1 Introduction

Revolutionary technological advancements have and will continue to play an important role in determining the power and utility of genomics and, in turn, its impact on all of biology just as was the case for molecular biology and recombinant DNA. PCR, DNA microarrays, and automated sequencers are only a few of many examples that dramatically changed how scientists conduct their experiments and what biological problems can be addressed. **Most recently, next generation sequencers (e.g., Illumina/Solexa Genome Analyzer or GA, Roche 454 GS FLX, ABI SOLiD) that produce enormous amount of sequence data are beginning to impact the field [19, 23].** Originally designed to provide higher throughput and lower cost for whole genome de novo sequencing or resequencing, it became readily apparent that these machines could be applied to other experimental questions and concepts [28]. Animal scientists will soon have great power to characterize genotypic variation and RNA levels within and across individuals and connect this variation with phenotypic trait variation.

Existing uses for next-generation sequencing include genome resequencing, which can detect SNPs and other small-scale sequence variation, and RNAseq, which is used to sequence entire RNA populations. Because of the large number of independent samples possible with these technologies, next-gen sequencing can be used quantitatively to count relative frequencies of SNPs, examine copy-number variation, and assay RNA abundance. Moreover, paired-end data can be generated with both small (300 bp) and medium (1-5 kb) insert sizes, permitting the detection of structural variation as well as RNA splice variants. Finally, selection protocols such as ChIP and array capture have been used to "filter" DNA populations prior to sequencing in order to provide quantitative information about protein binding and chromatin states, as well as targetted "deep sequencing" of specific genomic regions. **Over the next few years, next-gen sequencing will offer insights into regulatory regions, small noncoding RNA, and epigenetic phenomena such as methylation and histone modification. Our goal is to address these various applications by building easy-to-use tools to take in and analyze next-gen sequencing data.**

Next-generation sequencers offer several advantages compared to capillary sequencers (e.g., ABI 3730), which were state-of-art less than a few years ago. These benefits include: (1) no need to prepare libraries, which saves time and money, and may allow for the characterization of sequences recalcitrant to cloning, (2) highly parallel processing that allows 1-300 million molecules to be run at the same time, (3) 10+ Gb, and rapidly increasing, of total sequence generated per run, and (4) many applications (see below).

The disadvantages of these platforms, besides the high initial cost to purchase the instrument are (1) relatively short read lengths (typically 35-75 bases), (2) potentially higher error rates compared to Sanger sequencing, and (3) uneven coverage [11].

However, the most critical hindrance to using next-generation sequencing effectively is the lack of easy-to-use computational tools to support data analysis. Specifically, (1) large datasets are not easily manipulated, (2) there is no standard pipeline to handle the data, (3) a long analysis time is required even with high-performance compute clusters, (4) there is a lack of user interfaces. More importantly, most of the existing tools are designed for human, mouse, and other biomedical models in mind. What may not be readily apparent is that due to the short read lengths and the algorithms used, a high quality reference genome is required. While some agricultural animals have genome sequence assemblies, it is likely that they will never reach the quality achieved already

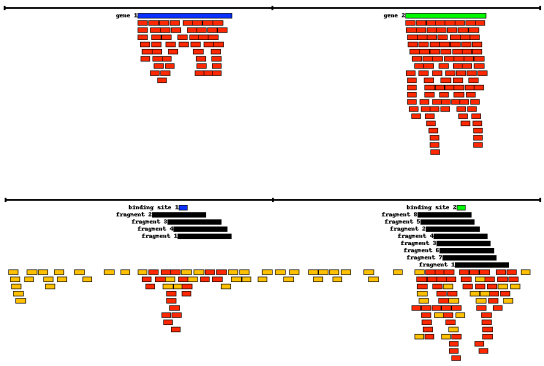

Figure 1: Two applications of next-generation sequencing to genomics. (a) RNA expression analysis; (b) ChIP-seq analysis.

for human and mouse. **In short, due to these problems, in order to utilize next-generation sequencing for agricultural animals, the community needs bioinformatics tools that regular bench scientists can use and that incorporate the unique needs and restrictions of agricultural animals.**

In this submission, we propose to fill this needed gap by building an open source analysis pipeline and associated Web interface to take in next-gen sequencing data, map it onto existing genomes and gene annotations, visualize the mappings, summarize the data, and extract digested data in formats that are easily manipulated and immediately useful to animal scientists. Input from the community and software evaluation will be conducted on a regular basis to ensure that our tools are widely disseminated and relevant to the user community.

1.1 Next-gen sequencing approaches offer immense opportunities to animal scientists.

Next-gen sequencing offers a systematic solution to many problems in agricultural animal genomics. Because next-gen sequencing can quantitatively sequence any population of DNA molecules, there are many potential applications; below are some of the applications most applicable to animal genomics.

First, next-gen sequencing can be used to resequence animals in order to assay known genomic variation and detect genomic sequence novelty, including single-nucleotide polymorphisms (SNPs) and the location of likely insertions or deletions (indels) [24].

Second, the deep sampling offered by current and future platforms enables quantitative analysis of genomic variation, e.g. providing an estimate of heterozygosity, minor allele frequency, and copy-number variation (CNV) [11].

Third, next-gen sequencing can be used to sequence RNA populations and define cSNPs, detect novel genes, find novel splice variants, and discover non-coding RNAs, all with high sensitivity. This is done by sequencing cDNA and mapping the reads back to known or predicted genes. Reads that do not map to known genes are candidates for novel genes, and can be used to determine putative exons [22]. This is extremely useful for genome annotation.

Fourth, quantitative RNA analysis with RNAseq can complement or replace expression microarrays. With the depth of sequencing already available, RNA species expressed in the 10s or 100s

of molecules per cell will yield multiple reads; more prevalent molecular species will yield more reads, and the ratios of the read counts (normalized for length of molecule) will represent the relative numbers of the molecules being compared (Figure 1 (a)) [22, 9].

Fifth, paired-end tag (PET) sequencing can be used to discover structural variation in the genome (large insertions, deletions, and inversions) [10].

Sixth, next-gen sequencing can be used to quantitatively measure chromatin immunoprecipitation pull-downs and methylation sites. By immunoprecipitating DNA bound by a particular protein (e.g. a transcription factor, or chromatin with a particular histone modification) and then sequencing it, the original genomic locations of the protein binding can be determined (Figure 1 (b)). As with RNA expression, the key is sequencing deeply enough that that the pulled-down DNA can be quantified [13].

Finally, next-gen sequencing offers exciting opportunities for comparative genomic analyses, where we can sequence the results of selective sweeps, examine conservation patterns across many animals, and cheaply assemble the genomes from phylogenetically closely-related species.

This list of examples clearly demonstrates how next-gen sequencing is already impacting animal genomics, and offers unprecedented opportunites for connecting structural and RNA variation with phenotypic traits.

1.1.1 Comparison to microarrays

A good illustration of the power of next-generation sequencing can be obtained by comparing it to microarrays, a previous technological revolution. In many ways, next-gen sequencing tackles problems not well addressed by microarrays. Because microarrays must be designed against a specific genome assembly, it is only possible for them to *detect* certain kinds of genomic novelty such as SNPs and indels; in contrast, because next-gen sequencing produces actual sequence reads, SNPs and indels can be completely *defined* by sequence. For another example, the detection of novel genes is possible with whole-genome tiling arrays, but defining novel splice variants is quite difficult with arrays because one needs to design oligos against specific splice sites. In contrast, deep sequencing of RNA populations provides reads that map across splice sites [22]. Next-gen sequencing already offers a wider dynamic range for RNA expression analysis and ChIP assays, where microarrays appear to saturate relatively quickly, leading to poor discrimination at very high or very low signals [13, 22]. Finally, **sequencing data can be kept and reanalyzed as new genome assemblies emerge and annotations are redefined, unlike microarrays, which must be redesigned**. This may be particularly important in agriculture, where a number of important genomes have not yet been finished and are being regularly updated.

So, while microarrays – especially expression arrays – are still cheaper then next-gen sequencing, it is likely that in the future, next-gen sequencing will largely replace microarrays.

1.1.2 Drawbacks to next-gen sequencing approaches

There are still a number of significant problems with next-generation sequencing data.

First, the reads are still relatively short. While the Illumina GA2 can produce reads that average 60-100 bases, this is not sufficient to do a de novo assembly of a large diploid genome or diverse RNA population. Instead, these reads must be mapped against known "reference" sequence, which engenders problems when the reference sequence is not complete or error-free. This is certainly

the case for existing agricultural animal genome assemblies.

Second, error characteristics and coverage biases for next-gen sequencing reads are machine specific and poorly understood; one study found that there was essentially no correlation between the different coverage biases present in ABI, Illumina, Roche, and Sanger reads [11], so statistical frameworks must be individually developed for different machines.

Third, a significant percentage of reads are "garbage" reads – often between 40% and 50% of reads cannot be mapped to the reference sequence [23, 11]. This is characteristic of the massively parallel colony approach where overlapping clusters of DNA molecules yield mixed sequence reads, and varies with the specifics of the source DNA and the sequencing protocol used and the expected number of garbage reads is not predictable. This causes problems when mapping reads to reference sequence: reads may not map because they are garbage, not because they represent novel sequence.

And fourth, the volume and characteristics of the data make it tricky to analyze.

1.1.3 Data analysis is challenging

Analyzing next-gen sequencing data is challenging for many reasons; **data analysis is the biggest hurdle in using this technology effectively.**

First, the volume of data from a single experiment is immense. Individual data sets are approaching and will soon surpass the size of the raw data from a typical mammalian whole genome shotgun sequencing project ($\approx$100 gb); storing and manipulating this data on personal computers is difficult and time consuming. For example, our recent ABI SOLiD dataset resequencing two inbred chicken lines was over 500 gb in sequence data, requiring transfer by hard drive!

Second, bias and error within next-gen sequencing data are poorly understood and individual to the machine type as well as the preparation protocol [11]; we've already seeen some of this in SNP detection in our own data, where ABI and Illumina show many different SNPs. It is not yet clear how big an impact this has on analysis results, but it will undoubtedly be as significant as the well-known inter-lab microarray bias. This especially highlights the need for simple tools to do comparative analysis between different data sets and different tools to determine bias and assess the reliability of conclusions drawn from next-gen sequencing data.

Third, no standard analysis pipelines are available. Although there are quite a few tools capable of mapping short reads to genome assemblies and cDNA data, this is only the beginning. Post-mapping analysis such as detecting SNPs, CNVs, and structural variation, or analyzing RNAseq data for novel genes and splice variants, must be done separately; correlation with existing annotations and already known sequence features is an additional step often requiring manual review.

While some labs have made post-mapping analysis software available [14], it is invariably difficult to run and aimed at only one kind of data [26]. In practice, this means that every lab invents a new pipeline or adapts another lab's pipeline in a non-standard way, as we have done for our own projects involving allele-specific expression and genome resequencing. Since mapping programs also typically have unique or idiosyncratic output formats, scientists must write their own conversion tools in order to compare the results of one mapping tool with another, which also decreases the ability to control for tool variation.

Every lab using next-gen sequencing data must currently develop their own analysis pipeline: in the past, genome assembly and analysis was primarily done at the big sequencing centers, which

could afford to develop in-house expertise; now every lab using next-gen sequencing data must do the same. Having common pipelines is important for analysis quality, standardization and comparison, and efficiency.

Fourth, mapping and analyzing next-generation sequencing data is computationally expensive. While the state of the art in mapping algorithms continues to improve – Bowtie is 10-50 times faster than maq, for example, requiring approximately 40 hours to map 1 billion reads to the human genome [16] – the volume of data is increasing faster than program speed is improving. Moreover, post-mapping analysis can be even more expensive: PEMer, a program for analyzing structural variants with paired-end tags, requires approximately 10,000 CPU hours to do a de novo mapping of 10 million reads from Roche's 454 platform [14]. This kind of compute power is only available in high-performance compute clusters such as Michigan State's HPCC, and it is not easy to use effectively.

Fifth, existing tools are generally not integrated with user interfaces, limiting the use of these tools to computationally adept users. Bioinformatics tools are often implemented on UNIX-based machines with command-line interfaces, for several reasons: physicists, mathematicians, and computer scientists are typically trained on UNIX; UNIX-based operating systems such as Linux and FreeBSD are freely available and come with an immense variety of scientific tools; command-line UNIX programs can usually be run in batch mode and across multiple compute nodes; and relatively few scientific programmers have the training or expertise to write a graphical user interface. Good or bad, this means that most bioinformatics tools are not accessible to experimental biologists but must instead be run by bioinformatics specialists. This in turn limits the overall use of tools to labs with the requisite expertise.

Sixth, existing tools and analytical approaches tend to be biased towards resequencing data from low-variation and high-quality "finished" genomes such as human and mouse, which limits their use in agricultural populations with substantial standing variation and incomplete assemblies. A driving force of the next-gen sequencing revolution is the 1000 Human Genome project, so the tools developed to deal with the data have focused either on sequence from lab organisms with low heterozygosity such as yeast or mouse, or on human data; their performance on genomes with reasonably high heterozygosity or unfinished assemblies will generally be poor. This is particularly important in agricultural animals such as chicken and cow, where large parts of the genome have incomplete coverage, are assembled into an "Un..."-chromosome, or have oddball contig orientation. Moreover, tools to map across significant error or variation between the sequencing data set and the reference genome do not exist; in particular, software to deal with short insertions and deletions, a source of substantial genomic variation, is not yet available. Because existing mapping algorithms are focused more on speed than on sensitivity of mapping, they may perform poorly on agricultural genomes [16]. We do not yet have an understanding of the likely impact of these problems.

Seventh, there are as yet no standards for data analysis, publication, or release, making it difficult to compare data between experiments and between labs. One lesson from microarray analysis is that **inter-lab comparison of data is important**: because early microarray data sets were not held to any format standards, and there were no standards for data analysis, data publication, and data release, platform and analysis pipeline comparisons were difficult. We should avoid repeating this mistake for next-gen sequencing. Labs should make the raw sequencing data available along with quality scores and mate-pair information, to facilitate critical re-analysis of

their results, provide tool builders with a wide range of data sets, and facilitate meta-analyses that integrate across multiple data sets. However, there are currently no standards for including meta-data such as sequencing protocol, sequencing machine revision, and mapping program parameters. This makes analysis replication and integration virtually impossible between labs.

Eighth, the pluses and minuses of existing mapping software, and the tradeoffs between the different parameter settings, are not clear. For most uses of next-gen sequencing data, the first task is to map the data back to a reference genome. A variety of programs to do this mapping exist, but they all use different algorithms, take different approaches to resolving idiosyncrasies (e.g. some randomly map reads to repetitive sequence, others discard such reads), and have widely varying command-line options. Most analyses so far have focused on differences in sequencing platforms, and not in mapping software; however, there is some evidence that errors in mapping contribute substantially to false positives predictions of variation [11]. **Comparing mappings and analyses based on these mappings will be important to determine biases introduced by mapping software in addition to the different sequencing platforms.**

Combined, these challenges present serious obstacles to immediate and effective widespread use of next-gen sequencing technologies, particularly in agricultural labs.

1.2 Genomics pipelines and user interfaces

Many of the difficulties in using next-generation sequencing to study variation and expression in agricultural genomes can be alleviated by building an analysis pipeline integrated with a user interface. This will encapsulate data handling and processing coordination within a point-and-click interface accessible to non-specialists. With an integrated analysis pipeline, users would only need to upload their data, indicate the type of sequencing data (DNA, cDNA, paired-end, etc.) and the source genome, and press "Go" to do their first analysis. The application would then handle raw data storage and management, marshal computational resources to do the analysis, correlate pre-existing genomic information and annotations with the analysis, and make the results available in both summary and drillable-down form via an interactive interface.

In addition, a pipeline can offer a standard interface to a compute cluster, making it possible for biologists to directly use high-performance computing (HPC) platforms on a demand basis and letting institutions better leverage their HPC resources. Because a pipeline runs a standard set of analysis tools with appropriate default settings, it can also offer a reference by which new data sets and approaches can be compared to "standard" analyses.

Even simple analysis pipelines such as the one we implemented in the Cartwheel application for comparative sequence analysis can offer dramatic advantages to scientists [5, 6].[1] Cartwheel allows researchers to upload sequences, analyze and annotate them with gene predictions and motifs, compare multiple sequences with different programs, and visualize and interact with the results. By making it possible to modify analysis parameters and visualize the results without worrying about the rote processing details, scientists spend their time focusing on the scientific problem. With Cartwheel, bench biologists can do the analyses themselves without going through the filter of a computational collaborator, who may in turn not fully appreciate the biological considerations involved in the problem. And, perhaps most importantly, scientists gain intuition about both the computational methods and the biological problem through visualization, and by exploring the

[1] See http://family.caltech.edu/tutorial/ for screenshots and a tutorial.

space of parameters and results within the application. **This results in a higher level of scientific data analysis with more potential for extracting information from the data. These same advantages can be realized for analyzing next-generation sequencing data.**

Cartwheel brought an easy-to-use interface to the problem of comparative sequence analysis, and in so doing turned a 3 year process into a 3 month process by providing the tools directly to bench biologists. We hope to do the same for next-gen sequencing data. Cartwheel continues to be used by dozens of people and is very popular among regulatory developmental biologists.

Given these advantages of easy-to-use pipeline applications, why aren't there more of them? First, scientific programmers generally avoid writing user interfaces because it requires different expertise. Second, relatively little in the way of funding is available for the development of new user interfaces in genomics; the most commonly UIs have come from center grants (e.g. UCSC Genome Browser, ENSEMBL, NCBI). When easy-to-use applications are created in an individual lab, they tend to require significant expansion in order to become generally useful to the community. And third, there is a narrow "sweet spot" in the tradeoff between the size of the biological community and the software engineering effort required to build and maintain an application: too small a community, and the effort isn't worthwhile; too big a community, and the needs are too diverse for a single application to cover. Next-generation sequencing is already widespread but has narrow and well-defined computational needs, so it is in the sweet spot for building a community analysis platform.

Several desktop viewers capable of handling next-generation sequencing data do already exist. EagleView displays genome assemblies, annotations, and mapped reads in an interactive desktop viewer [12]. MapView efficiently loads in mappings and allows the user to examine individual points as well as search for likely variants based on user-specified criteria [3]. Because neither EagleView nor MapView integrate an analysis pipeline or provide an interface to compute resources, so users must build the mappings themselves, they only represent a solution to the problem of data visualization.

1.3 A sequence analysis pipeline for next-generation sequence data

We propose to build a sequence analysis pipeline to map next-generation sequencing data onto reference genomes and do post-mapping analysis. We will integrate existing analysis tools with a user interface and a compute platform so that scientists can make immediate use of their data. This model was successfully applied by Dr. Brown in the Cartwheel application.

Our application will have an easy-to-use Web interface for mapping and analysis, a back-end compute system that distributes mapping and analysis jobs across multiple computers, and computational interfaces to permit remote control of the pipeline. Our software will be designed for installation and use at other institutions and companies, and we will develop it openly and solicit input from both users and developers. This will ensure that our pipeline is of use to as many people in the agricultural community as possible. Finally, our Web site will store users' data and analyses as they need, allowing users to revisit analyses, alter parameters, and explore mappings across multiple data sets.

For example, to analyze Illumina resequencing data from a chicken strain using our software, a user would:

1. provide the data to our Web site, either by uploading it, entering a Web or FTP address for

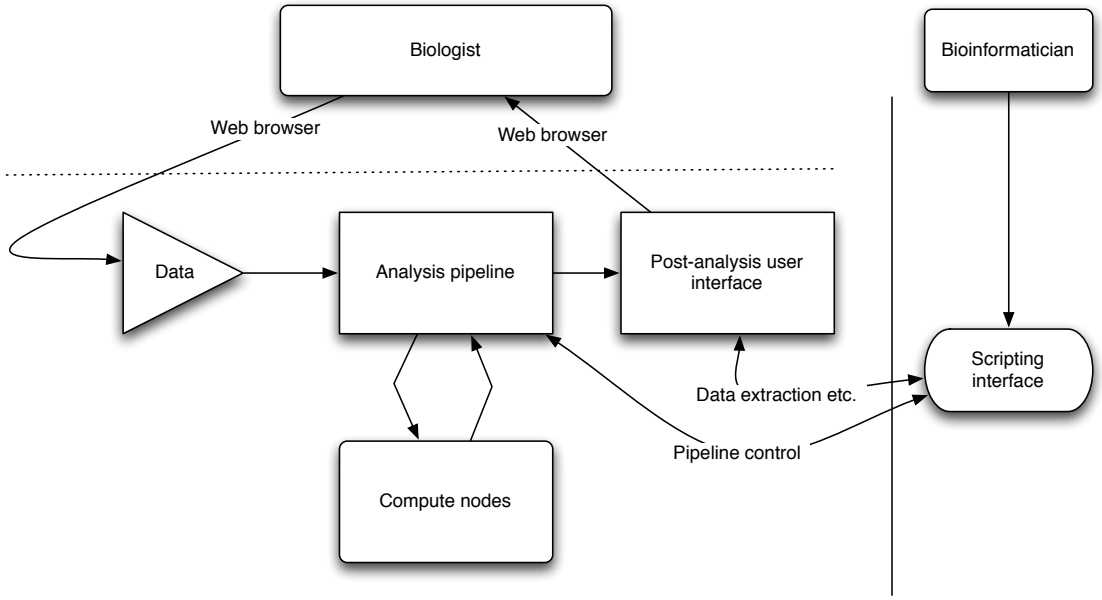

Figure 2: Layout of the proposed analysis pipeline.

download, or mailing us a 1 tb hard drive;
2. select "genomic mapping" to the "chicken" genome with a tool, e.g. Bowtie;
3. optionally define mapping parameters;
4. submit the job to the queue of jobs and wait until it is done;
5. examine the results, either by requesting a mapping summary; asking to visualize detected SNPs, CNVs, and indels on a per-chromosome basis; extracting data containing the genomic locations of variation into an Excel-compatible file; rank regions of a specified size with low or high variation; etc.

Behind the scenes, the Web server and analysis pipeline software would:

1. store the uploaded data by user or lab;
2. serve the genomic mapping request to one or more compute nodes with either default or updated mapping parameters;
3. when the mapping is complete, parse the mapped data into an efficient on-disk lookup structure (see cnestedlist, below);
4. analyze the mapping for locations where significant numbers of reads reported the same SNP, or failed to detect sequence (CNVs or indels);

The pieces to this pipeline already exist in separate form, but they need to be integrated into a single piece of software with a user-friendly interface built for agricultural genomes. **This is a non-trivial software-engineering problem, independent of the scientific problems involved in next-generation sequencing analysis.**

Our analysis pipeline will have a "point-and-shoot" user interface that provides an easy-to-use wrapper for existing mapping programs and post-mapping analysis packages, together with programmatic "glue" to permit the programs to talk to each other and to make use of the current chick,

cow, pig, and sheep genome assemblies and annotations. Our initial set of functionality will be targetted at providing an interface for the "low-hanging" fruit: locating SNPs and CNVs in genomic data, and extracting splice variants and expression ratios from RNA sequencing data; these tasks are relatively straightforward computationally, rely on existing command-line tools, and can be of immediate use to scientists. Our next set of functionality will focus on the more difficult task of using next-gen sequencing data for detection of indels and larger-scale variation through the adaptation of existing mapping algorithms and the use of paired-end sequencing data; this is a less well-defined problem and will require some significant software development to achieve. Our third set of functionality will focus on analyzing data from ChIP-seq and epigenetic filtering: these are likely future applications that address long-term USDA goals, but are unlikely to be major sources of agricultural data in the short-term. For all of these goals, we will focus first on providing an initial user interface to a simple set of functionality, followed by multiple iterations to add features prioritized by users.

In addition to a simple and well-documented user interface for doing mapping and post-mapping analysis with a variety of programs, our pipeline will provide programmatic hooks to allow more computationally adept users to integrate their own mapping and post-mapping analysis programs, define their own processing and post-processing instructions, and install the pipeline on a variety of high-performance computing (HPC) cluster systems. Moreover, our software will be developed in an open manner with regular full releases of the source code, permitting full use, re-use, and adaptation by both academic and commercial entities. **Our goal is to build a system that is of immediate use (within 12 months) to agricultural biologists with next-gen sequencing data, but is also extensible by computational biologists.**

1.4 Personnel and data

Our team is very capable of implementing the proposed software.

Dr. Brown is an experienced genomicist and software developer, with a proven track of building usable and useful open source software. Moreover, he has substantial experience in developing academic software using rapid iterations and controlling development deadlines and software functionality with agile development techniques, an area in which he has consulted for several companies. Dr. Brown is also a member of both a Computer Science department and a Molecular Genetics department, with access to a broad range of graduate students to support interdisciplinary research. Finally, MSU has recently invested $5m in internal funding for an Institute for Cyber-Enabled Research, which will support Dr. Brown's work with a half-time postdoc and help integrate his post-docs with the broader computational community at MSU. **This will leverage USDA funds by effectively providing cost-sharing for the post-doc.**

Dr. Cheng, a research geneticist at ARS, will provide a "bench scientist's" view to complement Dr. Brown's expertise in computational biology. Dr. Cheng's group has datasets generated on the Roche 454, Illumina GA, and ABI SOLiD. He has been applying these sequences for de novo sequencing of viral genomes, resequencing of chicken genomes, ChIP seq, and transcriptome profiling. Consequently, Dr. Cheng has had considerable experience in using several programs for the various analyses. Dr. Cheng and Brown meet frequently to address various computer issues and applications, which has proven very beneficial to both labs.

We believe that our request for support of two full-time post-doctoral fellows, one on the biology side and one on the computational side, is the minimum necessary to do the proposed work. Both

post-docs will be cross-trained in computational approaches and biological problems, expanding the number of people able to tackle the genomics problems of the future. We are also asking for funding for a single graduate student who will engage in algorithmic research on indel detection, which is a relatively unexplored and very challenging problem.

Initial test data sets

We are in possession of several data sets that we will use for our initial analyses. As part of one USDA NRI-funded project, our goal is to identify cis-acting elements and genes in chicken that responded to Marek's disease virus (MDV) challenge. We have sequenced two inbred lines (6 and 7) that differ in genetic resistance to MD on an ABI SOLiD machine. Both lines were sequenced with mate pairs (i.e. both ends of $\approx$3 Kb fragments) which provides additional sequence and the power to detect genome rearrangements such as copy number variation. Plans are already in place to evaluate ABI and Illumina with random and mate-pair libraries from the same chicken. **This information will be valuable in assessing coverage biases and mate-pair utility.**

As part of the same USDA project, RNA from uninfected and MDV-infected chickens is being run on an Illumina GA2 to identify cSNPs and give preliminary evidence for allelic imbalance; many more samples will be processed later with the Illumina GoldenGate assay. In addition, a different USDA NRI-funded grant incorporates sequencing of chromatin immunoprecipitated DNA fragments (ChIP-seq). In this case, genomic fragments (both chicken and MDV) that bind either MDV Meq or chicken c-Jun will be identified.

Our experience with these data sets provided the motivation for this proposal.

2 Rationale and Significance

Next-generation sequencing offers significant opportunities to tackle genomic variation in agricultural animals, and software tools to enable analysis and extraction of information from next-generation sequencing data addresses many short-, medium-, and long-term goals of the USDA CSREES; see attached "Meeting Program Goals" appendix for specifics.

Existing sequencing platforms offer the ability to gather immense amounts of data that can be used to address many USDA goals – indeed, it is already being used for similar work in human [15, 21]. However, the data analysis and integration side of the science is simply not ready. Unlike whole genome shotgun data, where only a few centers were capable of doing the initial sequencing and hence data analysis could efficiently be concentrated at those centers, next-gen sequencing data is already available on a much broader scale. The approaches, tools, and expertise to make use of this data are severely lacking, and our proposed software will quickly open up many new avenues for agricultural genomics. **Moreover, the needs of the agricultural community are unlikely to be given significant consideration in other funding mechanisms;** we will accelerate agricultural science by building tools tailored to specific agricultural goals and the types of genomic analysis needed by the agricultural community.

The lack of easy-to-use software with intuitive user interfaces, encapsulation of high-end computing resources is a perennial problem in bioinformatics [25, 26]. One of the central reasons for this is that other bioinformatics funding mechanisms have been focused on developing new bioinformatics algorithms rather than making better use of existing approaches and data. This leads to a "valley" where widespread biological data analysis significantly trails the leading edge of bioin-

formatics techniques; our proposal is an opportunity for agriculture to narrow this valley and make immediate use of next-gen sequencing data. By providing a common tools infrastructure for others to build on, we can provide support to multiple other USDA grantees and decrease the cost of data analysis for future grants.

Agricultural genomics also competes with many other priorities in sequencing centers and genome sites. New genome assemblies often take a long time to be re-annotated posted to the central genome sites. By providing an alternative interface for making use of these genomes, especially in the context of big RNAseq and genomic variation data sets, we can help agricultural scientists make use of new genome assemblies more quickly and help drive new and better genomics-based science.

Finally, while not a specific objective of this project, we hope that individual users will be able to help identify errors in the reference genome assemblies. In other words, efficient analysis of individual genomes should translate into better quality genome builds. We have already established such a relationship with WUSTL for the chicken genome (see letter of support).

3 Research Methods

A lack of powerful, easy-to-use tools for data analysis will soon be the central problem preventing us from using next-generation sequencing to address genomic variation in animal genomics. Below, we outline how we will tackle the basic integration of the pipeline, discuss a software stack to integrate existing software into a single easy-to-use application, and describe our approach to effective and adaptable software development.

3.1 The basic analysis pipeline

We will start by providing a basic Web interface capable of mapping uploaded next-gen sequencing data to the chicken, cow, sheep, and pig genomes using Bowtie and maq [16, 18]. These are two of the most commonly used mapping programs, and they are both open source and freely available. Mapping will be performed on an attached compute cluster for the user (initially, on our own - see Budget Justification). The mapping data will be stored in the cnestedlist binary database format (see Software Architecture, below), which is a highly optimized sequence feature storage and retrieval system.

Post-mapping analysis options will include SNP calling, CNV detection, RNAseq analysis, and peak detection for ChIP-seq.

Detecting SNPs: Detecting SNPs, even in heterozygous samples, is straightforward with deep next-generation sequencing [11, 24]. The primary problem is ensuring sufficient coverage to eliminate false positives due to sequencing bias; Harismendy et al. (2009) found that due to local biases in sequencing coverage, an average coverage of > 50x was required to exhaustively determine variation [11]. Calling SNPs is straightforward: we will query the mapping for overlapping reads that call the same base, count how many reads share the same call, and use a user-specified cutoff to eliminate calls with too few reads. Because false positives (miscalled bases) are often dependent on local sequence characteristics (such as repeats and indels) that cause incorrect mapping, our SNP-calling interface will also report average coverage in the immediate neighborhood of SNPs. We will also report overlap with known gene, position in genome, etc.

Detecting CNVs: Basic CNV analysis is straightforward – slide a window of a given size across the genome and measure per-base relative coverage of unique genomic regions in mapped reads – the tradeoffs between window size, coverage, and statistical significance of number of reads is still unclear for large genomes [17, 11]. For large windows, biased and random variation in sequencing coverage will average out, but sensitivity to detect small CNVs will decrease; for small windows, variation in coverage will have a more substantial effect. Because biased variation in sequencing coverage varies from sequencing platform to sequencing platform, we will start by providing a basic interface that ranks likely CNVs by probability based on an unbiased model, and then work to develop models for bias once we obtain sequencing data from each new platform.

RNAseq structure analysis: Our RNAseq analysis interface will be built around TopHat, which maps reads onto existing genes, and then maps otherwise unmapped reads onto the genome to recover likely exons based on clustering of reads [27]. TopHat then tries to map reads across the ends of the new exons to determine probable new splice junctions. Because TopHat already relies on the Bowtie and maq mapper software and is open source, it will be straightforward to integrate it into our pipeline. The analysis summary output for RNAseq analyses will simply be a list of likely new exons, along with the genes to which they probably belong, together with detected splice junctions not present in annotations. More advanced summaries include a simple search-by-gene-name, and PFAM protein domains present in the newly discovered exons.

RNAseq expression analysis: Once we have a TopHat analysis of the RNAseq data, we will simply count the number of reads belonging to a particular molecule and treat that as the unnormalized expression level of that gene. RNAseq expression analysis output will be given in digital counts per gene or exon, as well as in log2 ratio of any two RNAseq experiments for compatibility with microarray expression analysis software.

ChIP-seq: Several "peak finder" tools exist for detecting the location of likely DNA binding events in ChIP-seq data. We will integrate the PeakFinder software, which is freely available and open source [13]. Our main addition to this software will be to integrate it with nearest-neighbor feature detection and basic motif searching, which will allow users to correlate likely DNA binding with likely binding sites and nearby genes.

Analyzing PET for structural variation: We will integrate an interface to the PEMer ("Paired End Mapper") software into our pipeline [14]. This software is a freely available package for analyzing paired-end tags; paired-end tag (PET) technology relies on sequencing mate pairs from in vitro size-selected and looped DNA molecules [10]. PEMer places reads individually, determines the best placement for matching ends, and then identifies and clusters outliers as candidates for indels or inversions. PEMer has been successfully applied to Roche 454, Illumina, and ABI data, using several different mapping strategies. PEMer is modular and supports multiple mapping inputs, making it simple for us to integrate.

By integrating PEMer into our pipeline and providing an interface to the output, we can solve several problems for users. The primary problem is that of *compute power*: the authors estimate that PEMer requires approximately 10,000 CPU hours to do a de novo mapping of 10 million reads from Roche's 454 platform. The mapping can be sped up linearly by distributing it across a compute cluster, which we will do automatically.

All of these tools will be integrated as post-mapping analysis tools with a point-and-click GUI

to enable people to immediately visualize and extract data from their analyses.

3.1.1 Insertions and deletions

Reliably detecting short insertions or deletions with short-read sequencing is difficult, although some effective approaches have been developed for Sanger resequencing data [4]. While paired-end tag analysis can detect large insertions, and CNV analysis can detect large deletions, neither has the resolution to find small 1-30 bp indels. Since these indels comprise a substantial part of mammalian variation, and they cannot be discovered easily except by resequencing this is an important problem to solve.

No mapping programs currently do indel matching. Using single (non-paired-end) reads to positively detect short indels is algorithmically challenging. For deletions, the computational time required to look for split matches to a single read scales polynomially with the size of the deletion. To detect insertions of novel sequence, mapping of high-coverage sequencing must first be done, following which reliable points of mismatch can be examined.

In addition to working on algorithms to do single-read indel matching, we will also investigate an alternative approach that relies on negative evidence: locations where shorter reads map more effectively than longer reads should represent "true" mismatches indicative of genomic novelty. This is because the longer read will generate mismatches with indels that the shorter read will ignore.

We will also investigate a "candidate indel" approach that detects locations with non-repetitive sequence that is uncovered but immediately adjacent to highly-covered regions. These are candidates for deletions.

Short indels may comprise 25% or more of the variation in a genome, and indel suppression may play an important role in regulatory sequence evolution [4, 7, 20]. Because indel analysis is so important, we regard it as a critical component of next-gen sequence analysis, despite the fact that it has largely been ignored. As such we believe investigating short indel detection will make an excellent thesis project for an interdisciplinary PhD student in Computer Science.

3.2 Additional functionality

In addition to integrating already existing packages and approaches for making use of next-generation sequencing data, and determining the best parameters for use in the chicken, pig, cow, and sheep genomes, we plan to add the following additional functionality:

- Import of unannotated genome assemblies. As sequencing centers refine existing genomes, they post new assemblies; however, building gene models and providing annotated assemblies can take many more months. In the meantime, these unannotated assemblies can be used to improve resequencing and RNAseq analyses, albeit with reduced functionality (e.g. no correlation with annotated features).
- Attempted assembly of leftover reads. Because of the large number of unmappable reads produced by the Illumina and ABI platforms, unmapped reads are typically discarded. In situations where substantial portions of the reference genome are missing (e.g. chicken [1]), or a nearby species is being sequenced, some of these unmappable reads may represent "real" sequence. We will provide an interface to the Velvet assembler to try de novo assemblies from unmapped reads [30].

- Mapping to "private" sequence. Institutions may have private sequence collections of ESTs and BACs or fosmids, or even entire genomes; we will provide a mechanism for upload of these sequence collections and their use as mapping substrates for RNAseq and resequencing efforts. This will let users leverage their internal data as well as public genome data.
- Mapping of private ESTs to existing assemblies. Many private EST collections exist; we will provide an interface to GMAP that will allow users to map these ESTs onto genome assemblies [29]. Again, this will let users leverage their internal data.
- Import of GFF3 feature files. Many existing bioinformatics programs produce genome annotations in GFF3 format files; in case users wish to use a program that we have not integrated into our pipeline, we will provide them with a GFF3 upload option.
- Handling of barcoded analyses. As sequencing depth increases, more users will want to do multiplex sequencing in which multiple barcoded samples are sequenced together [8]. The most straightforward way to deal with these samples is to separate them into distinct samples based on their barcode, and analyze them distinctly; however, barcoding also offers an opportunity to examine error rates in the protocol by using the barcode sequence as an internal control.
  We believe that our proposal to provide this additional functionality is unique in the agricultural community.

3.2.1 Programmatic interface

We will also provide a programmatic interface to all of our software that will be accessible via an RPC mechanism (e.g. XML-RPC). This will provide a cross-language, cross-platform mechanism for customizing our parameters, uploading and downloading data, and interacting with our visualization interface.

3.3 Software architecture

We will follow the well-proven model of database-backed Web sites, where an Internet-accessible Web server acts as a front-end to a large collection of data. Expensive computation will be performed on separate compute servers (e.g. our local Beowulf cluster) and the results will be stored with the raw data and made accessible to the Web server. The basic software architecture will be a standard stack of Linux + PostgreSQL + a Python back-end to do data manipulation; our front-end Web interface will be written in jQuery, a commonly-used JavaScript toolkit used for interactive "Web 2.0" applications. Raw read data and mappings will be stored in a specialized cnestedlist-based "worldbase" database installed locally to the Web server (see http://code.google.com/p/pygr for details on worldbase; unpublished).

3.3.1 Distributing mapping across multiple computers

Mapping reads to reference sequence is a "pleasantly parallel" process: each read can be considered individually for mapping. Thus the mapping process scales linearly with the number of compute nodes used. In practice, for such problems a "Beowulf" cluster of individually capable machines is used to split the problem across as many compute nodes as are available. The challenge then becomes one of data division and integration, dispersing the raw sequence data across multiple nodes and then collecting the mapped data from multiple nodes and integrating it into a single mapping. A variety of approaches have been used for this kind of computation, including

a Linda tuple space approach (in which computational problems are "posted" for consumption by machines) and Google's MapReduce formalism (which uses two functional programming primitives, 'map' and 'reduce', to organize the process). We will adapt our existing Linda tuple space implementation (c.f. Cartwheel) to the problem of mapping, with an additional data reduction layer on top to integrate the mappings across individual node results. This implementation works on top of existing cluster computing software, including Torque and Oscar, two commonly used task scheduling systems, and it can easily be adapted to other systems.

3.3.2 Storing and querying mappings

We need to be able to retrieve mapped sequences by querying locations in the reference genome; this is a challenging algorithmic problem, especially for large data sets with many sequence-to-sequence relationships. We will use the cnestedlist data structure, which stores sequence relationships in a nested hierarchy of intervals; cnestedlist supports O(N logN) retrieval of relationships from a database of N relationships [2]. An existing open source implementation of cnestedlist is built into the pygr open source software package and is available for immediate use. We also believe we can improve cnestedlist performance dramatically for short reads by using a position-local array instead of a nested list, but that is a secondary goal.

3.3.3 Storing and retrieving individual reads by name or sequence

We have already built a simple, scalable read retrieval system called 'screed'.[2] This database system stores sequence records and metadata (e.g. quality scores, PET information) in a fast write-once-read-many hash table that scales to over a billion reads. This is already integrated with the cnestedlist implementation in pygr.

3.3.4 Building a Web interface

Web interfaces are an excellent way to present data and permit interactive manipulation of the data. From a software engineering perspective they are much easier to maintain and support than client-side graphical interfaces (GUIs), which require the user to install and run new software on their own computer; in contrast, everyone has a Web browser! Until recently, however, Web interfaces were less powerful than client-side GUIs; this is one reason why most genome database displays are relatively static. With the advent of new JavaScript implementations and asynchronous HTTP connection functionality (e.g. AJAX), Web interfaces have become coequal with GUIs in functionality. This makes it possible to implement a rich user interface in a Web site.

The basic mapping visualization interface will be a GMOD-style sequence-and-features window, as is used by the UCSC Genome Browser and Ensembl. This display will show the basic scaffold track together with annotations, user-specified information, and mapping information.

3.4 Software development methodology and collaboration plan

To ensure that the latest version of the software is always functioning, and that new features are immediately testable, we will use an agile-based software development methodology, in which multiple short iterations of software development are followed by functional releases. Because users evaluate the software on a regular basis, the next iteration can take into account user comments and feedback immediately. Agile methodologies also use automated testing to keep the

[2] Available at http://github.com/ctb/screed/.

software as a whole functioning; a thorough automated test suite also supports continued development, deployment, maintenance, and platform migration, by making sure that the software works across different installations and development environments. A formal development practice is necessary for any big software project, and agile methodologies have a proven track record of efficiently creating useful and functional software.

3.4.1 Tentative development timeline

Our development plan is based on 3 month timelines during which significant new features will be developed, tested, documented, and released to the public. Each 3 month period will contain 8 two-week iterations during which users will have access to new features, but we do not expect new people to start using the feature sets produced during each iteration, because each iteration will contain poorly documented new features. Instead, at each three month release we will spend an iteration doing exploratory testing and building documentation and tutorials, so that new users can ease into using the software.

- Months 1-3: integrate existing mapping tools (maq, Bowtie) with our scheduling system. Within three months, we will be able to take uploaded data, transfer the data to compute nodes, run maq and Bowtie, and parse the results into a cnestedlist data structure for querying and data extraction. This simple pipeline will also have a basic Web user interface that supports external users.
- Months 3-6: post-mapping analysis. We will develop a user interface to extract SNPs and likely CNVs from mappings.
- Months 6-9: RNAseq data analysis. cSNP, splice junction, and novel RNA molecules; raw expression profiling extraction.
- Months 9-12: Statistical assessments of coverage and reliability of predicted variation; allelic variation analysis in mixed samples.
- Year 2: interfaces to existing assemblies and annotations; data mining associations with observed variation. "Additional functionality", above, with a focus on the import of new genome assemblies and private data.
- Year 3: structural variation analysis with PEMer, etc. Barcoded analysis. Interface expansion.
- Year 4: Iterate through user feedback and improve software; emphasis on exporting software to external sites.

3.4.2 Documentation, tutorials, and screencasts

In our experience with Cartwheel and FamilyRelations, we have found that users just don't read documentation; instead they prefer brief tutorials on solving their specific problem. We will focus our documentation effort on technical aspects of the software pipeline, e.g. statistical methodologies and mapping parameters, and instead provide written tutorials and video "screencasts" demonstrating the function of the software.

3.4.3 Incorporating community needs

A key part of our proposal is to adapt to the needs of the community. Accordingly, we will have a public mailing list to which all users can subscribe, so that we can conduct a dialog with them.

This model has worked well in the case of the Velvet assembler, for example, with many user questions and helpful responses being posted every week. The biology post-doctoral fellow will also maintain a wiki containing suggestions and comments for upcoming features. We will be proactive in engaging our user community for feedback and evaluation.

Should we receive funding, we will immediately contact the cow, pig, sheep and aquaculture communities in order to determine what data is or will soon be available.

We feel that input from many users is critical to making our tools useful to the agricultural animal community. We hope to work with the major genome centers (e.g. WUSTL; see letter of support) to make sure that all available sequence information is incorporated into genome assemblies and annotation.

Finally, we will also contact the USDA Bioinformatics and NRSP-8 species coordinator (see letter of support) work with the ANGENMAP mailing list and Web site in order to identify additional possible users.

3.4.4 Evaluation of software

Next-generation sequencing is a fast-moving target and we cannot outline exactly what the needs will be in two years. We will evaluate the software function and usability in several ways suited to adapting to the needs of the agricultural community.

First, we will work closely with colleagues and collaborators who have or are generating data. We already have ChIP-seq and RNAseq data from chicken, and we will soon have 20x coverage of several strains of chicken from a commercial collaborator. We also have collaborators at Purdue who are very interested in a user interface for analyzing next-gen sequencing data. If we can analyze our own data, then others can analyze theirs.

Second, starting in 2010, we will use the software in two sets of local courses: local intramural courses at Michigan State University, where Dr. Brown teaches bioinformatics courses in Microbiology and Molecular Genetics, as well as in bioinformatics summer courses at the Kellogg Biological Station. These latter courses are open to external students. During the courses we will assign "fuzzy" objectives (e.g. "determine all points of variation near gene X in this population", or "correlate gene expression with cSNP variation in this RNAseq sample") and then evaluate the software by watching the students tackle the problem. Since the goal is to build the software and user interface so that scientists with knowledge of the biological goals can simply analyze their data without any further effort than watching a brief video tutorial, this kind of student evaluation will be very effective.

Third, starting at the end of the first year, we will run demos and tutorials at conferences (e.g. Plant and Animal Genome) to introduce users to our software's interface and functionality and receive comments.

3.4.5 Development tools, open development, and software availability

We will develop the code in the open, both to encourage ideas and participation and to encourage casual use and re-use of our software. This approach has a long history of working well in the open source software world and has been used in software as diverse as Linux, OpenOffice, R, and BioPerl.

To allow for the maximum community participation, we will use a common open source devel-

opment stack for development: a public developer's mailing list for technical communication, a Trac-based Wiki and issue-tracking system for process management, a github-based source code repository for our version control system, and buildbot continuous integration system to publicize our software's status. We will also use a number of common automated testing tools, including nose, Selenium, and twill, in order to write test code that can be executed by anyone.

All software will be made available under the Apache Software Foundation Open Source license, which permits both commercial and non-commercial use, re-use, and distribution of unmodified and derived software. This will encourage people and companies to make use of our software in whatever way best suits them, without any encumbrances.

3.4.6 Collaboration plan

MSU and the ADOL are neighbors, and we will meet every 2-week iteration. Dr. Brown and Dr. Cheng will jointly supervise one biology post-doc, shared between the two labs, whose responsibility will be to drive and direct the biological goals of the project. Dr. Brown will also supervise a computational post-doc housed in his lab; this post-doc will be partially supported by the new Institute for Cyber-Enabled Research (ICER) at MSU, and will participate in the ICER community. Their responsibility will be to drive the software development of the project. In addition, Dr. Brown will recruit an interdisciplinary graduate student to Computer Science through the Quantitative Biology Initiative at Michigan State; this graduate student will work with Dr. Cheng and Dr. Brown as well as with both post-docs to tackle novel research in the context of next-gen sequence analysis, in particular indel detection.

3.4.7 Deliverables and post-grant maintenance

We will support and extend our own public installation of the analysis Web site for the duration of the grant. We also expect to continue running a Web site indefinitely, albeit potentially at reduced functionality for the public, because we will need to continue using it for our own research. This will not be an undue burden: for example, Dr. Brown continues to run the Cartwheel Web site for comparative sequence analysis, which is now three years past funding.

We hope to contribute to "best practices" and standardization of analysis protocols in the currently rather chaotic field of next-gen sequence analysis. Our choice of an open development model based on unencumbered open source figures prominently in our proposal because we hope others will derive new and better analysis software from ours.

Our software will be easy to install in the hopes that sequencing centers and companies can make use of our work directly. Institutions with these sequencing machines badly need interfaces for data storage, analysis, and retrieval; by providing our software with source code, documenting it and providing tutorials, and supporting/encouraging external contributions to our development process, we hope that these institutions will be able to make maximal use of our software.

We also expect that the local MSU Research Technology facility will run a public version of our site for MSU users and collaborators. This may be made more widely public, as well.

Finally, once our software has proven to be useful, we will apply for continued funding from NIH's Software Maintenance grant program and/or the NIH Research Resources funding program.