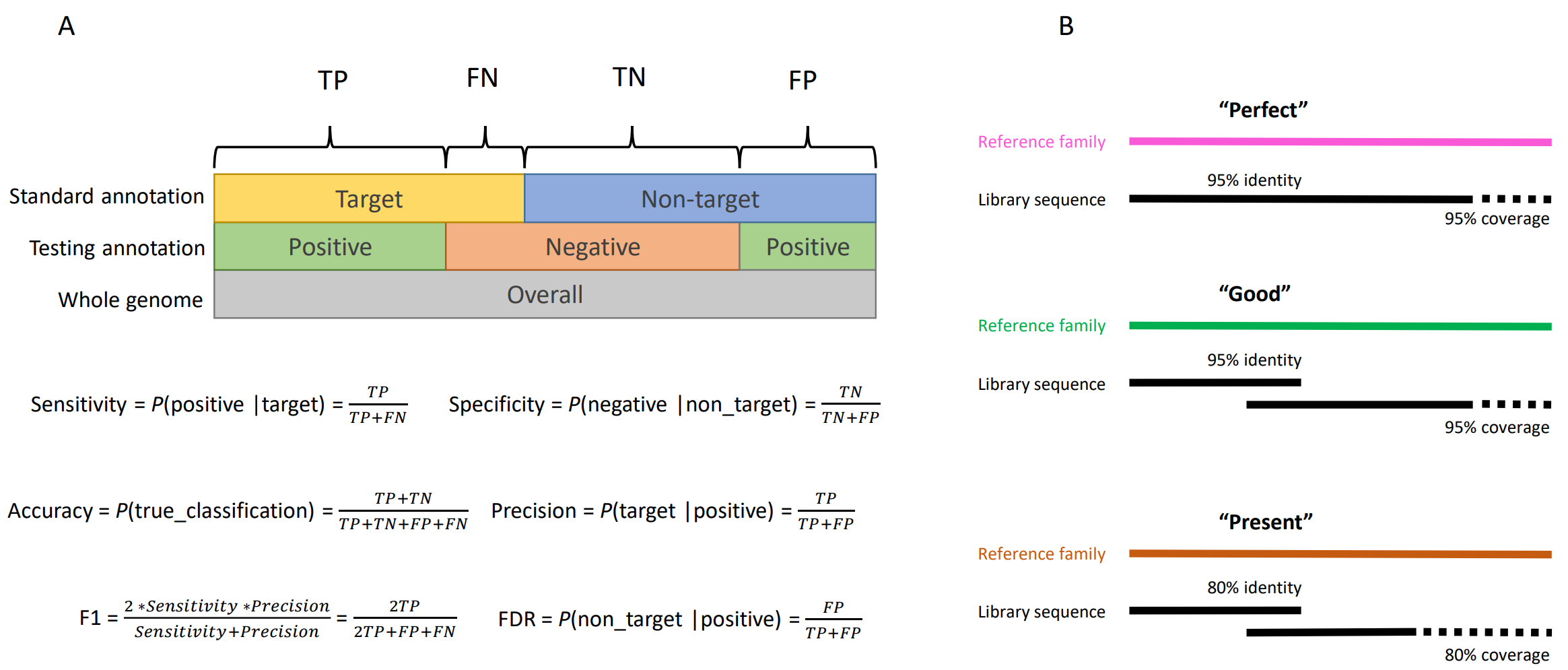
## Additional File 1. Supplementary Materials

### Setting up benchmarking methods for TE library evaluation

To fairly and comprehensively measure the quality of TE libraries generated by different TE identification tools, we use the benchmarking methods released in two recent studies, EDTA and RepeatModeler2. For convenience, we hereafter refer to the benchmarking methods of EDTA and RepeatModeler2 as BM\_EDTA and BM\_RM2, respectively.

As shown in Fig. S1(A), the BM\_EDTA evaluates the performance of various tools by annotating the genome with the gold standard TE library and the tested TE library generated by these tools. Based on the total number of genomic DNA bases, six metrics, including sensitivity, specificity, accuracy, precision, FDR, and F1, are used to characterize the annotation performance of the tested library. The BM\_EDTA can display detailed metrics, including the rates of false positives, which are common in many TE identification methods. However, it cannot reflect the integrity of the TE models. All general repeat identification programs, even those with many fragments and unclear boundaries, still performed well in benchmarking. For example, while a 1 kbp intact TE sequence and ten 100 bp fragments may obtain the same performance, the former is obviously more valuable in terms of TE integrity and biological significance.

As shown in Fig. S1(B), the BM\_RM2 aligns the tested TE library with the gold standard library and divides the gold standard sequences into four levels: “Perfect”, “Good”, “Present”, and “Not found”. “Perfect” families are those for which one sequence in the tested library matches with >95% sequence similarity and >95% length coverage to a family consensus in the gold standard library. “Good” families are those in which multiple overlapping sequences in the tested library match with >95% similarity and >95% coverage to the curated consensus. A family is considered “present” if one or multiple library sequences align with >80% similarity and >80% coverage to the reference consensus sequence. Below these thresholds, a family is considered “not found”. The BM\_RM2 takes the integrity of the sequence into consideration. Intact TE models usually get a perfect level, while fragments can only get a good, present, or even not found level. However, it cannot display the rate of false positives in the tested TE library. By combining the two complementary benchmarking methods, we can accurately evaluate the integrity of TE models and the rate of false positives in the whole TE library.



**Fig. S1.** Schematic representation of benchmarking methods. A. EDTA benchmarking methods. B. RepeatModeler2 benchmarking methods.

### Kmer-Based De Novo TE Searching Algorithm Description

Algorithm 1 describes the RepeatMasking algorithm, where *G* is genome assembly, *k* is the size of the *k*-mer, *L* is the length of divided genome segments, and *g* is the maximum length of the gap between adjacent repeat regions; *R* is the set of candidate repeat regions; The *buildRTable(·)* function is used to construct the hash table of repeated *k*-mers, the *cutSegments(·)* function is used to divide the whole genome into genome segments, the *cutKmers(·)* function cuts the genome segments into *k*-mers, the *queryRtable(·)* function is used to judge whether *k*-mers are repeated by querying the repeated *k*-mers hash table, and the *maskSequence(·)* is a function to mark repeated sequences, The *skipGaps(·)* function connects adjacent repeat sequences to skip small gaps, while the *extractRepeats(·)* function is used to extract candidate repeat sequences from masked sequences.

To analyze the time complexity of RepeatMasking, we noticed that DSK is a highly efficient tool that can process a mammalian genome in a few minutes, so the *buildRTable(·)* function actually takes very little time. The *cutSegments(·)* function divides the whole genome assembly into *N/L* segments, where *N=length(G)*. Both *cutKmers(·)* and *maskSequence(·)* have *O (L)* time complexity, while *queryRtable(·)* has *O (1)* time complexity. The total time complexity of algorithm 1 is *O(N·L)*, which is a function of *N* and *L*. In general, *L* is set as a fixed constant. In addition, since we use multiprocessing technology to accelerate the program, the running time of RepeatMasking can be reduced to *t* times the original, where *t* is the number of processes. Therefore, this algorithm has high efficiency in actual application.

|  |
| --- |
| **Algorithm 1** RepeatMasking |
| **Inputs:** G, k, L, g  **Outputs:** R |
| 1. **function** REPEATMASKING(G, k, L, g) 2. *H* ← *buildRTable(G, k)* 3. *S* ← *cutSegments(G, L)* 4. *R* ← *Φ* 5. **for** *i* = 0 → *length(S)* **do** 6. *s* ← *S[i]* 7. *P* ← *cutKmers(s, k)* 8. **for** *j = 0* → *length(P)* **do** 9. *p* ← *P[j]* 10. *isRepeated* ← *queryRtable(H, p.kmer)* 11. **if** *isRepeated* **then** 12. *s'* ← *maskSequence(s, p.start, p.end)* 13. **end if** 14. **end for** 15. *s''* ← *skipGaps(s', g)* 16. *r* ← *extractRepeats(s'')* 17. *R* ← *R*∪*r* 18. **end for** 19. **end function** |

### Example description of Fault-tolerant Mapping Expansion Algorithm

Due to the existence of insertion, deletion, and multiple TE sequences, multiple subsequence alignments will be generated in the candidate repeat area, as shown in manuscript Fig. 1(k): ***a, b, c, d, e, f***. The algorithm can be simply described as the following process:

1. We start by setting an extended threshold value p, then sort the alignments by starting and ending positions.

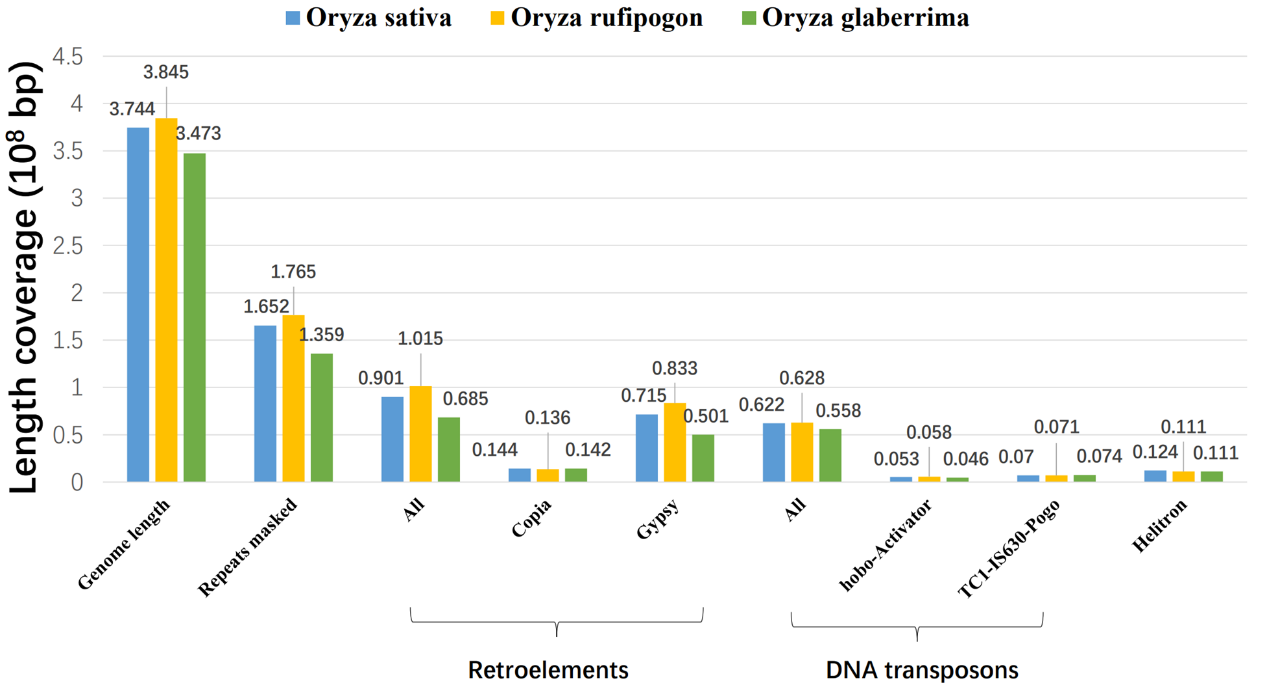
2. For each alignment, judge whether its adjacent alignment can expand the length of sequence. For example, the first is the alignment of subsequence ***a***, whose starting and ending positions are ***s1*** and ***e1***; the starting and ending positions of subsequence ***b*** are ***s2*** and ***e2***. Since ***e2 > e1*** and ***s2-e1<p***, it means that adding ***b*** can expand the length of the current subsequence, so we connect the subsequences ***a*** and ***b***. Similarly, connect subsequences ***c*** and ***d***. However, since ***s5-e4>p***, it indicates that the subsequence ***e*** is too far from sequence ***d*** to cross the gap in the middle, which should belong to two different TE instances. The TE instances in the above example are TE sequence 1 (starting ***s1***, terminating ***e4***) composed of subsequences ***a***, ***b***, ***c*** and ***d*** and TE sequence 2 (starting ***s5***, terminating ***e6***), corresponding to subsequences ***e*** and ***f***.

3. Since the query will be aligned to multiple different targets, we will get a set with overlapped sequences. We think that two sequences in the overlapped set have more than 95% overlap, and they are considered to be copies of each other. A representative sequence is used to represent all copies with overlap, and the boundary of the representative sequence is updated to include all copy sequences. Finally, we get a collection of non-overlapping repeats.

### Contribution of TEs to genome size

The amplification or contraction of transposable elements, affected by environmental stressors, is closely related to the genome size[1, 2]. LTR retrotransposons, especially the Ty3-gypsy elements, which are the major component in most plants, play an important role in the genome size variation across the Oryza genus[3].

By applying HiTE to several common rice subspecies, Oryza sativa, Oryza rufipogon, and Oryza glaberrima, we observed that there was significant genome size variation among these rice genus, and the main source of genome size difference is the Gypsy transposon (Fig. S2), as previously documented. The TE libraries of rice genomes are generated by HiTE using the default parameters. RepeatMasker is then used to generate the length coveage based on these TE libraries.



**Fig. S2.** The length coverage distribution of different types of transposons based on the analysis of rice genus.

## References

1. Canapa A, Barucca M, Biscotti M A, et al. Transposons, genome size, and evolutionary insights in animals. Cytogenetic and genome research, 2015, 147(4): 217-239.

2. Zhang X, Qi Y. The landscape of Copia and Gypsy retrotransposon during maize domestication and improvement. Frontiers in plant science, 2019, 10: 1533.

3. Zuccolo A, Sebastian A, Talag J, et al. Transposable element distribution, abundance and role in genome size variation in the genus Oryza[J]. BMC Evolutionary Biology, 2007, 7(1): 1-15.