11-17-2022

Whole genome duplication analysis of the invasive Lonicera maackii (Amur honeysuckle)

Gaochan Wang
gw1956@rit.edu

RIT

Whole genome duplication analysis of the invasive *Lonicera maackii* (Amur honeysuckle)

By

Gaochan Wang

A thesis Submitted in Partial Fulfillment of Requirements for the Degree of Master of Science in Bioinformatics

Department of Bioinformatics

College of Science

Thomas H. Gosnell School of Life Science

Rochester Institute of Technology

Rochester NY

November 17th, 2022

Rochester Institute of Technology
Thomas H. Gosnell School of Life Sciences
Bioinformatics Program

To: Head, Thomas H. Gosnell School of Life Sciences

The undersigned state that Gaochan Wang, a candidate for the Master of Science degree in Bioinformatics, has submitted his thesis and has satisfactorily defended it.

This completes the requirements for the Master of Science degree in Bioinformatics at Rochester Institute of Technology.

Thesis committee members:

Name **Date**

____ ____________________
Michael V. Osier, Ph.D.
Thesis Advisor

____ ____________________
Andre O. Hudson, Ph.D.

____ ____________________
Susan S. Pagano, Ph.D.

____ ____________________
Eli J. Borrego, Ph.D.

____ ____________________

Abstract

Invasive *Lonicera maackii* (*L. maackii*) is one of the highly successful and problematic bush honeysuckles found in the central and eastern of United States of America, which has been reported to pose a threat to native ecosystems by decreasing biodiversity. The mechanism by which *L. maackii* negatively impact environments is typically through either the direct effect of increased dominance or the indirect effect of territory modification. Numerous studies have documented the negative effects of *L. maackii* on native biota and the key traits such as seed dispersal, phenology, resistance to herbivory, rapid growth and environmental plasticity that contribute to invasion of *L. maackii*. In past decades, the studies mainly focused on negative effects and management of *L. maackii* invasion, and little was done to explore the genetic traits contributing to devastate the native ecosystem. Chloroplast-based genomic and chemical diversity in *L. maackii* has been reported. However, the whole genomic diversity in *L. maackii* has not been reported due to the availability of whole genome sequence of *L. maackii*. The advances in whole genome sequencing technologies and bioinformatic tools allow for studying the genomic diversity of *L. maackii* at the whole genome level. Genome duplication is a key evolutionary mechanism providing new genetic materials and new gene functions for plants, which play important roles in speciation and adaptation to biotic/abiotic stress. Given the fact that *L. maackii* is closely related to *L. japonica*, and whole genome duplication of *Lonicera japonica* (*L. japonica*) has been reported (Pu et al., 2020; Yu et al., 2022), we hypothesize that a whole genome duplication is present in *L. maackii*. In this study, we aim to investigate whether there is a genome duplication in *L. maackii* with the purpose of exploring the genomic diversity in *L. maackii*. We also conducted a comparison of genome duplication among the species in *Lonicera* genus. With the completion of whole genome assembly of *L. maackii* (Kesel et al., 2022), we conducted the gene prediction using Exonerate and gene duplications analysis using MCScanX in *L. maackii*. As a result, we predicted 32,642 genes and identified 5,668 genes, 24,911 genes, 703 genes, 902 genes, and 458 genes deriving from Singleton, Dispersed, Proximal, Tandem, and WGD modes, respectively. To our knowledge, this is the first genome duplication analysis that has been reported in *L. maackii*. Compared to *L. japonica*, a higher prevalence of Singleton and Dispersed modes of gene duplication was observed in *L. maackii*. The different genome duplication patterns between *L. maackii* and *L. japonica* may result from the difference of whole genome assembly format. The future directions should focus on improving the chromosome-scale genome assembly and whole genome annotation, promoting our understanding on the genome diversity and evolutionary traits in *L. maackii* and controlling the expansion of *L. maackii*.

Table of Contents

List of Figures

List of Tables

Acknowledgement

This study was conducted under the supervision of Dr. Michael Osier. I would like to thank Dr. Osier for his help and guidance throughout the project and the completion of my thesis.

I would like to thank my committee members, Dr. Eli Borrego, Dr. Andre Hudson, and Dr. Susan Pagano, for their critical suggestions and comments on the project and thesis.

Introduction

Invasive Lonicera maackii

Lonicera maackii (*L. maackii*) and other several species of bush honeysuckles found in central and eastern of North America (**Figure 1**) are characterized as one of the most problematic invasive species (Kang et al., 2018), which can cause harm to environment, the economy and human health (Kumar Rai and Singh, 2020). The global trade was one of important causes that introduced invasive plants into a new environment where they become established and their negative effects will likely accelerate in the future (Oswalt CM et al., 2015). Invasive plants have been a problem in the United States for years. A study report found that invasive plants are more widely distributed than non-invasive plants across the continental United States (Bradley et al., 2015). Another recent study estimated that the economic losses resulting from biological invasion reaches 26.8 billion US dollars per year in the United States (Fantle-Lepczyk et al., 2022). Invasive plants have intrinsic advantages, broader climate tolerances and stronger competitive abilities, to out-compete native plant species. Of the invasive plants, invasive shrubs are a particular threat to eastern forests in North America (Webster et al., 2006). Unlike plants in North America, it was reported that shrubs originating from East Asia did not experience thousands of years of shortened growing seasons and are better able to utilize shoulder seasons during the longer growing season of the Holocene (Fridley, 2012), which contributes to the invasion of these shrubs when introduced into North America.

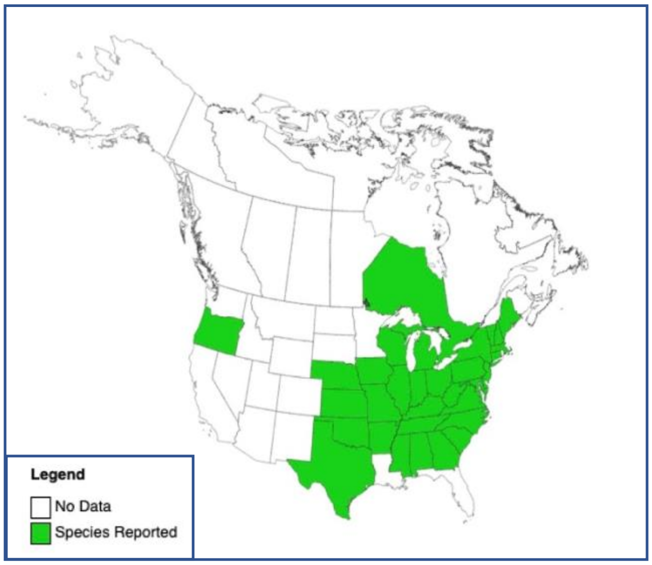

Figure 1. Distribution of *L. maackii* by state in North America. (Source: EDDMapS)

L. maackii ([Rupr.] Herder, Caprifoliaceae), also known as Amur Honeysuckle with white to yellow flowers (**Figure 2**), is a large, upright shrub native to China, Japan, and Korea (Rocha et al., 2014). This species of honeysuckles was first imported into North America as soil stabilization and as a food source for wildlife. However, *L. maackii* was proven to be less effective for either purpose due to its shallow roots and low-nutrition berry fruit. After introduction into North America, this species proves excellent at escaping and expanding its range.

L. maackii is a highly successful invasive shrub (McNeish and McEwan, 2016). Plenty of work has been conducted aiming to address the invasiveness of *L. maackii*. A previous report stated that the phenotype traits of *L. maackii* contribute to their invasion (McNeish and McEwan, 2016). Its rapid growth and plasticity in response to changing environmental conditions contribute to successful colonization in new habitats.

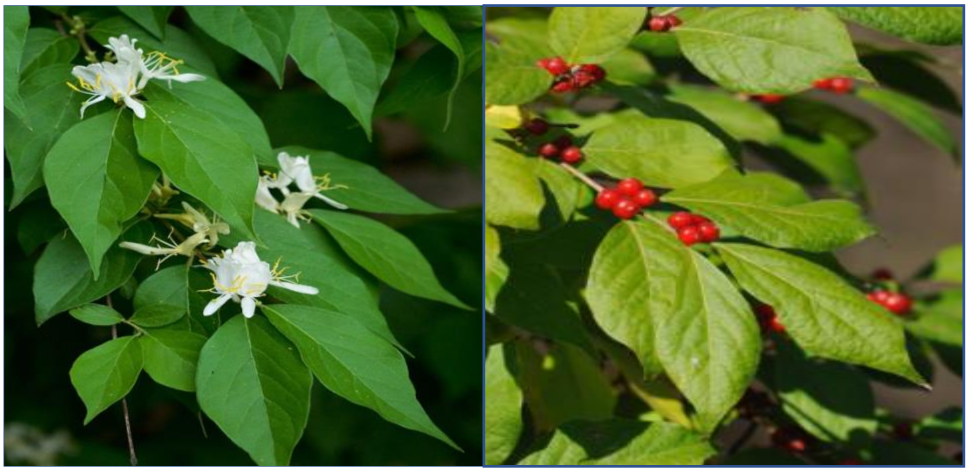

Figure 2. *L. maackii* flower (left) and fruit (right). (Source: New York Invasive Species Information)

L. maackii also has a highly competitive growth pattern. It grows faster than native plant species and can overtake the habitat by forming a dense shrub layer that crowds and shades out native plant species (McNeish et al., 2017; Mounger et al., 2021). Seeds and fruit traits (**Figure 2**) can be relevant for invasiveness since they are related to *L. maackii* dispersal, germination, and fitness. It was documented that *L. maackii* is a plant of long-distance dispersal and abundant propagule production (McNeish and McEwan, 2016). The seeds of *L. maackii* can germinate in various light, temperature, and soil condition. Consumption of *L. maackii* berries by birds, white-tailed deer, and possibly small mammals promotes the long-distance dispersal of its seeds. *L. maackii* also has a longer growing season. The development and expansion of its leaves take place 2 to 3 weeks earlier than native plants and its leaves are resistant to cold condition and the leaf abscission is later than native species. The accumulated studies have demonstrated that *L. maackii* can exhibit biochemical effects on predators through production of allelochemicals. Its leaf extracts contain two main allelopathic chemicals – luteolin and apigenin derivatives. Its root and shoot extracts were shown to reduce germination of several native herbaceous plants.

How does the ecosystem respond to *L. maackii* invasion? Upon the successful invasion, *L. maackii* produce negative effects on native plants, ecosystem, and human health (McNeish and McEwan, 2016). Numerous studies demonstrated that *L. maackii* can outcompete native species by its ability acquire nutrients, grow, and reproduce resulting in decreased biodiversity. The dense shrub and berries provide poor habitats and food to birds and other insects, resulting in an ecological trap that can reduce avian success. The carotenoid (rhodoxanthin) pigment (Jones et al., 2010) identified from non-native *L. morrowii* and *L. tatarica* berries can brighten and redden the color of birds' feather, which caused plumage in Yellow-breasted Chat, Kentucky Warbler, White-throated Sparrow, Baltimore Oriole and Northern Flicker (Hudon and Mulvihill, 2017). Genome annotation (Kesel, 2021) suggested that *L. maackii* may produce rhodoxanthin. Putative carotenoid cleavage proteins were detected by using BLAST and exonerate annotation methods. Therefore, further study using such as genome annotation and biochemical methods may contribute to answer whether *L maackii* can produce rhodoxanthin and influence bird plumage color. There is other evidence indicating *L. maackii* invasion can affect the abundance and survival of birds and amphibians. For example, a study led by Packett and Dunning found a positive correlation between bird density and Amur Honeysuckle density (Packett and Dunning, 2009).

Furthermore, another study found that sites dominated by Amur Honeysuckle had higher density of understory birds and lower density of canopy birds compared to sites that contained native shrub and saplings. In human, *L. maackii* has negative consequences on human-related disease vectors such as mosquito (Gardner et al., 2017; Shewhart et al., 2014).

Prevention and rapid response are the most effective attitude to control invasive species from establishing in the first place. Once established, control effort could become less effective. Currently, the prevention and control methods are mostly mechanical and chemical methods. Hand removal of Amur Honeysuckle seedlings and young plants is a regular option where the population are at lower level. Systemic herbicides can be employed if the shrub grow in full sun, or the bush honeysuckle population is large. Glyphosate and triclopyr have been used to control *L. maackii* population. It is also suggested that prescribed burning is an option to control *L. maackii* growing in open habitats. Given the fact that *L. maackii* can spread rapidly through birds eating seeds and guanos containing eaten seeds, control of *L. maackii* can be not only expensive but also time-consuming, and sometimes inefficient. It is suggested that effective control of *L. maackii* should be started in late summer or early fall before the seeds are ready for dispersal (McNeish and McEwan, 2016). Herbicides application is an example of chemical methods for controlling *L. maackii*. Foliar, stem injection, and cut stump application of herbicides are the common practices. Basal bark herbicide treatment of *L. maackii* was reported (Kleiman et al., 2018). A recent study on exploring the application of Glyphosate and an herbicide adjuvant derived from fungi on controlling *L. maackii* was reported (Rivera et al., 2022). However, the effectiveness of herbicide applications depends on the size and distribution of *L. maackii (Rathfon and Ruble, 2007)*. Combination of chemical and mechanical methods could be more effective in removal of *L. maackii.* Like the presence of resistance to antibiotic, invasive plants can develop resistance to herbicides. Biological controls could be an opportunity to manage *L. maackii* population. Classical biological controls are characterized using herbivores or pathogens to reduce or maintain densities of target plants below some threshold levels, and biological control of invasive plants were reported (Seastedt, 2015). However, there are still concerns about the biological controls. So far, there are no known biological controls of *Lonicera spp.*

Genetic analysis of invasive plants and its applications

Genetic analysis of invasive species is of great interest to study its success in new habitat, which could provide insight into mechanism of invasions. Previous studies suggested that the invasion success might depend more heavily on the ability of invasive species to response to natural selection (EunmiLee, 2002). The founder effect explains that invasive species have a low intra-population genetic diversity but have a high inter-population differentiation in introduced ranges compared to those of the region of its origin (Maebara et al., 2020). But the degree of genetic diversity and differentiation of introduced populations varies for each invasion event. Some populations of invasive species lose genetic diversity during invasion through founder effects (Dlugosch and Parker, 2008), but many have higher genetic diversity outside their native range (Wilson et al., 2009), which could result from interbreeding among divergent source populations (Rius and Darling, 2014), hybridization (Parepa et al., 2014), rapid mutation

(Exposito-Alonso et al., 2018), and exposure of cryptic genetic variation (Dlugosch et al., 2015). Such increases in genetic diversity can enhance colonization success and adaptive potential in invasive species. Genetic analysis of invasive plant populations has many applications such as predicting population response to biological or chemical control measurement based on diversity levels, identifying source populations, tracking introduction routes, and elucidating mechanisms of local spread and adaptation. This information can be used to develop more effective target strategies for managing existing plant invasions and preventing new ones (Teem et al., 2020).

Rapid evolution could also contribute to invasions (Molina-Montenegro et al., 2018). Based on Darwin's theory of natural selection, the individuals with heritable traits better suited to the environment will survive. Consequences of natural selection differ depending biologically on genetic structure. Many invasive species rely on rapid evolutionary change to adapt to their environments, which is accompanied by mechanisms like hybridization and polyploidy. Interspecific hybridization results in new genetic combinations that can be acted upon by natural selection and is a well-used mechanism of adaptive rapid evolution, especially in invasive plants. Hybridization increases genetic diversity and overall fitness through the generation of novel phenotypes. Polyploidy, also known as WGD, resulting in more than two sets of chromosomes in the genome, is a common phenomenon in plants, suggesting evolutionary advantage. In plants, it is usually associated with tolerance to a broad range of ecological conditions and has also been linked to higher levels of asexual reproduction, an increased resistance to pathogens, and changes to seed germination and dormancy. These characteristics, along with an increase in genetic diversity, can greatly influence the fitness of polyploid invaders.

In addition, a third possible explanation for plant invasion is related with a high pre-existing genetic diversity in the native range and the later filtering of genotypes that could be locally pre-adapted in the new range. Whole genome sequencing contributes to understand the complete gene information, the regulatory elements that control the functions of genes, genetic diversity within and between species, and identification and tracking of genetic variants.

Protein-coding gene prediction

Rapid and cost-effective next-generation sequencing (NGS) technologies provide large volumes of DNA sequencing data. Generating an assembly for a plant species is merely the first step in the elucidation of the genome. In order to better study the genome diversity for a given species, genome annotation is essentially important in identifying the functional elements from whole genome. The prediction of protein-coding genes is one of the most critical steps in genome annotations. Dozens of tools or pipelines have been developed for gene prediction. So far, three gene prediction methods (*ab initio* gene prediction, homology-based gene prediction, and transcriptome-based prediction) have been documented (Ejigu and Jung, 2020; Keilwagen et al., 2018; Scalzitti et al., 2020).

Ab initio gene prediction is an intrinsic method that utilizes the properties of DNA sequences alone to predict locations of genes, which relies on two types of sensors: signal and content sensors (Wang et al., 2004). Signal sensors refer to short sequence motifs, such as splice sites, branch points, polypyrimidine tracts, start codons and stop codons. Content sensors exploit the coding versus non-coding sequence features, such as exon or intron lengths or nucleotide composition. *Ab initio* methods usually use statistical models, such as Support Vector Machines, hidden Markov models or Neural Network (Scalzitti et al., 2020). *Ab initio* gene predictors such as Genscan (Burge and Karlin, 1997), GlimmerHMM (Salzberg et al., 1999), Augustus (Stanke and Waack, 2003), and GeneMark-ES (Lomsadze et al., 2005) can be used to identify previously unknown genes or genes that have evolved beyond the limits of similarity-based approaches. However, Automatic *ab initio* gene prediction algorithms often make substantial errors and result in inaccurate subsequent analyses such as functional annotations, identification of genes involved in important biological process, evolutionary studies. This is especially true in the case of large “draft” genomes, where the researcher is generally faced with an incomplete genome assembly, low coverage, low quality, and high complexity of the gene structures. Other important challenges that have attracted interest recently include the prediction of small proteins/peptides coded by short open reading frames (sORFs) or the identification of events such as stop codon recoding (Scalzitti et al., 2020). These atypical proteins are often overlooked by the standard gene prediction pipelines, and their annotation requires dedicated methods or manual curation.

Based on molecular evolution principle, gene sequences that are useful for survival and other crucial functions are conserved, especially in closely related species (Ejigu and Jung, 2020). Homology-based gene prediction exploits this fact to predict genes or transcripts in a newly sequenced target genomes by identifying significant matches (similarity to known genes) from reference genomes, typically the well annotated genomes by using alignment tools such as BLAST. But, increased evolutionary distance between the target species of interest and the reference species reduces the accuracy of homology-based gene finding. Furthermore, BLAST does not explicitly account for the exon–intron structure of genes. Searching for proteins or coding sequences in complete genomes, long and variable introns might be a problem for BLAST. To circumvent this problem, several approaches have been proposed for combining smaller, local hits of high similarity to parts of a given gene into larger, complete gene models. Exonerate (Slater and Birney, 2005), Genewise (Birney and Durbin, 2000), GeneMapper (Chatterji and Pachter, 2006) and GeMoMa (Keilwagen et al., 2019) are tools developed for homology-based gene prediction. Exonerate is a highly complete generic tool for pairwise sequence comparison as well as exon prediction. It can use various alignment models, exhaustive dynamic programming, or a variety of heuristics, depending on data types. For example, the protein2genome model allows alignment of a protein sequence to genomic DNA, which is similar to those used by Genewise. All models contain a built-in intron model to account for the spliced introns in the alignments.

Apart from *ab initio* and homology-based methods, transcriptome-based gene prediction is another useful method for gene prediction by using RNA-seq technology (Grandaubert et al., 2015). The expression data, in the form of ESTs and cDNAs, represents the sequences of spliced mRNAs found in the living cells. After mapping the RNA-seq data to the genome of interest via established tools, such as HISAT2 or Stringtie, the resulting matched RNA-seq reads can be used to identify genes by a gene finder. Tools utilizing transcriptome tend to be among the most accurate gene finder. However, transcriptome-based methods suffer from some practical problems. For example, due to the length of RNA-seq reads, they usually cover only a portion of a gene. Assembling the matched overlapping reads may result in false positive gene predictions that are not present in the transcriptome.

In summary, gene prediction suffers from several challenges, such as the sequencing errors in raw data, the quality of the assembled sequence, overlapping genes, or handling short reads. Combination of three methods may produce more accurate predicted genes.

Gene duplications and detection methods

Unlike other eukaryotic genomes, plant genome structure is complex and contains many types of sequences including non-coding sequences that may have important roles in genome functions and regulations. Moreover, plant genomes tend to evolve at higher rate, leading to higher genome diversity and creation of genetic novelties that contribute to evolve and adapt to changing environment. Gene duplications is one type of genomic changes that can lead to evolutionary novelties and development of new functions in plants. For example, gene duplication has been reported to play important roles in nutrient transport under stress conditions, in protection against heat, cold, or salty environments, in the resistance to drugs and pesticides, but also in the adaptation to domestication.

In a genome, duplicated genes can be generated by various mechanisms. So far, five types of duplications are documented based on mechanisms from previous studies. In the first mechanism, duplicated genes arise from whole genome duplication (WGD), leading to two copies of each gene from the genome. This type of duplication has been well documented in plants and defined as polyploidization, which is an essential source of genetic novelty that can lead to evolutionary innovations (Panchy et al., 2016; Qiao et al., 2019). Plants are especially prone to experience polyploidization. It was estimated that several WGD events took place during the evolution of plant species. The consequence of polyploidization is the large genome size in plants. Tandem duplication occurring at a smaller scale is the second mechanism, which creates an additional copy of a gene next to it producing tandemly arrayed genes (TAGs). The molecular mechanism of tandem duplications is unequal crossing over, which can produce regions containing one or more several genes. These unequal crossing overs result from homologous recombination between sequences non-homologous recombination by replication-dependent chromosome breakage. When multiple unequal crossing overs happen, it might lead to the increase or decrease of copy numbers in gene families. Transposable elements (TEs)-mediated duplication is the third mechanism. TEs are repeated sequences with the ability to move from one position to another along and across the chromosome. There are two TE-mediated mechanisms promoting the generation of duplicated genes: the retro-position and the trans-duplication. The retro-position mechanism consists of the reverse transcription of a messenger RNA from a host gene into a cDNA then inserted in another location of the genome by the action of the enzymes of a retrotransposon. The trans-duplication happens when DNA transposons incorporate un-spliced fragments of different genes-, although the true mechanism is still unknown. Segmental duplication is defined as long stretches of duplicated sequences that can span between 1 to 200 kb and share a sequence identity higher than 95%. The segmental duplications result from the replicative transpositions of small portion of chromosomes. However, the exact mechanism is still unclear.

After their formation, duplicated genes are subjected to different fates. Some duplicated genes might not be essential to cell function and will degenerate to pseudogenes or complete deletion. The pseudogenes can be conserved in genomes. For example, *Arabidopsis thaliana* and the rice contain many pseudogenes in their genomes. While some duplicated genes may evolve novel functions via neo-functionalization, or sub-functionalization. Mutations can provide a new allele giving rise to new functions for the genes. If these functions are beneficial, they will be subjected to fixation in the population through distinct selection pressures, which is a process termed neo-functionalization. A wide transcriptomic analysis in maize estimated that 13% of all gene pairs generated by WGD have been submitted to regulatory neo-functionalization in leaves. In addition to pseudogenes and neo-functionalization, the duplicated genes are also subjected to sub-functionalization. In this process, the subdivision of the ancestral gene function among the duplicated genes results in the different expression patterns. Although the mechanisms that form duplicated genes affect the fates of duplicated genes, gene function is a vital factor that determine the fate of the duplicated gene. In plants, most duplicated genes were derived from WGD and tandem duplication. Moreover, the mechanisms and fates of duplicated genes make bioinformatic identification of duplicated genes more difficult.

Given the importance of duplicated genes in evolution and adaptation of plants, many bioinformatic tools have been developed for identification of duplications within and between genomes. There are many variations among these tools ranging from the aims, computation costs to use of these tools. In summary, there is no stand-alone tool that can solve all the problems and the use of these tools depends on computer skills but also on the genomes being compared and the biological questions being asked. So far, 3 types of algorithms used by duplication detection tools are reported: paralog detection, detection of syntenic blocks, and detection of tandem arrayed genes. Many bioinformatic tools have been developed for this purpose and these approaches can be used for different aims and computation cost (Lallemand et al., 2020). For example, some of them are more suitable to identify a particular duplication event, some of them are more optimized for large genome, which can handle multiple genomes or deal with genome that has undergone multiple duplication and rearrangement events. The more details about the duplication detection tools can be found from a review paper (Lallemand et al., 2020). MCScanX is one of the most used tools aiming at searching syntenic blocks. Plant Duplicate Gene Database (plantDGD, http://pdgd.njau.edu.cn:8080) contains the types of gene duplications in 141 sequenced plant species including *Arabidopsis thaliana* and *Solanum lycopersicum*.

The MCScanX algorithm (Wang et al., 2012) containing 3 core programs and 12 downstream programs was developed from MCScan (Tang et al., 2008). Whole-genome BLASTP results are used to compute collinear blocks for all possible pairs of chromosomes and scaffolds. The analysis occurs in three main steps. The first step uses the results of an all-against-all comparison using BLASTP to find collinear blocks. BLASTP matches are sorted based on their genomic position. To handle tandem regions, all consecutive genes with a BLASTP match that are separated by less than five genes, are collapsed into a single representative. Then, the highest scoring chains of collinear gene pairs are searched for using dynamic programming. Non-overlapping chains involving at least five collinear gene pairs are saved. In a pair of collinear blocks, two distinct genomic locations with aligned collinear genes are assigned as anchors. The second step is to determine the modes of gene duplications, all genes are first assigned to the singleton mode. Genes with BLASTP hits to other genes are assigned to the dispersed duplicates mode. If the hits are close enough, they are assigned to the proximal duplicate mode. If the hits are neighboring, they are assigned to the tandem duplicate mode. To finish, anchored genes are assigned to the WGD/segmental mode. In the last step, twelve downstream analyses can be performed using different scripts and correspond to the computation of the nonsynonymous and synonymous rates (Ka and Ks), the generation of various plots, the construction of gene families with associated analyses, the detection of collinear tandem arrays, the computation of the number of intra- and inter-species collinear blocks at each locus of reference genomes, and the display of statistics on gene numbers at different duplication depths. MCScanX was widely used, and tool based on MCScanX algorithm was reported (Qiao et al., 2019).

Gene copy number variations and detection

The gene copy number is the number of copies of a particular gene in the genotype of an individual. In the plant genome, copy numbers of specific genes may vary because of deletions or duplications, which is a universal biological phenomenon named copy number variation (CNV). CNV is genomic rearrangements resulting from deletion, insertion, and duplication at specific regions on certain chromosome. The defined minimum length of a CNV is typically 1 kb, although many studies include smaller variants of as few as 50 base pairs (bps). Several mechanisms have been postulated to explain the formation of CNVs. The first mechanism is known as non-homologous end joining (NHEJ). NHEJ requires very low level of sequence similarity at the breakpoints and results from aberrant repair of uneven double-stranded breaks produced by DNA damage (Dolatabadian et al., 2017). The second proposed mechanism is non-allelic homologous recombination (NAHR) between DNA segments. Unlike NHEJ, NAHR requires high sequence similarity at the breakpoints (Dolatabadian et al., 2017). The third mechanism is fork stalling and template switching (FoSTeS) that is caused by DNA replication errors (Zmienko et al., 2014). FoSTeS events may generate insertions, deletions, and more complex rearrangements. Other mechanisms such as single-strand annealing, transposable elements and polyploidization were reported to produce CNVs (Hastings et al., 2009). CNV was first studied in human followed by animals and has been reported to play important roles in many human disorders and promoting tumorigenesis and resistance to drugs (Dolatabadian et al., 2017; Shlien and Malkin, 2009; Zhang et al., 2009). In contrast, CNVs in plants have not been so thoroughly studied. However, the massive genomic data suggests the existence of CNV and highlighted the contribution of CNV to natural diversity on the genomic level. CNVs are generally thought to be deleterious. Deletion CNVs can lead to loss of function (LOF), whereas duplication CNVs affecting entire protein-coding genes can be deleterious if they affect dosage-sensitive genes. On the other side, CNVs also appear to be an important driver that can enable adaptation under the natural selection pressure. One of the well-known examples is herbicide-resistance glyphosate identified in Palmer amaranth (Zmienko et al., 2014). Increased copy number of EPSPS (5-enolpyruvylshikimate3-phosphate synthase) gene was found to associate with increased EPSPS transcript and protein levels as well as increased glyphosate dose survival rate. The higher production of EPSPS enzyme due to the increased gene copy number enables those plants to overcome the inhibitory effect of glyphosate, most likely by providing enough enzyme molecules to bind the physiological substrate PEP, even in presence of glyphosate. Another example is that CNVs have been found to be associated with nucleotide binding leucine-rich repeat (NB-LRR) genes and receptor-like kinase (RLK) genes, known to be involved in plant defense-related mechanisms (Dolatabadian et al., 2017). Over the years, accumulating studies have demonstrated that CNV was involved in control of reproduction, insect/disease resistance, RNA interference, responses to environmental stress (Lye and Purugganan, 2019; Zmienko et al., 2014). These discoveries shed light on modern agriculture, as they can potentially provide guidance for plant genetic modifications.

At present, several methods have been developed for detect CNVs (Dolatabadian et al., 2017; Mounger et al., 2021): quantitative and digital PCR, in situ fluorescent hybridization, the paralogue ratio test, multiplex amplifiable probe hybridization, and multiplex ligation-dependent probe amplification. Most of these methods are capable of high-throughput genotyping of a particular variant in multiple DNA samples. However, they are not suitable for a genome-scale analysis and have limited use in CNVs discovery. NGS data provide an opportunity for a genome-wide study of CNVs in plants. Two genome-scale methods, array-based comparative genome hybridization (CGH) and reference genome-based NGS, were widely reported. CGH is based on the comparison of fluorescence signals of a test and reference sample hybridized to a microarray of tiled probes covering an entire genome. The use of smaller probes increases the specificity of CNV detection in this method. However, CGH is more accurate in detecting deletions than duplications. NGS-based methods allow for the detection of CNVs using short read sequencing data. Three NGS-based approaches are commonly used in CNV detection: the read depth (RD) approach, the read pair (RP) approach, and the split read (SR) approach (Lye and Purugganan, 2019). RD methods detect CNVs by comparing normalized read depth from short-read sequence data aligned to a reference genome. Low or zero RD is interpreted as a deletion and increased RD is interpreted as an increase in copy number. RP methods are based on the idea that read pairs should map to a reference separated by approximately the same distances as the insert size. If read-pairs map farther away from each other than expected, a deletion is detected; if they are too close together, an insertion is detected. SR methods use pair end reads and detect CNVs by aberrant mapping to a reference genome. For example, when only half of a read-pair maps to a genome, a CNV breakpoint is identified. Each method comes with a different set of biases. RP methods are less effective in repetitive regions and their accuracy is dependent on the size of the insert. SR methods are biased to detect smaller CNVs. RD methods typically have higher false positive rates and are biased towards detecting large variants. The common steps behind the most of computational methods available include pre-processing the sequence reads, mapping reads to reference, and CNVs calling.

The effectiveness of these methods is also dependent on sample read depth. Previous studies demonstrated that current CNV calling tools showed a high portion of false-positive rate. Due to the lack of a gold standard CNV set, the results are limited and incomparable. Moreover, certain programming skills are required for users to use some of these software. Due to these shortcomings, CNV studies using NGS data typically combine multiple computational approaches to minimize false positives. However, detection of CNVs from NGS data is still challenging due to the GC-content bias and the short-read lengths resulting from the NGS technology.

Hecaton is a framework specifically designed for plant genomes that detects CNVs using short paired-end Illumina reads (Wijfjes et al., 2019). CNVs are called by integrating existing structural variant callers through a machine-learning model and several custom post-processing scripts. In general, Hecaton includes the following steps (**Figure 3**): aligning reads to a reference genome using bwa mem; calling CNVs using the structural variants caller Delly, GRODSS, LUMPY, and Manta; post-processing each set of CNVs to remove false positives; merging all sets of CNVs into one large set; classifying CNVs in this large set as true or false positives using a random forest model; and last optional filtering of CNVs based on read depth using duphold and the presence of nearby gaps. Hecaton can be installed locally or can be run through a Docker image. However, Hecaton can only be installed on Linux systems and requires git, gawk, python-dev, python3-dev, and Anaconda/Miniconda (Python 3.6+) to be installed on the system. Hecaton was developed to identify CNVs such as deletion, insertion, tandem duplications, and dispersed duplications. But the performance of detecting CNVs larger than 1Mb was not documented. Yet, Hecaton can only work with Illumina paired end reads.

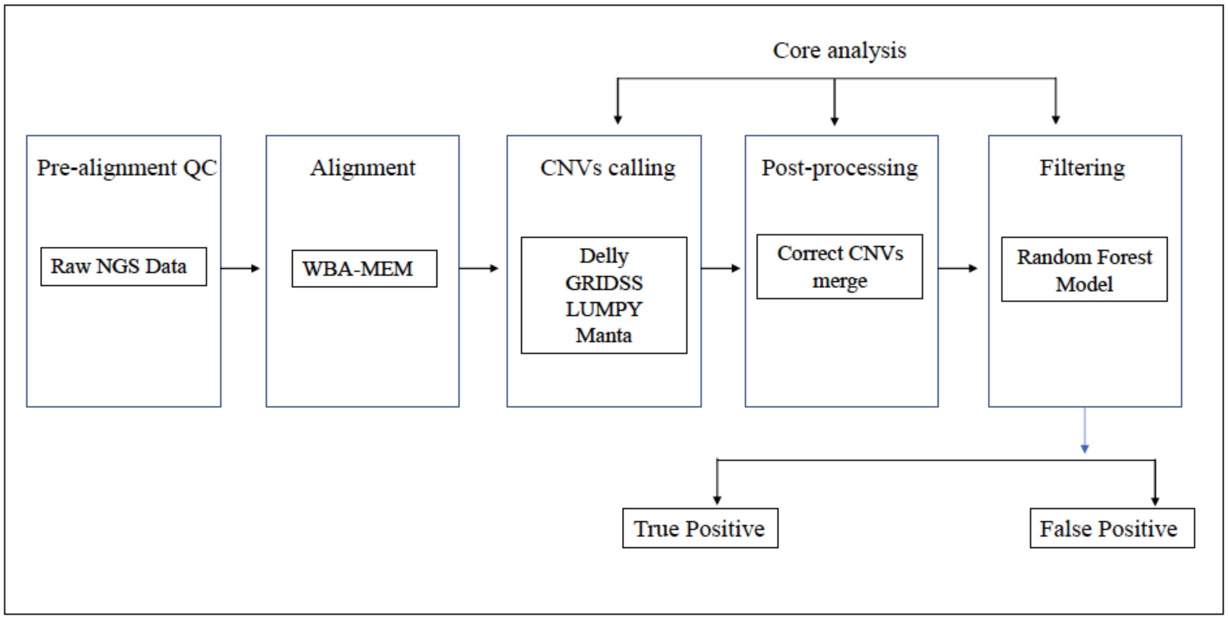

Figure 1. Overview of Hecaton pipeline.

Project purpose

Due to the advanced development of sequencing technology, especially the next generation sequencing, and the availability of computational tools, whole genome-wide analysis of invasive plants become feasible, which provides a valuable tool to further study invasive plants and their response to environment stress and biocontrol of invasive plants. *L. maackii* is a problematic invasive plant widely distributed in central and eastern USA. An investigation report from Maryland Department of Agriculture in 2016 concluded that the weed risk assessment for *L.*

maackii was High Risk. Its invasiveness and the outcomes resulting from its expansion in USA is likely to cause economic, ecological, environmental harm and harm to human health. It is time-consuming and costly to control *L. maackii*. Genetic traits such as gene duplication and copy number variation have been reported to associated with invasion of invasive species (Smith et al., 2020) and genetic biocontrol strategies have been proposed to invasive species control (Teem et al., 2020). Distinct from other species, plant genome was prone to evolve at a higher rate (Panchy et al., 2016), resulting in higher genome diversity. Recently, the whole genome duplication was reported in *L. japonica* (Pu et al., 2020; Yu et al., 2022), and the whole genome assembly of *L. maackii* was completed (Kesel et al., 2022). Given that *L. maackii* and *L. japonica* are closely related species, we hypothesize that a genome duplication is also present in *L. maackii*. Through this study, we aim to detect the genome duplication in *L. maackii* with the purpose to explore the genomic diversity in *L. maackii* in North America and to compare the genome duplications between *L. maackii* and *L. japonica*.

Methods and Materials

Testing datasets for evaluating the performance of MCScanX

Three publicly available datasets (**Table 1**) were selected to evaluate the performance of MCScanX in identifying gene duplications from whole proteome. *Arabidopsis thaliana* (*A. thaliana*) is a widely used plant model for genomic study (Arabidopsis Genome, 2000). *Vitis vinifera* (*V. vinifera*) and *Solanum lycopersicum* (*S. lycopersicum*) are also common domestic plant species (Jaillon et al., 2007; Tomato Genome, 2012). The gene duplication information about these 3 plant species can be found in PlantDGD

Table 1. Testing datasets used for evaluating MCScanX performance.

Species	Version	Source
A. thaliana	TAIR10	Phytozome 11
S. lycopersicum	iTAGv2.3	Phytozome 11
V. vinifera	Genoscope.12X	Phytozome 11

Testing datasets for evaluating the performance of Hecaton

To evaluate the performance of Hecaton on detection of CNVs, we used publicly available short read datasets of *A. thaliana*. **Table 2** provided the NCBI Short Read Archive accession numbers for the datasets used in this study. The genomic assembly and annotations of *A. thaliana* (version TAIR10) can be found from Phytozome 11.

Table 2. Testing datasets used for evaluating Hecaton.

Species	SRA Accession Number
A. thaliana	SRR1946567
A. thaliana	SRR1946568
A. thaliana	SRR1946569
A. thaliana	SRR17867641
A. thaliana	ERR2173372
A. thaliana	ERR8666067

Whole genome assembly of L. maackii and L. japonica

The invasive species of *L. maackii*, and two publicly available datasets of *L. japonica-Lj1017428* (Pu et al., 2020), *L. japonica-Sijihua* (Yu et al., 2022) were included in this study (**Table 3**). The whole genome assembly of *L. maackii* (Kesel et al., 2022) can be found at NCBI-assembly (GenBank assembly accession: GCA_023512865.1). The assembled genome and gene annotation of *L. japonica-Lj1017428* can be found from the Genome Warehouse in National Genomics Data Center with the BioProject ID (PRJCA001719) at https://bigd.big.ac.cn/gwh. The genome assembly of *L. japonica-Sijihua* can be collected from GenBank with the accession number SAMN24662184, and the proteome can be freely available at figShare (https://doi.org/10.6084/m9.figshare.18092708.v6). The gene annotation of *L. japonica-Lj1017428* and *L. japonica-Sijihua* were also used as reference proteome to predict protein-coding genes in *L. maackii.*

Table 3. Geographical information about the plant species in this study.

Species	Location
L. maackii	New York, USA
L. japonica-Lj1017428	Beijing, China
L. japonica-Sijihua	Shandong, China

Protein coding genes prediction using Exonerate

To prepare protein data as input files for MCScanX, protein-coding genes within the assemble genome were identified using Exonerate, a tool for pairwise alignment, by annotating the assembled contigs with reference proteomes, *Helianthus annuus* (UniProt proteome ID UP000215914) and *Lonicera japonica* (**Table** 4).

Table 4. Reference proteomes used in this study.

Reference Proteome	Number of genes	reference
Helianthus annuus	51,240	(Badouin et al., 2017)
Lonicera japonica- Lj10107428	33,961	(Pu et al., 2020)
Lonicera japonica- Sijihua	39,320	(Yu et al., 2022)

The protein2genome model was used to perform pairwise alignment between the reference proteomes as the query and the assembled contigs as the target. This alignment model considers gaps and frameshifts when performing the alignment, allowing for the prediction of gene location, coding regions, introns, and exon boundaries. The --showtargetgff flag was used to convert the alignments to GFF format. The flag --showalignment was set to ‘no’ to reduce the size of the output file. The parameter --fsmmemory was set to 500 to supply 0.5 GB of memory for the finite state machine's heuristic analyses. The parameter --seedrepeat was set to 100. However, the value of –seedrepeat parameter can be adjusted based on the exonerate output. The gene information (mRNA attribute) was then extracted from the Exonerate output file. Here are the commands for running exonerate locally.

```
#Run exonerate in local mode
#-q option for reference proteome input
exonerate –-model protein2genome -q
reference_proteome.fasta -t Lmaackii_assembly.fasta –-
showalignment no –-showtargetgff yes –fsmemory 500 –
seedrepeat 100 > exonerate_gene_predict.out
```

Removal of identical genes and merging the overlapping genes

To clean and remove the genes with identical or overlapping nucleotide sequences (**Figure 4**). A C++ program (**Appendix A_1**) was developed to remove duplicated genes and merge the genes with overlapping regions to form a new gene (remove_clean_gff.cpp). First, the genes originating from the positive strand ("+") and the negative strand ("-") were separated and subsequently subjected to remove the identical genes or merge the overlapping genes. Finally, the predicted genes were stored in a BED file containing 6 columns (contig, gene_start_position, gene_end_position, gene_name, score, and strand).

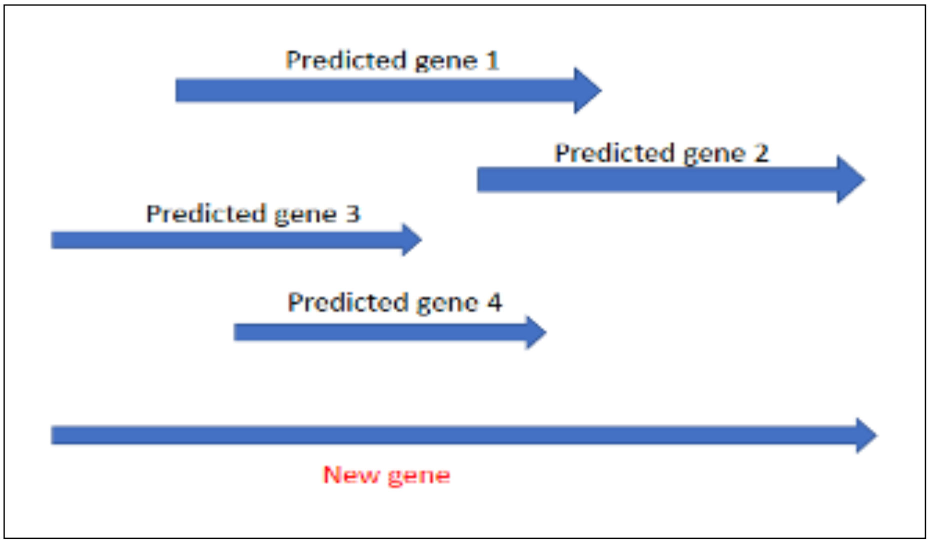

Figure 2. The approach to merge overlapping predicted genes in *L. maackii*.

Extraction of gene nucleotide sequence using bedtools

The nucleotide sequence of predicted genes was extract from *L. maackii* genome using bedtools. The *L. maackii* genome assembly and BED file containing the gene information were used as input files. If the feature of a predicted gene is negative strand, the nucleotide sequence was reversely complemented by using -s option. The command for running bedtools was described as below:

```
#Extracting the nucleotide sequences of predicted genes
using bedtools
bedtools getfasta -fi Lmaackii-genome_assembly.fasta -bed
gene_position_info.bed -fo genes_seq.fasta -s
```

Protein sequences extraction using Exonerate

The Exonerate was used to generate the protein sequences of predicted genes based on the gene nucleotide sequence. Protein sequences from 3 reading frames were subjected to comparison to identify the “best” protein sequence using a C++ script (**Appendix A_2**). The “best” protein was defined by having the least number of stop codons in the protein sequences. The command for extracting protein sequence from nucleotide sequence using Exonerate tool was provided below:

```
#The first reading frame
fastatranslate -F 1 genes_seq.fasta >
genes_prot_seq_1stFrame.fasta
#The second reading frame
fastatranslate -F 2 genes_seq.fasta >
genes_prot_seq_2ndFrame.fasta
#The third reading frame
fastatranslate -F 3 genes_seq.fasta >
genes_prot_seq_3rdFrame.fasta
```

Identification of gene duplication by MCScanX

The MCScanX package can be installed using the command below:

```
#Download MCScanX package from Github
unzip MCscanX.zip
cd MCScanx
make
```

Pre-computed BLAST results (Lmaackii.blast) and gene location information (Lmaackii.gff) are required for running MCScanX successfully. The Lmaackii.gff file was prepared containing the gene location information: contig_id, gene_id, gene_start_position, and gene_end_position (tab separated) and was extracted from the BED file. The *L. maackii* protein sequences were used to create the BLAST database and the all-versus-all local BLASTP (E value $< 1 \times 10^{-10}$, top five matches) was used to generate the blast file. The Lmaackii.blast file was produced by running the commands below:

```
#Preparation of the protein database for blastp
makeblastdb -in Lmaackii-protein.fasta -dbtype prot title
Lmaackii-proteinDB -parse_seqids -out Lmaackii-proteinDB
#run blastp
blastp -query Lmaackii-protein.fasta -db Lmaackii-
proteinDB -num_thread 6 -evalue 1e-10 -max_target_seqs 5
-outfmt 6 -out Lmaackii.blast
```

When the input files are ready, the MCScanX package is run using default setting (**Table 5**) and the command below:

```
#run MCScanX
./MCScanX Lmaackii
#run Duplicate_gene_classifier
./duplicate_gene_classifier Lmaackii
```

Table 5. MCScanX parameters.

-k	MATCH_SCORE, Final_score=MATCH_SCORE+NUM_GAPS*GAP_PENALTY (default: 50)
-g	GAP_PENALTY, gap penalty (default: -1
-s	MATCH_SIZE, number of genes required to call synteny (default: 5)
-e	E_VALUE, alignment significance (default: 1e-05)
-u	UNIT_DIST, average intergenic distance (default: 10000)
-m	MAX_GAPS, maximum gaps (one gap=UNIT_DIST) allowed (default: 20)
-a	only builds the pairwise blocks (.synteny file)
-b	patterns of syntenic blocks. 0: intra- and inter-species (default); 1: intra-species; 2: inter-species
-h	print this help page

Identification of CNVs by Hecaton

Hecaton was run locally on Linux system in this study. All the prerequisites were installed except Nextflow. Since all steps of Hecaton are run using the Nextflow workflow language, we installed Nextflow using the command below:

```
#Download and install Nextflow
wget -qO- https://get.nextflow.io | bash


#Add Nextflow to $PATH
export PATH=$PATH:directory/to/nextflow
```

After Nextflow was installed, Hecaton can be installed through a bash script (**Appendix A_3**). Hecaton also requires dependencies (grids, picard, speedseq) to be installed separately, which was achieved by running another bash script (**Appendix A_4**). Before running Hecaton (**Appendix A_5**), a bash script (**Appendix A_6**) should be executed to ensure that Hecaton can be run correctly. The parameters for running Hecaton were listed in **Table 6**. In this study, whole genome assembly and annotation of *A. thaliana* (version TAIR10) was used as the reference genome, and 6 NGS datasets were used to test the performance of Hecaton on identifying CNVs. Here, the default cutoff value of 0.7 was used because of the good balance of sensitivity and precision. The cutoff can be changed through –cutoff parameter.

Table 6. The specific parameters for running Hecaton.

Required parameters	
--genome_file	reference genome (processed by preprocess.sh) in FASTA format
--reads	location of a set of paired end reads in FASTQ format
--manta_config	config file that will be passed to the Manta tool
--output_dir	output directory to which all results will be written
--model_file	random forest model that will be used to filter CNVs.
Optional parameters	
-w	the working directory to which intermediate results will be written
-resume	to resume the task from the point of failure

Results

Evaluation of MCScanX on identification of gene duplications

We first evaluate the performance of MCScanX on identification of gene duplications using 3 testing datasets (*A. thaliana, V. vinifera* and *S. lycopersicum*). The MCScanX tool can identify and count the number of genes originating from each type of duplications (Singleton, Dispersed, Proximal, Tandem and WGD or Segmental). The tandem duplication type was selected to evaluate the performance of MCScanX. Among the 3 test datasets, 3584 genes were identified as tandem duplications in *A. thaliana*, 3448 tandem duplicated genes were found in *V. vinifera*, and 4143 tandem duplications were detected in *S. lycopersicum* (**Table 7**). At PlantDGD, there are 3525 tandem duplicated genes in *A. thaliana,* 3439 tandem duplicated genes in *V. vinifera*, and 4075 tandem duplicated genes in *S. lycopersicum* (**Table 7**)

Table 7. Number of tandem duplications were found in testing datasets.

Species	Total genes	PlantDGD	MCScanX
A. thaliana	27,416	3,525	3,584
V. vinifera	26,346	3,439	3,448
S. lycopersicum	34,727	4,075	4,143

To calculate the specificity and sensitivity of MCScanX performance, the tandem duplicated genes that are present in both PlantDGD and MCScanX output are defined as true positive (TP). The tandem duplicated genes that are present only in PlantDGD are defined as false negative (FN). The tandem duplicated genes that are present only in MCScanX output are defined as false positive (FP). The genes that are not detected as tandem duplications by both PlantDGD and MCScanX are defined as true negative (TN). The performance metrics are computed: sensitivity defined as TP/(TP + FN), specificity defined as TN/(TN + FP), positive predictive value (PPV) defined as TP/(TP + FP), negative predictive value (NPV) defined as TN/(TN + FN), false negative rate (FNR) defined as FN/(FN + TP), false positive rate (FPR) defined as FP/(FP + TN). The computing parameters are listed in **Table 8**.

Table 8. Comparison of number of tandem duplicated gene found by MCScanX and PlantDGD. TP, true positive; FP, false positive; FN, false negative; TN, true negative.

Species	TP	FP	FN	TN
A. thaliana	3,507	77	18	23,814
V. vinifera	3,393	55	46	30,562
S. lycopersicum	4,053	90	22	22,852

By computing the sensitivity and specificity of MCXcanX on identifying the tandem duplicated genes, we found that MCXcanX can detect the tandem duplications with high sensitivity (>98%) and specificity (>99%) as shown in **Table 9**.

Table 9. Performance of MCScanX on detection of tandem duplications using test datasets.

PPV, positive predictive value; NPV, negative predictive value; FPR, false positive rate; FNR, false negative rate.

Species	Sensitivity	Specificity	PPV	NPV	FPR	FNR
A. thaliana	0.9949	0.9968	0.9785	0.9992	0.0032	0.0051
V. vinifera	0.9866	0.9982	0.9840	0.9985	0.0018	0.0134
S. lycopersicum	0.9946	0.9981	0.9784	0.9990	0.0039	0.0054

We next analyze the performance of MCScanX on detection of tandem pairs in 3 testing datasets. Among the 3 datasets, 2097 tandem pairs in *A. thaliana*, 2035 tandem pairs in *V. vinifera*, and 2445 tandem-pairs in *S. lycopersicum* were identified with MCScanX (**Table 10**). At PlantDGD, there are 2063 tandem-pairs in *A. thaliana*, 2034 tandem-pairs in *V. vinifera*, and 2408 tandem-pairs in *S. lycopersicum* (**Table 10**).

Table 10. Comparison the number of tandem duplication pairs in PlantDGD and MCScanX.

Species	genes	chromosome	gene pairs	PlantDGD	MCScanX
A. thaliana	27,416	7	27,409	2,063	2,097
V. vinifera	26,346	19	26,327	2034	2,035
S. lycopersicum	34,727	12	34,715	2,408	2,445

By analysis, there are 2047 matched tandem-pairs in *A. thaliana*, 1998 matched tandem-pairs in *V. vinifera*, and 2387 matched tandem pairs in *S. lycopersicum* between PlantDGD database and MCScanX output (**Table 11**). By computing the evaluation metrics, MCScanX resulted in more than 98% sensitivity and 97% PPV (**Table 11**). Given the larger number of gene pairs (**Table 10**), it is reasonable to estimate that higher specificity of MCScanX on identifying tandem duplication pairs was observed using testing datasets.

Conclusively, the results of evaluating the performance of MCScanX using testing datasets suggested that MCScanX is an accurate tool for detection of gene duplications in *L. maackii*.

Table 11. Evaluation of MCScanX on identifying tandem duplication pairs.

TP, true positive; FP, false positive; FN, false negative; PPV, positive predictive value.

Species	TP	FP	FN	Sensitivity	PPV
A. thaliana	2,047	50	16	0.9922	0.9762
V. vinifera	1,998	37	36	0.9823	0.9818
S. lycopersicum	2,387	58	21	0.9913	0.9763

Evaluation of Hecaton on detection of CNVs

The default cutoff used by the random forest model of Hecaton (0.7) resulted in a good balance of sensitivity and precision: Hecaton attained at least 80 % precision for all the different types of CNVs (Wijfjes et al., 2019). We used the default cutoff value in our testing datasets. For the Hecaton output, we manually parsed the VCF files and picked the tandem duplication as the factor to evaluate the performance of Hecaton on detection of CNVs.

The reason we chose tandem duplication as the factor to test Hecaton was that we chose the tandem duplications from PlantDGD as the positive control. As shown in **Figure 5**, The number of tandem duplications identified from testing datasets was far lower than the number in PlantDGD, and the number of tandem duplications identified varies among testing datasets. However, it should be noted that the sequencing depth and coverage varies among testing datasets. Ideally, the sequencing reads were equally distributed along the reference genome. However, it was not the situation in a real world. Additionally, the tandem duplications in PlantDGD were detected by using protein. By using Hecaton, the tandem duplication was identified directly from

NGS data. For a given plant, Hecaton cannot be used to identify all the tandem duplication based on one set of NGS data.

In conclusion, we decided not to use Hecaton detect CNVs in *L. maackii*.

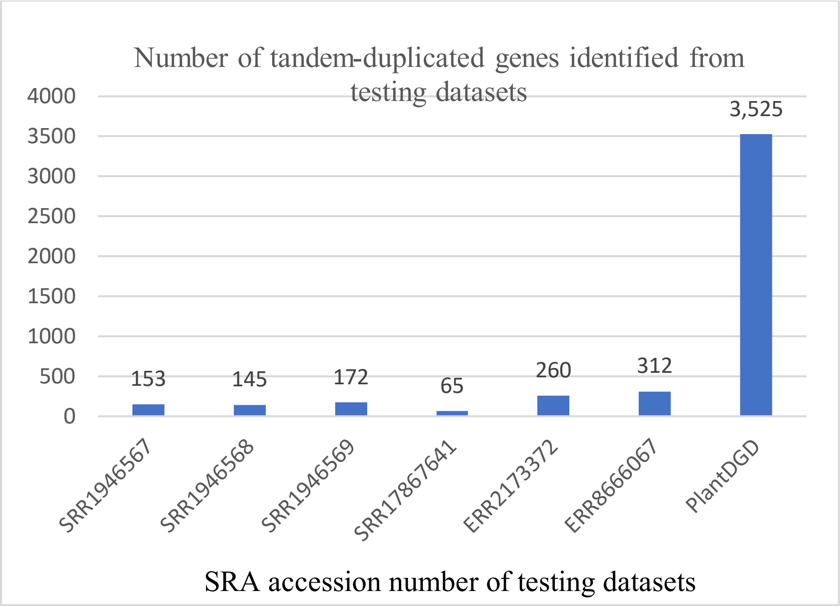

Figure 3. Comparison of the tandem duplication identified in testing datasets.

Protein-coding gene prediction in L. maackii

Prediction of protein-coding genes was based on homology-based prediction. Exonerate was used to predict genes using amino acid sequences from *H. annuus* (UniProt proteome ID UP000215914), *L. japonica*-Lj10107428, *L. japonica*-Sijihua as reference proteomes. We chose *H. annuus* as a reference proteome because it is closely related to *L. maackii,* and its genome is well annotated. *L. japonica* and *L. maackii* belong to the same *genus* of *Lonicera*, and the whole genomes of *L. japonica* were recently published (Pu et al., 2020; Yu et al., 2022). To better predict the protein-coding genes, two values (75, 100) of –seadrepeat parameter was used. Increasing value of –seedrepeat can speed up searches and result in lower number of predicted genes. Based on the nucleotide position of predicted genes, duplicated genes were removed and overlapping genes were merge into new genes. The results were listed in **Table 12**. Given the number of genes found in *L. japonica*, it is estimated that the number of predicted genes in *L. maackii* ranges between 30,000 and 40,000. We found that the possible number of predicted genes resulted from two sources: *L. japonica*-Lj10107428 –seedrepeat 75 and *L. japonica*-Sijihua –seedrepeat 100.

Table 12. The number of predicted genes using exonerate in this study.

Reference proteome	--seedrepeat	Number of predicted genes
H. annuus	75	8,268
L. japonica-Lj10107428	75	32,642
L. japonica-Sijihua	75	56,881
H. annuus	100	4,261
L. japonica-Lj10107428	100	20,925
L. japonica-Sijihua	100	37,429

Protein sequences of predicted genes in L. maackii

The nucleotide sequences of predicted genes were extract from the *L. maackii* genome assembly using the bedtools. Three reading frames of protein sequences of each gene were translated using exonerate tool. For a given gene, we next parsed the protein sequence of each reading frame, the protein sequence was selected based on the number of stop codon found in the protein sequence. For a "perfect" protein sequence, there was zero stop codon in the protein except the end of a protein sequence. We parsed the protein sequences and found that using *L. japonica* (Lj10107428) as reference proteome and setting the value of –seedrepeat to 75 produced the best results (**Table 13**).

Table 13. Comparison of gene prediction among different reference proteomes and parameters.

Reference Proteome	--seedrepeat	Total protein	Perfect protein	Ratio
H. annuus	75	8,268	6,575	0.795
L. japonica-Lj10107428	75	32,642	29,070	0.891
L. japonica-Sijihua	75	56,881	39,293	0.670
H. annuus	100	4,261	3,530	0.828
L. japonica-Lj10107428	100	20,925	18,893	0.903
L. japonica-Sijihua	100	37,429	25,665	0.686

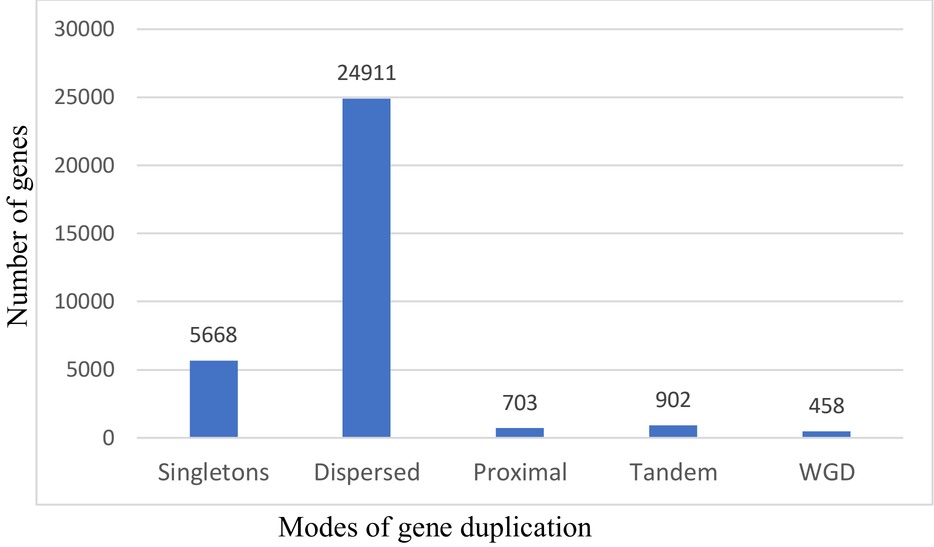

Figure 4. Duplicated genes identified in *L. maackii*. WGD: whole genome duplication.

Genome-wide identification of different modes of gene duplication

The local all-against-all BLASTP algorithm-based detection of collinear blocks was conducted using protein sequences (32,642) to search populations of potential duplicated genes. The gene duplication pool contained 32,642 genes (100% of all genes). We attempted to search the five modes of duplicated genes, respectively, derived from WGD/segmental, singleton, dispersed, proximal, and tandem. As a result, we successfully identified 5,668 genes, 24,911 genes, 703 genes, 902 genes, and 458 genes deriving from Singleton, Dispersed, Proximal, Tandem, and WGD modes, respectively (**Figure 6**).

By comparison, we also analyzed the gene duplication in *L. japonica-Lj10107428* and *L. japonica-Sijihua*. As shown in **Figure 7**, the number of genes derived from different origins varies among the 2 species. Compared to *L. japonica*, we noted that a higher number of genes derived from Singleton and Dispersed modes, respectively, was observed in *L. maackii*. On the contrary, a higher number of genes derived from Proximal, Tandem, and WGD modes, respectively, was found in *L. japonica*. Although the duplicate genes from 5 modes differed in *L. japonica-Lj10107428* and *L. japonica-Sijihua,* the difference was not obvious compared with the results in *L. maackii.* We suspected that the whole genome assembly resulted in the difference among *L. maackii*, *L. japonica-Lj10107428* and *L. japonica-Sijihua.* The whole genome of *L. maackii* is contig-scale genome assembly. The whole genome of *L. japonica-Lj10107428* and *L. japonica-Sijihua* are chromosome-scale genome assembly. Compared to contig, chromosome is a much longer assembly. In addition, the special feature of MCScanX is that each chromosome (contig) is used as a reference. In a word, the difference of genome assembly and MCScanX algorithm may explain the difference of gene duplications observed in *L. maackii*, *L. japonica-Lj10107428* and *L. japonica-Sijihua.* Furthermore, the different results observed in *L. japonica-Lj10107428* and *L. japonica-Sijihua* suggested that the genome annotation also affect the result of MCScanX software package.

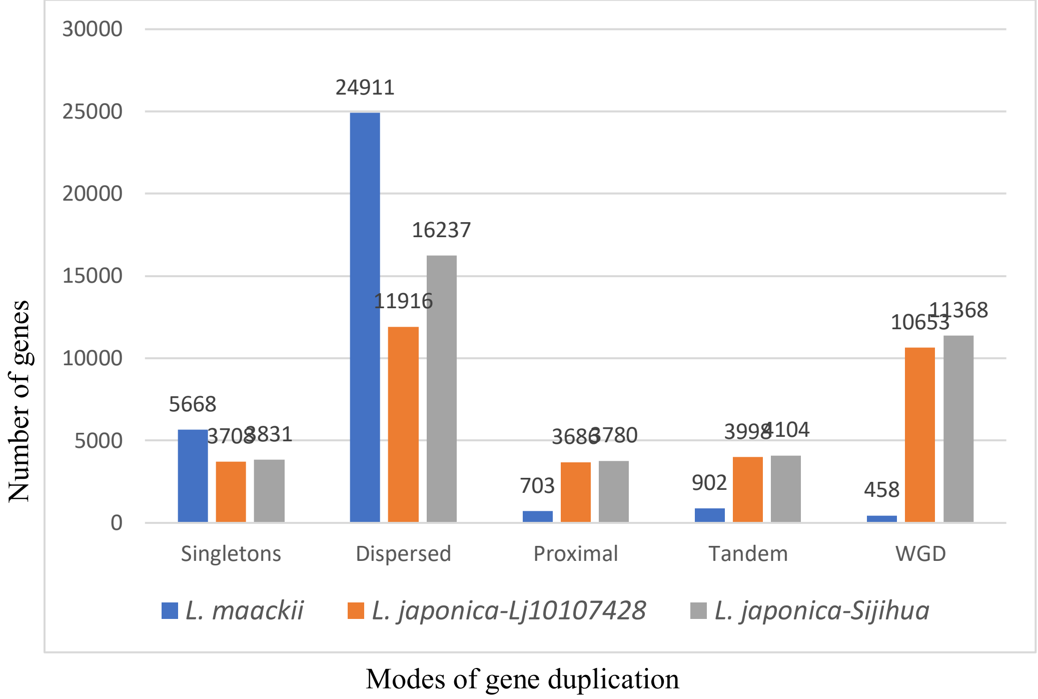

Figure 5. Comparison of gene duplications in *L. maackii* and *L. japonica*. WGD: whole genome duplication.

Discussion and Future Direction

Here we performed protein-coding gene prediction and gene duplication analysis in *L. maackii*. By using homology-based approach, we predicted 32,642 protein-coding genes from *L. maackii* contig-scaled assembly, and detected 5,668 genes, 24,911 genes, 703 genes, 902 genes, and 458 genes deriving from Singleton, Dispersed, Proximal, Tandem, and WGD modes, respectively. We also performed the comparison of gene duplication between *L. maackii* and *L. japonica*, we found that the gene duplications in *L. maackii* differ from that in *L. japonica*. Higher populations of Singleton and Dispersed duplications were observed in *L. maackii*. The genes derived from modes of Proximal, Tandem, and WGD duplication were much less in *L. maackii*.

Genome size and complexity, transposable elements, heterozygosity, polyploidy, gene content and gene families, non-coding RNAs, and widely distributed repetitive sequences are the factors that make plant genome assembly challenging. (Li and Harkess, 2018). Over the past 20 years, the sequences of over 1000 plant genomes have been reported (Sun et al., 2022). Most of the genome assemblies available from the NCBI were generated predominantly using short-read sequencing technology. Short-read sequencing can yield draft assemblies sufficient for estimates of gene space and repeat content, but of limited utility for investigations of chromosomal organization. The assembly of plant genomes with large number of repetitive sequences is much more difficult with only short read sequences. Repetitive sequences are abundant in species with larger genomes and have always been a major challenge for genome assemblies. Repeats longer than read length would lead to gaps in the genome assembly due to uncertainty in assembly of these regions. This would break down the genome into pieces, leading to the loss of linkage information among genetic markers. In addition, repeats may also be led to mis-assembly where two unlinked regions were joined together and resulted in higher than usual read coverages. In the case of repetitive sequences containing genes, such as tandemly duplicated genes and retrogenes, such mis-assembly would reduce the gene copy number estimation. These missing genes not only make it challenging to account for all the genes in a genome but also create problems for functional genomic studies by impacting gene expression level estimates or loss-of-function studies. With the development of sequencing technologies, the addition of long-read data can improve the contiguity of assemblies based on the value of short-read genome assemblies. Advanced scaffolding strategies can also simplify genome assembly, enabling access to more chromosome-scale assemblies of plant species with increasing genome complexity and size. The combination of short read sequencing and long read sequencing has resulted in the recent reporting of many high-quality chromosome-level genome sequences. In this study, the *L. maackii* genome assembly was generated by using short read sequencing. Although the quality of assembly is good, our results suggested that the contig-level assembly affected the downstream analysis of gene duplications compared to chromosome-level assemblies of *L. japonica*. For the future work, the long read sequencing can be employed to generate a chromosome-level assembly.

Whole genome annotation is highly important to study *L. maackii*. Here, the gene prediction in this study was performed using a reference proteome. We selected *L. japonica* proteome that is the most closely related species to *L. maackii* as the reference proteome. The limitation is that the reference proteome was recently reported, which might introduce bias in gene prediction study for *L. maackii*. To improve the genome annotation, combination of *ab initio* gene prediction, homology-based gene prediction, and the transcriptome sequencing (short-read/long-read RNA-seq technology) can be used.

Gene duplication that is the source of genetic variation in invasive plants has been widely studied as an important factor in evolution for a long time, which is closely associated with adaptive evolution, such as the genes related to immunity, development, and reproduction. Some populations of invasive species lose genetic diversity during invasion through founder effects. But many have higher genetic diversity outside their native range. In plants, WGD can produce thousands of duplicated genes. The fate of those duplicated genes is determined by several factors such as the mode of duplication, functions of duplicated genes, and protein interactions. Some of those duplicated genes were lost during evolution and some were maintained. From evolutionary perspective, the duplicated genes that contributed to adapt to environmental stresses would be kept. A recent study (2019) conducting gene duplication analysis on 141 sequenced plants genomes suggested that WGD genes were more conserved and tandem and proximal duplicated genes were prone to develop new functions (Qiao et al., 2019). The GO enrichment analysis to investigate the functional roles of tandem and proximal genes in model plant *A. thaliana* was conducted. The results indicated that tandem and proximal duplicated genes shared several enriched GO terms such as defense response, drug binding, endomembrane system, monooxygenase activity, oxidoreductase activity, and oxygen binding. However, proximal duplicates are enriched in GO terms associated with apoptotic processes, cell death, programmed cell death, immune response, and signaling receptor activity. Tandem duplicated genes are enriched in GO terms involved in "binding," such as tetrapyrrole binding, iron ion binding, heme binding, and cofactor binding, and "activity" such as transferase activity, hydrolase activity, electron transfer activity, and catalytic activity. Investigation of gene duplications in *L. maackii* would promote our understanding on genetic responses to new environment and/or the evolution after introduction into North America. In this study, we conducted the analysis on gene duplication in *L. maackii*. However, we did not perform gene functional annotation. For future work, we can align the predicted protein-coding gene sequences against public functional databases such as GO and KEGG using BLAST, which could help us identify genes that contribute to invasiveness of *L. maackii* and/or genes that could be targets for controlling *L. maackii*. Given the fact that there is no publicly available data about native *L. maackii*, we performed the comparison of gene duplication between invasive *L. maackii* and native *L. japonica*. However, the results might be biased due the difference in whole genome assemblies in *L. maackii* and *L. japonica.*

Although accurate detection of gene duplications is still difficult, different computational approached and pipelines have been developed to identify gene duplications at whole-genome level. In this study, we employed MCScanX software package to perform genome-wide identification of gene duplications in *L. maackii*. MCScanX was developed to detect gene duplication in whole genome at protein level by using dynamic programming algorithm. The special feature and strength of MCScanX is that each chromosome is used as a reference. Furthermore, compared to DNA, proteins are more conserved during the evolution. Therefore, the gene duplications by using MCScanX software package are biased by the whole genome assembly and the quality of available genome annotation. In our study, the first version of genome annotation of *L. maackii* was generated by suing homology-based approach, which suggested that improvements of genome annotation and genome assembly were needed to produce less biased gene duplications in *L. maackii*.

Our findings here were only a start. The future work should aim to generate a chromosome-level genome assembly, improve the protein-coding gene prediction, and conduct gene functional analysis, which might promote our control on *L. maackii* invasiveness.

References

Arabidopsis Genome, I. (2000). Analysis of the genome sequence of the flowering plant Arabidopsis thaliana. Nature *408*, 796-815.

Badouin, H., Gouzy, J., Grassa, C.J., Murat, F., Staton, S.E., Cottret, L., Lelandais-Briere, C., Owens, G.L., Carrere, S., Mayjonade, B.*, et al.* (2017). The sunflower genome provides insights into oil metabolism, flowering and Asterid evolution. Nature *546*, 148-152.

Birney, E., and Durbin, R. (2000). Using GeneWise in the Drosophila annotation experiment. Genome Res *10*, 547-548.

Bradley, B., Early, R., and Sorte, C. (2015). Space to invade? Comparative range infilling and potential range of invasive and native plants. Global Ecology and Biogeography *Vol. 24, No. 3/4*.

Burge, C., and Karlin, S. (1997). Prediction of complete gene structures in human genomic DNA. J Mol Biol *268*, 78-94.

Chatterji, S., and Pachter, L. (2006). Reference based annotation with GeneMapper. Genome Biol *7*, R29.

Dlugosch, K.M., Anderson, S.R., Braasch, J., Cang, F.A., and Gillette, H.D. (2015). The devil is in the details: genetic variation in introduced populations and its contributions to invasion. Mol Ecol *24*, 2095-2111.

Dlugosch, K.M., and Parker, I.M. (2008). Invading populations of an ornamental shrub show rapid life history evolution despite genetic bottlenecks. Ecol Lett *11*, 701-709.

Dolatabadian, A., Patel, D.A., Edwards, D., and Batley, J. (2017). Copy number variation and disease resistance in plants. Theor Appl Genet *130*, 2479-2490.

Ejigu, G.F., and Jung, J. (2020). Review on the Computational Genome Annotation of Sequences Obtained by Next-Generation Sequencing. Biology *9*, 295.

EunmiLee, C. (2002). Evolutionary genetics of invasive species. Trends in Ecology & Evolution *17*.

Exposito-Alonso, M., Becker, C., Schuenemann, V.J., Reiter, E., Setzer, C., Slovak, R., Brachi, B., Hagmann, J., Grimm, D.G., Chen, J*., et al.* (2018). The rate and potential relevance of new mutations in a colonizing plant lineage. PLoS Genet *14*, e1007155.

Fantle-Lepczyk, J.E., Haubrock, P.J., Kramer, A.M., Cuthbert, R.N., Turbelin, A.J., Crystal-Ornelas, R., Diagne, C., and Courchamp, F. (2022). Economic costs of biological invasions in the United States. Sci Total Environ *806*, 151318.

Fridley, J.D. (2012). Extended leaf phenology and the autumn niche in deciduous forest invasions. Nature *485*, 359-362.

Gardner, A.M., Muturi, E.J., Overmier, L.D., and Allan, B.F. (2017). Large-Scale Removal of Invasive Honeysuckle Decreases Mosquito and Avian Host Abundance. Ecohealth *14*, 750-761.

Grandaubert, J., Bhattacharyya, A., and Stukenbrock, E.H. (2015). RNA-seq-Based Gene Annotation and Comparative Genomics of Four Fungal Grass Pathogens in the Genus Zymoseptoria Identify Novel Orphan Genes and Species-Specific Invasions of Transposable Elements. G3 (Bethesda) *5*, 1323-1333.

Hastings, P.J., Lupski, J.R., Rosenberg, S.M., and Ira, G. (2009). Mechanisms of change in gene copy number. Nat Rev Genet *10*, 551-564.

Hudon, J., Driver, R.J., Rice, N.H., Lloyd-Evans, T.L., Craves, J.A., and Shustack, D.P. (2016). Diet explains red flight feathers in Yellow-shafted Flickers in eastern North America. The Auk *134*, 22-33.

Hudon, J., and Mulvihill, R. (2017). Diet-induced plumage erythrism as a result of the spread of alien shrubs in North America. North American Bird Bander *42(4)*.

Jaillon, O., Aury, J.M., Noel, B., Policriti, A., Clepet, C., Casagrande, A., Choisne, N., Aubourg, S., Vitulo, N., Jubin, C*., et al.* (2007). The grapevine genome sequence suggests ancestral hexaploidization in major angiosperm phyla. Nature *449*, 463-467.

Jones, T.M., Rodewald, A.D., and Shustack, D.P. (2010). Variation in Plumage Coloration of Northern Cardinals in Urbanizing Landscapes. The Wilson Journal of Ornithology

Kang, K.B., Kang, S.J., Kim, M.S., Lee, D.Y., Han, S.I., Kim, T.B., Park, J.Y., Kim, J., Yang, T.J., and Sung, S.H. (2018). Chemical and genomic diversity of six Lonicera species occurring in Korea. Phytochemistry *155*, 126-135.

Keilwagen, J., Hartung, F., and Grau, J. (2019). GeMoMa: Homology-Based Gene Prediction Utilizing Intron Position Conservation and RNA-seq Data. Methods Mol Biol *1962*, 161-177.

Keilwagen, J., Hartung, F., Paulini, M., Twardziok, S.O., and Grau, J. (2018). Combining RNA-seq data and homology-based gene prediction for plants, animals and fungi. BMC Bioinformatics *19*, 189.

Kesel, E., Hudson, A.O., and Osier, M.V. (2022). Whole-Genome Sequence, Assembly and Annotation of an Invasive Plant, Lonicera maackii (Amur Honeysuckle). Plants (Basel).

Kleiman, L.R., Kleiman, B.P., and Kleiman, S. (2018). Successful Control of Lonicera maackii (Amur Honeysuckle) with Basal Bark Herbicide Ecological Restoration *36*.

Kumar Rai, P., and Singh, J.S. (2020). Invasive alien plant species: Their impact on environment, ecosystem services and human health. Ecol Indic *111*, 106020.

Lallemand, T., Leduc, M., Landes, C., Rizzon, C., and Lerat, E. (2020). An Overview of Duplicated Gene Detection Methods: Why the Duplication Mechanism Has to Be Accounted for in Their Choice. Genes (Basel) *11*.

Li, F.W., and Harkess, A. (2018). A guide to sequence your favorite plant genomes. Appl Plant Sci *6*, e1030.

Lomsadze, A., Ter-Hovhannisyan, V., Chernoff, Y.O., and Borodovsky, M. (2005). Gene identification in novel eukaryotic genomes by self-training algorithm. Nucleic Acids Res *33*, 6494-6506.

Lye, Z.N., and Purugganan, M.D. (2019). Copy Number Variation in Domestication. Trends Plant Sci *24*, 352-365.

Maebara, Y., Tamaoki, M., Iguchi, Y., Nakahama, N., Hanai, T., Nishino, A., and Hayasaka, D. (2020). Genetic Diversity of Invasive Spartina alterniflora Loisel. (Poaceae) Introduced Unintentionally Into Japan and Its Invasion Pathway. Front Plant Sci *11*, 556039.

McNeish, R., Benbow, M., and McEwan, R. (2017). Removal of the Invasive Shrub, Lonicera maackii (Amur Honeysuckle), from a Headwater Stream Riparian Zone Shifts

Taxonomic and Functional Composition of the Aquatic Biota. Invasive Plant Science and Management *10(3), 232-246*.

McNeish, R.E., and McEwan, R.W. (2016). A review on the invasion ecology of Amur honeysuckle (Lonicera maackii, Caprifoliaceae) a case study of ecological impacts at multiple scales. The Journal of the Torrey Botanical Society.

Molina-Montenegro, M.A., Acuna-Rodriguez, I.S., Flores, T.S.M., Hereme, R., Lafon, A., Atala, C., and Torres-Diaz, C. (2018). Is the Success of Plant Invasions the Result of Rapid Adaptive Evolution in Seed Traits? Evidence from a Latitudinal Rainfall Gradient. Front Plant Sci *9*, 208.

Mounger, J., Ainouche, M.L., Bossdorf, O., Cave-Radet, A., Li, B., Parepa, M., Salmon, A., Yang, J., and Richards, C.L. (2021). Epigenetics and the success of invasive plants. Philos Trans R Soc Lond B Biol Sci *376*, 20200117.

Oswalt CM, S., F., Q., G., BV., I.I., SN., O., BC., P., and KM, P. (2015). A subcontinental view of forest plant invasions. NeoBiota.

Packett, D., and Dunning, J. (2009). Stopover Habitat Selection by Migrant Landbirds in a Fragmented Forest-Agricultural Landscape. The Auk *126*.

Panchy, N., Lehti-Shiu, M., and Shiu, S.H. (2016). Evolution of Gene Duplication in Plants. Plant Physiol *171*, 2294-2316.

Parepa, M., Fischer, M., Krebs, C., and Bossdorf, O. (2014). Hybridization increases invasive knotweed success. Evol Appl *7*, 413-420.

Pu, X., Li, Z., Tian, Y., Gao, R., Hao, L., Hu, Y., He, C., Sun, W., Xu, M., Peters, R.J.*, et al.* (2020). The honeysuckle genome provides insight into the molecular mechanism of carotenoid metabolism underlying dynamic flower coloration. New Phytol *227*, 930-943.

Qiao, X., Li, Q., Yin, H., Qi, K., Li, L., Wang, R., Zhang, S., and Paterson, A.H. (2019). Gene duplication and evolution in recurring polyploidization-diploidization cycles in plants. Genome Biol *20*, 38.

Rathfon, R., and Ruble, K. (2007). HERBICIDE TREATMENTS FOR CONTROLLING INVASIVE BUSH HONEYSUCKLE IN A MATURE HARDWOOD FOREST IN WEST-CENTRAL INDIANA. e-Gen Tech Rep SRS–101 US Department of Agriculture, Forest Service, Southern Research Station.

Rius, M., and Darling, J.A. (2014). How important is intraspecific genetic admixture to the success of colonising populations? Trends Ecol Evol *29*, 233-242.

Rivera, B., Meilan, R., Scharf, M., Karve, R., and Jenkins, M. (2022). THE EFFECT OF A NOVEL HERBICIDE ADJUVANT IN TREATING AMUR HONEYSUCKLE (LONICERA MAACKII). Invasive Plant Science and Management *1-23*.

Rocha, O.J., McNutt, E., and Barriball, K. (2014). Isolation and characterization of microsatellite loci from Amur honeysuckle, Lonicera maackii (Caprifoliaceae). Appl Plant Sci *2*.

Salzberg, S.L., Pertea, M., Delcher, A.L., Gardner, M.J., and Tettelin, H. (1999). Interpolated Markov models for eukaryotic gene finding. Genomics *59*, 24-31.

Scalzitti, N., Jeannin-Girardon, A., Collet, P., Poch, O., and Thompson, J.D. (2020). A benchmark study of ab initio gene prediction methods in diverse eukaryotic organisms. BMC Genomics *21*, 293.

Seastedt, T.R. (2015). Biological control of invasive plant species: a reassessment for the Anthropocene. New Phytol *205*, 490-502.

Shewhart, L., McEwan, R.W., and Benbow, M.E. (2014). Evidence for facilitation of Culex pipiens (Diptera: Culicidae) life history traits by the nonnative invasive shrub Amur honeysuckle (Lonicera maackii). Environ Entomol *43*, 1584-1593.

Shlien, A., and Malkin, D. (2009). Copy number variations and cancer. Genome Med *1*, 62.

Slater, G.S., and Birney, E. (2005). Automated generation of heuristics for biological sequence comparison. BMC Bioinformatics *6*, 31.

Smith, A.L., Hodkinson, T.R., Villellas, J., Catford, J.A., Csergo, A.M., Blomberg, S.P., Crone, E.E., Ehrlen, J., Garcia, M.B., Laine, A.L*., et al.* (2020). Global gene flow releases invasive plants from environmental constraints on genetic diversity. Proc Natl Acad Sci U S A *117*, 4218-4227.

Stanke, M., and Waack, S. (2003). Gene prediction with a hidden Markov model and a new intron submodel. Bioinformatics *19 Suppl 2*, ii215-225.

Sun, Y., Shang, L., Zhu, Q.H., Fan, L., and Guo, L. (2022). Twenty years of plant genome sequencing: achievements and challenges. Trends Plant Sci *27*, 391-401.

Tang, H., Wang, X., Bowers, J.E., Ming, R., Alam, M., and Paterson, A.H. (2008). Unraveling ancient hexaploidy through multiply-aligned angiosperm gene maps. Genome Res *18*, 1944-1954.

Teem, J.L., Alphey, L., Descamps, S., Edgington, M.P., Edwards, O., Gemmell, N., Harvey-Samuel, T., Melnick, R.L., Oh, K.P., Piaggio, A.J.*, et al.* (2020). Genetic Biocontrol for Invasive Species. Front Bioeng Biotechnol *8*, 452.

Tomato Genome, C. (2012). The tomato genome sequence provides insights into fleshy fruit evolution. Nature *485*, 635-641.

Wang, Y., Tang, H., Debarry, J.D., Tan, X., Li, J., Wang, X., Lee, T.H., Jin, H., Marler, B., Guo, H.*, et al.* (2012). MCScanX: a toolkit for detection and evolutionary analysis of gene synteny and collinearity. Nucleic Acids Res *40*, e49.

Wang, Z., Chen, Y., and Li, Y. (2004). A brief review of computational gene prediction methods. Genomics Proteomics Bioinformatics *2*, 216-221.

Webster, C.R., Jenkins, M.A., and Jose, S. (2006). Woody Invaders and the Challenges They Pose to Forest Ecosystems in the Eastern United States. Journal of Forestry *104*, 366-374.

Wijfjes, R.Y., Smit, S., and de Ridder, D. (2019). Hecaton: reliably detecting copy number variation in plant genomes using short read sequencing data. BMC Genomics *20*, 818.

Wilson, J.R., Dormontt, E.E., Prentis, P.J., Lowe, A.J., and Richardson, D.M. (2009). Something in the way you move: dispersal pathways affect invasion success. Trends Ecol Evol *24*, 136-144.

Yu, H., Guo, K., Lai, K., Shah, M.A., Xu, Z., Cui, N., and Wang, H. (2022). Chromosome-scale genome assembly of an important medicinal plant honeysuckle. Sci Data *9*, 226.

Zhang, F., Gu, W., Hurles, M.E., and Lupski, J.R. (2009). Copy number variation in human health, disease, and evolution. Annu Rev Genomics Hum Genet *10*, 451-481.

Zmienko, A., Samelak, A., Kozlowski, P., and Figlerowicz, M. (2014). Copy number polymorphism in plant genomes. Theor Appl Genet *127*, 1-18.

Appendix

Scripts and Programs

SCRIPT/PROGRAM NAME:

remove_clean_gff.cpp

PURPOSE:

To remove duplicated predicted genes and parse the overlapped predicted genes to form new gene.

APPLICATION IN THIS PROJECT:

By using exonerate to predict genes, there are plenty of duplicated genes or overlapped genes based on the nucleotide position of the predicted gene. This C++ program was used to remove the duplicated genes and merge the overlapped gene to form a new gene. The output was a GFF file containing the gene information.

```
###########################A_1 remove_clean_gff.cpp#############################

#include <iostream>
#include <string>
#include <fstream>
#include <algorithm>
#include <vector>
#include <sstream>
using namespace std;

int main()
{
    ifstream infile;
    ofstream outfile;
    string line, line2;
    vector<string> contigName;
    vector<string> geneInfo;

    infile.open("Ljaponica_seedrepeat75_PosStrand.txt");
    while (getline(infile, line))
{
        geneInfo.push_back(line);
        stringstream ss(line);
        string seqid, strand, attribute;
        int start_pos, end_pos;
        ss >> seqid >> start_pos >> end_pos >> strand >> attribute;
        contigName.push_back(seqid);
    }

    cout << contigName.size() << "  " << geneInfo.size() << endl;
    infile.close();
    cout << "Gene info input done!" << endl;

    sort(contigName.begin(), contigName.end());
    contigName.erase(unique(contigName.begin(),contigName.end()), contigName.end());
    int M = contigName.size();
    int N = geneInfo.size();
cout << M << " -> " << N << endl;

    outfile.open("Ljaponica_seedrepeat75_PosStrand.bed");

    vector<pair<int, int> > myVector;
    vector<pair<int, int> > New_myVector;

    for (int i = 0; i < M; i++)
    {
        string contig = contigName[i];
        for (int j = 0; j < N; j++)
        {
            if (geneInfo[j].find(contig) != -1)
            {
                stringstream ss(geneInfo[j]);
                string seqid, strand, attribute;
                int start_pos, end_pos;
                ss >> seqid >> start_pos >> end_pos >> strand >> attribute;
                myVector.push_back(make_pair(start_pos, end_pos));
            }
        }
        sort(myVector.begin(), myVector.end());
        myVector.erase(unique(myVector.begin(), myVector.end()), myVector.end());
```

```
            New_myVector.push_back(make_pair(myVector[0].first, myVector[0].second));
            //cout << myVector.size() << endl;

            for (int a = 1; a < myVector.size(); a++)
            {
                int x1 = myVector[a].first;
                int y1 = myVector[a].second;
                int x2 = New_myVector[New_myVector.size() - 1].first;
                int y2 = New_myVector[New_myVector.size() - 1].second;
                if (y2 >= x1)
                {
                    New_myVector[New_myVector.size() - 1].second = max(y1, y2);
                }
                else
                {
                    New_myVector.push_back(make_pair(x1, y1));
                }
            }

            for (int b = 0; b < New_myVector.size(); b++)
            {
                outfile << contig << "\t" << New_myVector[b].first << "\t" << New_myVector[b].second
<< "\t" << "+" << endl;
            }

            myVector.clear();
            New_myVector.clear();
        }

        cout << "Work Done!" << endl;
```

```
    return 0;
}
```

```
####################remove_clean_gff.cpp########################################
```

SCRIPT/PROGRAM NAME:

pick_best_reading_frame.cpp

PURPOSE:

To select the “best” protein sequences.

APPLICATION IN THIS PROJECT:

The protein sequences of predicted genes in this project were generate from the nucleotide sequence based on the gene position. This C++ program was used to compare the protein sequence derived from three reading frames and select the best protein sequence from the three reading frames. The output was a fasta file containing the protein sequences.

```
########################A_2 pick_best_reading_frame.cpp#########################

    #include <iostream>
    #include <string>
    #include <fstream>
    #include <algorithm>
    #include <vector>
    #include <sstream>
    #include <cstdbool>

    using namespace std;

    int main()
    {
      // Read the first file
      string line, name, content;
      vector<string> nameFrame1;
      vector<string> contentFrame1;
      ifstream fin;

fin.open("/Users/wang2034/Desktop/MCScanX/test/Lmaackii/Exonerate_out/Ha100proteinF1.fasta");

      while (getline(fin, line))
      {
        if (line[0] == '>')
        {
          if (!name.empty())
          {
            nameFrame1.push_back(name);
            contentFrame1.push_back(content);
          }
```

```
                name = line.substr(1, line.length() - 1);
                content.clear();
            }
            else
            {
                if (line[line.size() - 1] == '\n' || line[line.size() - 1] == '\r')
                {
                    content += line.erase(line.length() - 1);
                }
                else if ((line[line.size() - 1] == '\n' && line[line.size() - 2] == '\r') || (line[line.size() - 1] == '\r' && line[line.size() - 2] == '\n'))
                {
                    content += line.erase(line.length() - 2);
                }
                else
                {
                    content += line;
                }
            }
        }
        if (!name.empty())
        {
            nameFrame1.push_back(name);
            contentFrame1.push_back(content);
        }

        fin.close();
        line.clear();
        name.clear();
        content.clear();
        cout << nameFrame1.size() << " " << contentFrame1.size() << endl;

        // Read the second file
        vector<string> nameFrame2;
        vector<string> contentFrame2;

fin.open("/Users/wang2034/Desktop/MCScanX/test/Lmaackii/Exonerate_out/Ha100proteinF2.fasta");
        while (getline(fin, line))
        {
          if (line[0] == '>')
      {
            if (!name.empty())
            {
              nameFrame2.push_back(name);
              contentFrame2.push_back(content);
            }

            name = line.substr(1, line.length() - 1);
            content.clear();
          }
          else
          {
            if (line[line.size() - 1] == '\n' || line[line.size() - 1] == '\r')
            {
              content += line.erase(line.length() - 1);
            }
            else if ((line[line.size() - 1] == '\n' && line[line.size() - 2] == '\r') || (line[line.size() - 1] ==
'\r' && line[line.size() - 2] == '\n'))
            {
              content += line.erase(line.length() - 2);
            }
            else
            {
              content += line;
            }
          }
        }
        if (!name.empty())
        {
          nameFrame2.push_back(name);
          contentFrame2.push_back(content);
        }

        fin.close();
        line.clear();
        name.clear();
        content.clear();
        cout << nameFrame2.size() << " " << contentFrame2.size() << endl;

        // Read the 3RD file
        vector<string> nameFrame3;
        vector<string> contentFrame3;

fin.open("/Users/wang2034/Desktop/MCScanX/test/Lmaackii/Exonerate_out/Ha100proteinF3.fasta");
        while (getline(fin, line))
        {
          if (line[0] == '>')
          {
            if (!name.empty())
            {
              nameFrame3.push_back(name);
```

```
                     contentFrame3.push_back(content);
                  }

                  name = line.substr(1, line.length() - 1);
                  content.clear();
               }
               else
               {
                  if (line[line.size() - 1] == '\n' || line[line.size() - 1] == '\r')
                  {
                     content += line.erase(line.length() - 1);
                  }
                  else if ((line[line.size() - 1] == '\n' && line[line.size() - 2] == '\r') || (line[line.size() - 1] ==
'\r' && line[line.size() - 2] == '\n'))
                  {
                     content += line.erase(line.length() - 2);
                  }
                  else
                  {
                     content += line;
                  }
               }
            }
            if (!name.empty())
            {
               nameFrame3.push_back(name);
               contentFrame3.push_back(content);
            }

            fin.close();
            line.clear();
```

```
        name.clear();
        content.clear();
        cout << nameFrame3.size() << " " << contentFrame3.size() << endl;

        //// pick the best frame that has the least stop codons in the protein sequences
        ofstream outputfile;

outputfile.open("/Users/wang2034/Desktop/MCScanX/test/Lmaackii/Exonerate_out/Ha100protein.fasta")
;

        for (int i = 0; i < contentFrame3.size(); i++)
        {
          string seq1 = contentFrame1[i];
          string seq2 = contentFrame2[i];
          string seq3 = contentFrame3[i];
          int Frame1_stopcodon_counts = 0;
          int Frame2_stopcodon_counts = 0;
          int Frame3_stopcodon_counts = 0;

          for (int a = 0; a < seq1.length(); a++)
          {
            if (seq1[a] == '*')
            {
              Frame1_stopcodon_counts += 1;
            }
          }

          for (int b = 0; b < seq2.length(); b++)
          {
            if (seq2[b] == '*')
            {
```

```
        Frame2_stopcodon_counts += 1;
    }
}

for (int c = 0; c < seq3.length(); c++)
{
    if (seq3[c] == '*')
    {
        Frame3_stopcodon_counts += 1;
    }
}

int MIN = min(Frame1_stopcodon_counts, min(Frame2_stopcodon_counts, Frame3_stopcodon_counts));

if (Frame1_stopcodon_counts == MIN)
{
    outputfile << ">" + nameFrame1[i] << endl;
    outputfile << contentFrame1[i] << endl;
}
else if (Frame2_stopcodon_counts == MIN)
{
    outputfile << ">" + nameFrame2[i] << endl;
    outputfile << contentFrame2[i] << endl;
}
else
{
    outputfile << ">" + nameFrame3[i] << endl;
    outputfile << contentFrame3[i] << endl;
}
```

```
    }
}
```

####################A_2 pick_best_reading_frame.cpp#################################

SCRIPT/PROGRAM NAME:

install_Hecaton.sh

PURPOSE:

To install Hecaton tool on Linux system for local use.

APPLICATION IN THIS PROJECT:

Hecaton can only be install on Linux system. Before installing Hecaton, all of the scripts of Hecaton should be added to $PATH. This bash script was used to download the Hecaton package, add the required script to $PATH environment variable first. Then, the script for installing Hecaton can executed.

```
###################A_3 install_Hecaton.sh#########################################
#!/bin/bash
#Download Hecaton package
git clone https://git.wur.nl/bioinformatics/hecaton.git

#Set permissions and add all scripts of Hecaton to $PATH:
cd hecaton
chmod +x scripts/collapse/* && \
chmod +x scripts/convert/* && \
chmod +x scripts/filter/* && \
chmod +x scripts/genotype/* && \
chmod +x scripts/gridss/* && \
chmod +x scripts/intersect/* && \
chmod +x scripts/predict/* && \
chmod +x scripts/process/* && \
export
PATH=$PWD/scripts/collapse:$PWD/scripts/convert:$PWD/scripts/filter:$PWD/scripts/genotype:$PWD/scripts/gridss:$PWD/scripts/intersect:$PWD/scripts/predict:$PWD/scripts/process:$PATH && \
export PYTHONPATH=$PYTHONPATH:$PWD/scripts
```

```
#Install Hecaton locally

bash install.sh
```

```
####################A_3 install_Hecaton.sh######################################
```

SCRIPT/PROGRAM NAME:

install_Hecaton_dependencies.sh

PURPOSE:

To install dependencies required for running Hecaton locally.

APPLICATION IN THIS PROJECT:

The successful execution of Hecaton requires the dependencies installed. This bash script was used to install GRIDSS, PICARD, and SPEEDSEQ, repectively, and add them the $PATH environment variable.

```
#########################A_4 install_Hecaton_dependencies.sh#############################
#!/bin/bash
#Install dependencies
mkdir hecaton_deps && \
cd hecaton_deps && \
wget https://github.com/PapenfussLab/gridss/releases/download/v2.0.1/gridss-2.0.1-gridss-jar-with-dependencies.jar && \
export GRIDSS_JAR=$PWD/gridss-2.0.1-gridss-jar-with-dependencies.jar && \
wget https://github.com/broadinstitute/picard/releases/download/2.18.23/picard.jar && \
export PICARD=$PWD/picard.jar && \
source activate hecaton_py2 && \
git clone --recursive https://github.com/hall-lab/speedseq && \
cd speedseq && \
make align && \
make sv && \
make config && \
export PATH=$PWD/bin:$PATH && \
source deactivate && \
cd ../..

############################# A_4 install_Hecaton_dependencies.sh########################
```

SCRIPT/PROGRAM NAME:

run_Hecaton.sh

PURPOSE:

To conduct CNVs analysis using Hecaton.

APPLICATION IN THIS PROJECT:

This bash script contained 3 commands for running Hecaton. The first command was used to generate the index of reference genome. This step only needs to be run once for every reference genome. The second command was used to run Hecaton on a set of paired-end reads. The output BEDPE files can be found in the random_forest_calls folder. The third command was for converting BEDPE output files to VCF format. In this study, the default cutoff (0.7) was used. The VCF format file can be manually parsed to screen for tandem duplication.

```
###############A_5 run_Hecaton.sh ###############################################
#!/bin/bash


#pre-processing the reference genome
bash bash/preprocess.sh genome.fa


#run Hecaton on a set of paired-end reads
nextflow run -c nextflow/nextflow.config -w hecaton_workdir nextflow/hecaton.nf --
genome_file genome.fa --reads "reads{1,2}.fq" --manta_config
docker/configManta_weight_1.py.ini --output_dir output --model_file
models/random_forest_model_concat_A_thaliana_ColxCvi_O_sativa_Suijing18_coverage_10x_
insertions_balanced_subsample.pkl
#convert BEDPE output to VCF
#output files can be found in the random_forest_calls folder
source activate hecaton_py3
scripts/convert/bedpe_to_vcf.py -i output.bedpe -o output.vcf -s name_of_your_sample
bgzip output.vcf
tabix output.vcf.gz
duphold -t number_of_threads -v output.vcf.gz -b alignment_of_this_sample.bam -f reference.fa
-o output_duphold.vcf
bgzip output_duphold.vcf
```

```
tabix output_duphold.vcf.gz

source deactivate
```

```
#########################A_5 run_Hecaton.sh ####################################
```

SCRIPT/PROGRAM NAME:

steps_before_running_Hecaton.sh

PURPOSE:

To add dependencies and environment language to $PATH to ensure that Hecaton can be run

APPLICATION IN THIS PROJECT:

Execution of Hecaton requires several dependencies and workflow language. This bash script was used to add these dependencies and workflow language to $PATH environment variable to ensure the Hecaton can be executed correctly.

```
#########################A_6 steps_before_running_Hecaton.sh ###################
#!/bin/bash


cd hecaton
export PATH=$PATH:/usr/local/bin/seqtk
export PATH=$PATH:/usr/local/bin/miniconda3/bin
export PATH=$PATH:/home/gwang
cd hecaton_deps
export PATH=$PWD/scripts/collapse:$PWD/scripts/convert:$PWD/scripts/filter:$PWD/scripts/genotype:$PWD/scripts/gridss:$PWD/scripts/intersect:$PWD/scripts/predict:$PWD/scripts/process:$PATH
export PYTHONPATH=$PYTHONPATH:$PWD/scripts
export GRIDSS_JAR=$PWD/gridss-2.0.1-gridss-jar-with-dependencies.jar
export PICARD=$PWD/picard.jar
cd speedseq
export PATH=$PWD/bin:$PATH
cd ../..



#################A_6 steps_before_running_Hecaton.sh ##########################
```