917-920 (2010)

17. F. Feinbube, B. Rabe, M. von Löwis, and A. Martin Polze, Proc. Ninth Int Parallel and Distributed Computing (ISPDC) Symp: 63-70 (2010)

18. Y. Huang, Y. Fan, C. Li, T. Tong, and H. Feng, Proc. 4th Int Bioinformatics and Biomedical Engineering (iCBBE) Conf: 1-4 (2010)

19. R. Jiang, X. Wu, F. Zeng, Z. Yu, and W. Zhang, Proc. Int. Joint Conf. Bioinformatics, Systems Biology and Intelligent Computing IJCBS '09: 70-76 (2009)

20. P. Kuchnio, and D. W. Capson, Proc. 16th IEEE Int Image Processing (ICIP) Conf: 2325-2328 (2009)

21. L. Ligowski, and W. Rudnicki, Proc. IEEE Int. Symp. Parallel & Distributed Processing IPDPS 2009: 1-8 (2009)

22. A. Maringanti, V. Athavale, and S. Patkar, Proc. Int High Performance Computing (HiPC) Conf: 438-444 (2009)

23. R. Riedmuller, C. Busch, M. Seeger, S. Wolthusen, and H. Baier, Proc. 2nd Int Security and Communication Networks (IWSCN) Workshop: 1-8 (2010)

24. W. Waizenegger, P. Kauff, I. Feldmann, and P. Eisert, Proc. 16th IEEE Int Image Processing (ICIP) Conf: 4301-4304 (2009)

25. X. Wang, F. Qiu, G. Chen, and S. Prasad, Proc. IEEE Int Parallel & Distributed Processing (IPDPS) Symp: 1-9 (2010)

26. X. Ye, N. Yuan, D. Fan, P. Ienne, and W. Lin, Proc. IEEE Int Parallel & Distributed Processing (IPDPS) Symp: 1-10 (2010)

27. Q. N. Tran, Proc. Seventh Int Information Technology: New Generations (ITNG) Conf: 7-12 (2010)

28. E. Hauksson, J. Stock, K. Hutton, W. Yang, J. A. Vidal-Villegas, and H. Kanamori, Pure Appl. Geophys., 168, 1255-1277 (2010)

29. X. Meng, Z. Peng, and P. Zhao, 2011 AGU Fall meeting, S21D-02 (2011)

30. R. S. Stein, V. Sevilgen, S. Toda and J. Lin, Report to the California Seismic Safety Commission (2010)

31. Z. Peng, J. E. Vidale, and H. Houston, Geophys. Res. Lett., 33, L17307 (2006)

32. Z. Peng, J. E. Vidale, M. Ishii, and A. Helmstetter, J. Geophys. Res., 112, B03306 (2007)

33. D. H. Von Seggern, and K. D. Smith, Seism. Res. Lett., 79(2), 325 (2008)

34. F. Waldhauser, Bull. Seism. Soc. Am., 99, 2736-2848 (2009)