Sequence analysis

TE-greedy-nester: structure-based detection of LTR retrotransposons and their nesting

Matej Lexa 1,2,†,*, Pavel Jedlicka 1,†, Ivan Vanat 2, Michal Cervenansky 1,2 and Eduard Kejnovsky 1

1Department of Plant Developmental Genetics, Institute of Biophysics of the Czech Academy of Sciences, 61200 Brno, Czech Republic
2Department of Machine Learning and Data Processing, Faculty of Informatics, Masaryk University, 60200 Brno, Czech Republic

†The authors wish it to be known that, in their opinion, the first two authors should be regarded as Joint First Authors.
*To whom correspondence should be addressed.

Abstract

Motivation: Transposable elements (TEs) in eukaryotes often get inserted into one another, forming sequences that become a complex mixture of full-length elements and their fragments. The reconstruction of full-length elements and the order in which they have been inserted is important for genome and transposon evolution studies. However, the accumulation of mutations and genome rearrangements over evolutionary time makes this process error-prone and decreases the efficiency of software aiming to recover all nested full-length TEs.

Results: We created software that uses a greedy recursive algorithm to mine increasingly fragmented copies of full-length LTR retrotransposons in assembled genomes and other sequence data. The software called TE-greedy-nester considers not only sequence similarity but also the structure of elements. This new tool was tested on a set of natural and synthetic sequences and its accuracy was compared to similar software. We found TE-greedy-nester to be superior in a number of parameters, namely computation time and full-length TE recovery in highly nested regions.

Availability and implementation: http://gitlab.fi.muni.cz/lexa/nested.
Contact: lexa@fi.muni.cz

Supplementary information: Supplementary data are available at Bioinformatics online.

1 Introduction

Genomes of most eukaryotic organisms contain repetitive sequences present as dispersed repeats created by different classes of transposable elements (TEs) (Kapitonov and Jurka, 1999; Smit, 1999). The dispersed repeats are produced throughout evolution by the activity of TEs, often in transposition bursts of various intensities, with many transposition events happening in a short evolutionary time frame, such as those after polyploidization events (Hirochika, 1997; Vicient and Casacuberta, 2017). In genomes with high TE content, some insertions necessarily result in the fragmentation of another transposon already present at that particular insertion locus. This leads to nesting where only the youngest full-length copies can be recognized by software not accounting for fragmentation. Previous estimates of TE nesting in plant genomes ranged from no nesting detected in Physcomitrella patens to 14.3% of TEs fragmented by a TE insertion in Oryza sativa (Gao et al., 2012). There are many tools and approaches searching for repeated sequences and their families (Bergman and Quesneville, 2007; Saha et al., 2008; Valencia and Grigis, 2019). An exhaustive list has been published in a recent review (Geornier-Poigny and Bourque, 2018). To discover nesting, people have come up with strategies to identify transposon fragments that may have originally been a part of a full-length element. Perhaps the most popular is RepeatMasker in its newer version http://www.repeatmasker.org. It identifies fragments based on sequence similarity to a library of known repeats and stitches together nearby fragments that look like they are continuations of each other when mapped to a model element. LTRtype (Zeng et al., 2017) identifies different types of structurally complex LTR retrotransposon elements as well as the nested configuration of these TEs. The system is capable of rapidly scanning large-scale genomic sequences characterizing eight complex types of LTR retrotransposon elements in addition to the common configuration of two LTR sequences positioned around an internal sequence with protein coding regions. The authors claim the program is able to correctly annotate a large number of structurally complex elements as well as nested insertions. It includes complex elements, e.g. those made of up to two internal sequences and three LTRs. REannotate (Pereira, 2008) processes RepeatMasker annotation for: automated defragmentation of dispersed repetitive elements; resolution of the temporal order of insertions in clusters of nested elements; and the estimation of the age of elements with long terminal repeats. Another specialized software tool is TEnest, for untangling nested insertions of LTR retrotransposons, also using sequence similarity and classification of
identified repeats into families (Krommiller and Wise, 2008, 2013). If nearby fragments belong to the same family, the software will recognize them as a valid power set and assign them to the same full-length element, thus establishing a nesting order. Greedier (Li et al., 2008) is an alignment-based software tool also used to detect nested insertions of transposons. Stitzer et al. (2019) mentioned a recursive approach of annotating nesting of TEs, although no software is provided or mentioned in their paper.

All available tools rely heavily on the evaluation of sequence similarity at some key step but it is evident that the structure of elements (order of domains and regulatory regions) can help to reconstruct fragmented elements. Structure-based tools are specifically available for certain classes of repetitive sequences, such as LTR retrotransposons (Ellinghaus, 2008; McCarthy and McDonald, 2003; Xu and Wang, 2007), however, none are capable of recognizing element nesting. Therefore, here, we present an alternative approach to detect nesting, using structure-based recognition of repetitive sequences, relying primarily on identification of component features of a typical transposon and their relative position.

2 Algorithm and implementation

We felt there is a possibility for improvement on TE detection by combining structure-based tools with a greedy algorithmic approach that would eliminate all detected TEs from analyzed sequences before going to the next round of detection. Initial rounds of analysis would help reconstruct many TEs that were fragmented by insertion of younger TEs but underwent little other change. Such an approach is inherently modular, it allows us to use an external, independently tested tool to detect LTR retrotransposons (or any other classes and tools in future modifications) and to separate full-length TE detection from their scoring and establishment of the most likely nesting order.

Besides developing the TE-greedy-nester (labeled as TE-g-nester in all figures and tables), we also used the common code base for creating an application that runs in the opposite direction, to generate sequences containing nested TEs (‘TE-generator’). The TE-generator can be used for the limited testing of TE-greedy-nester. The TE-greedy-nester, TE-generator and other softwares are available from our GitLab project homepage at https://gitlab.fimuni.cz/lexa/nested. It is written in Python and will run on most Unix/Linux platforms. It requires prior installation of LTR FINDER (Xu and Wang, 2007), BLAST (Altschul et al., 1990) and GenomeTools (Gribanov et al., 2013). An installation script and a link to an Ubuntu virtual machine with the latest version installed are provided.

We have also previously packaged our tools for integration and easier deployment. A Snakemake pipeline was created to run TE-greedy-nester on larger datasets and store results of the analysis in GFF files and also in a relational database http://hedron.fimuni.cz/TEDb/index.html. A Linux Mint virtual machine has been created to enable users not only to work with the above pipeline avoiding potential installation issues but also to modify it to meet their specific requirements http://hedron.fimuni.cz/TE-nester_Mint.zip.

2.1 TE-greedy-nester

Our main goal was to design an application capable of processing sequences automatically and finding nested TEs in reasonable time. We needed to address specific problems related to the correct detection of element nesting. First, while sensitive enough, the procedure should be resistant to detecting false positives. To this end, we incorporate a greedy algorithm that evaluates several possible candidates for full-length TEs but ultimately picks only the best ones, based on the presence of typical full-length TE sequence features. As a result, false positives are quite rare initially and may become more frequent at later stages which, however, can be stopped at that point. To support precision, we chose to use LTR FINDER (Xu and Wang, 2007), a TE detection tool that showed low false-positive results and high precision in our experience as well as recent tests (Valencia and Girgis, 2019). Another requirement is the ability to detect deep nesting. In such cases, the oldest elements are barely recognizable because of ageing and the procedure must allow for imperfections without compromising the ability to detect the partly eroded elements.

Several rounds of design produced a procedure according to the following pseudocode (see also Fig. 1A) where capitalized terms in parentheses represent typical components of an LTR retrotransposon, non-coding sequences (LTR, PBS, PPT and TSD) detected by LTR Finder and conserved protein-coding sequences often called protein domains (GAG, PROT, RH, RT, INT and CHD) detected using BLASTX.

Evaluation of full-length TE candidates is done by constructing a weighted directed graph, where nodes represent required sites in a full-length element (such as domains, PBS, PPT and TSD) (Fig. 1B). The program is designed to find a path from the left LTR to the right LTR, whilst visiting every required node in the correct order (domains are ordered differently in Gypsy and Copia families, some, like ENV are family-specific, all components are optional). By assigning weights to the edges, we prioritize a path that has as complete a structure as possible. At the same time, we allow alternative paths with respective penalties if there is a missing node, or an incorrect order of available nodes. The graph structure is quite universal and is open for future refinements.

We also need a way to recover various subsequences of the analyzed sequence, such as the original unfragmented sequences of older TEs fragmented by nesting and the identified features annotated to the analyzed sequence. This is achieved by a procedure where the removed sequences are virtually returned to their positions in the genome and the coordinates of TEs and their features are adjusted for the inserted elements. Once all TEs that have been removed in the first phase are processed, we generate a GFF3 file with coordinates that map to the analyzed sequence. The final GFF output file can be used to visualize all the identified features with specialized software, such as Genome Tools Annotation Sketch (Fig. 2) (Gremme et al., 2013), a genome browser, such as IGV (Robinson et al., 2011; Thorvaldsdottir et al., 2013), or to extract sequences for certain features using e.g. bedtools (Quinlan and Hall, 2010). In addition, the sequences of all TEs detected by TE-greedy-nester are clipped out of the genomic matrices and stored in FASTA format in a separate directory.

```python
algorithm TE-greedy-nester is
    input: DNA sequences

while changed(export_TEs) = TRUE do
    foreach DNA sequence do
        with LTR Finder detect module = (LTRs|PBS|PPT|TSD)
        save TE
        with BLASTX get domain = (GAG|PROT|RH|RT|INT|CHD)
        for each TE do
            hsno_TE = calculate_score(TE, domain, module)
            //using greedy algorithm, scoring graph
            save hsno_TE//highest-score_non-overlapping TE
            removed_positions = positions(hsno_TE)
            move removed_positions to export_TEs
        save removed_positions //to adjust export_GFF
        export_GFF = adjust(export_GFF, removed_positions)
    output: export_TEs, export_GFF
```
2.2 TE-generator
To carry out tests of the software, especially its ability to recover nested sequences, we used TE-generator, part of the code that is designed to carry out virtual insertions of TE sequences from a library into a background sequence. The precise position of each inserted sequence in the resulting test sequence is recorded, to compare the generated GFF3 file with results of analysis of the same sequence by TE-greedy-nester.

3 System and methods

3.1 Testing data
3.1.1 Randomly inserted sequences
A first group of testing sequences was prepared as a random mixture of full-length sequences from maize. A database of full-length TEs were downloaded from Maize TE Database at http://people.oregonstate.edu/~fowlerjo/MaizeRepeatDatabases/uniqueTEDB_1526.fasta.txt in 2018 and Mips-REdat database at ftp://ftp.mips.helmholtz-muenchen.de/plants/REdat/mipsREdat_9.3p_ALL.fasta.gz (Nussbaumer et al., 2013) for maize and rice full-length TEs, respectively. The generator module of TE-greedy-nester was used to generate 10 Mbp sequences with 10 and 90% TE sequence content. Settings were calculated from average TE length (see commandsUsed.txt in Supplementary Materials).

3.1.2 Deeply nested sequences (Russian doll/Matryoshka)
To test the depth of nesting discoverable by existing tools, as well as TE-greedy-nester, we wrote a short program (see file matryoshka.pl) which creates annotated sequence files with an arbitrary depth of mutually nested TE elements. It complements TE-generator, which also creates pockets of multiply nested TEs in the generated sequence, however, TE-generator is biased towards shallow nested sets. Matryoshka is written in Perl (code available in Supplementary Materials). The program takes a multifasta file of LTR retrotransposon sequences as input, samples it i times (i can be set on the command line), nesting each consecutive sequence into a random position of the previously inserted sequence. FASTA and BED files are written as output. Two sets of sequences containing 20 TEs taken from either Zea mays TE database (zea_matryoshka.fa) or O.sativa (oryza_matryoshka.fa) were generated.

3.1.3 Maize adh1 neighborhood sequence
To test TE-greedy-nester on biological sequence which is rich in nested LTR retrotransposons with known position and annotation, the Z.mays alcohol dehydrogenase 1 gene (SanMiguel et al., 1998) (adh1-F gene accession number: AF050457.1) flanked with 150 kbp on both 5’ and 3’ regions, was analyzed. The maize genome was downloaded from Phytozome 12.0 https://phytozome.jgi.doe.gov/pz/portal.html (Goodstein et al., 2012). TE sequences obtained by TE-greedy-nester were annotated using the maize specific retrotransposon database multifasta http://people.oregonstate.edu/~fowlerjo/MaizeRepeatDatabases/uniqueTEDB_1526.fasta.txt and BLASTN tool (Altschul et al., 1990). Thereafter, the GT Annotation Sketch figure from TE-greedy-nester was rearranged following TE orientation given in the study by SanMiguel et al. (1998) as later adapted by Fedoroff (2012).

3.1.4 Maize sequence (first 2 MB from Chr10)
The second biological sequence from the maize genome, Chromosome10: 0–2 Mb, was used previously to compare LTType, REannotate and TEnest tools (Zeng et al., 2017). We used this sequence to maintain continuity and to obtain reliable data for users of TE-greedy-nester.

3.2 Testing software and data analysis
Performance of TE-greedy-nester and other software able to detect TE nesting (TEnest, LTType and REannotate) were tested on three different types of data: (i) synthetic data with known randomly inserted sequences from maize and rice (TE-generator and matryoshka); (ii) thoroughly studied and annotated adh1 locus from maize and (iii) biological data analyzed by other similar software (2 MB from maize Chr10).
Testing on synthetic data was done by comparing GFF files produced by TE-greedy-nester with TE-generator and matryoshka GFF files describing the introduced TEs. We calculated the number of intervals representing TEs that overlap by a predefined percentage of length on both sides using the python script compGffs2Generator.py (Supplementary Materials).

Testing on the adh1 locus was done by visual inspection of published adh1 annotations (Fedoroff, 2012) and our own visualization with GT Annotation Sketch.

Testing on biological data was carried out by counting the number of detected full-length TEs in assembled or partly assembled genomes of 18 species downloaded from Phytozome or other sources (where applicable): Arabidopsis thaliana, Arabidopsis lyrata, Azolla filiculoides, Brachypodium distachion, Chlamydomonas reinhardtii, Dunaliella salina, Glycine max, Gossypium raimondii, Lotus japonicus, Medicago truncatula, Micromonas pusilla, Musa acuminata, O.sativa, P.patens, Populus trichocarpa, Pseudotsuga menziesii, Solanum lycopersicum and Sorghum bicolor.

To confirm a quality of TEs retrieved by TE-greedy-nester, their long terminal repeat (LTR) sequences were cut with bedtools package (Quinlan and Hall, 2010) and subsequently LTR identity was measured using global alignment by STRETCHER tool (Emboss 6.6.0; Rice et al., 2000).

3.3 Performance measures
Performance measures used here were calculated according to the formulas used in the study by Ou and Jiang (2018)

\[
\begin{align*}
\text{Sensitivity} & = \frac{TP}{TP + FN} \\
\text{Specificity} & = \frac{TN}{FP + TN} \\
\text{Accuracy} & = \frac{TP + TN}{TP + TN + FP + FN} \\
\text{Precision} & = \frac{TP}{TP + FP}
\end{align*}
\]

where true positive (TP) stands for matches of coordinates of TE found by tested tools and corresponding element given by TE-generator within tolerance ≥5% of reference TE length at the start and end positions. Correspondingly, false positive (FP) is a TE with no match with any TEs in the generated sequences, a false negative (FN) are TEs which were present in generated sequence and absent in output GFFs from given tools, true negative (TN) is estimated TE count from number of bases which are without TE in both GFF from TE-generator and also from tested tools.

3.4 Speed and RAM performance
Resource utilization of the four different software tools was compared on all the datasets used in this article. For evaluations, the programs were run on a dedicated Linux machine running no other job, using the GNU time command to obtain ‘real time’ and ‘maximum resident set size’ as an average of three runs. The machine had 12 GB physical RAM, a 4 core Intel i5 CPU. The running process was monitored for signs of memory swapping and process pruning.

### 4 Results
We developed the TE-greedy-nester software for finding nested LTR retrotransposons using a combination of a greedy algorithm and identification of full-length TEs. In contrast to comparable software relying mostly on sequence similarity, TE-greedy-nester is based on identification of structural features of LTR retrotransposons. We compared the performance of TE-greedy-nester with four other related software tools. The number of features detected by TE-greedy-nester on different types of data is reported in Table 1. We found that TE-greedy-nester detected the highest number of TEs in all tested sequences in comparison with other examined tools.

Moreover, despite the higher rate of false-positive identification, TE-greedy-nester also retrieved the highest count of TEs matched with elements present in annotated sequences (number of TEs matched, Table 1). To better evaluate the performance of TE-greedy-nester, we carried out a deeper analysis using both synthetic and biological data.

4.1 Annotated synthetic data (TE-generator and matryoshka)
To test TE-greedy-nester on synthetic data, we generated artificial sequences using TE-generator. The TE-generator sequences had medium levels of nesting. We also prepared sequences with extreme depth of nesting, only inserting new sequences into previously inserted ones and call them ‘matryoshka’. After running TE-greedy-nester on these sequences, we evaluated sensitivity, selectivity, precision and accuracy (see Section 3). Sensitivity and precision measure the ability to detect sequences, selectivity measures the ability to reject false positives and accuracy is a combination of both. Calculated comparative performance values are shown in Figure 3. While TE-greedy-nester showed higher sensitivity for all available data, its comparative accuracy gave mixed results, with lower specificity (higher false-positive rate) on synthetic data with high TE density (90%) (Fig. 3B). On the other hand, TE-greedy-nester was superior to all existing software on synthetic data with deep nesting (matryoshka). While TEnest could compete with TE-greedy-nester when provided with the proper TE database (maize) (Fig. 3C), TE-greedy-nester was the only software that could detect deep nesting correctly on mixed-origin TE data (Fig. 3D), showing an even higher accuracy on mixed data than maize data. TE-greedy-nester was also one to two orders of magnitude faster than TEnest on all datasets.

Because the above performance evaluations on annotated data partly depend on the definition of successful TE identification, we examined the effect of position tolerance on performance measures in the same four annotated synthetic datasets (Fig. 4). It can be seen that low copy datasets, such as matryoshka, produced the same number of TEs at the lowest used tolerance of 1% (Fig. 4C and D). TE rich data from TE-generator show an increased TE discovery at higher tolerance level, most likely as a result of increased false-positive rate, this was more apparent in sequences with 90% TE coverage (Fig. 4B) than in those with 10% TE coverage (Fig. 4A).

| Table 1. Number of detected or expected full-length LTR retrotransposons by TE-greedy-nester, TEnest, LTRtype and REannotate on natural (rows 1–2) and artificial (rows 3–6) testing sequences |
|------------------------------------------|--------------|---------------|--------------|---------------|--------------|---------------|---------------|
| Sequence name | Seq. length (bp) | No. of reference TEs | No. of TEs found | No. of TEs matched (tolerance = 0.01) |
|------------------------------------------|--------------|---------------|--------------|---------------|--------------|---------------|
| Zm_adh1 | 302987 | 21 | 15 | 11 | 7 | 6 |
| Zm_Chr10_2Mb | 200000 | — | 157 | 46 | 60 | 21 |
| Zm_synth_10 | 1014128 | 260 | 78 | 77 | 42 | 30 |
| Zm_synth_90 | 10271500 | 2329 | 774 | 177 | 250 | 193 |
| Zm_matryoshka_20 | 120219 | 20 | 16 | 14 | 2 | 2 |
| Os_synth_10 | 10190963 | 100 | 90 | 16 | 6 | 3 |
| Os_synth_90 | 9562856 | 900 | 729 | 71 | 44 | 24 |
| Os_matryoshka_20 | 198560 | 20 | 14 | 4 | 0 | 0 |

4.2 Artificial synthetic data (TE-generator and matryoshka)
To test TE-greedy-nester on artificial data, we generated artificial sequences using TE-generator. The TE-generator sequences had low levels of nesting. We also prepared sequences with extreme depth of nesting, only inserting new sequences into previously inserted ones and call them ‘matryoshka’. After running TE-greedy-nester on these sequences, we evaluated sensitivity, selectivity, precision and accuracy (see Section 3). Sensitivity and precision measure the ability to detect sequences, selectivity measures the ability to reject false positives and accuracy is a combination of both. Calculated comparative performance values are shown in Figure 3. While TE-greedy-nester showed higher sensitivity for all available data, its comparative accuracy gave mixed results, with lower specificity (higher false-positive rate) on synthetic data with high TE density (90%) (Fig. 3B). On the other hand, TE-greedy-nester was superior to all existing software on synthetic data with deep nesting (matryoshka). While TEnest could compete with TE-greedy-nester when provided with the proper TE database (maize) (Fig. 3C), TE-greedy-nester was the only software that could detect deep nesting correctly on mixed-origin TE data (Fig. 3D), showing an even higher accuracy on mixed data than maize data. TE-greedy-nester was also one to two orders of magnitude faster than TEnest on all datasets.

Because the above performance evaluations on annotated data partly depend on the definition of successful TE identification, we examined the effect of position tolerance on performance measures in the same four annotated synthetic datasets (Fig. 4). It can be seen that low copy datasets, such as matryoshka, produced the same number of TEs at the lowest used tolerance of 1% (Fig. 4C and D). TE rich data from TE-generator show an increased TE discovery at higher tolerance level, most likely as a result of increased false-positive rate, this was more apparent in sequences with 90% TE coverage (Fig. 4B) than in those with 10% TE coverage (Fig. 4A).
While Figure 4 shows the overall performance values of the different tools tested, their abilities to discover nested TEs remain partly hidden in the lump sum numbers. We therefore divided the counts of identified TEs based on their nesting status, including their nesting level (how many successive nesting events occurred within the given TE internal region) (Fig. 5). The most striking finding is that all tools overestimate the number of non-nested elements and underestimate the number of nested ones (Fig. 5A and B). TEnest was less prone to this error in high-density TE data (90% TEs; Fig. 5B), in accordance with its higher specificity on the same data (Fig. 3B). LTRtype and REannotate missed more TEs than the other two tools. Only TEnest and TE-greedy-nester were able to resolve the majority of the deeply nested matryoshka dataset (Fig. 5C and D). The same tendency could be seen at different nesting levels in 90% TE-generator data. TEnest and TE-greedy-nester always had much higher counts than the other tools at nesting levels II and deeper. TEnest gave higher counts than TE-greedy-nester at nesting levels IV and higher, however, most of those above nesting level VI were false positives (Fig. 5B). The best performance of TE-greedy-nester on mixed origin matryoshka data (generated with TE sequences from multiple plant species) observed in Figure 5D can be seen here as well.

For a better perspective of individual tool performance, we also show the data from this analysis in the Integrated Genome Browser (Robinson et al., 2011) (Supplementary Fig. S1). While both TE-greedy-nester and to a limited extent also TEnest had a tendency to overestimate certain types of TEs (false positives), in the overall visualization, TE-greedy-nester results render best the overall density and nesting depth distribution of TEs along the sequence.

### 4.2 Biological data

After testing on synthetic data, we applied the compared tools to biological data, namely (i) the well-studied adh1 region from maize and (ii) a 2-MB region from maize chromosome 10 used in a previous comparative study by Zeng et al. (2017).

#### 4.2.1 Adh1 region

TE-greedy-nester as well as the other three compared tools was tested on the adh1 region in which 21 full-length LTR retrotransposons (of which 12 are nested) were found by SanMiguel et al. (1998). TE-greedy-nester detected 15 (7 nested), TEnest detected 11 (1 nested), LTRtype 7 (0 nested) and REannotate 6 (1 nested) (Fig. 6). Only four of these full-length LTR retrotransposons were identified by all tools, while six were common for TEnest and TE-greedy-nester results, as can be seen in Supplementary Figure S2A.

To compare adh1 outputs from TE-greedy-nester with the original SanMiguel report on family level, TEs from TE-greedy-nester were additionally annotated using maize-specific TE database http://people.oregonstate.edu/~fowlerjo/MaizeRepeatDatabases/uniqueTEDB_1526.fasta.txt and a locally installed BLASTN (Altschul et al., 1990) tool (Supplementary Fig. S3). Although the TE annotations from TE-greedy-nester and Fedoroff (2012) do not fully match, we counted 12 families that were correctly recognized and placed within the segment.

#### 4.2.2 2 MB region of maize Chr10

We used all compared software tools for an analysis of LTR retrotransposons in the initial 2 MB region of maize chromosomes 10. The maize genome was chosen because of its high content of LTR retrotransposons that are often nested and that it was previously tested by other authors on the same sequence. The results for this maize segment (Supplementary Fig. S2B) were in line with results on 90% TE-generator data. TE-greedy-nester found the most TEs, TEnest was the most conservative in the number of non-nested TEs and LTRtype and REannotate were not able to find full-length TEs beyond nesting level I.

### 4.3 Performance tests

To compare the four evaluated software tools also by computational performance and requirements, we recorded computation times and peak physical memory usage on the data described above (Table 2). TEnest, which performed very well in nesting accuracy tests above, was the slowest and most memory hungry of the four (worst case 8719 s, 12GB RAM). With the largest datasets, it caused the system to swap memory and kill processes (shown as >12GB RAM in 2) resulting also in a steep increase of computation time. Our TE-greedy-nester was comparable with LTR Type and REannotate in terms of speed and memory usage (worst case 886 s, 628 MB). While it performed better with RAM on extremely small datasets, LTR Type used slightly more RAM (752 MB versus 500–600 MB) and was also slower to compute the results on small datasets (36s versus 13–21 s). It should be noted that TEnest was set to use four
processors, while TE-greedy-nester used multithread BLASTX searches (3 processors). LTR Type and REannotate relied on RepeatMasker output, however, the time to generate that output was included and tabulated as well. For further details, please, see Section 3.

### 4.4 Plant genomes

Finally, we applied TE-greedy-nester to 18 plant genomes available mostly from Phytozome (see Section 3) and provide the results in GFF files (http://hedron.fi.muni.cz/TEgnester/). In Table 3, we show the classification and the main characteristics of TE nesting in these 18 species. We demonstrated that 96.4% of detected TEs (i.e. 98 187 out of 10 1887), have LTR identity 80% or higher (Supplementary Fig. S4). TE-greedy-nester also found at least one protein domain in 29.8–88.4% of identified LTR retrotransposons in vascular plants but only 0–1.9% in non-vascular plants. The percentage of TEs found in nested configuration was between 19.6 and 54% in vascular plants and 0 and 17.6% in non-vascular plants. The highest nesting rates were observed in *S.bicolor* and *G.max*.

![Fig. 5. Number of correctly identified TEs at different nesting levels by the four software tools; (A) zea_10%; (B) zea_90%; (C) zea_matryoshka; (D) oryza_10%; (E) oryza_90% and (F) oryza_matryoshka](image)

![Fig. 6. Number of correctly identified TEs at different nesting levels by the four software tools in biological data; (A) zea_adh1 and (B) zea_2MB](image)
Table 2. Time and memory requirements of tested software when analyzing input data of different types

| Species   | Sequence | TE-greedy-nester | TEnest | LTRtype | REAnnotate |
|-----------|----------|------------------|--------|---------|------------|
| Z.mays   | synt10   | 174              | 83     | 297     | 338        |
|           | synt90   | 650              | 8917   | 740     | 657        |
|           | matryoshka | 16              | 544    | 26      | 11         |
|           | adh1     | 14               | 1052   | 36      | 21         |
|           | Chr10_2Mbp | 331             | 7866   | 136     | 122        |
| O.sativa | synt10   | 187              | 6562   | 322     | 330        |
|           | synt90   | 886              | 3072   | 145     | 525        |
|           | matryoshka | 13              | 57     | 36      | 15         |
| Maximum memory usage (Mbytes) |          |                  |        |         |            |
| Z.mays   | synt10   | 591.5            | 146.4  | 740.5   | 489.0      |
|           | synt90   | 628.1            | >12000 | 740.7   | 536.3      |
|           | matryoshka | 74.8            | 18.2   | 739.5   | 482.4      |
|           | adh1     | 77.1             | 351.5  | 741.2   | 482.2      |
|           | Chr10_2Mbp | 152.7           | 11800.0| 752.2   | 482.1      |
| O.sativa | synt10   | 595.7            | >12000 | 740.4   | 488.0      |
|           | synt90   | 603.7            | >12000 | 740.4   | 517.9      |
|           | matryoshka | 80.2            | 16.8   | 740.7   | 482.2      |

Note: Time and memory usage were measured as ‘real time’ and ‘maximum resident set size’ using the Linux time command on a dedicated machine with 12 GB RAM, Intel i5 processor with four cores, Fedora Linux installed and no other jobs running.

5 Discussion

Here, we present a new bioinformatic tool TE-greedy-nester for the detection of nested LTR retrotransposons that is faster and more efficient in finding deeply nested elements than existing tools. These advantages are due to the combination of structure-based retrotransposon detection with recursive sequence removal. This results in comparatively low memory usage and high computation speed, as seen in performance tests presented herein. With the default settings our tool is competitively sensitive although can produce higher rates of false positives in some instances, as seen in both synthetic and real biological data. We are exploring ways to reduce the number of false-positive calls where several directions of action are possible, such as (i) optimizing the search parameterization and scoring of candidate TEs; (ii) improving TE annotation, especially by accounting for TSD sequences that should be present in bonafide full-length TEs and (iii) abandoning the greedy approach by introducing extra passes that would pre-score elements or explore multiple sequence fragmentation scenarios.

Another advantage of TE-greedy-nester is its ability to identify nesting in different species without the need for species-specific TE databases. This is in sharp contrast to the other three compared tools that lack performance in cross-species applications. Their results also differ significantly with the size and sparsity of the used TE database.

While the focus now is on the nesting of LTR retrotransposons, the approach is modular and can be expanded to other classes of repetitive sequences by simply employing additional TE detection tools alongside LTR Finder. However, even without expansion to other TE classes, the algorithm in TE-greedy-nester can detect short foreign insertions in full-length LTR retrotransposons. This is the advantage of using structural information where the most important signal is the order of required TE components, while distance and sequence similarity is secondary.

Both synthetic and biological data were chosen to represent different TE densities and levels of nesting to identify the strengths and weaknesses of the tested tools. In this respect, we found that TE-greedy-nester had the highest sensitivity across the board of different tests. As expected, high sensitivity typically comes with a higher proportion of false positives, and so it is important to look at accuracy and precision for an objective comparison of different tools. TE-greedy-nester showed markedly lower precision with 90% TE-generator data, suggesting that it could benefit from parameter fine-tuning depending on the TE density of the data being analyzed. Identifying the best parameter combinations for different types of data is beyond the scope of this article, where only fixed or default settings were used. TE-greedy-nester also found nested full-length TEs one to two orders of magnitude faster than TEnest, which gave the best results of the three compared tools across different tests.

6 Conclusion

In this work, we present a new software tool for the recovery of full-length LTR retrotransposons fragmented by the nesting of other elements. We used a recursive approach in combination with structure-based detection of TEs with LTR Finder and implemented it in a Python tool called TE-greedy-nester. We tested this computational approach on synthetic and natural DNA sequences. Testing showed that TE-greedy-nester gave high-quality results faster than existing tools and is superior in selected parameters, especially in its ability to recover full-length LTR retrotransposons in deeply nested regions. Moreover, we analyzed 18 plant genomes and showed that TE-greedy-nester could be used in studies of TE life cycle and genome evolution, especially in areas where relative insertion times are important.

Acknowledgements

The authors thank Christopher Johnson for critical reading of the manuscript.
### Table 3. TE-greedy-nester result summary on genome sequences from organisms across the plant kingdom ordered by the detected rate of nesting

| Species           | Class      | Family              | Genome size (Mbp) | All TEs | TEs with domain<sup>a</sup> | Annotation rate (%) | Nesting Solo rate (%) | LTRs | Reported TEs |
|-------------------|------------|---------------------|-------------------|---------|-----------------------------|---------------------|-----------------------|------|-------------|
| Selaginella moellendorfii | Isoetopsida | Selaginellaceae     | 212.7             | 6704    | 3187                        | 9891                | 2489                  | 1756 | 4245        | 42.9 | 58.6 | 3247 ~941<sup>b</sup> |
| S. bicolor       | Monocots   | Poaceae             | 732.2             | 19662   | 10128                       | 29790               | 9947                  | 8479 | 18426       | 61.9 | 54    | 8706 2391<sup>c</sup> |
| G. max           | Eudicots   | Fabaceae            | 978.5             | 19325   | 11203                       | 30528               | 7128                  | 8610 | 15738       | 51.6 | 45.3 | 20266 3237<sup>d</sup> |
| M. cuminumata    | Monocots   | Musaceae            | 390.6             | 3304    | 3599                        | 6903                | 1270                  | 1842 | 3112        | 45.1 | 40.8 | 4917 ~1440<sup>e</sup> |
| P. trichocarpa   | Eudicots   | Salicaceae          | 422.9             | 4455    | 3467                        | 7922                | 1039                  | 2106 | 3145        | 39.7 | 33    | 2747 1479<sup>f</sup> |
| P. patens        | Bryopsida  | Funariaceae         | 473.2             | 3353    | 6017                        | 9370                | 2685                  | 5598 | 8283        | 88.4 | 32.4 | 24643 ~2426<sup>g</sup> |
| Brachypodium     | Monocots   | Poaceae             | 271.2             | 1668    | 2803                        | 4471                | 848                   | 1847 | 2695        | 60.3 | 31.5 | 5174 569<sup>h</sup> |
| A. thaliana      | Eudicots   | Brassicaceae        | 119.1             | 677     | 831                         | 1508                | 196                   | 429  | 625         | 41.4 | 31.4 | 422 376<sup>i</sup> |
| S. lycopersicum  | Eudicots   | Solanaceae          | 823.9             | 6895    | 9494                        | 16389               | 2770                  | 7234 | 10004       | 61   | 27.7 | 22591 2086<sup>j</sup> |
| L. japonicus     | Eudicots   | Fabaceae            | 462.5             | 1923    | 2706                        | 4629                | 490                   | 1534 | 2024        | 43.7 | 24.2 | 1974 ~8930<sup>k</sup> |
| A. lyrata        | Eudicots   | Brassicaceae        | 206.7             | 1746    | 2820                        | 4566                | 547                   | 1713 | 2260        | 49.5 | 24.2 | 11318 ~940<sup>l</sup> |
| O. sativa        | Monocots   | Poaceae             | 374.3             | 3361    | 5163                        | 8524                | 929                   | 3066 | 4135        | 48.5 | 22.5 | 6668 7043<sup>m</sup> |
| M. truncatula    | Eudicots   | Fabaceae            | 411.8             | 2685    | 4578                        | 7263                | 462                   | 1703 | 2165        | 29.8 | 21.3 | 4569 12250<sup>n</sup> |
| G. raimondii     | Eudicots   | Malvaceae           | 761.4             | 5064    | 9835                        | 14899               | 1611                  | 6620 | 8231        | 55.2 | 19.6 | 18393 3297<sup>o</sup> |
| D. salina        | Chlorophyceae | Dunaliellaceae    | 343.7             | 20002   | 4318                        | 24316               | 81                    | 378  | 459         | 1.9  | 17.6 | 5386 ~12372<sup>p</sup> |
| C. reinhardtii   | Chlorophyceae | Chlamy-domonadaceae | 107.1            | 2360    | 1356                        | 3716                | 0                     | 47   | 47          | 1.3  | 0     | 181 ~2230<sup>q</sup> |
| A. filiculoides  | Polyphyloidsa | Salvinaceae        | 750               | 238     | 270                         | 508                 | 0                     | 0    | 0           | 0    | 0    | 459 17484<sup>r</sup> |
| M. pusilla       | Mamiello   | Mamiellaceae        | 22                | 20      | 41                          | 61                  | 0                     | 0    | 0           | 0    | 0    | 116 ~65<sup>s</sup> |

<sup>a</sup>Counted from TEs with annotated protein domain.

<sup>b</sup>Vanburen et al. (2018).

<sup>c</sup>Jiang and Ramachandran (2013).

<sup>d</sup>Du et al. (2010).

<sup>e</sup>Hribova et al. (2010).

<sup>f</sup>Cossu et al. (2012).

<sup>g</sup>Lang et al. (2018).

<sup>h</sup>Stritt et al. (2019).

<sup>i</sup>Peterson-Burch et al. (2004).

<sup>j</sup>Xu and Du (2014).

<sup>k</sup>Holligan et al. (2006).

<sup>l</sup>Civan et al. (2011).

<sup>m</sup>Wang and Liu (2008).

<sup>n</sup>Xu et al. (2017).

<sup>o</sup>Goodstein et al. (2012).

<sup>p</sup>Li et al. (2018).

---

**Funding**

This research was supported by the Czech Science Foundation [grant number 18-00258S to E.K. and M.L.].

**Conflict of Interest:** none declared.

**References**

Ahmed, M. and Liang, P. (2012) Transposable elements are a significant contributor to tandem repeats in the human genome. *Comp. Funct. Genomics*, 199, 1–7.

Altschul, S.F. et al. (1990) Basic local alignment search tool. *J. Mol. Biol.*, 215, 403–410.

Bergman, C.M. and Quesneville, H. (2007) Discovering and detecting transposable elements in genome sequences. *Brief. Bioinform.*, 8, 382–392.

Civan, P. et al. (2011) On the coevolution of transposable elements and plant genomes. *J. Bot.*, 2011, 893546.

Cossu, R.M. et al. (2012) A computational study of the dynamics of LTR retrotransposons in the *Populus trichocarpa* genome. *Tree Genet. Genomes*, 8, 1–7.

Du, J. et al. (2010) SoyTEdb: a comprehensive database of transposable elements in the soybean genome. *BMC Genomics*, 11, 113.

Ellinghaus, D. et al. (2008) LTRharvest, an efficient and flexible software for de novo detection of LTR retrotransposons. *BMC Bioinformatics*, 9, 18.

Fedoroff, N.V. (2012) Presidential address. Transposable elements, epigenetics, and genome evolution. *Science*, 338, 758–767.

Gao, C. et al. (2012) Characterization and functional annotation of nested transposable elements in eukaryotic genomes. *Genomics*, 100, 222–230.

Goermer-Potvin, P. and Bourque, G. (2018) Computational tools to unmask transposable elements. *Nat. Rev. Genet.*, 19, 688–704.

Goodstein, D.M. et al. (2012) Phytome: a comparative platform for green plant genomics. *Nucleic Acids Res.*, 40, D1178–1186.

Gremme, G. et al. (2013) GenomeTools: a comprehensive software library for efficient processing of structured genome annotations. *IEEE/ACM Trans. Comput. Biol. Bioinform.*, 10, 645–656.
Hirochika, H. (1997) Retrotransposons of rice: their regulation and use for genome analysis. *Plant Mol. Biol.*, 35, 231–240.

Holligan, D. et al. (2006) The transposable element landscape of the model legume *Lotus japonicus*. *Genetics*, 174, 2215–2228.

Hribo, E. et al. (2010) Repetitive part of the banana (*Musa acuminata*) genome investigated by low-depth 454 sequencing. *BMC Plant Biol.*, 10, 204.

Jiang, S.-Y. and Ramachandran, S. (2013) Genome-wide survey and comparative analysis of LTR retrotransposons and their captured genes in rice and sorghum. *PLoS One*, 8, e71118.

Kapitonov, V.V. and Jurka, J. (1999) Molecular paleontology of transposable elements from *Arabidopsis thaliana*. *Genetica*, 107, 27–37.

Kronmiller, B.A. and Wise, R.P. (2013) TEnest 2.0: computational annotation and visualization of nested plant transposable elements. *Plant Physiol.*, 164, 45–59.

Kronmiller, B.A. and Wise, R.P. (2008) TEnest: automated chronological annotation and visualization of nested transposable elements. *Methods Mol. Biol.*, 1057, 303–319.

Lang, D. et al. (2018) The *Physcomitrella patens* chromosome-scale assembly reveals moss genome structure and evolution. *Plant J.*, 93, 515–533.

Li, X. et al. (2008) A novel genome-scale repeat finder geared towards transposons. *Bioinformatics*, 24, 468–476.

Li, F.W. et al. (2018) Fern genomes elucidate land plant evolution and cyanobacterial symbioses. *Nat. Plants*, 4, 460–472.

McCarthy, E.M. and McDonald, J.F. (2003) LTR_STRUC: a novel search and identification program for LTR retrotransposons. *Bioinformatics*, 19, 362–367.

Nussbaumer, T. et al. (2013) MIPS PlantsDB: a database framework for comparative plant genome research. *Nucleic Acids Res.*, 41, D1144–D1151.

Ou, S. and Jiang, N. (2018) LTR_retriever: a highly accurate and sensitive program for identification of long terminal repeat retrotransposons. *Plant Physiol.*, 176, 1410–1422.

Peitera, V. (2008) Automated paleontology of repetitive DNA with REannotator. *BMC Genomics*, 9, 614.

Peterson-Burch, B.D. et al. (2004) Genomic neighborhoods for Arabidopsis retrotransposons: a role for targeted integration in the distribution of the Metaviridae. *Genome Biol.*, 5, R78.

Quinlan, A.R. and Hall, I.M. (2010) BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics*, 26, 841–842.

Rice, P. et al. (2000) EMBOSS: the European Molecular Biology Open Software Suite. *Trends Genet.*, 16, 276–277.

Robinson, J.T. et al. (2011) Integrative genomics viewer. *Nat. Biotechnol.*, 29, 24–26.

Saha, S. et al. (2008) Computational approaches and tools used in identification of dispersed repetitive DNA sequences. *Trop. Plant Biol.*, 1, 85–96.

SanMiguel, P. et al. (1998) The paleontology of intergene retrotransposons of maize. *Nat. Genet.*, 20, 43–45.

Smit, A.F. (1999) Interspersed repeats and other mementos of transposable elements in mammalian genome. *Curr. Opin. Genet. Dev.*, 9, 657–663.

Stitzer, M.C. et al. (2019) The genomic ecosystem of transposable elements in maize. *eLife*, 8, doi:10.1101/559922.

Stritt, C. et al. (2019) Diversity, dynamics and effects of long terminal repeat retrotransposons in the model grass *Brachypodium distachyon*. *N. Phytol*, 10.1111/nph.16308, 1–13.

Thorvaldsdottir, H. et al. (2013) Integrative Genomics Viewer (IGV): high-performance genomics data visualization and exploration. *Brief. Bioinform.*, 14, 178–192.

Valencia, J.D. and Gregis, H.Z. (2019) LtrDetector: a modern tool-suite for detecting long terminal repeat retrotransposons de-novo on the genomic scale. *BMC Genomics*, 20, 450.

Vanburen, R. et al. (2018) Extreme haplotype variation in the desiccation-tolerant clubmoss Selaginella lepidophylla. *Nat. Commun.*, 9, 8.

Vicent, C.M. and Casacuberta, J.M. (2017) Impact of transposable elements on polyploid plant genomes. *Ann. Bot. Lond.*, 120, 195–207.

Wang, H. and Liu, J.S. (2008) LTR retrotransposon landscape in *Medicago truncatula*: more rapid removal than in rice. *BMC Genomics*, 9, 382–382.

Xu, Y. and Du, J. (2014) Young but not relatively old retrotransposons are preferentially located in gene-rich euchromatic regions in tomato (*Solanum lycopersicum*) plants. *Plant J.*, 80, 582–591.

Xu, Z. et al. (2017) GTEdb: the first web-based database of transposable elements in cotton (*Gossypium raimondii*). *Database*, 2017, 1–7.

Xu, Z. and Wang, H. (2007) LTR_FINDER: an efficient tool for the prediction of full-length LTR retrotransposons. *Nucleic Acids Res.*, 35, W265–268.

Zeng, F. et al. (2017) LTRtype, an efficient tool to characterize structurally complex LTR retrotransposons and nested insertions on genomes. *Front. Plant Sci.*, 8, 402.