Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Benchmarking short-read genotyping tools across rice pangenome graph representations

Long Ma
Taiyuan University of Technology
Weizheng Yue
Taiyuan University of Technology
Xubin Chen
Taiyuan University of Technology
Li Zhou
Taiyuan University of Technology
Weihua Pan
Agricultural Genomics Institute at Shenzhen
Dongxi Li
dxli0426@126.com

Taiyuan University of Technology

Research Article

Keywords: Rice pangenomics, pangenome graph representations, graph-based genotyping, short-read variant analysis, benchmarking

Posted Date: November 23rd, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11632025/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Abstract

Background

Compared with a single linear reference genome, pangenome graphs can improve variant detection by representing sequence diversity within a population. Yet the performance of variant-analysis tools has not been compared systematically across graph representations. Graph-based variant analysis includes graph-based genotyping and graph-enhanced variant discovery. Here, we focus on graph-based genotyping and evaluate four tools on rice pangenome graphs.

Results

We constructed three pangenome graphs, MC, PGGB, and minigraph, from 36 rice genome assemblies. We evaluated VG, GraphTyper2, PanGenie, and BayesTyper for detecting single-nucleotide polymorphisms, small insertions and deletions, and structural variants. The benchmark included graph-included and graph-excluded samples at four Illumina short-read depths: 5×, 10×, 15×, and 20×. Minigraph was used mainly for structural-variant analysis. VG could not be evaluated on PGGB because graph-index construction failed. Performance varied across graph representations and variant classes. For single-nucleotide polymorphism and small insertion-and-deletion detection, VG with MC performed best at 20× and 15×. Across the six samples, the mean F1 scores were 0.653 and 0.550 at 20×, and 0.642 and 0.535 at 15×, respectively. At 10× and 5×, GraphTyper2 with PGGB performed best. The mean F1 scores were 0.612 and 0.516 at 10×, and 0.563 and 0.473 at 5×, respectively. For structural-variant detection, VG with MC achieved the highest mean F1 scores across the six samples at 10×, 15×, and 20×. The scores were 0.528, 0.529, and 0.4931, respectively. At 5×, PanGenie with PGGB performed best, with a mean F1 score of 0.396. Resource use also varied. GraphTyper2 required 36.47 hours on PGGB. BayesTyper used 96.16 gigabytes of peak memory on minigraph.

Conclusions

Across the various conditions evaluated in this study, no single tool consistently performed the best. Therefore, when selecting a tool, one should consider the graph structure and its compatibility with the specific workflows of the tools under consideration.

Introduction

Rice is an important food crop and a classic model plant in the field of crop genomics. Its germplasm resources harbor a wealth of sequence variation, including single-nucleotide polymorphisms (SNPs), small insertions and deletions (indels), and structural variations (SVs) [1, 2]. These variants can affect gene structure, gene regulation, and agronomic traits, so their accurate characterization is important for studying rice genome evolution, population diversity, and trait-associated variation.

Most variant analyses still rely on a single linear reference genome. This approach can miss population-specific sequences and complex variants, and may bias read alignment when the sample differs substantially from the reference [3, 4]. A pangenome graph combines sequences from multiple individuals as alternative paths, providing a broader reference for read alignment and variant analysis across different genomic backgrounds. Different construction methods produce different topologies and path organizations, and they alter the number and types of candidate variants represented in the graph. Genotyping performance should therefore be assessed in relation to the graph structure, candidate-variant representation, and analysis tool.

Systematic evaluations of different rice pangenome graph representations in combination with genotyping tools remain limited. Using 36 rice genome assemblies, we constructed three pangenome graphs, MC, PGGB, and minigraph, and paired them with VG, GraphTyper2, PanGenie, and BayesTyper. We assessed the performance of each tool and graph combination for SNPs, small insertions and deletions, structural variants, and computational resource use. The VG and PGGB combination could not be evaluated because construction of the PGGB graph index failed.

Materials and Methods

Overall Study Design

We constructed three pangenome graphs, MC, PGGB, and minigraph, from 36 rice genome assemblies and evaluated VG, GraphTyper2, PanGenie, and BayesTyper with each graph using short-read sequencing data. The benchmark included both samples used to build the graphs and samples that were excluded from graph construction. For each sample, we tested four sequencing depths: 5×, 10×, 15×, and 20×. For each tool, we supplied the inputs required by its workflow, including the relevant graph structure or index, candidate-variant VCF file, reference genome, and sorted BAM file. The resulting VCF files were normalized and compared with assembly-derived truth sets within intervals supported by alignments between each sample assembly and the reference genome. The complete workflow is summarized in Fig. 1.

Pangenome Graph Construction

NIP-T2T served as the common coordinate system throughout the study [5]. We represented every pangenome graph and candidate variant VCF in this coordinate system for read alignment, variant representation, and accuracy assessment.

The MC, PGGB, and minigraph graphs were constructed from the same 36 rice genome assemblies: 534M, GP58, GP117, GP119, GP505, GP523, GP524, GP543, GP622, GP635, GP680, Gla4, HP436, HP519, NIP-T2T, OL2296, OL3101, OM1952, W0108, W0133, W0169, W0596, W0639, W1668, W1677, W1723, W1732, W1735, W1777, W1787, W1970, W2005, W2108, W3033, W3071, and W3092 [2, 6, 7, 8, 9]. We generated MC with the minigraph-cactus pipeline (v3.0.0) [10], which represents multipath sequence

relationships. PGGB (v0.7.4) [11] was used for whole-genome alignment, graph construction, graph smoothing, and variant extraction. We constructed the minigraph graph with minigraph (v0.21-r606) [12], using NIP-T2T as the reference and adding the other 35 assemblies progressively. MC and PGGB were assessed for SNPs, small indels, and SVs. Because its available candidate set consisted mainly of SV records, minigraph was assessed primarily for SVs.

Samples and Sequencing Data

Six rice samples were included in the benchmark. W0169, W1735, and W3033 were part of the 36-genome panel used for graph construction. The graph-excluded samples were 9311 [13, 14], IR64 [15], and Zhonghua11 [16, 17, 18]. All samples had 150-bp Illumina paired-end reads.

For each sample, we synchronously and randomly subsampled the original R1 and R2 reads with seqtk (v1.5-r133) [19] to approximate 5×, 10×, 15×, and 20× coverage, using the length of the NIP-T2T reference genome for the calculation. Read pairing was retained, and each sample–depth combination received a deterministic random seed. We did not perform additional adapter trimming, quality trimming, or error correction beyond the preprocessing applied to the original data. Target depths, numbers of subsampled read pairs, and random seeds are reported in **Supplementary Table S1**.

Assembly-derived truth sets

None of the six samples used in this study had an experimentally validated truth set. We therefore aligned each sample assembly to the NIP-T2T reference genome and constructed assembly-derived truth sets.

Dipcall (v0.3) [20] was used to generate truth sets for SNPs and small insertions and deletions. Because only one consensus assembly was available for each sample, we supplied the same assembly as both haplotype inputs to Dipcall. This produced a pseudo-diploid representation for genotype-aware comparisons and generated the corresponding SNP and small-indel truth sets. We then processed the results with bcftools (v1.24) [21] by checking REF-allele consistency, normalizing and decomposing variants, and sorting and indexing the VCF files. Records with “.” in the FILTER field were retained. Under the VCF specification, “.” indicates that no filter was applied, whereas “PASS” indicates that the record passed all applied filters [22].

Dipcall also produced a BED file for each sample, listing intervals supported by reliable assembly-to-reference alignments. The same intervals were used as a common evaluation mask for SNPs, small indels, and SVs. We then harmonized the chromosome names with those of the common reference genome while retaining the standard 0-based, half-open BED coordinate format.

We generated SV truth sets with minimap2 (v2.31-r1302) [23] and SVIM-asm (v1.0.3) [24]. The sample assemblies were aligned to NIP-T2T with minimap2 using the asm5 preset, and the cs tag was retained.

This tag records matches and sequence differences between the assembly and the reference, including substitutions, insertions, and deletions. After sorting and indexing the alignments with samtools (v1.24) [25], we ran SVIM-asm in haploid mode. The resulting VCF records were standardized with respect to chromosome names and reference-contig fields, and were then formatted, sorted, and indexed. We retained only records with FILTER=PASS. Breakend(BND) records were excluded because their paired-breakend representation was not directly comparable with the sequence-resolved SVs evaluated here. The truth sets represented sequence-resolved, assembly-supported SV alleles relative to the reference.

Variant genotyping and VCF processing

We evaluated VG (v1.73.0) [26], GraphTyper2 (v2.7.7) [27], PanGenie (v4.2.1) [28], and BayesTyper (v1.5) [29]. VG aligns reads to graph indexes, aggregates read support, and calls variants within the graph-based workflow. We used it with MC and minigraph. We also attempted to use VG with PGGB: conversion to GFA and generation of the XG, GBWT, and GBZ indexes completed, but construction of the distance index (.dist) stopped after VG reported an oversized snarl containing 818,250 nodes and the runtime error “node length is too large.” The distance, minimizer, and zipcode indexes were consequently incomplete or unavailable, so VG was not evaluated on PGGB. GraphTyper2 uses the reference genome, sample BAM files, and candidate VCFs; we used its genotype mode for SNPs and small indels and genotype_sv for SVs. PanGenie infers genotypes from chromosome-specific pangenome indexes, candidate variant VCFs, and sample reads; chromosomes were processed separately and the calls were merged afterward. BayesTyper first counts k-mers and builds a Bloom filter from the reads, then clusters and genotypes the candidate variant panel.

Because tool-generated VCF files may differ in chromosome names, reference-sequence information, filtering status, genotype representation, and multiallelic organization, we standardized them before evaluation. For each call set, we harmonized chromosome and reference-sequence information, retained FILTER=PASS records with complete genotypes containing at least one non-reference allele, checked REF alleles against the reference, normalized and decomposed variants, and sorted, compressed, and indexed the VCF files. Records that failed reference-sequence checks were removed. Truth VCFs followed the pipeline-specific criteria described above, which were not identical to the criteria applied to tool calls. The sample-specific BED files were used as evaluation masks.

We applied the same variant definitions to all tool calls and truth records. SNPs had single-base REF and ALT alleles that differed. Small indels were explicit-sequence insertions or deletions with an absolute REF–ALT length difference below 50 bp. SVs were sequence-resolved variants with explicit REF and ALT alleles and an absolute length difference of at least 50 bp. We defined the threshold using the absolute difference in REF and ALT allele lengths, rather than applying separate thresholds to the two alleles. This threshold includes large insertions, deletions, and other length-changing events, but leaves some SV classes outside the analysis.

Accuracy evaluation

For each sample, calls were evaluated only within intervals supported by the assembly-to-reference alignment. The same BED file defined the evaluation region for SNPs, small indels, and SVs. We supplied the BED files to RTG vcfeval (v3.13) [30] with -e for SNP and small-indel evaluation and to Truvari (v5.4.0) [31] with --includebed for SV evaluation. In both tools, the BED files defined the evaluable coordinate range and were not used to pre-filter records in the truth VCFs.

RTG vcfeval compared SNPs and small indels using the same NIP-T2T reference, sample mapping (--sample syndip,<sample>), and diploid setting (--sample-ploidy 2). The resulting genotype-aware precision, recall, and F1 score measure concordance with the assembly-derived pseudo-diploid representation, rather than accuracy against independently resolved diploid genotypes.

We evaluated SVs with Truvari using identical matching parameters for all comparisons. Truvari matches variants at the variant level using position, size, and sequence similarity, and calculates precision, recall, and F1 score without using genotype information. The metrics quantify sequence-level matching to the haploid-assembly SV truth sets. BND variants were excluded. GraphTyper2 initially produced symbolic ALT alleles, so we sequence-resolved them before evaluation to make the representation comparable with the other tools. Because SNP and small-indel metrics are genotype-aware whereas SV metrics use variant-level matching, the two sets of metrics were interpreted separately.

Computational resource profiling

We profiled resource use with the 5× Illumina dataset from sample 9311. The assessment covered 11 completed tool–graph workflows: VG on MC and minigraph, and GraphTyper2, PanGenie, and BayesTyper on each of the three graph representations. VG on PGGB was excluded because graph-index construction failed. Each workflow was configured with 16 threads. We recorded wall-clock time, mean CPU core usage, and peak memory. Resource measurements refer to sample 9311 at 5× coverage.

Results

Graph Size and Candidate Variant Space

The three pangenome graph representations differed in both structural scale and candidate-variant space (Fig. 2).

The MC and PGGB files were both much larger than the minigraph file, and the MC graph contained the greatest numbers of nodes and edges. MC and PGGB included candidate alleles representing SNPs, small indels, SVs, and other record types, whereas the candidate alleles in minigraph were mainly associated with SVs. These candidate variants reflect the variation represented by each graph, with PGGB representing the largest number of variants. To summarize variant counts and the proportions of different variant classes, we decomposed multiallelic records into individual ALT alleles and found that SNPs accounted for the largest proportion in the MC and PGGB graphs.

SNP Genotyping Performance

SNP genotyping performance differed among graph representations and tools (Fig. 3).

On the MC graph, at sequencing depths of 10× and above, VG had higher F1 scores and recall than the other three tools. For graph-included samples, its F1 score rose from 0.577 at 10× to 0.626 at 20×, and recall rose from 0.438 to 0.491. The graph-excluded group showed the same pattern, with F1 scores of 0.641 and 0.680 and recall values of 0.496 and 0.534. BayesTyper had the highest precision in both groups, ranging from 0.915 to 0.943 in graph-included samples and from 0.930 to 0.941 in graph-excluded samples. Its lower recall results in an F1 score that is lower than that of VG.

On the PGGB graph, GraphTyper2 had higher F1 scores, precision, and recall than PanGenie and BayesTyper in both sample groups at every sequencing depth. For graph-included samples, GraphTyper2's F1 score ranged from 0.553 to 0.613, precision from 0.866 to 0.890, and recall from 0.401 to 0.475. For graph-excluded samples, the corresponding ranges were 0.573 to 0.636, 0.928 to 0.936, and 0.413 to 0.484.

Small Indel Genotyping Performance

For small indels, the relative performance of the tools changed between MC and PGGB, and also varied with sequencing depth and sample group (Fig. 4).

On the MC graph, VG was the most sensitive to increasing sequencing depth. Except at 5×, VG had the highest F1 score in both sample groups. And its F1 score also increased with depth. In graph-included samples, the score rose from 0.354 at 5× to 0.544 at 20×. In graph-excluded samples, it increased from 0.374 to 0.556. PanGenie was relatively stable in graph-included samples, with F1 scores ranging from 0.464 to 0.468. It had the highest F1 score in this group at 5×.

On the PGGB graph, GraphTyper2 achieved the highest F1 score and precision in both sample groups at all sequencing depths. For graph-included samples, its F1 score increased from 0.470 at 5× coverage to 0.527 at 20×; for graph-excluded samples, it increased from 0.478 to 0.536.

SV Matching Performance

For the sequence-resolved, length-changing SVs included in the analysis, Truvari results varied across both tools and graph representations (Fig. 5).

On the MC graph, VG had the highest F1 score at sequencing depths of 10× and above. In graph-included samples, its F1 score increased from 0.493 at 10× to 0.533 at 20×, while recall rose from 0.369 to 0.419 and precision remained close to 0.74. In graph-excluded samples, the F1 score increased from 0.493 to 0.524, recall from 0.392 to 0.435, and precision remained around 0.66. At 5×, PanGenie had the highest F1 score among graph-included samples (0.453). Its performance changed little with sequencing depth, whereas GraphTyper2 consistently produced lower F1 scores. BayesTyper reached an F1 score of 0.443 at 15× in graph-included samples, but this fell to 0.385 at 20×, mainly because recall decreased.

On the PGGB graph, BayesTyper had the highest F1 score among graph-excluded samples at all sequencing depths, ranging from 0.353 to 0.406. Among graph-included samples, its F1 score ranged from 0.471 to 0.480 at depths of 10× and above; at 5×, its score was slightly lower than that of PanGenie (0.439 versus 0.447). GraphTyper2 performed the worst.

On the minigraph graph, VG performed best at depths of 10× and above. Its F1 score increased from 0.496 to 0.547 in graph-included samples and from 0.413 to 0.443 in graph-excluded samples. Recall increased from 0.399 to 0.480 and from 0.352 to 0.408, respectively. At 5×, PanGenie had a higher F1 score than VG among graph-included samples (0.396 versus 0.320), whereas BayesTyper performed better than VG among graph-excluded samples (0.318 versus 0.280). PanGenie remained relatively stable across sequencing depths. BayesTyper's F1 score decreased as sequencing depth increased, while GraphTyper2 improved but remained the lowest-performing tool.

We examined whether GraphTyper2's symbolic ALT alleles might explain its lower performance. We therefore sequence-resolved its output and found that the results were nearly identical to those obtained with the original output (**Supplementary Table S5**). To maintain consistency with the other three tools, the figure shows results from the sequence-resolved output.

Computational resource use under a single profiling condition

Figure 6. **Computational resource use across supported tool–graph workflows.** (**A**) Wall-clock time. (**B**) Mean CPU core usage. (**C**) Peak memory. Rows represent graph representations, and colors identify the analysis tools. Within each graph, the measurements are joined by lines to aid visual comparison. **Supplementary Table S6** gives the complete set of measurements.

Regarding the actual runtime of various tool and pangenome graph combinations, GraphTyper2 took the longest when paired with the PGGB graph, requiring 36.468 hours, while PanGenie was the fastest, running on minigraph in only 0.427 hours. Although minigraph is clearly smaller than the MC or PGGB graphs, which would theoretically imply shorter runtimes, BayesTyper took 2.657 hours with minigraph, longer than with either of the other two graph types. VG showed similar runtimes on MC and minigraph, at 1.452 hours and 1.437 hours, respectively.

In terms of average CPU core usage, GraphTyper2 consistently used the fewest cores across the three graphs, suggesting limited parallel processing capability. In contrast, BayesTyper used the most cores on the MC and PGGB graphs, at 10.72 and 10.86, respectively.

Peak memory usage varied across the different tool-graph combinations. The most significant difference occurred on minigraph, where GraphTyper2 and BayesTyper consumed 1.238 GB and 96.161 GB, respectively. GraphTyper2 required 8.531 GB with the MC graph and 34.76 GB with the PGGB graph. PanGenie's peak memory usage remained relatively consistent across the three graphs: 24.287 GB on

MC, 24.209 GB on PGGB, and 23.209 GB on minigraph. VG showed relatively high memory consumption, reaching 79.237 GB on the MC graph and 69.516 GB on minigraph.

Discussion

Graph Structure and Tool Compatibility

MC and PGGB are larger in scale than minigraph, containing a greater number of nodes, edges, and candidate variants. Their increased structural complexity, however, has not translated into consistent improvements in accuracy. The graph structure alone cannot account for the performance differences observed among these tools; rather, such differences likely stem from the interplay of various factors, including graph topology, the organization of paths, the representation of candidate variants, and the specific genotyping workflows employed by each tool.

The four evaluated tools differed in how they used graph information. VG relied directly on graph indexes for read alignment and variant analysis. Its performance depended on the graph structure and on successful index construction. VG performed well on MC, but index construction did not finish on PGGB. Tool and graph compatibility therefore involved both accuracy and computational feasibility. GraphTyper2, PanGenie, and BayesTyper all combined candidate variants with sequencing evidence, but they processed candidate variants and inferred genotypes in different ways. On PGGB, GraphTyper2 was suitable for single-nucleotide polymorphism and small-indel analysis but performed less well for structural variants. BayesTyper performed better for structural-variant analysis on the same graph. A workflow suited to small-variant analysis may not be equally suitable for structural-variant analysis.

Tools and graph structures should be selected as a combination. In the tasks evaluated here, VG with MC and GraphTyper2 with PGGB were more suitable for small-variant analysis. BayesTyper with PGGB and VG with MC or minigraph were more suitable for structural-variant analysis. Graph representation should therefore be chosen together with the intended analysis tool and target variant class.

Sample inclusion and sequencing depth

Whether graph-included samples performed better than graph-excluded samples depended on the tool, pangenome graph representation, and variant class. Since the two groups contained different accessions, we use these comparisons only to describe the observed patterns. The independent effect of graph inclusion cannot be separated from the effects of genetic background, assembly quality, or truth-set characteristics. Therefore, inclusion in pangenome graph construction should not be regarded as a guarantee of improved genotyping performance.

Increasing sequencing depth improved the F1 score for most tool-graph combinations because additional reads provide more evidence for alleles represented in the graph or candidate variant set. This effect was not consistent, however. For BayesTyper, both precision and recall decreased at 20×, resulting in a lower F1 score. Greater sequencing depth may amplify the effects of sequencing errors and

ambiguous alignments, thereby increasing erroneous calls. Sequencing depth should therefore be selected together with the analysis tool, graph representation, and target variant class, rather than assuming that higher depth will always improve genotyping performance.

Computational resources

GraphTyper2 had the longest runtime on PGGB. This may be related to the graph's higher structural complexity and GraphTyper2's weak parallel processing capability. BayesTyper had the highest peak memory usage on minigraph, but minigraph was smaller in scale and contained fewer candidate variants. This result indicates that computational demand is also affected by each tool's internal data processing and genotyping workflow.

All workflows were configured to use 16 threads. In practice, the tool–graph combinations did not achieve the same degree of parallel execution. Task allocation, the level of parallelization, and serial processing steps may account for the observed differences in CPU use and runtime. Tool selection should consider variant-analysis performance, runtime, and memory requirements together. The resource assessment used 5× sequencing data from sample 9311 and covered only the specific tool and graph combinations evaluated in this study.

Limitations and future directions

Accuracy in this study was evaluated against truth sets derived from genome assemblies; these truth sets were not independently validated as diploid truth sets. For SNPs and small indels, the consensus assembly of each sample was treated as two haplotypes and provided as input to dipcall to construct a pseudo-diploid truth representation. This representation supports genotype-aware comparison but lacks independent information for each homologous chromosome and cannot resolve the true heterozygous genotypes. The SV truth sets were constructed from haploid assemblies; Truvari was used primarily to evaluate variant-level matching of sequence-resolved SVs, and diploid genotype concordance was outside the scope of the evaluation. The analysis focused on sequence-resolved, length-changing SVs, defined as variants with an absolute difference of at least 50 bp between the REF and ALT lengths, and excluded BND records. Accordingly, the SNP and small-indel metrics reflect concordance with the assembly-derived pseudo-diploid truth representation, whereas the SV metrics reflect matching performance for the specified sequence-resolved SVs. These results should not be generalized as overall accuracy across all variant classes or SV types. In addition, the evaluation regions were derived from assembly-to-reference alignments and used as a common evaluation mask across variant classes; they were not specifically generated according to SV breakpoint confidence.

Future benchmarks can extend this framework to more accessions, independently validated truth sets, and long-read or hybrid sequencing data. Evaluations that vary graph construction and candidate-variant generation separately could distinguish their effects from those of the genotyping tools. Extending the benchmark to additional SV classes and breakpoint representations would also show how well the observed patterns generalize beyond the sequence-resolved, length-changing variants assessed here.

Together, these steps would provide a broader basis for choosing graph-based short-read analysis workflows for rice.

Conclusions

Our benchmark evaluated the short-read variant analysis performance of combinations of three rice pangenome graph representations and four genotyping tools across four sequencing depths and three variant types. No single combination of tool and graph representation consistently outperformed the others across all conditions. At higher sequencing depths, the combination of VG and MC is recommended for detecting all variant types evaluated in this study, while the combination of VG and minigraph serves as a robust choice for structural variant detection. When the pangenome graph is large and constructed using PGGB, VG may fail during the index-building process; in such cases, GraphTyper2 can be used for the analysis of single-nucleotide polymorphisms and small insertions/deletions, while BayesTyper is suitable for structural variant analysis. Beyond these recommendations, the selection of a graph-based pangenome genotyping tool should also take into account the available computational resources. Future research should leverage larger-scale population datasets, include more diverse graph construction workflows, include more diverse genotyping workflows, and further validate the generalizability of these findings.

Declarations

Ethics approval and consent to participate

Not applicable. This study analyzed publicly available rice genome assemblies and sequencing data. It involved neither human participants nor newly collected plant material.

Consent for publication

Not applicable.

Competing interests

No conflict of interest is declared.

Funding

This work has received funding from the Science and Technology Project of the Ministry of Agriculture and Rural Affairs of China (0304); the National Key R&D Program of China (2025YFC3410300); the National Natural Science Foundation of China (Grant No. 32470678); the Agricultural Science and Technology Innovation Program (CAAS-ZDRW202503); the Youth Innovation Program of the Chinese Academy of Agricultural Sciences (Y2025QC36); the Innovation Program of Chinese Academy of

Agricultural Sciences (CAAS-CSIAF-202301); the Basic Research Programs of Shanxi Province (202303021211069); and Key Research and Development Programs of Shanxi Province (202402020101008).

Author Contribution

W.P., D.L. and L.Z. conceived and designed the study. W.Y. acquired and curated the data. X.C. installed and configured the analysis software. L.M. implemented the benchmarking workflows, performed the analyses, and drafted the manuscript. All authors read and approved the final manuscript.

Acknowledgement

We thank Yanchun Li for guidance and assistance during the preparation of the manuscript.

Code availability.

The code supporting the analyses in this study is available in the GitHub repository [32].

Data Availability

The 36 rice genome assemblies used to construct the MC, PGGB, and minigraph graphs, including the graph-included samples W0169, W1735, and W3033, were obtained from publicly available rice pangenome resources. The associated sequencing and genomic data are available from the European Nucleotide Archive (PRJEB73710; https://www.ebi.ac.uk/ena/browser/view/PRJEB73710), the National Genomics Data Center (PRJCA024131; https://ngdc.cncb.ac.cn/bioproject/browse/PRJCA024131), Figshare (https://doi.org/10.25452/figshare.plus.25697817.v1), and RicePanDB (http://ricepandb.ncgr.ac.cn).For the graph-excluded samples, the sequencing data and genome assembly of 9311 are available from NGDC (CRA023020; https://ngdc.cncb.ac.cn/gsa/browse/CRA023020) and Genome Warehouse assembly record 88980 (https://ngdc.cncb.ac.cn/gwh/Assembly/88980/show), respectively. The IR64 genome assembly is available from NCBI Assembly (GCA_055690995.1; https://www.ncbi.nlm.nih.gov/datasets/genome/GCA_055690995.1/), with associated sequencing data under BioProject PRJNA1311239 (https://www.ncbi.nlm.nih.gov/bioproject/1311239). For Zhonghua11, the Illumina short-read data are available from NCBI SRA (SRR10567117; https://www.ncbi.nlm.nih.gov/sra/?term=SRR10567117), and the genome assembly is available from NGDC under BioProject PRJCA037115 (https://ngdc.cncb.ac.cn/bioproject/browse/PRJCA037115).The NIP-T2T reference genome is available from NCBI Assembly (GCA_034140825.1; https://www.ncbi.nlm.nih.gov/datasets/genome/GCA_034140825.1/). The analysis code is available at https://github.com/malong0318/rice-pan-genotyping-benchmark. Assembly-derived truth-set manifests,

evaluation results, and supplementary tables are provided as supplementary files accompanying this article.

References

1. Wang W, Mauleon R, Hu Z, et al. Genomic variation in 3,010 diverse accessions of Asian cultivated rice. Nature. 2018;557:43–9. https://doi.org/10.1038/s41586-018-0063-9.
2. Guo D, Li Y, Lu H, et al. A pangenome reference of wild and cultivated rice. Nature. 2025;642:662–71. https://doi.org/10.1038/s41586-025-08883-6.
3. Eizenga JM, Novak AM, Sibbesen JA, et al. Pangenome graphs. Annu Rev Genomics Hum Genet. 2020;21:139–62. https://doi.org/10.1146/annurev-genom-120219-080406.
4. Liao WW, Asri M, Ebler J, et al. A draft human pangenome reference. Nature. 2023;617:312–24. https://doi.org/10.1038/s41586-023-05896-x.
5. National Center for Biotechnology Information. Nipponbare genome assembly AGIS1.0, accession GCA_034140825.1. NCBI Datasets. https://www.ncbi.nlm.nih.gov/datasets/genome/GCA_034140825.1/. Accessed 29 Oct 2026.
6. National Center for Gene Research, Chinese Academy of Sciences. A pangenome reference of wild and cultivated rice: sequencing data, BioProject PRJEB73710. European Nucleotide Archive. https://www.ebi.ac.uk/ena/browser/view/PRJEB73710. Accessed 28 October 2026.
7. National Center for Gene Research, Chinese Academy of Sciences. A pangenome reference of wild and cultivated rice, BioProject PRJCA024131. CNCB-NGDC BioProject. https://ngdc.cncb.ac.cn/bioproject/browse/PRJCA024131. Accessed 28 October 2026.
8. Zhao Y. A pangenome reference of wild and cultivated rice. Figshare. 2025. https://doi.org/10.25452/figshare.plus.25697817.v1
9. National Center for Gene Research, Chinese Academy of Sciences. RicePanDB: wild and cultivated rice pangenome data. http://ricepandb.ncgr.ac.cn. Accessed 28 October 2026.
10. Hickey G, Monlong J, Ebler J, et al. Pangenome graph construction from genome alignments with Minigraph-Cactus. Nat Biotechnol. 2024;42:663–73. https://doi.org/10.1038/s41587-023-01793-w.
11. Garrison E, Guarracino A, Heumos S, et al. Building pangenome graphs. Nat Methods. 2024;21:2008–12. https://doi.org/10.1038/s41592-024-02430-3.
12. Li H, Feng X, Chu C. The design and construction of reference pangenome graphs with minigraph. Genome Biol. 2020;21:265. https://doi.org/10.1186/s13059-020-02168-z.
13. National Genomics Data Center. GSA project CRA023020. CNCB-NGDC. https://ngdc.cncb.ac.cn/gsa/browse/CRA023020. Accessed 28 October 2026.
14. National Genomics Data Center. Genome Warehouse assembly record 88980. CNCB-NGDC. https://ngdc.cncb.ac.cn/gwh/Assembly/88980/show. Accessed 28 October 2026.
15. National Center for Biotechnology Information. IR64 genome assembly GCA_055690995.1 and BioProject PRJNA1311239. NCBI Datasets and BioProject.

https://www.ncbi.nlm.nih.gov/datasets/genome/GCA_055690995.1/; https://www.ncbi.nlm.nih.gov/bioproject/1311239. Accessed 28 October 2026.

16. Yao X, Sun Z, Xu X, et al. Complete genome assembly of japonica rice variety Zhonghua 11. Plant Commun. 2025;6:101463. https://doi.org/10.1016/j.xplc.2025.101463.
17. National Center for Biotechnology Information. Zhonghua11 Illumina sequencing reads, run SRR10567117, BioProject PRJNA592760. NCBI Sequence Read Archive. https://www.ncbi.nlm.nih.gov/sra/?term=SRR10567117. Accessed 28 October 2026.
18. Yao X, Sun Z, Xu X et al. Zhonghua 11 genome assembly, BioProject PRJCA037115. CNCB-NGDC BioProject. https://ngdc.cncb.ac.cn/bioproject/browse/PRJCA037115. Accessed 28 October 2026.
19. Li H. seqtk: toolkit for processing sequences in FASTA/Q formats. GitHub. https://github.com/lh3/seqtk. Accessed 29 Oct 2026.
20. Li H. dipcall: call variants from diploid assemblies. GitHub. https://github.com/lh3/dipcall. Accessed 29 Oct 2026.
21. Danecek P, Bonfield JK, Liddle J, et al. Twelve years of SAMtools and BCFtools. GigaScience. 2021;10(2):giab008. https://doi.org/10.1093/gigascience/giab008.
22. Danecek P, Auton A, Abecasis G, et al. The variant call format and VCFtools. Bioinformatics. 2011;27:2156–8. https://doi.org/10.1093/bioinformatics/btr330.
23. Li H. Minimap2: pairwise alignment for nucleotide sequences. Bioinformatics. 2018;34:3094–100. https://doi.org/10.1093/bioinformatics/bty191.
24. Heller D, Vingron M. SVIM-asm: structural variant detection from haploid and diploid genome assemblies. Bioinformatics. 2020;36(22–23):5519–21. https://doi.org/10.1093/bioinformatics/btaa1034.
25. Li H, Handsaker B, Wysoker A, et al. The Sequence Alignment/Map format and SAMtools. Bioinformatics. 2009;25:2078–9. https://doi.org/10.1093/bioinformatics/btp352.
26. Garrison E, Sirén J, Novak AM, et al. Variation graph toolkit improves read mapping by representing genetic variation in the reference. Nat Biotechnol. 2018;36:875–9. https://doi.org/10.1038/nbt.4227.
27. Eggertsson HP, Kristmundsdottir S, Beyter D, et al. GraphTyper2 enables population-scale genotyping of structural variation using pangenome graphs. Nat Commun. 2019;10:5402. https://doi.org/10.1038/s41467-019-13341-9.
28. Ebler J, Ebert P, Clarke WE, et al. Pangenome-based genome inference allows efficient and accurate genotyping across a wide spectrum of variant classes. Nat Genet. 2022;54:518–25. https://doi.org/10.1038/s41588-022-01043-w.
29. Sibbesen JA, Maretty L, Krogh A. Accurate genotyping across variant classes and lengths using variant graphs. Nat Genet. 2018;50:1054–9. https://doi.org/10.1038/s41588-018-0145-5.
30. Cleary JG, Braithwaite R, Gaastra K, et al. Comparing variant call files for performance benchmarking of next-generation sequencing variant calling pipelines. bioRxiv. 2015;023754. https://doi.org/10.1101/023754.

31. English AC, Menon VK, Gibbs RA, Metcalf GA, Sedlazeck FJ. Truvari: refined structural variant comparison preserves allelic diversity. Genome Biol. 2022;23:271. https://doi.org/10.1186/s13059-022-02840-6.
32. Ma L. Rice pan-genotyping benchmark. GitHub. https://github.com/malong0318/rice-pan-genotyping-benchmark. Accessed 29 Oct 2026.

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- SupplementaryTableS1Sampleandsequencingdata.xlsx
- SupplementaryTableS4SVmatchingperformance.xlsx
- SupplementaryTableS5ComparisonofGraphTyper2SVrepresentations.xlsx
- SupplementaryTableS6Computationalresourceuse.xlsx