Contig Farmer: A tool for extracting maximal-length contiguous sequences from a database of short sequence reads

Ryan C. Thompson, Paul J. Rushton, Tom W. Laudeman, and Michael P. Timko

University of Virginia Department of Biology

June 2009

Contents

Abstract

Here we present Contig Farmer, a tool for improving the length and depth of coverage of contigs gener- ated from a database of short sequence reads. Contig Farmer works without assembling the entire database and has only modest hardware requirements. The underlying methodology of Contig Farmer is iterative growth of seed contigs using repeated search and assembly. The utility of Contig Farmer is demonstrated on the sequences in TOBFAC, the database of tobacco transcription factors. Contig Farmer successfully grew the TOBFAC contigs, both in length and in depth of coverage, to yield a larger, higher-quality set of contigs.

1 Introduction

The invention of DNA sequencing technology has caused a massive shift in modern biology. The nucleotide sequences of genes have become primary objects of study. This shift has had far-reaching effects; for example the fact that we can now know the physical difference between two alleles of the same gene (that is, the altered base) is a constant source of confusion to new students attempting to learn the once-clear distinction between phenotype and genotype. Many species' genomes have already been fully-assembled, including our own, and the rate of sequencing continues to grow exponentially, thanks to new 2nd-generation sequencing technologies such as Roche 454 and Illumina sequencers. However, regardless of the rate of generation of new sequences, the fact remains that the *de novo* sequencing, scaffolding, and assembly of a full eukaryotic genome remains a daunting task — difficult, time-consuming, and expensive.

In many cases, the research goal or question is focused on a particular aspect of biology. Often, with such focused goals, study of the full genome sequence is unnecessary and a more cost and labor efficient approach may be taken. Biologists studying one or several genes do not require the entire genome sequence. Rather, they need only the sequence of their several genes of interest, along with several thousand base pairs (bp) of flanking sequence on each side. This region will usually contain all the genomic elements involved in the transcriptional regulation of the gene. If this relatively small contiguous region is all that is required, then scaffolding and assembling a full genome sequence for the organism under study is unnecessary.

Alternative strategies exist for obtaining sequence for a majority of expressed genes in a genome at a fraction of the time, cost, and effort. One such strategy is "gene space sequencing" (GSS) [1, 2]. The technique takes advantage of one mechanism by which organisms, especially plants, control transcription: cytosine methylation. Higher plants often possess large amounts of repetitive DNA sequences. For example, the tobacco species Nicotiana tabacum resulted from an interspecific hybridization between two closely-related species with equal chromosome numbers [3]. This means that large parts of the tobacco genome are approximately duplicated. In order to manage this duplication, higher plants have evolved extensive systems of cytosine methylation to suppress duplicate expression [4]. The extent of cytosine methylation

is much greater in plants than in other kingdoms, owing to the greater incidence of genomic duplication events in the plant kingdom.

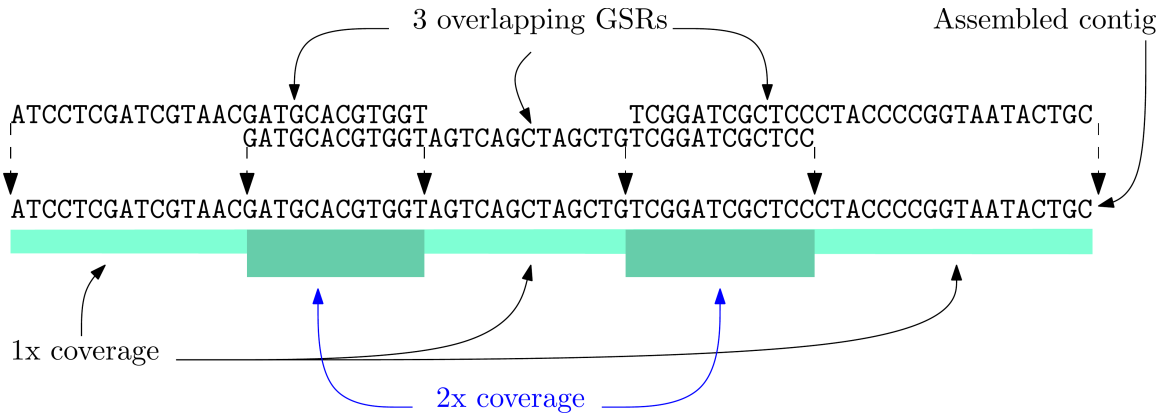
Plant biologists have long known about and taken advantage of differential methylation in plant genomes in order to enrich for the actively-expressed member of a set of duplicate genes. Gene space sequencing consists of cloning plant DNA into strains of E. coli that restrict methylated DNA (methylation filtration), and then sequencing what remains [4]. The effect is that the sequence database produced in this way is significantly enriched in hypomethylated regions of the plant's genome, precisely the regions containing the majority of expressed genes. These re- gions are called the "gene-rich space" or "gene space," from which the technique, gene space sequencing (GSS), takes its name. Our lab is currently using GSS data from the tobacco and cowpea genomes.

While generating and using a GSS database in lieu of a fully assembled genome savesa lot of time and money, it introduces its own set of problems that must be overcome in order to effectively carry out genomic studies. In generating a GSS database, a large fraction of the genome is necessarily excluded from sequencing. This makes it impossible to assemble or scaffold anything approaching a full genome. The raw sequencing data is simply that: a set of sequencing reads, with no associated mapping information. In particular, it cannot be known, a priori, whether any two sequencing reads from the database map to adjacent or overlapping locations, or whether they map to different chromosomes entirely. Any contigs must be assembled purely based on exact matches of their sequences at one end. This process is complicated by errors introduced by the process of reading a DNA sequencing trace [5, 6]. These errors force the assembler to accept less-than-perfect alignments as true overlaps, which in turn leads to erroneous assembly of similar but distinct sequences.

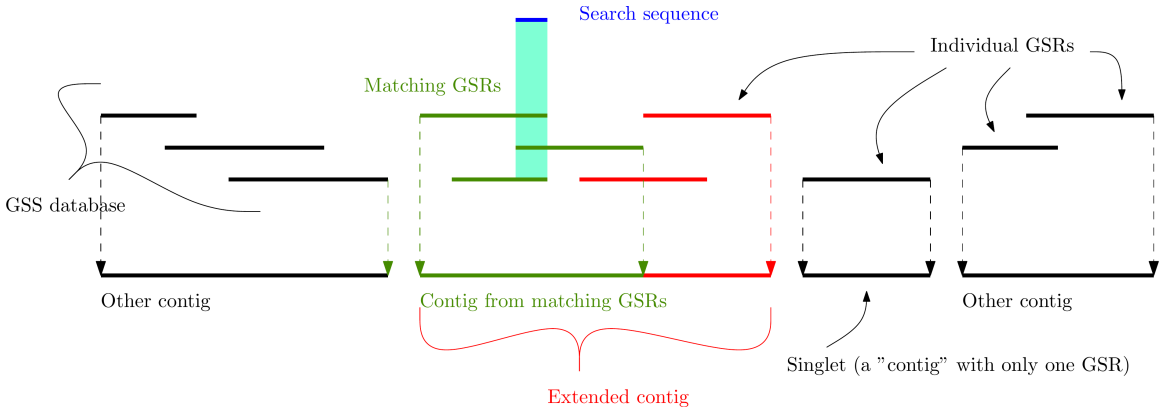
A "read" is simply the result of a single sequencing run. If that read is part of a GSS database, it is called a gene space read (GSR). A contig is a set of individual sequencing reads that have overlapping sequences. Taken together, they form one large sequence that is longer than any individual read (Fig. 1a). Another useful aspect of a contig is that wherever the individual sequences overlap and agree, we can be more confident that the contig sequence is correct, because that sequence is independently confirmed by two (or more) reads [7]. The number of reads that overlap a position in the contig is referred to as the depth of coverage of that position, and is a measure of confidence. See Fig. 1b for an explanatory example.

Manually assembling a contig from GSS reads around every sequence of interest represents a monumental effort, but having a large contig containing the sequence of interest is highly desirable, for the reasons mentioned above. Individual reads are, on average, 600bp in length. This means that without a contig, we have no knowledge of what lies further than 600bp either up- or downstream of the sequence of interest. A 600bp stretch of DNA is almost never sufficient to contain more than a single exon of a gene, and will often miss essential regulatory elements, such as the promoter region.

Thus, contig assembly and curation represented a major time investment in the preparation of TOBFAC, the database of tobacco transcription factors [3]. The primary method of finding sequences in the tobacco GSS database was by tblastn searches with amino acid sequences from Arabidopsis thaliana. The results from



(a) An example of 3 overlapping sequencing reads that can be joined into a longer contiguous sequence called a contig. Note that the diagram does not show complementary base pairing, but alignment of the ends of sequences to each other. Where the sequences overlap, the coverage improves, as does the confidence in the correctness of the sequence.



(b) An example of the method of searching a database for reads (green) that match a sequence of interest (blue). Notice that this method fails to find some reads that could yield a longer contig (red).

Figure 1: Explanatory examples of contigs.

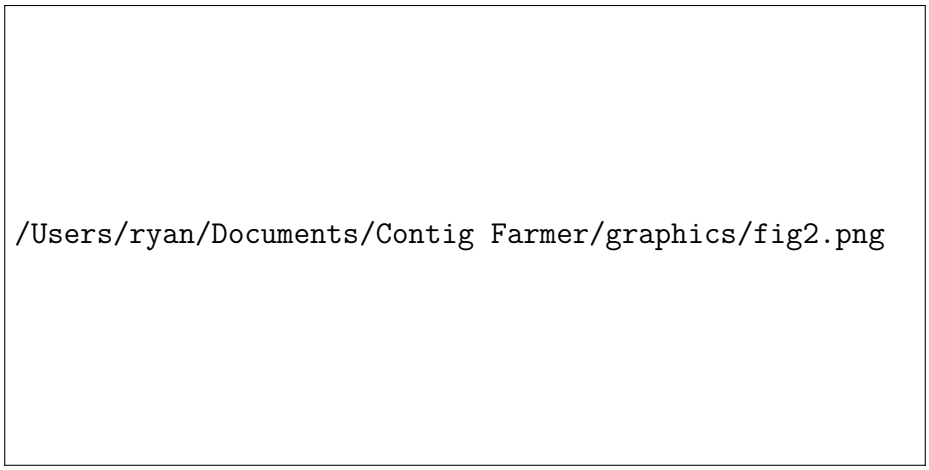


Figure 2: Illustration of an intervening intron preventing full assembly of a coding sequence. If the sequence of interest (blue) spans an intron boundary and the intron is longer than a single read, the matching GSRs (green) may assemble into two contigs instead of one (green, below), even if GSRs exist in the database that could bridge the gap (red). This is because the search sequence does not match the intronic GSRs, so they are not included in the assembly.

these searches were assembled into contigs and then curated. As such, the sequences in TOBFAC represent GSS database sequences encod- ing the DNA-binding domains of transcription factors in the tobacco genome. Since every input sequence for the assembly step necessarily contained a DNA-binding domain, the same 600bp limit still applied to the contig sequences in TOBFAC, since the contig sequence is limited by the input sequences. Accordingly, the majority of TOBFAC contigs are less than 1200bp in length (600bp upstream and 600bp downstream).

As a result, the authors note that many of the sequences in TOBFAC do not represent full genes, or even full DNA-binding domains [3]. Instead, these sequences only include the 5'- or 3'-ends of the DNA-binding domain. The other end is typically missing due to an intervening intron that is longer than the 600bp limit (Fig. 2). Some of these 5'- and 3'-fragments will represent both ends of the same genes, while others will represent separate genes. This makes it impossible to give an exact number of transcription factors in TOBFAC. Instead, a range must be given, based on the minimum and maximum possible number of 5'-to-3' pairings.

In addition to transcription factors, our lab also studies their targets — the regulatory elements in the promoter regions of other genes. A gene's promoter region can often map to a site several kilobases upstream of the coding sequence, so a 600bp read containing the coding sequence is insufficient. These and other similar problems led us to seek a way to automate the creation, and more importantly the extension and improvement of coverage, of contigs derived from GSS databases. This was the genesis of Contig Farmer.

Here we present Contig Farmer, a software tool designed for "growing" seed reads into the largest possible contigs, by pulling homologous sequences from the database

and assembling only the results. In so doing, it avoids having to assemble the entire database at once, which would require a large, powerful computer and a long time, and which may not yield desirable results. The key innovation, which allows it to grow contigs beyond the 600bp barrier, is that Contig Farmer executes multiple rounds of searching and assembly in an automated fashion. Each round potentially extends a contig further, as the growing ends of the contig match new sequences from the database. Critically, Contig Farmer does this while assembling only a tiny fraction of the entire database. This makes it suitable for running on a typical web server, desktop, or laptop, taking at most a few hours, usually much shorter.

The completion of a working implementation of Contig Farmer opens up many new opportunities for our lab. The production of an improved version of TOBFAC will be greatly accelerated by the automa- tion afforded by Contig Farmer. Perhaps more important than the increased length is the improved depth of coverage and corresponding increase in confidence that the sequences are assembled correctly and represent the correct primary sequence. Similarly, we expect the use of Contig Farmer to become a regular step in the study of new genes of interest from the tobacco or cowpea GSS data. Its speed, simplicity, and accuracy make it easy to do so.

2 Algorithm

The Contig Farmer algorithm takes, as input, a database of unassembled genomic sequencing reads (usually GSRs), and a subset of the reads in the database to be used as "seeds." The seeds may optionally be assembled into contigs. It is understood that the seeds should represent all the members of some family of sequences. The seed reads are designated "round zero" for the purposes of indexing in the steps described below, as well as to indicate that as the input, they represent the result of zero rounds of iteration. In addition to input sequences, the algorithm also requires a search program and an assembly program.

To generate the contigs for round N, where N is any positive integer, search the database for sequences homologous to each read from round N — 1. Note that every seed will necessarily match itself in the database, so the search results should invariably include every seed. Combine the results from all the searches and use the assembly program to assemble them into contigs. Finally, discard every contig that does not contain at least one seed read. Successive rounds may be generated until the contigs are as long as desired, or until the output of one round repeats exactly the output of a previous round. See Fig. 3.

3 Genesis, Rationale, & Comments

The basis for the Contig Farmer algorithm came from the data-mining strategy used in constructing TOBFAC, the tobacco transcription factor database [3]. Briefly, the strategy entailed searching with a number of sequences of interest, pooling the resulting sequences, and then assembling the sequences into contigs. This strategy worked

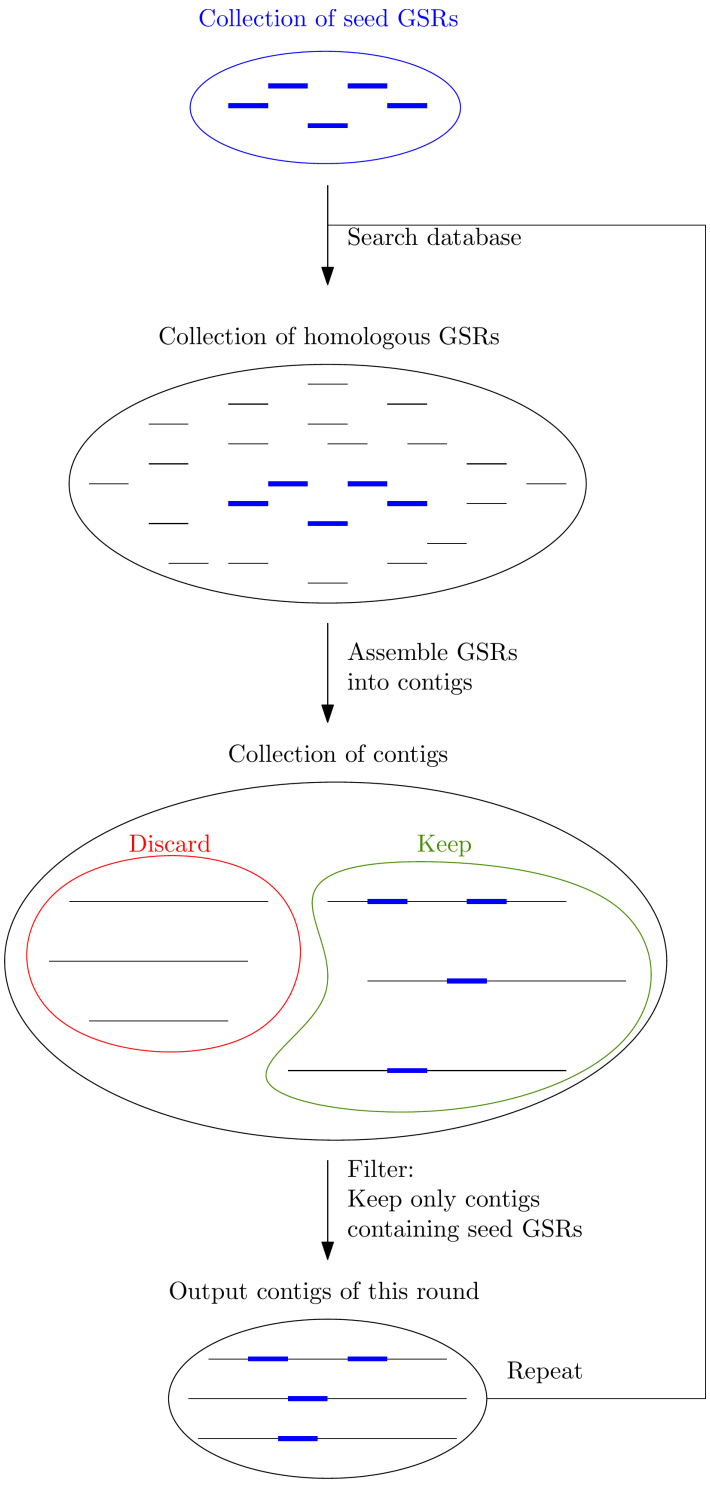


Figure 3: Flowchart of Contig Farmer algorithm. First, a database is search with seed GSRs. The results are assembled into contigs, and contigs that do not contain seed GSRs are discarded. Those that remain are the output of the round, and become the input to the search step of next round. Since each round produces larger contigs, the search results of each round will be expanded relative to the previous round. Thus contigs are progressively grown.

well, but was limited, because every result from the searches had to contain part of a sequence of interest. Since average length of the reads from the TGI data is about 600bp, the approximate maximum length for any contig is 1200bp: 600bp both up- and downstream from the sequence itself. Indeed, the majority of TOBFAC sequences are shorter than this limit — only 14 (out of 64) families have a 3rd quartile length greater than 1200bp, and only 5 have a 3rd quartile length greater than 1300bp.

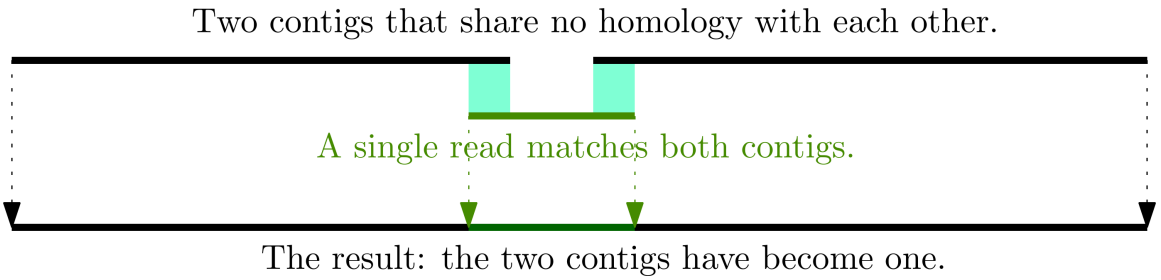
To remove this limit, our first thought was to assemble the entire set of reads into contigs, and then to search those contigs. However, we quickly learned that a GSS database could not be reliably assembled without a pre-existing scaffold, and we also lacked the computing resources to assemble it. In order to select more sequences to assemble without pulling in the entire database, we posited that repeating the original search-pool-assemble procedure using the result contigs as the new search sequences would yield yet larger contigs. This could be repeated an arbitrary number of times, until the set of contigs stabilizes, and additional iteration does not extend any contigs. That is, in principle, the procedure could be iterated to stability, though in practice 10 rounds is usually more than sufficient.

In our initial testing, we quickly realized that our algorithm was missing an important step: filtering. The search step returns only sequences that have homology to the seed sequences, but there is no guarantee that the seed sequences are the best match for every search result. Many of the reads are in fact more homologous to each other than to the seeds. During assembly, these reads will form their own orphan contigs that have no direct relation to the seeds. Worse, in subsequent rounds these derelict contigs could grow and spawn additional derelicts. These unrelated contigs needed to be filtered out after the assembly step of each round.

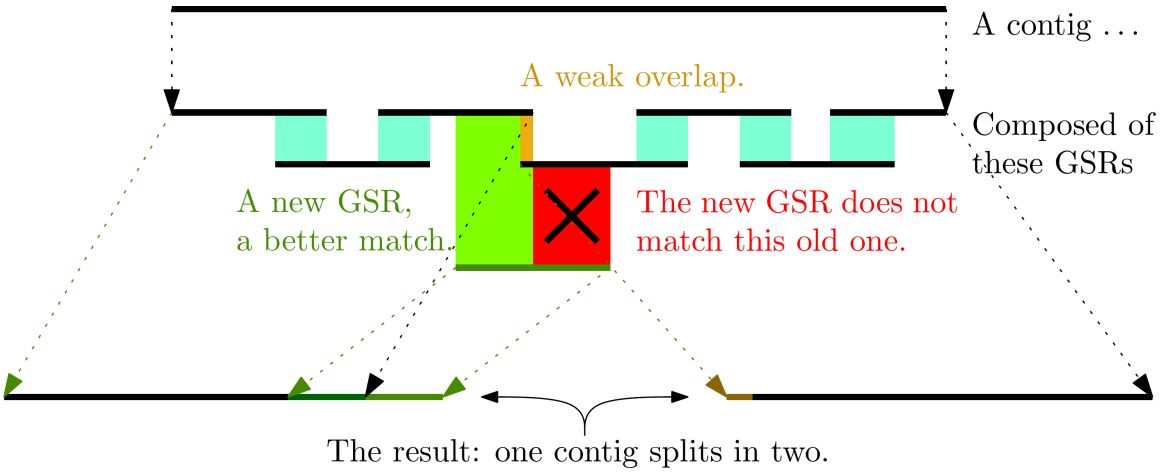
In the TOBFAC procedure, the contigs were manually curated after assembly, and ones that seemed wrong were discarded [3]. This kind of repeated manual curation was a clear impediment to automation, so a filtering algorithm had to be devised to systematically select acceptable contigs and discard the rest, all without human input. We settled on a simple algorithm: reads from the initial contigs would be marked as privileged, and after each round, any contig that did not contain at least one such read would be discarded.

Inherent in this filtering algorithm is a subtle but important assumption. Essentially, the filter assumes that at least part of every "sequence of interest" was already in the seed data. That is, it assumes that searching will not turn up any additional sequences of interest. If any such sequences were turned up during a search, they would assemble into independent contigs that would not contain any seed reads. These contigs, and the new data that they contain, would therefore be erroneously flagged as orphans and excluded by this filtering algorithm. Thus, the Contig Farmer algorithm truly works as its name suggests, and only grows contigs from existing seeds. In principle, a different filtering algorithm could be substituted, based on stringent homology alignment to the seeds, rather than exact matching, if more seeds were expected to be found.

Note that the algorithm does not make direct use of the contigs from the previous round. The previous round's contigs are only used to select the query reads for the search step. This ensures that a round's assembly step is not biased toward previous



(a) If two contigs assemble with opposite ends of the same GSR (green), they will merge into one large contig.



(b) If two of the GSRs in a contig have a sub-optimal overlap, they may be assembled incorrectly (yellow). If the correct GSR is found (green), it will out-compete the sub-optimal overlap and displace the bad GSR (red). The result is the split of the original incorrectly assembled contig into two smaller contigs.

Figure 4: Schematic illustrations of contigs splitting and merging as a result of newly-found GSRs.

assemblies — newly discovered reads and reads from the previous round compete equally during the assembly step. If a new read assembles in a position that displaces an old read, the result will be a contig that splits into two (or more) smaller contigs, relative to the previous round (Fig. 4b). Conversely, if a read is found that matches two previous contigs, one on each end, the result will be a joined contig comprising the original two contigs from the previous round, along with the bridging read (Fig. 4a). Thus, while all the original seed reads are preserved, the original contigs can merge and split throughout a run. Both splitting and merging of contigs are desirable — a split eliminates what was likely an incorrect assembly, while a merge yields a longer contig.

While the Contig Farmer algorithm could be run on a single seed GSRs at a time, running it on a set of related GSRs has a number of advantages. Foremost is the potential for joining multiple contigs together as described above, which grows larger contigs in fewer rounds. Additionally, combining the search results from multiple searches gives each GSR more chances to match and assemble with others. The com-

bining of search results also eases the computational burden, because many of the GSRs are expected to appear in multiple searches, since the seed GSRs are homologous to each other. If the search results were not combined, the assembly step would have to consider these repeated search results as many times as they appear. But with combining, each GSR is considered only once in the assembly step, regardless of the number of searches in which it appears. Lastly, the pooling of search results into a single assembly guarantees that the resulting set of contigs is internally consistent, because the assembler will not allow the same GSR to be incorporated into multiple non-overlapping contigs. Thus, the contigs in a particular round will represent disjoint sets of GSRs, as expected.

Relative to contigs from a fully-assembled database, the contigs produced by Contig Farmer are expected to be as long or longer, provided that the search step is sufficiently sensitive and enough rounds are carried out. For any given read, every read that could possibly assemble with it should be found in the search step, just as in a full-database assembly. However, since Contig Farmer excludes a large part of the database, the included reads have much less competition and therefore have more chances to assemble with each other, when the correct assembly would be with a read that was not in the search results. Thus, the Contig Farmer algorithm should be at least as sensitive as a full assembly, with some loss in selectivity, while greatly reducing the required computation relative to a full assembly.

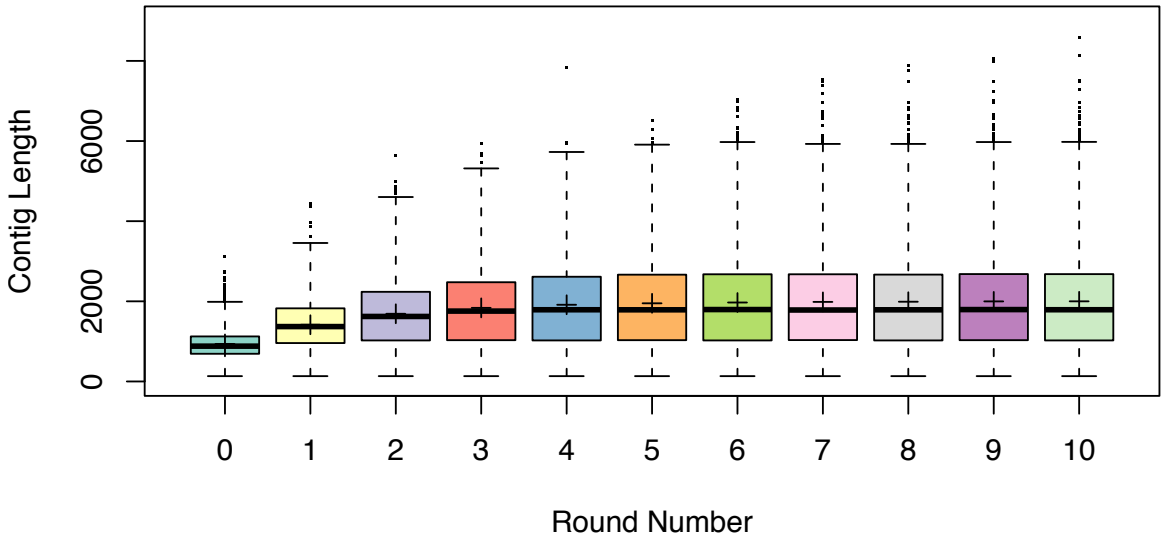
4 Results

At this time, Contig Farmer has only been applied to the sequences in TOBFAC. Though Contig Farmer was run independently on each of the 65 families represented on TOBFAC, the results are pooled together here for the purpose of summarization.

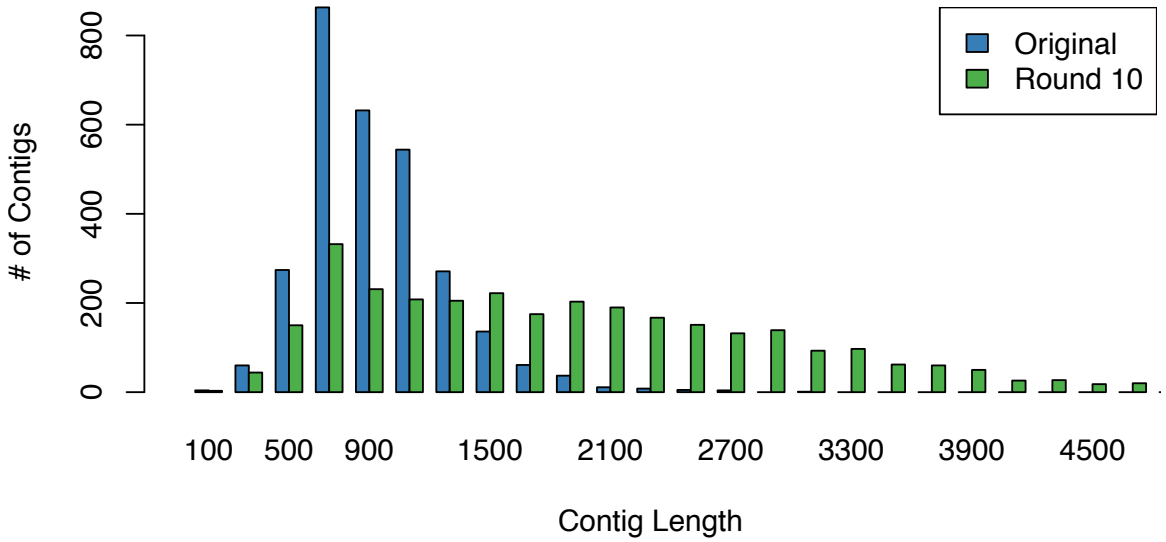
4.1 Contig Growth

Contig Farmer has greatly improved the length of most contigs in the TOBFAC data set. Fig. 5a shows the distribution of contig lengths in each round of the Contig Farmer run. The median length has increased from 881 to 1790bp. The data show that nearly all contigs were finished growing at round 5. However, the few upper outliers continue to grow linearly at a rate near 500bp per round all the way through round 10. This suggests that each contig tends to grow linearly at a rate of one half-read-length (250 to 300bp) per round on each end, until that contig reaches a round in which the search finds no additional GSRs that assemble onto it. At that point, the contig's length stabilizes. A plot of the number of contigs still growing vs round number would likely resemble the survival curve of an aging population, starting out large and decaying to zero. Unfortunately, the identities of the contigs in one round are lost in the next round, making it difficult to track individual contigs through multiple rounds. As a result, generating such a "survival curve" is not feasible.

Fig. 5b gives a more detailed comparison of the length distribution of the TOBFAC input data compared with the round 10 output of Contig Farmer. The output



(a) Box-and-whiskers plots of contig length vs. round number for the TOBFAC data. Each of the families of transcription factors on TOBFAC was run through 10 rounds of Contig Farmer. The boxplots are for the combined data. Black bars in the center of boxes represent the median contig lengths. Boxes represent the interquartile range (IQR), whiskers represent 2 IQR above or below the quartiles, and single points represent outliers. Plus signs represent the mean lengths.



(b) Side-by-side histogram of TOBFAC sequence length distribution before (blue) and after (green) Contig Farmer.

Figure 5: Two representations of Contig Farmer's improvements in contig length.

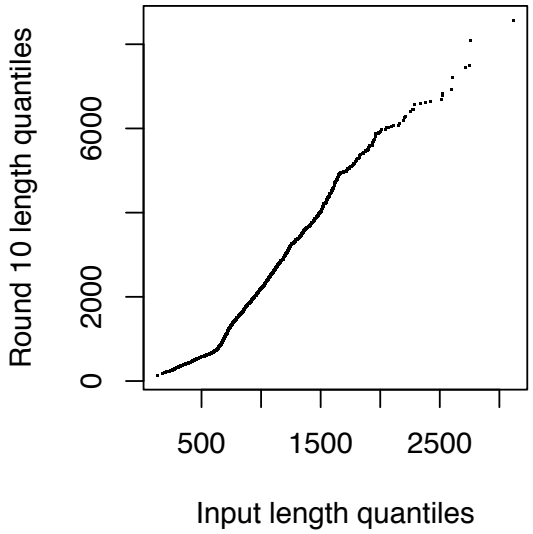


Figure 6: Quantile-quantile plot of contig length distributions. On the x-axis are the quantiles of the original TOBFAC contigs, and on the y-axis are the quantiles of the round 10 output of Contig Farmer. An approximately straight line indicates a similar distribution (accounting for location and scaling factors).

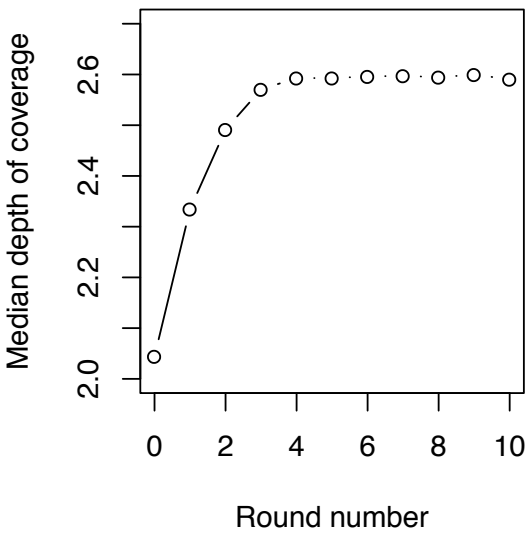
distribution has a very fat right-hand tail, indicating that while the median is only about 1800bp, many contigs exceed the median by a large margin. A quantile-quantile plot of the input and output distributions (Fig. 6) reveals that the output distribution output is mostly just a scaled-up version of the input distribution, except that the left tail is left unscaled. The unscaled region includes all the lengths less than about 600bp, the mean length of one contig. The quantiles below 600bp represent singlet GSRs that cannot be assembled with any other sequences in the database. It is therefore expected that Contig Farmer was unable to grow them at all.

4.2 Depth of Coverage

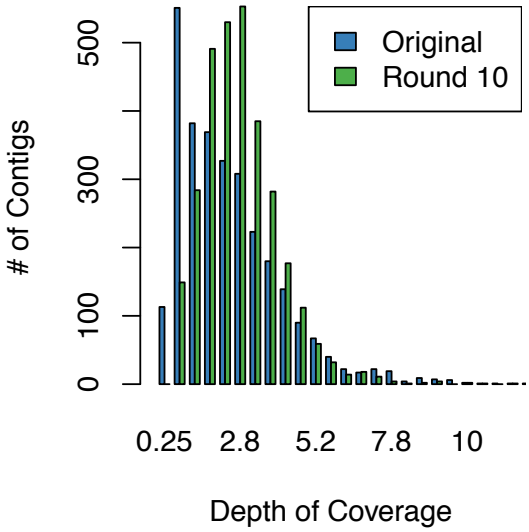
In addition to growing the TOBFAC contigs laterally, Contig Farmer was also able to improve the depth of coverage, though the improvement was not as dramatic (Fig. 7). The median depth of coverage was initially 2.0, indicating that on average, each base was overlapped by two GSRs. The median depth of coverage increased to 2.6 by round 4 and remained there through round 10. In this it shows similar behavior to the median contig length, which also stabilized in 4 to 5 rounds.

4.3 Splits and Joins

Since the Contig Farmer algorithm is designed to allow contigs to split and join as necessary, we also decided to investigate the proportions of contigs in TOBFAC that split and joined during the run of Contig Farmer. The results for all 65 families are listed in Appendix A. In total, 170 out of 2911 contigs (5.8%) joined with others, while 379 contigs split (13.0%). The number of output contigs always equals the



(a) A plot of median depth of coverage vs round number.



(b) Side-by-side histogram of the depth of coverage of the original TOBFAC contigs vs. that of Contig Farmer's round 10 output.

Figure 7: Two representations of Contig Farmer's improvements in depth of coverage. Coverage improvements were marginal compared to length improvements, yet still significant.

starting number plus the number of splits less the number of joins. A plot of output vs input number of contigs is shown in Fig. 8. Since there are more splits than joins, the data lie above the identity line.

Some families exhibit interesting patterns of splits and joins. For example, the LFY (also called LEAFY) family had 7 contigs initially, and 5 of them joined with other ones, leaving only two contigs in the final output. This is consistent with the fact that the LEAFY gene in A. thaliana (Accession `NP_200993`) contains a repeated domain. The tobacco homologs of LEAFY likely also contain repeated domains, separated by introns, and Contig Farmer successfully bridged these introns. Similarly, each family's pattern of splits and joins will depend on the typical intron-exon structure and other properties of that family's gene sequences.

5 Implementation

5.1 Development

Initially, as a proof of concept, we manually carried out several rounds of the algorithm on a few small families from TOBFAC, using BLAST as the search tool and Phrap as the assembly tool. This yielded encouraging results. The resulting contigs were longer and had better coverage. Furthermore, we observed a few instances of the expected splitting and joining of contigs.

We next implemented the algorithm in Perl and applied it to all 65 families,

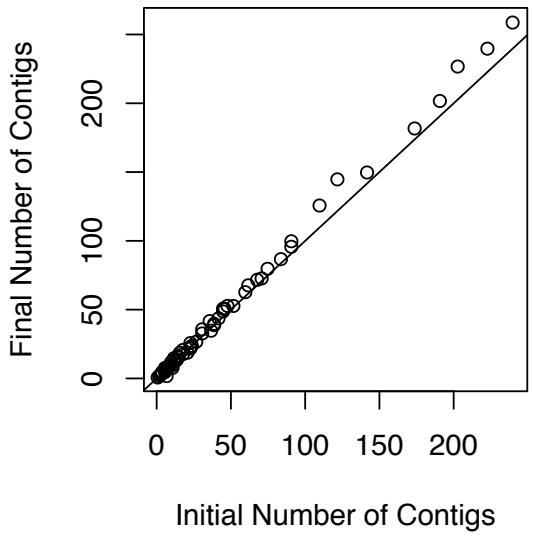


Figure 8: A plot of the number of output contigs vs. the number of input contigs for each family in TOBFAC. The identity line is graphed for reference.

again using BLAST and Phrap for search and assembly, respectively. The script ran successfully on most of the families, with similarly impressive results. However, in several cases, the script failed inexplicably, and the Perl code had become too complex to debug, so for the release version, we decided to rewrite Contig Farmer using the R programming language.

5.2 Release Version

The version of Contig Farmer included with this work is written in the R programming language. The code was developed and tested on Linux, and will likely not work on non-UNIX-like platforms. The code will be made available online at http://people.virginia.edu/∼rct2c/contig-farmer/.

Contig Farmer uses NCBI's Mega BLAST tool for the homology search step, and Phrap for the contig assembly step [8, 7]. Mega BLAST was selected over BLAST because it is designed for aligning sequences that are only expected to differ by sequencing errors, which is exactly appropriate to this application. It was also selected because it has a similar interface to BLAST, the previously used tool. This allowed some code reuse that simplified the development of the final version. Compared to BLAST, we found that Mega BLAST returned much more relevant results, making the filtering step less computationally intensive and reducing the time required to run Contig Farmer. Phrap was selected because we already have a license for it, and for consistency with the TOBFAC sequences, which were assembled using Phrap. However, nothing about the design of Contig Farmer limits it to these two tools. In principle, any sequence alignment tool and assembly tool could be substituted. Furthermore, the code is modularly designed, so any such future modifications would require minimal changes to the code.

Because Contig Farmer uses Mega BLAST as its search tool, the target database

must be formatted by NCBI's formatdb tool (the same tool used to prepare databases for normal BLAST searches). All files with sequences in them are stored in FASTA format, with the additional requirement that each sequence's header line must contain an element of the following form that specifies the database reads that compose it: `[component _gsr=GSR_ID1|GSR_ID2|GSR_ID3|...]`. This is required even if the sequence is itself a read from the database, rather than a contig composed of such sequences. This restriction may be removed in future versions.

When run, Contig Farmer will automatically create a result directory with a name derived from that of the input file, and all output produced will go inside this directory. (In particular, CF will append "cfresults.d" to the input file name to generate the name of the result directory.) After a successful run, the results directory will contain a fasta file for each round (including a copy of round zero, the original sequences), a file called "cfarmer.rda" containing all the data in R's binary format, and a log file, containing all the output produced by the program. Note that Contig Farmer keeps a copy of each round primarily for its own internal purposes, and most users will only be interested in analyzing the final round (by default, round 10). The data file contains both the options passed to the program on the command line as well as the data for each round. This data file can later be loaded in an interactive R session and analyzed. (For example, the figures in this document were prepared from saved data files.) The log file also includes the options passed to the program, as well as an entry for each completed round, with some summary statistics computed on the contigs for that round. Each log entry is timestamped, to facilitate accurate record-keeping.

Contig Farmer will automatically detect and attempt to resume from a pre-existing results directory. This can save time when running additional rounds on a case-by-case basis after a batch run ona large number of files. When it resumes, Contig Farmer will log the new options and a summary of all the previously completed rounds before continuing where the previous run left off. It does not read the options from a previous run, nor does it check that the previous run's options are the same as the current run. This may be addressed in later versions, although it could also be considered a feature that enhances the flexibility of the program by allowing the user to change options after several rounds. For example, if a new updated database with more reads becomes available, it may be advantageous to resume farming from the latest round while switching to the updated database.

6 Discussion

In evaluating the effectiveness of Contig Farmer as a tool for improving contig quality, it is important to be confident that the output is truly an improvement. One useful metric in the area has already been treated: depth of coverage. Even though an increase from 2.0 to 2.6 in median coverage may seem modest, the confidence in the correctness of a sequence goes up exponentially with the depth of coverage, so this increase is magnified.

As a more qualitative metric, inspection of individual contigs from the output of Contig Farmer often reveals that they better match the typical sequence motifs of the

family to which they belong, compared to the original contigs. This likely results from resolution of incorrect assemblies by splitting and joining. In a particularly striking example, Contig Farmer found a previously-characterized WRKY-family transcription factor called WIZZ (Accession AB028022, [9]) in the TOBFAC contigs. WIZZ was previously believed not to exist in the database, but in fact all the GSRs containing the sequence for WIZZ were in TOBFAC, but in separate contigs, assembled incorrectly. Contig Farmer successfully split these GSRs out of their original contigs and put them together to recover WIZZ.

The Contig Farmer program has only recently been finished, and we have yet to apply it to its full potential by integrating it into our existing genomics pipelines and procedures. However, the preliminary data from applying it to the sequences in TOBFAC are very encouraging. Now that the base algorithm has been implemented, the potential exists for a number of improvements, optimizations, and additions. These can be divided broadly into improvements on the underlying algorithm and technical improvements on the implementation.

On the algorithmic side, we would like a way to relax the "seed" property of Contig Farmer. That is, we want a way to use the Contig Farmer data in a "discovery" mode in addition to the current "farming" mode. One way to do this is to relax the filtering criterion. Instead of requiring contigs to exactly include a seed read, they could be kept on the basis of mere homology. The threshold level of homology would have to be carefully tuned to balance sensitivity and selectivity of the contig extension.

An alternative to changing the filtering criteria would be to keep the discarded contigs from each round in a separate data set from the accepted ones. After the last round, all the "discard bins" could be pooled together and assembled to yield a set of "second-hand" sequences. These could be automatically searched for homology to the seed reads, sorted based on the best search results, and manually inspected for potential additional sequences of interest.

Another algorithmic improvement relates to the performance of the assembly program, phrap, which can be greatly improved by providing certain metadata on the input sequences. In particular, phrap can use quality values generated from sequencing traces to resolve mismatches in assembly — in case of a mismatch, it can take the base with a higher quality value [7]. Additionally, phrap can use the fact that the same sequence is often read from both ends — a forward and a reverse read, to yield a more accurate confidence measure. This feature requires that the sequence's names reflect whether they are forward and reverse reads. Both of these types of meta data are likely available, and now that we know Contig Farmer works, the next step is to obtain this information for our GSS data and use it to improve the assembly.

Additionally, phrap also produces some useful information that Contig Farmer does not make use of at present. Many sequencing reads are identical to others, except that a segment is missing from the middle of the sequence [7]. These are called "deletion reads," and they can interfere with the assembly if they are not detected and culled. Phrap detects and reports these, and future versions of Contig Farmer should take this information into account, possibly by "blacklisting" those reads and excluding them from subsequent rounds.

On the technical side, one of the first priorities should be to implement the early-

termination part of the algorithm. Contig Farmer will currently always perform the requested number of rounds, even if one of those rounds is identical to a previous round, indicating that all further rounds will be repeats of previous rounds and are hence unnecessary. In the future, Contig Farmer will test for repeated rounds and automatically stop to avoid doing extra work. A possible improvement on the early termination

method would be to selectively exclude contigs from later rounds once they stop extending from round to round. The analysis of the length distributions revealed that very few contigs continue growing for more than 5 rounds. Selective exclusion of nongrowing contigs would allow Contig Farmer to quickly finish the last few growing contigs in later rounds without getting bogged down by the rest. Since the computational burden of the algorithm currently increases with each round as the contigs grow, this improvement would be a welcome addition, as it would counteract that trend. With these and other optimizations, Contig Farmer can become even faster and lighter than it currently is.

Another useful technical improvement would be an overhaul of the interface. The best way to restructure the command-line interface would be to use Perl to write a wrapper script that reads the command-line options and passes them to the real program in a simple format. This would allow the command-line interface to be much more flexible, as Perl's option-parsing routines are far more advanced than R's. Furthermore, we plan to create a paste-and-go web form frontend to Contig Farmer, similar to the NCBI BLAST search page that everyone is familiar with.

Contig Farmer provides a quick, simple way to extract relevant sequence information from a sequence database without the initial overhead of having to assemble the whole database. As gene space sequencing becomes a more common approach to plant genomics, the usefulness of Contig Farmer, especially in our lab, will continue to grow. Furthermore, since Contig Farmer is designed for assembling contigs from a database of short reads, it may have an application in extracting longer sequences from 2nd-generation sequencing technologies, which produce very large numbers of short sequences.

While a full genome sequence is better suited for many purposes, producing one still requires a great amount of computation and labor. Contig Farmer's niche is to provide quick, cost-effective analysis with minimal hardware requirements for queries focused on a particular class or family of sequences.

References

[1] Xianfeng Chen et al. "CGKB: an annotation knowledge base for cowpea (Vigna unguiculata L.) methylation filtered genomic genespace sequences". In: *BMC Bioinformatics* 8.1 (2007), p. 129. ISSN: 14712105. DOI: 10.1186/1471-2105-8-129.

[2] Michael P Timko et al. “Sequencing and analysis of the gene-rich space of cowpea”. In: *BMC Genomics* 9.1 (2008), p. 103. ISSN: 14712164. DOI: 10.1186/1471-2164-9-103.

[3] Paul J Rushton et al. “TOBFAC: the database of tobacco transcription factors”. In: *BMC Bioinformatics* 9.1 (2008), p. 53. ISSN: 1471-2105. DOI: 10.1186/1471-2105-9-53.

[4] Mary Gehring and Steven Henikoff. “DNA methylation dynamics in plant genomes”. In: *Biochimica et Biophysica Acta (BBA) - Gene Structure and Expression* 1769.5-6 (May 2007), pp. 276–286. ISSN: 01674781. DOI: 10.1016/j.bbaexp.2007.01.009.

[5] Philipp L Wesche, Daniel J Gaffney, and Peter D Keightley. “DNA Sequence Error Rates in Genbank Records Estimated using the Mouse Genome as a Reference”. In: *DNA Sequence* 15.5-6 (2004), pp. 362–364. DOI: 10.1080/10425170400008972.

[6] B Chevreux. “MIRA: an Automated Genome and EST Assembler”. PhD thesis. German Cancer Research Center Heidelberg, 2005, p. 322. URL: https://books.google.com/books?id=ac0zOAAACAAJ.

[7] Phil Green. *Phrap*. http://www.phrap.org/. 2009.

[8] Zheng Zhang et al. “A Greedy Algorithm for Aligning DNA Sequences”. In: *Journal of Computational Biology* 7.1-2 (Feb. 2000), pp. 203–214. ISSN: 1066-5277. DOI: 10.1089/10665270050081478.

[9] K. Hara et al. “Rapid systemic accumulation of transcripts encoding a tobacco WRKY transcription factor upon wounding”. In: *MGG - Molecular and General Genetics* 263.1 (Feb. 2000), pp. 30–37. ISSN: 0026-8925. DOI: 10.1007/PL00008673.