RAA 2020 Vol. **20** No. **6**, 91(8pp) doi: 10.1088/1674–4527/20/6/91

http://www.raa-journal.org *http://iopscience.iop.org/raa*

Research in Astronomy and Astrophysics

A PRESTO-based parallel pulsar search pipeline used for FAST drift scan data

Qiu-Yu Yu (余秋雨)[1], Zhi-Chen Pan (潘之辰)[2], Lei Qian (钱磊)[2], Shen Wang[2], You-Ling Yue[2], Meng-Lin Huang[2], Qiao-Li Hao[2], Shan-Ping You[1], Bo Peng[3], Yan Zhu[4], Lei Zhang[2] and Zhi-Jie Liu[1]

[1] Guizhou Normal University, Key Laboratory of Information and Computing Science, Guiyang 550001, China; *liuzj@gznu.edu.cn*

[2] National Astronomical Observatories, Chinese Academy of Sciences, Beijing 100101, China; *panzc@bao.ac.cn*

[3] School of Information Engineering, Southwest University of Science and Technology, Mianyang 621010, China

[4] School of Physics, University of Electronic Science and Technology of China, Chengdu 610054, China

Received 2019 November 14; accepted 2019 December 29

Abstract We developed a pulsar search pipeline based on PulsaR Exploration and Search TOolkit (PRESTO). This pipeline simply runs dedispersion, Fast Fourier Transform (FFT) and acceleration search in process-level parallel to shorten the processing time. With two parallel strategies, the pipeline can highly shorten the processing time in both normal searches and acceleration searches. This pipeline was first tested with Parkes Multibeam Pulsar Survery (PMPS) data and discovered two new faint pulsars. Then, it was successfully applied in processing the Five-hundred-meter Aperture Spherical radio Telescope (FAST) drift scan data with tens of new pulsar discoveries up to now. The pipeline is only CPU-based and can be easily and quickly deployed in computing nodes for testing purposes or data processing.

Key words: methods: data analysis — pulsars: general — star: neutron

1 INTRODUCTION

A typical pulsar search process consists of (1) selecting and marking the Radio Frequency Interferences (RFIs), (2) dedispersion, (3) transforming the time domain data to a frequency domain signal, for example, using Fast Fourier Transform (FFT), (4) searching for periodic signals and (5) finding pulsar candidates from search results. One of the goals of pulsar search software is to accelerate these steps in order to shorten the data processing time. One of the basic ideas is finding a suitable pulsar searching plan. For example, reducing the number of dispersion trails is equivalent to reducing the total computational effort and thus reducing the data processing time. Hardware like Graphics Processing Units (GPUs) also accelerate pulsar searches (e.g., the code peasoup[1], Barr et al. in prep). In dedispersion, a GPU/multicore-based code can easily reduce the data processing time to 50% of the original value or even shorter (Sclocco et al. 2016). Using multi-core Central Processing Units (CPUs) and GPUs, the coherent dedispersion pipeline of the Giant Metrewave Radio Telescope (GMRT) is better than realtime (De & Gupta 2016). For the LOw Frequency ARray (LOFAR), GPUs speed up dedispersion over 2.5 times faster than the previous Fermi code with rough estimation (Serylak et al. 2013). For periodic signal searches, GPU-based acceleration search codes (e.g., Luo et al. in prep) also reduce the searching time. For the Southern High Time Resolution Universe (HTRU) pulsar survey, the GPU-accelerated dedispersion and periodicity search codes are nearly 50 times faster than the previously used pipeline (Morello et al. 2019). The Fourier Domain Acceleration Search (FDAS) of the Square Kilometre Array (SKA) has achieved better than realtime performance (Dimoudi & Armour 2017). For single pulse search, GPU-based Heimdall[2] (Barsdell et al. 2010) and ROACH2-based (ROACH2[3]) hardware, or even a CPU-based code (Lee et al. in prep) can process data in realtime. As an example at the Green Bank Telescope (GBT), GPU-accelerated codes are now available to optimize this dedispersion task, and to search for transient pulsed radio emission (Walsh & Lynch 2018).

[1] `https://github.com/ewanbarr/peasoup`

[2] `https://sourceforge.net/projects/heimdall-astro/`

[3] `https://casper.ssl.berkeley.edu/wiki/ROACH2`

As a special situation, for a drift scan survey (e.g., CRAFTS[4]) with the Five-hundred-meter Aperture Spherical radio Telescope (FAST, Nan et al. 2011; Jiang et al. 2019), we are going to process many small files (e.g., less than 100 MB) with the goal of searching in realtime. This is a challenge for both computing and disk input/output. For example, at the frequency of ∼300 MHz, the beam size of FAST (∼13 arcmin) corresponds to a transit time of ∼52 s. With 50% beam overlap, we have to process one file within 26 s on average in order to process the data in realtime. If the number of channels is 256, the sampling time is 0.2 ms, and with 8-bit sampling, the size of a data file is ∼60 MB, as typical pulsar search requires frequent reading and writing.

GPUs are suitable for computing rather than reading or writing. Thus, a GPU-based code for processing small files may hardly reach high performance. Secondly, we do not have GPUs for processing FAST drift scan data during the first two years of FAST commissioning time. In order to search for new pulsars, we need a CPU-based pulsar search pipeline. In addition, with many pulsar search examples in the package, PulsaR Exploration and Search TOolkit (PRESTO, Ransom 2001; Ransom et al. 2002, 2003) is an open source code that has already been applied to discover many pulsars. It has been updated and widely employed till now (Swiggum & Gentile 2018). So, we started our work with PRESTO.

In this paper, we present our PRESTO-based pipeline for FAST drift scan pulsar search data. The details of the pipeline are presented in Section 2. The testing results on both the Parkes Multibeam Pulsar Survey (PMPS, Manchester et al. 2001) data and real FAST data are given in Section 3. Sections 4 and 5 feature data processing results and discussion, respectively. Section 6 is the conclusion.

2 THE PIPELINE

The pulsar search relies on CPU, memory and disk Input/Output operations Per Second (IOPS) to some degree. For large data files (e.g., with size ≥1 GB), the processing time needed for dedispersion, FFT or the search phase depends largely on the processing power of the CPU. If acceleration search or phase modulation search is implemented, a large size of memory may be required. For small files, normal pulsar search pipelines based on PRESTO or SigProc[5] and `https://github.com/SixByNine/sigproc` (Lorimer 2011) require writing and reading files frequently. As an improvement, some GPU-based codes, such as peasoup, provide a one-step pulsar search with only one command (which can be very long) and with the candidate list being the only output file. For the FAST drift scan pulsar survey, we utilize a cluster with 20 nodes (480 cores in total, Intel E5 2680 v3 2.5 GHz CPU). For data processing, we prepared a CPU-based pulsar search pipeline with PRESTO routines.

A typical PRESTO-based pulsar search pipeline (for example, PRESTO provides several examples: GBNCC_search.py, full_analysis.py and PALFA_presto_search.py) processes the data with the following steps:

(1) finding and masking RFIs (using **rfifind**);

(2) dedispersing the data into time series with many dispersion measure (DM) values (using **prepdata** or **prepsubband**);

(3) transforming the time series data to frequency domain data (using **realfft**). This step is not necessary, but **accelsearch** can also perform this transformation;

(4) searching for periodic signals from the frequency domain data (using **accelsearch**);

(5) sifting the search result and obtaining candidates (using **ACCEL_sift.py**);

(6) folding the data with the periods and DMs of the candidates (using **prepfold**);

(7) searching for single pulses from the time series **(using single_pulse_search.py *.dat)**.

For steps 2, 3, 4 and 6, the corresponding routines (**prepdata** or **prepsubband**, **realfft**, **accelsearch** and **prepfold**) or the same (**realfft**) will be run multiple times with different options. Those routines can run in parallel to save time. Along with this idea, we utilize Linux shell scripts, and C language and Python codes to reorganize the PRESTO-based pulsar search pipeline, so that the pipeline starts multiple processes to dedisperse the data, generate FFT, search and fold at the same time, thus reducing the total time cost. For dedispersing, we apply **prepdata**, because this is the routine utilized for dedispersion. In fact, it is not always a good choice. We will discuss this in Section 5. The name of the pipeline is RPPPS[6], which is the abbreviation for Re-analysing Pipeline for Parkes Pulsar Survey, since the pipeline was originally designed for reprocessing PMPS data.

There are two key values for RPPPS. One key parameter for RPPPS is the maximum number of processes that run simultaneously (called parallel number hereafter). The other one is applied to control how the code runs in parallel (called parallel version hereafter). The pipeline may run routines round by round (hereafter V2), or one by one according to the number of running processes (hereafter V3)[7]. For example, if the parallel number is 128, with op-

[4] `http://crafts.bao.ac.cn`

[5] `http://sigproc.sourceforge.net`

[6] `https://github.com/qianlivan/RPPPS`

[7] V1 is for the non-parallel testing pipeline and will never be used again

Table 1 Details on Test Files

	PMPS File	FAST File
File Name	PM0064_00311.sf	FAST_test.fits
File Size	∼100 MB	∼51 MB
Number of Channels	96	200
Bandwidth	288 MHz	50 MHz
Band	1231.5–1516.5 MHz	290–340 MHz
Width of Channels	3 MHz	0.25 MHz
Observing time	2100.224 s	52.4288 s

tion V2, the code will start 128 processes (for dedispersing, calculating FFT and acceleration search in serial) simultaneously, count until all the processes are finished and then start another 128 processes. With option V3, the code will start one process and check how many processes are running in total. If the number of processes is less than 128, it will start one more. If the number of processes is equal to or larger than 128, it will wait for several seconds and check the number of processes again.

With such parallel strategies, the PRESTO-based pulsar search pipeline can be accelerated a lot.

3 TEST AND RESULTS

3.1 Test Hardware Platform and Data Files

To test our pipeline, we performed tests on two computers with different configurations. The configurations of these computers are

1) Two Intel Gold 6130 × 2, 2.1 GHz, 32 cores in total, 192 GB DDR4 RAM

2) Two Intel E5-2680 v3 × 2, 2.5 GHz, 24 cores in total, 512 GB DDR3 RAM, which is the configuration of the computing nodes for the FAST drift scan pulsar search.

As for the data files used for the test, we randomly selected a PMPS file from the public Parkes database, PM0064_00311.sf, as the test data file. In order to avoid the test results being affected by different pulsar searching results from different files, all tests use the same file. Another test file is a real observational file from FAST (also randomly selected). The details of these two files are presented in Table 1.

Another setting related to the amount of calculation is the DM value for dedispersion. For the PMPS file, we choose approximately 0 to 3500 pc cm^{-3} as the dispersion range. For FAST data, the DM range is approximately 0 to 800 pc cm^{-3}. Then the **DDplan.py** program in PRESTO was utilized to generate a dedispersion scheme. The dedispersion plans for PMPS data and FAST data are provided in Tables 2 and 3.

3.2 Differences in Storage, Hyper-Threading and Implementing the Turbo Boost Technique

Currently used storage environments include mechanical hard disk drives (HDDs) and solid-state disks (SSDs), as well as network file systems. In addition, the computer's memory (RAM) can be divided into storage space for temporary usage, e.g., the /dev/shm partition under Linux is a virtual storage from memory, which has a much faster read/write speed than HDDS or SSDs. In order to test if different storage environments will affect the speed of data processing, we selected the HDDs and the /dev/shm partition of Linux for testing. We processed the same PMPS data file in the memory virtual partition and the HDD. In the test, we applied the default settings of the server, which are enabling Turbo Boost Technology (TBT) and disabling hyper-threading (HT). In order to cause more IOP pressure, we use V2 as the parallel version, and set zmax value for **accelsearch** and parallel number to be 0 and 128/1024, respectively. We repeated this process 10 times for each storage. The platform we utilized is the first one which is mentioned in Section 2.1.

The result is that whatever data processing method was running on HDD or the /dev/shm folder, the time cost is between 250 s and 270 s when the number of parallel processes was set to be 128 or 1024, though in fact there are only 32 real CPU cores in total. So, an HDD can support running such a parallel pulsar search pipeline. The reason for the HDD and SDD having similar performance is probably because of Linux filesystem caching on these machines, which have lots of RAM.

The HT feature can double the number of cores in the operating system. In common situations, e.g., a desktop for daily usage, the HT is turned on, but it is suggested to be turned off for high performance computing. Since we are applying a simple parallel approach, it is necessary to see if enabling HT will affect the data processing time or not. TBT is a feature that allows the CPU to automatically raise its clock frequency. According to this principle, it seems that enabling this function should be more conducive to shortening the data processing time.

Table 4 displays the average time cost of whether the TBT and/or HT will be used or not. We utilized the V2 parallel version and set the parallel number to be 128 for this test. In order to prevent errors, we ran 10 repeated trials and obtained one averaged value. The numbers in bold are the shortest time cost. It is obvious that for a non-acceleration search, TBT and HT should be enabled, while for acceleration search, TBT should be enabled and HT should be disabled.

According to the tests above, we will enable TBT and HT, and use the shared network file system for further tests

Table 2 Details on Dedispersion Scheme for PMPS Data

No.	DM Start ($\mathrm{pc\,cm^{-3}}$)	DM End ($\mathrm{pc\,cm^{-3}}$)	DM Step ($\mathrm{pc\,cm^{-3}}$)	Down Sample Factor	Number of DMs
1	0	114.0	0.5	1	228
2	114.0	266.0	1.0	2	152
3	266.0	476.0	2.0	4	105
4	476.0	856.0	5.0	8	76
5	856.0	2376.0	10.0	16	152
6	2376.0	3896.0	20.0	32	76

Table 3 Details of Dedispersion Scheme for FAST Ultra-wideband Drift Scan Data

No.	DM Start ($\mathrm{pc\,cm^{-3}}$)	DM End ($\mathrm{pc\,cm^{-3}}$)	DM Step ($\mathrm{pc\,cm^{-3}}$)	Down Sample Factor	Number of DMs
1	0	31.5	0.1	8	315
2	31.5	62.9	0.2	16	157
3	62.9	139.4	0.5	32	153
4	139.4	278.4	1.0	64	139
5	278.4	556.4	2.0	128	139
6	556.4	784.4	3.0	128	76

Table 4 Influence of HT and TBT on Time Cost

TBT.	HT	ZMAX	Average Time Cost (s)
Enable	Enable	0	**226.5**
		50	431.1
	Disable	0	267.2
		50	**424.8**
Disable	Enable	0	297.3
		50	465.9
	Disable	0	304.0
		50	491.7

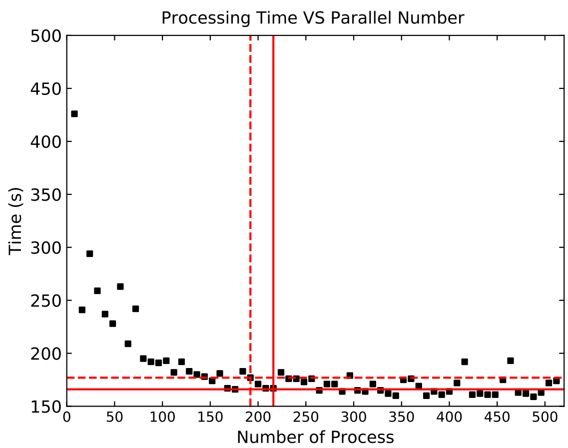

Fig. 1 Processing time reduces when the number of parallel processes increases. *Vertical* and *horizontal dashed red lines* represent the parallel number 192 and processing time 177 s, respectively. *Vertical* and *horizontal solid red lines* signify the parallel number 216 and processing time 166 s, respectively.

on the computing node which would be utilized for FAST drift scan pulsar search.

3.3 The Relationship between the Number of Processes and Processing Time

The time cost of our pipeline may be affected by the following rules:

(1) When the number of processes running in parallel is less than the number of CPU cores, the CPU usage is not 100%. The time cost should be longer than expected.

(2) When the number of processes running in parallel is equal to or several times larger than the number of CPU cores, the total computation time is close to the shortest.

(3) When the number of processes running in parallel is much larger than the number of CPU cores, the total time cost changes little due to the computer not being able to provide more computing resources. Moreover, this will cause the computer to stop responding and thus the time cost will increase.

In order to detect these situations, we processed the same FAST file with different numbers of parallel processes on the computing nodes (the second one mentioned in Section 2.1), to see if the processing time was shortened when the number of parallel processes increased. The number of processes started at 8, in steps of 8, until 1024.

Figure 1 demonstrates the time cost decreases when the number of processes increases. We finally selected 192 (177 s) and/or 216 (166 s) as the number of processes for the FAST drift scan pulsar search. In a real situation, the total time costs for processing FAST drift scan data with the parallel numbers of 192 or 216 are almost same.

4 RESULTS

We then applied the RPPPS as the pulsar search pipeline to find possible new pulsars from archived PMPS data and FAST drift scan data. This resulted in two new pulsar discoveries from PMPS data and tens of new pulsars discovered from FAST data.

4.1 PMPS Data Reprocessing

We processed ∼16% of the PMPS dataset (8240 files) for pulsar search testing, aiming to find new faint pulsars missed by previous processing. It cost ∼14 days to process all the files with the acceleration search in which the zmax value was 50, which resulted in 410 535 candidates. All those candidates were viewed by human beings and 2238 good candidates were finally selected. Among them, the sixteen best candidates were then re-observed by Parkes in April of 2018 and July of 2019. Two were confirmed to be new pulsars (middle panels of Fig. 2). They are very faint pulsars with medium DM values. The pulsar J1900–04 is relatively bright in both PMPS data and our confirmation observation data. The pulsar J1808–12 is so faint that we cannot search and find it in our confirmation data. If we do not know its period and DM from PMPS data reprocessing, we cannot find it. The details of discovering these two new pulsars will be in another paper (Pan et al. in prep).

The PMPS data reprocessing indicates that the pipeline has a good performance on pulsar search and is potentially good at finding faint pulsars.

4.2 FAST Drift Scan

From August 2017, this pipeline was employed to process FAST drift scan pulsar search data. The ultra wide-band receiver covers a band of 290–1760 MHz. We utilized the 290–340 MHz band (200 channels) for pulsar search with RPPPS. The characteristics of every file are the same as the one we used to test the pipeline (see details in Table 1). With the cluster mentioned before, processing 12-hour drift scan data will cost approximately the same time, which means that the data can be processed in real-time.

With RPPPS, the first FAST pulsar candidate was discovered on 2017 August 6, from FAST's August 4 drift scan data. Figure 2 (upper) features the discovery plot. It is a relatively strong pulsar and there is no doubt that it should be found. This candidate was then confirmed by FAST on August 6. This is also the first real pulsar discovery by RPPPS.

As a cross check between Parkes data and FAST data, the first confirmed FAST pulsar (J1900–0134, Qian et al. 2019) can also be re-discovered from PMPS data by RPPPS. Figure 3 (upper) displays the re-discovery of this pulsar by RPPPS from the PMPS data. The plot from corresponding FAST data can be found in Qian et al. (2019). This demonstrates that the RPPPS can be suitable to process data from different telescopes, and locate new bright or faint pulsars.

The first millisecond pulsar (MSP) candidate was also detected by RPPPS (bottom panel of Fig. 2). Approximately one third of the band was removed to reduce the RFIs, yet this MSP is still very bright and manifests scintillation features. The RPPPS also identified faint candidates from the drift scan data, such as the one in the bottom panel of Figure 3.

Till now, more than 40 candidates have been ascertained with RPPPS, including high DM pulsars, MSPs and binary pulsars. Most of them are already confirmed to be new pulsars.

5 DISCUSSIONS

5.1 Realtime Data Processing:

According to the test result, one computing node can process approximately 26 s of observation data in ∼170 s. Within 20 nodes, processing one-night (10–12 hours) of drift scan data should be finished within ∼4 hours (33% of the observation time). In fact, the data processing time reaches ∼80% of the observing time. This is mainly due to the bottleneck from the shared network file system IOPS. The computing nodes have very limited disk space, thus we cannot run the whole search on local disks. Otherwise, we may firstly copy the data file to the local disk of the computing node and move the search result back to the shared file system after data processing is finished, and thus accelerate the pulsar search even more.

In our previous test, the time costs when utilizing different local storages are almost same. When employing clusters and when many computing nodes were reading and writing at the same time, the shared network file system should be high performance.

5.2 Using HT

After running for several months, we upgraded the cluster and enabled HT for all the computing nodes, because after optimizing by Sugon, enabling HT will shorten the time cost more than disabling it. So, enabling the HT or not seems to depend on the software settings, while the TBT can no doubt help shorten the total time cost.

5.3 Possibility of Missing New Pulsars

For PMPS data, we only processed 16% of the data. Assuming that the discovery rate is same, we may find 12 new pulsars in total. Compared with processing the PMPS data in recent years (e.g., Knispel et al. 2013; Eatough et al. 2013), one key point for our discovery can be that all the

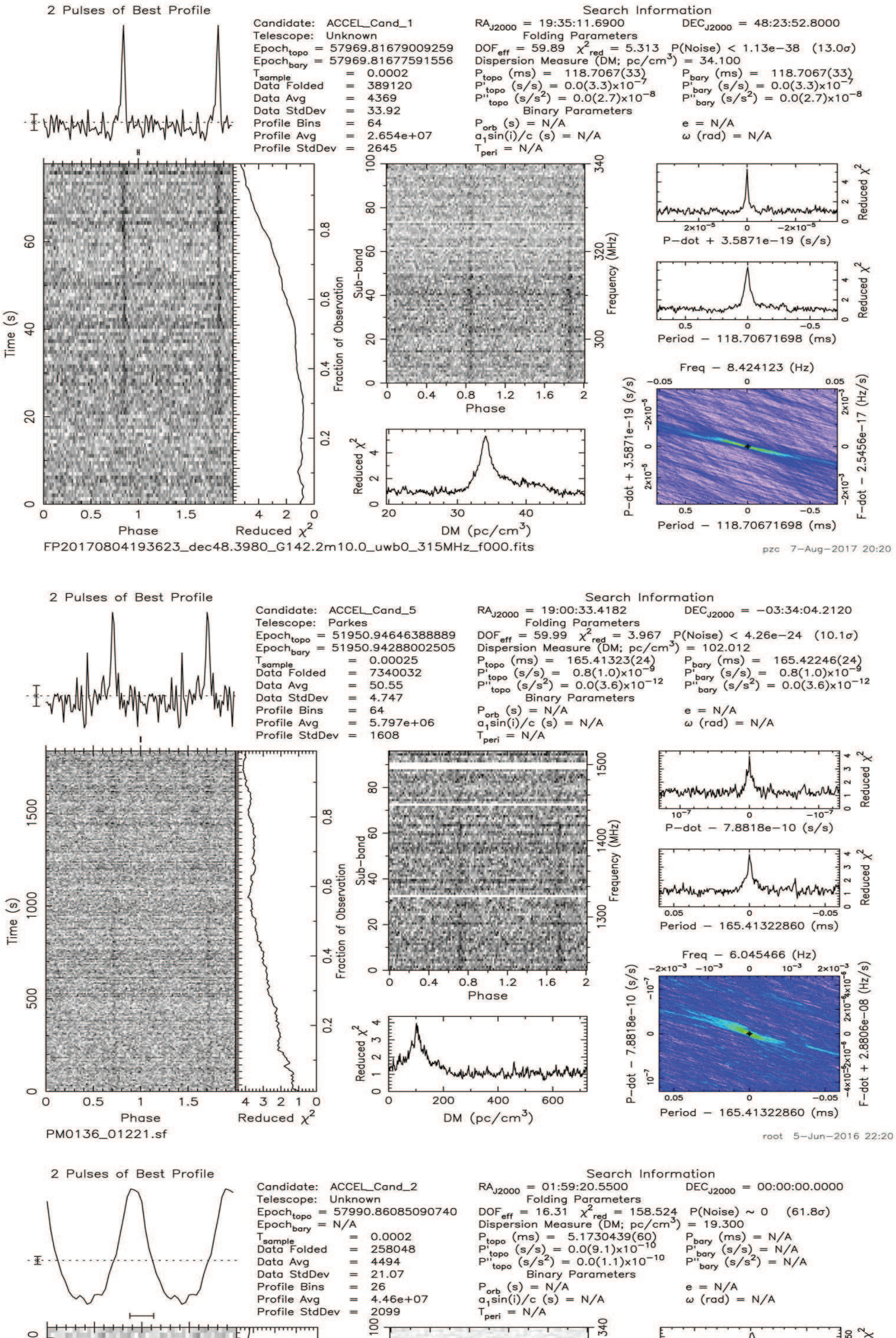

Fig. 2 *Upper*: the discovery plot of the first FAST pulsar candidate (now confirmed), designated as C1; *Middle*: The one is J1900–04, the confirmed new pulsars from reprocessing of PMPS data, discovery details will be in Pan et al. (in prep); *Bottom*: the first FAST MSP candidate (now confirmed), discovered by RPPPS, designated as C8.

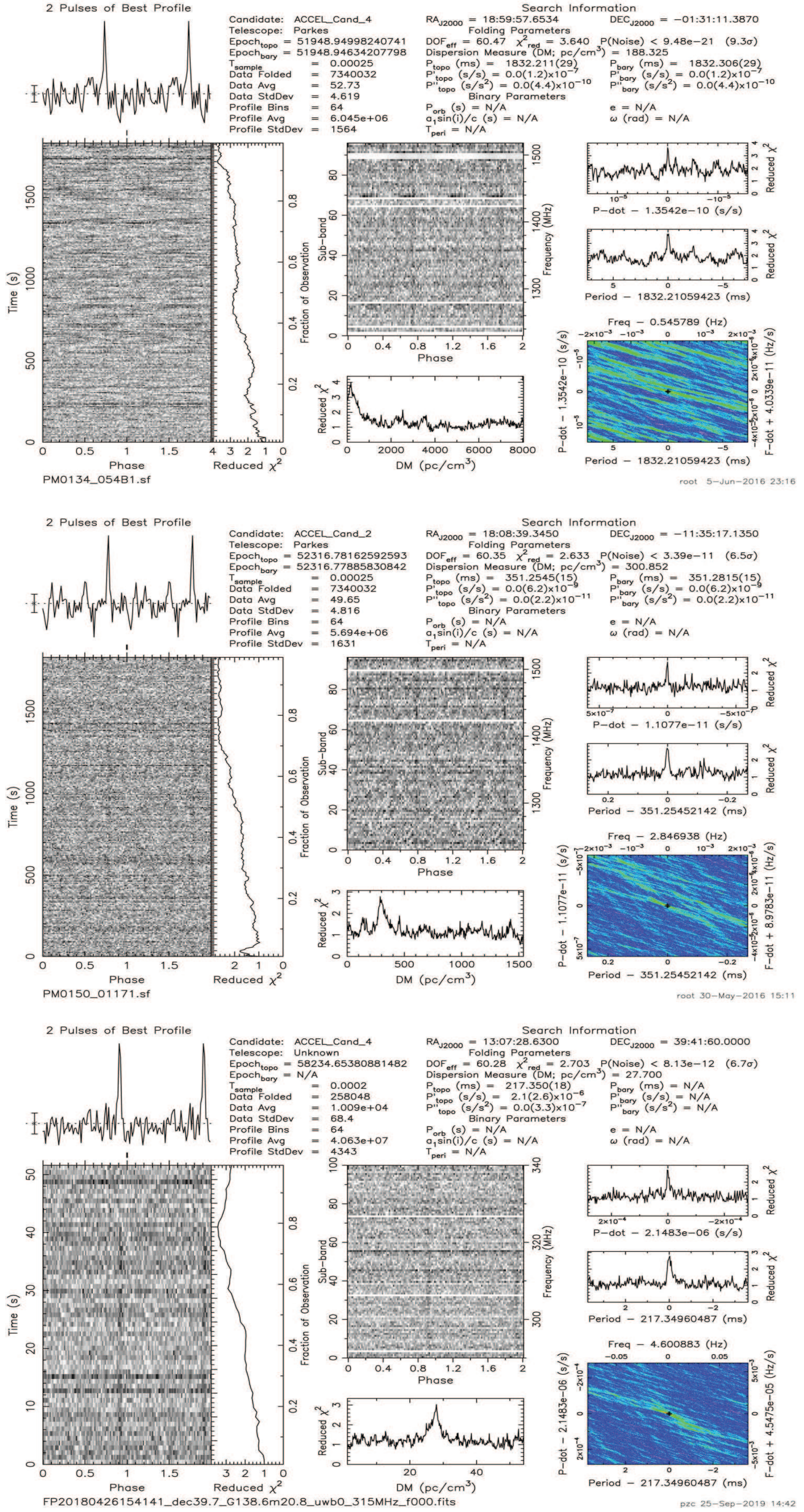

Fig. 3 *Upper*: the PMPS re-detection of the first FAST confirmed pulsar (J1900-0134), see Qian et al. (2019) for the plot from corresponding FAST data; *Middle*: the right one is J1808–12; *Bottom*: a very faint FAST candidate discovered by RPPPS, designated as C59.

candidates were checked by human beings. We will describe these discoveries and publish the human-checked results in Pan et al. (in prep) soon.

For the FAST data, the discovered new pulsars have low DM values (less than 100), since the data are from the 290–340 MHz band. We have no doubt that we may miss high DM pulsars. The data are only for 52 s of observation and thus we indeed missed long period pulsars (with periods approximately longer than 2 s). Those pulsars were detected by a single pulse search pipeline (Zhu et al. in prep).

We have carefully tuned the search parameters before we start searching for new pulsars in PMPS data or FAST data, but some new pulsars will still be missed.

5.4 Using Other Routines

The PRESTO routine **prepdata** is simply applied for dedispersing data with one DM value. We utilize it in the pipeline. Another routine, **prepsubband**, can be employed for dedispersing data with a set of DM values. We also tested it and compared the time cost. When we execute it for dedispersing a data file with a relatively small size (e.g., less than 10 GB) and then run pulsar search in parallel, it is faster or even much faster than running **prepdata** in parallel. This is due to saving time from reading data files. The situation is similar to when the MPI routines **prepsubband** and **mpiprepsubband** are utilized.

6 CONCLUSIONS

In order to search for pulsars from FAST drift scan data (many small files, only CPU available), we present the RPPPS package. Our conclusions are as follows:

(1) With the RPPPS package, we successfully processed FAST ultra-wideband pulsar search data in realtime with 20 computing nodes (480 cores in total). The RPPPS package is suitable for processing small files when only CPUs are available for computing, and can achieve high efficiency.

(2) Till now, two new pulsars from PMPS data and tens of new pulsars from FAST drift scan data have already been discovered with RPPPS. The two new pulsars from PMPS data are very faint, indicating that the RPPPS can be suitable for finding faint pulsars, including faint new pulsars missed by previous PMPS data processing. The new pulsars from FAST data have highly varied parameters (or properties).

Acknowledgements We thank Scott Ransom and an anonymous reviewer for useful comments and suggestions. This work is supported by the National Key R&D Program of China (2017YFA0402600), State Key Development Program for Basic Research (2015CB857100), National Basic Research Program of China (973 program, No. 2015CB857101) and National Natural Science Foundation of China (Grant Nos. 11703047, 11773041, U1631132 and U1831131). ZP is supported by the CAS "Light of West China" Program, and by the CAS Key Laboratory of FAST, National Astronomical Observatories, Chinese Academy of Sciences. LQ is supported in part by the Youth Innovation Promotion Association of CAS (id.2018075). This work made use of data from the Five-hundred-meter Aperture Spherical radio Telescope (FAST). FAST is a Chinese national mega-science facility, built and operated by the National Astronomical Observatories, Chinese Academy of Sciences.

References

Barsdell, B. R., Barnes, D. G., & Fluke, C. J. 2010, MNRAS, 408, 1936
De, K., & Gupta, Y. 2016, Experimental Astronomy, 41, 67
Dimoudi, S., & Armour, W. 2017, in Astronomical Society of the Pacific Conference Series, 512, Pulsar Acceleration Searches on the GPU for the Square Kilometre Array, eds. N. P. F. Lorente, K. Shortridge, & R. Wayth, 599
Eatough, R. P., Kramer, M., Lyne, A. G., & Keith, M. J. 2013, MNRAS, 431, 292
Jiang, P., Yue, Y., Gan, H., et al. 2019, Science China Physics, Mechanics, and Astronomy, 62, 959502
Knispel, B., Eatough, R. P., Kim, H., et al. 2013, ApJ, 774, 93
Lorimer, D. R. 2011, SIGPROC: Pulsar Signal Processing Programs, Astrophysics Source Code Library. Record Ascl: 1107.016
Manchester, R. N., Lyne, A. G., Camilo, F., et al. 2001, MNRAS, 328, 17
Morello, V., Barr, E. D., Cooper, S., et al. 2019, MNRAS, 483, 3673
Nan, R., Li, D., Jin, C., et al. 2011, International Journal of Modern Physics D, 20, 989
Qian, L., Pan, Z., Li, D., et al. 2019, Science China Physics, Mechanics, and Astronomy, 62, 959508
Ransom, S. M. 2001, New Search Techniques for Binary Pulsars, PhD Thesis, Harvard University
Ransom, S. M., Cordes, J. M., & Eikenberry, S. S. 2003, ApJ, 589, 911
Ransom, S. M., Eikenberry, S. S., & Middleditch, J. 2002, AJ, 124, 1788
Sclocco, A., van Leeuwen, J., Bal, H. E., & van Nieuwpoort, R. V. 2016, Astronomy and Computing, 14, 1
Serylak, M., Karastergiou, A., Williams, C., et al. 2013, in IAU Symposium, 291, Neutron Stars and Pulsars: Challenges and Opportunities after 80 Years, ed. J. van Leeuwen, 492
Swiggum, J. K., & Gentile, P. A. 2018, AJ, 156, 190
Walsh, G., & Lynch, R. 2018, in American Astronomical Society Meeting Abstracts, 231, 243.04