PSRDP: A Parallel Processing Method for Pulsar Baseband Data Postprint

Authors: Ya-Zhou Zhang, Hai-Long Zhang, Jie Wang, Xin-Chen Ye, Shuang-Qiang Wang, Xu Du, Han Wu, Ting Zhang, Shao-Cong Guo and Meng Zhang

Date: 2024-02-01T14:50:09+00:00

Abstract

To address the challenge of real-time processing of ultra-wide bandwidth pulsar baseband data, we designed and implemented a pulsar baseband data processing algorithm (PSRDP) based on GPU parallel computing technology. PSRDP is capable of performing operations including baseband data unpacking, channel separation, coherent dedispersion, Stokes detection, phase and folding period prediction, and folding integration within GPU clusters. We validated the algorithm using J0437-4715 pulsar baseband data generated by the CASPSR and Medusa backends of Parkes, as well as J0332+5434 pulsar baseband data produced by the self-developed backend of the Nanshan Radio Telescope, from which we obtained corresponding pulse profiles. Experimental analysis reveals that the pulse profiles generated by the PSRDP algorithm presented in this work are essentially consistent with the processing results from the Digital Signal Processing Software for Pulsar Astronomy (DSPSR), thereby confirming the effectiveness of the PSRDP algorithm. Furthermore, comparative analysis using identical baseband data demonstrates that PSRDP achieves processing speeds at least comparable to DSPSR. The theoretical insights and technical expertise acquired through this research on the PSRDP algorithm establish a foundational framework for the real-time processing of ultra-wide bandwidth pulsar baseband data from the QTT (Qitai Radio Telescope).

Full Text

Preamble

Research in Astronomy and Astrophysics, 24:015025 (11pp), 2024 January

https://doi.org/10.1088/1674-4527/ad0e99

PSRDP: A Parallel Processing Method for Pulsar Baseband Data

Ya-Zhou Zhang[1,2], Hai-Long Zhang[1,2,3,4], Jie Wang[1,4], Xin-Chen Ye[1,2,4], Shuang-Qiang Wang[1], Xu Du[1,2], Ting Zhang[1,2], Shao-Cong Guo[5], and Meng Zhang[6]

[1] Xinjiang Astronomical Observatory, Chinese Academy of Sciences, Urumqi 830011, China; zhanghailong@xao.ac.cn
[2] School of Cyber Science and Engineering, Qufu Normal University, Qufu 273165, China

Received 2023 October 20; revised 2023 November 13; accepted 2023 November 20; published 2024 January 16

Abstract

To address the challenge of real-time processing for ultra-wide bandwidth pulsar baseband data, we have designed and implemented a GPU-based parallel computing algorithm called PSRDP (PulSaR baseband Data Processing). PSRDP performs baseband data unpacking, channel separation, coherent dedispersion, Stokes detection, phase and folding period prediction, and folding integration entirely within GPU clusters. We validated the algorithm using J0437-4715 pulsar baseband data from the CASPSR and Medusa backends at Parkes Observatory, as well as J0332+5434 data from the self-developed backend of the NanShan Radio Telescope (NSRT). The resulting pulse profiles demonstrate that PSRDP produces results essentially identical to those from the Digital Signal Processing Software for Pulsar Astronomy (DSPSR), confirming its effectiveness. Furthermore, speed comparisons using identical baseband data show that PSRDP' s processing speed is comparable to or better than DSPSR. The theoretical insights and technical experience gained from this PSRDP research establish a foundation for real-time processing of ultra-wide bandwidth pulsar baseband data from the QTT (Qi Tai radio Telescope).

Key words: (stars:) pulsars: general –methods: data analysis –techniques: miscellaneous

1. Introduction

Advances in manufacturing and information technology have driven radio telescope receiving systems toward ultra-wide bandwidth and phased array feed (PAF) architectures. Multi-beam and ultra-wide bandwidth systems have dramatically increased observational data volumes. For instance, the low-frequency array processes approximately 14 Tb of raw astronomical signals per second at a 200 MHz sampling rate (van Haarlem et al. 2013). The FAST multi-beam backend achieves a data sampling rate of 12.8 GB s^{-1} with 100 s time resolution, 4000 channels, and dual polarization (Li et al. 2018). The QTT currently under construction in Xinjiang, China, will generate annual archive volumes exceeding 10 PB with its ultra-wide bandwidth and PAF receiving system (Wang

et al. 2023). Real-time processing of such ultra-wide bandwidth pulsar data represents an urgent challenge and motivates this study.

As astronomical research progresses and digital technology advances, performance requirements for digital backend systems continue to escalate. Modern backends must achieve high-speed sampling, real-time analysis, and data preprocessing across wider bandwidths with higher frequency resolution. Handling high-speed data streams from PAF and multi-beam systems while performing real-time processing on heterogeneous platforms has become a critical operational challenge for radio observatories. Due to storage limitations, massive astronomical signals must be processed and analyzed in real-time, placing stringent demands on processing hardware (Wei 2019).

Contemporary digital backend systems typically employ hybrid FPGA+CPU+GPU architectures. A key technical challenge in such heterogeneous systems is achieving high-speed data flow between CPU and GPU while minimizing errors and packet loss during massive data transfers. A practical solution involves creating ring buffers in memory for real-time high-speed caching, then transferring buffer data to GPU memory for processing.

GPU acceleration has proven highly effective for scientific analysis and simulations. Internationally, CPU+GPU heterogeneous platforms have become mainstream for real-time pulsar signal processing. Leveraging GPU computational units reduces CPU workload and significantly enhances system data flow processing efficiency. As digital backend technology advances, real-time signal processing poses serious challenges to conventional computing, while GPU clusters offer a viable solution for managing the massive data streams generated by radio astronomy observations.

The Parkes ultra-wide bandwidth receiver system employs an FPGA+GPU architecture for data preprocessing and recording. It divides the 704–4032 MHz signal into three analog RF bands for sampling. During FPGA preprocessing, the data is further divided into 26 continuous subbands of 128 MHz each using polyphase filter bank (PFB) technology (Zhang et al. 2023b). The FPGA packages data into VDIF format and transfers it to GPU nodes via UDP protocol. To handle high-speed network flows, each GPU node uses PSRDADA to establish high-speed ring buffers for temporary data storage before copying data to GPU memory for further processing (Hobbs et al. 2020).

The Green Bank Telescope (GBT) employs a ROACH2+GPU architecture in its VEGAS (Versatile GBT Astronomical Spectrometer) backend for pulsar signal processing. VEGAS' s pipeline software uses multiple concurrent threads: network threads transmit data from ROACH2 to a shared buffer via 10 GbE links, reading threads copy data to GPU memory, and after processing, disk-writing threads store results (Chennamangalam et al. 2014).

The FAST backend combines ROACH2, GPUs, and high-speed networking for pulsar search and timing observations (Nan & Li 2014; You et al. 2021). The Shanghai Tian Ma Radio Telescope Digital Backend System uses a hy-

brid FPGA+GPU architecture with dual-polarization input. FPGA output is transmitted via 10 GbE to eight high-performance GPU computers for coherent and incoherent dedispersion (Yan et al. 2017). The Yunnan Astronomical Observatory operates a 512 MHz bandwidth pulsar system with 8-bit sampling, using ROACH2 for baseband data acquisition and DSPSR for processing (Xu et al. 2015). This system provides 128 channels (4 MHz each) for decoding, coherent dedispersion, polarization computation, and folding, storing results in PSRFITS format (Hotan et al. 2004). The Haoping Radio Telescope (HRT) also uses a ROACH2+GPU architecture, transmitting ROACH2 output via 10 GbE to eight GPU nodes supporting both incoherent and coherent dedispersion modes (Luo et al. 2020).

2. Parallel Processing Algorithm for Pulsar Baseband Data Based on GPU

To meet the real-time or near-real-time processing requirements for ultra-wide bandwidth pulsar baseband data, we designed and implemented the PSRDP algorithm using GPU parallel computing technology. GPUs offer far more cores than CPUs and can execute numerous threads simultaneously, making them ideal for the massive computational tasks involved in ultra-wide bandwidth pulsar data processing.

The PSRDP algorithm flow is illustrated in Figure 1 [Figure 1: see original paper]. The process begins with reading pulsar baseband data on the HOST, copying it to the DEVICE, then performing channelization, coherent dedispersion, Stokes detection, folding integration, and other operations. Upon completion, results are copied back to the HOST for storage.

2.1. Unpack Baseband Data

After copying baseband data to GPU memory, unpacking is the first required operation. For CASPSR backend data with dual-polarization 8-bit sampling, the two polarizations are stored alternately every four data points (real numbers). As shown in Figure 2 [Figure 2: see original paper], the dual-polarization data are unpacked and recombined, grouping data from the same polarization together to facilitate subsequent processing. To enable GPU parallel unpacking, Equation (1) establishes the relationship between indices before and after unpacking:

$$j = \left\lfloor \frac{i}{\text{npad}} \right\rfloor + \text{polsize} \times (i \bmod \text{npad})$$

where i is the index before unpacking, j is the index after unpacking, $\text{npad} = 4$ (since dual polarizations are stored alternately every four data points), and if the data block length is N, then $\text{polsize} = N/2$.

For Medusa backend data with dual-polarization 32-bit sampling, the dual polarizations are stored alternately every 2048 data points (1024 complex numbers equal 2048 real numbers), as shown in Figure 3 [Figure 3: see original paper]. The unpacking process is similar to CASPSR, with npad = 2048, and requires offset binary decoding. Since each datum is independent, GPU parallel technology can fully accelerate the unpacking process.

2.2. Multi-channel Coherent Dedispersion

During propagation to Earth, pulsar signals undergo dispersion due to the interstellar medium (ISM), causing high-frequency signals to arrive before low-frequency signals. The ISM acts as a filter with a specific transfer function. To restore the original pulsar signal characteristics, dedispersion is necessary. Coherent dedispersion filters the observed signal through the inverse of the ISM transfer function, and in theory can completely eliminate dispersion effects (Zhang et al. 2023a).

As shown in Figure 4 [Figure 4: see original paper], after performing FFT on the time-domain data sequence, the signal is further channelized into multiple subbands. Based on each subband' s central frequency, bandwidth, and dispersion measure (DM), chirp coefficients (the inverse function of the ISM transfer function) are generated according to Equation (2):

$$H(f) = \exp\left(-i \cdot 2\pi \cdot D \cdot \frac{\Delta f^2}{f_{\text{ref}}^2}\right)$$

where the reference frequency f_{ref} is set as the central frequency, Δf represents the difference between a specific frequency point f and f_{ref}, and D is the dispersion constant equal to $4.15 \times 10^3\,\text{MHz}^2\,\text{pc}^{-1}\,\text{cm}^2\,\text{s}$. The generated chirp coefficients for each subband are multiplied point-by-point with the subband data to achieve coherent dedispersion. Since the data have no interdependencies, this process can be efficiently parallelized on GPU.

2.3. Stokes Detection

After coherent dedispersion, dual-polarization data from two orthogonal channels (pol0 and pol1) are processed to calculate the four Stokes parameters: I, Q, U, and V (Sutinjo et al. 2022). For dual-linear feeds, the Stokes parameters are calculated using Equation (3):

$$\begin{aligned} I &= P_{AA} + P_{BB} \\ Q &= P_{AA} - P_{BB} \\ U &= 2\,\text{Re}(P_{AB}) \\ V &= 2\,\text{Im}(P_{AB}) \end{aligned}$$

For dual-circular feeds, Equation (4) is used:

$$\begin{aligned} I &= P_{AA} + P_{BB} \\ Q &= 2\,\mathrm{Re}(P_{AB}) \\ U &= 2\,\mathrm{Im}(P_{AB}) \\ V &= P_{AA} - P_{BB} \end{aligned}$$

where P_{AA} and P_{BB} are the powers of the two orthogonal polarization channels pol0 and pol1, respectively, and P_{AB} is the cross-correlation product of the dual polarizations, with $\mathrm{Re}(P_{AB})$ and $\mathrm{Im}(P_{AB})$ representing its real and imaginary parts. The Stokes parameters are completely independent, and as shown in Figure 5 [Figure 5: see original paper], each datum uses a separate GPU thread for calculation. GPU parallel technology significantly reduces resource overhead compared to traditional single-threaded CPU processing and improves computational efficiency.

2.4. Folding and Integrating

Different pulsars have varying periods, and combined with different sampling rates, the number of sampling points per period often cannot be represented as an integer power of two (which would optimize FFT execution). This causes the final remaining data to be discarded or require special handling during integration. Our algorithm combines folding and integration operations and processes them simultaneously.

In the CUDA programming model, calculations are organized in a three-level hierarchy. Each CUDA kernel invocation creates a new Grid composed of multiple Blocks, with each Block containing up to 1024 separate Threads (Hijma et al. 2023). As shown in Figure 6 [Figure 6: see original paper], the Grid can control the number of Blocks by setting three dimensions: `gridDim.x`, `gridDim.y`, and `gridDim.z`. Similarly, Block can control the number of Threads through `blockDim.x`, `blockDim.y`, and `blockDim.z`.

By default, the final folded profile contains 1024 data points. Therefore, for n channels and four Stokes parameters, a space of size $4 \times 1024 \times n$ must be pre-allocated to store the generated profile data. For multi-channel, multi-polarization data, since the recording start time is identical across channels, the corresponding profile sequence index for each group is also the same, requiring only one index sequence to be generated. The index sequence generation method is shown in Figure 7 [Figure 7: see original paper].

Pulsar periods are extremely stable, but observatory location, pulse dispersion, solar system orbital parameters (Romer delay, Shapiro delay, and Einstein delay), and binary system effects can all influence pulse arrival times, thereby affecting the folding period (Pennucci 2015). To accurately fold pulse profiles

and determine phases at specific timestamps, it is necessary to predict both the folding period and the pulsar phase at that timestamp (Zhang et al. 2023a).

When processing pulsar baseband data, the data must be divided into blocks. To accurately fold the pulse profile, we predict the folding period and the relative phase value (between 0 and 1) of the first sample point in each block based on the block' s start time. After determining the phase value of the first sample point, the phase values of all sample points in the block can be calculated. The phase value of the ith sample point is given by Equation (5):

$$\text{phase}_i = \text{phase}_1 + i \times \text{phase\_per\_sample} - \lfloor \text{phase}_1 + i \times \text{phase\_per\_sample} \rfloor$$

where phase_1 is the phase value of the first sampling point obtained through prediction, the floor function represents rounding down, and phase_per_sample represents the phase interval per sampling point, expressed by Equation (6):

$$\text{phase\_per\_sample} = \frac{1}{\text{nbin}}$$

The same indexes in the index sequence may not be clustered together, as a piece of data may span more than one period. For folding operations, corresponding points from each period are summed and averaged, while integration averages multiple consecutive points. The pulse profile index generation diagram is shown in Figure 7 [Figure 7: see original paper]. We combine folding and integration operations, adding and averaging data corresponding to the same index.

During algorithm implementation, direct addition of GPU multi-threaded calculation results leads to errors. While atomic addition operations are often used in such cases, their locking mechanism for data protection at the 底层 seriously slows down computation. To address this, we designed and implemented a GPU-based multi-threaded folding integration algorithm based on the method described in Section 7.1 of the *Handbook of Pulsar Astronomy* (Lorimer & Kramer 2012). The core idea is that for N-channel, M-polarization data, assuming the final profile data size generated by each polarization is nbin, GPU multi-threading technology launches $N \times M \times$ nbin threads. The Grid dimensions are set as `gridDim.x = nchan` (number of channels) and `gridDim.y = npol` (number of polarizations), while Block dimensions are `blockDim.x = 1024` (number of pulse profile bins), `blockDim.y = 1`, and `blockDim.z = 1`, creating a total of $N \times M \times 1024$ threads. As shown in Figure 8 [Figure 8: see original paper], data with index 1 in a particular polarization and channel uses a single thread to perform serial loop addition, avoiding the resource overhead caused by atomic addition.

3. Experimental Validation

3.1. Parkes Baseband Data Processing

3.1.1. CASPSR Baseband Data Processing We tested PSRDP using baseband data from the CASPSR backend at Parkes Observatory. The target was pulsar J0437-4715, observed with 400 MHz bandwidth from 1182–1582 MHz for 8 seconds, producing 12.8 GB of data. PSRDP divided the data into 1024 channels, performing Stokes detection, coherent dedispersion, integration, and folding on each channel. Extracting the Stokes I parameter from each channel yields Figure 9 [Figure 9: see original paper], clearly showing dispersion delay between channels. After incoherent dedispersion, the subband profiles align correctly, as shown in Figure 10 [Figure 10: see original paper]. The phases and amplitudes of the two profiles are essentially identical, verifying the effectiveness of PSRDP' s period and phase prediction methods.

We processed the same baseband data with DSPSR, also outputting 1024 channels. Comparing the final profile with that from PSRDP, Figure 11 [Figure 11: see original paper] shows the two profiles have consistent phases and amplitudes, further confirming PSRDP' s effectiveness.

3.1.2. Medusa Baseband Data Processing We also tested PSRDP using Medusa backend data from Parkes, again targeting J0437-4715. This dataset had 128 MHz bandwidth from 704–832 MHz, observed for approximately 9.6 seconds, producing 9.6 GB of data. The baseband data was divided into 128 MHz subbands and processed similarly to Section 3.1.1. After incoherent dedispersion, Figure 12 [Figure 12: see original paper] reveals clear radio frequency interference (RFI) in some channels. After removing the affected channels, Figure 13 [Figure 13: see original paper] shows a clear pulse profile.

Processing the same data with DSPSR and outputting 128 channels, then comparing the final profile (after interference removal) with PSRDP's output, Figure 14 [Figure 14: see original paper] shows the two profiles have essentially identical phase and amplitude.

3.2. NSRT Baseband Data Processing

Using our self-developed pulsar backend (employing three custom FPGA development boards with PFB technology to divide the 964–1732 MHz signal into six 128 MHz subbands, each transmitted to a GPU server for processing), we generated baseband data from NSRT' s L-band 1 GHz receiver. The target was pulsar J0332+5434, producing single-channel 128 MHz bandwidth, dual-polarization, 16-bit data from 1220–1348 MHz during a 5-minute observation. Using the same processing method as Sections 3.1.1/3.1.2, we obtained Figure 15 [Figure 15: see original paper]. After removing some interfered channels, the resulting dynamic spectrum is shown in Figure 16 [Figure 16: see original paper].

3.3. PSRDP and DSPSR Speed Comparison

Regarding algorithm execution time, we conducted speed tests on both PSRDP and DSPSR using identical CASPSR baseband data. We tested channel numbers of 2, 4, 8, 16, 32, 64, 128, 256, 512, and 1024, calculating average processing times over multiple runs. The results, shown in Figure 17 [Figure 17: see original paper], demonstrate that PSRDP' s execution time is not slower than DSPSR.

4. Conclusion

In response to QTT's ultra-wide bandwidth pulsar data processing requirements, we designed and implemented PSRDP based on GPU parallel technology. After unpacking on the HOST, PSRDP reads baseband data and transfers it to the DEVICE. The wideband signal is further subdivided using FFT, with each subband undergoing coherent dedispersion, Stokes detection, periodic phase prediction, and folding integration before being transferred back to the HOST for scientific storage.

We systematically tested PSRDP using J0437-4715 data from Parkes' CASPSR and Medusa backends and J0332+5434 data from NSRT' s self-developed backend. PSRDP outputs multiple subbands, aligns profiles through incoherent dedispersion, and generates final profiles that show no obvious differences from DSPSR in phase or amplitude, with slightly better processing speed. These comparisons verify PSRDP' s effectiveness and establish a technical foundation for real-time processing of QTT' s ultra-wide bandwidth pulsar baseband data.

Acknowledgments

This work is supported by the National Key R&D Program of China (Nos. 2021YFC2203502 and 2022YFF0711502); the National Natural Science Foundation of China (NSFC) (12173077 and 12003062); the Tianshan Innovation Team Plan of Xinjiang Uygur Autonomous Region (2022D14020); the Tianshan Talent Project of Xinjiang Uygur Autonomous Region (2022TSYCCX0095); the Scientific Instrument Developing Project of the Chinese Academy of Sciences (grant No. PTYQ2022YZZD01); China National Astronomical Data Center (NADC); the Operation, Maintenance and Upgrading Fund for Astronomical Telescopes and Facility Instruments, budgeted from the Ministry of Finance of China (MOF) and administrated by the Chinese Academy of Sciences (CAS); and the Natural Science Foundation of Xinjiang Uygur Autonomous Region (2022D01A360).

References

Chennamangalam, J., Scott, S., Jones, G., et al. 2014, PASA, 31, e048
Hijma, P., Heldens, S., Sclocco, A., van Werkhoven, B., & Bal, H. E. 2023, ACM Comput. Surv., 55, 239
Hobbs, G., Manchester, R. N., Dunning, A., et al. 2020, PASA, 37, e012
Hotan, A. W., van Straten, W., & Manchester, R. N. 2004, PASA, 21, 302

Li, D., Wang, P., Qian, L., et al. 2018, IMMag, 19, 112
Lorimer, D. R., & Kramer, M. 2012, *Handbook of Pulsar Astronomy* (Cambridge: Cambridge Univ. Press)
Luo, J.-T., Gao, Y.-P., Yang, T.-G., et al. 2020, RAA, 20, 111
Nan, R., Li, D., Jin, C., et al. 2011, IJMPD, 20, 989
Nan, R., & Li, H. 2014, SSPMA, 44, 1063
Pennucci, T. T. 2015, PhD thesis, Univ. Virginia
Sutinjo, A. T., Ung, D. C. X., & Sokolowski, M. 2022, A&A, 664, A102
van Cappellen, W. A., Oosterloo, T. A., Verheijen, M. A. W., et al. 2022, A&A, 658, A146
van Haarlem, M. P., Wise, M. W., Gunst, A. W., et al. 2013, A&A, 556, A2
Wang, N., Xu, Q., Ma, J., et al. 2023, SCPMA, 66, 289512
Wei, D. 2019, PhD thesis, Univ. Chinese Academy of Sciences
Xu, Y., Li, J., Zhang, Y., et al. 2015, AR&T, 12, 480
Yan, Z., Shen, Z.-Q., Wu, Y.-J., Zhao, R.-B., & Liu, Q.-H. 2017, in 2017 XXXI-Ind General Assembly and Scientific Symp. of the Int. Union of Radio Science (URSI GASS), 1 (Piscataway, NJ: IEEE)
You, S.-P., Wang, P., Yu, X.-H., et al. 2021, RAA, 21, 314
Zhang, H.-L., Zhang, Y.-Z., Zhang, M., et al. 2023a, RAA, 23, 015023
Zhang, M., Zhang, H.-L., Zhang, Y.-Z., et al. 2023b, RAA, 23, 085012
Zhang, X., & Duan, R. 2022, Proc. SPIE, 12190, 1219032

ORCID iDs

https://orcid.org/0000-0001-6046-2950 Ya-zhou Zhang
https://orcid.org/0000-0003-0380-6395 Hai-Long Zhang
https://orcid.org/0000-0001-6448-0822 Jie Wang
https://orcid.org/0000-0002-1234-5678 Xu Du

Note: Figure translations are in progress. See original paper for figures.