Summer 6-30-2014

Node-Oriented Workflow (NOW): A Command Template Workflow Management Tool for High Throughput Data Analysis Pipelines

Eric B. Lipsky
Sigfried and Janet Weis Center for Research, Geisinger Health System

Brian R. King
Bucknell University, brk009@bucknell.edu

Gerard Tromp
Sigfried and Janet Weis Center for Research, Geisinger Health System, gctromp@geisinger.edu

Recommended Citation

Lipsky, Eric B.; King, Brian R.; and Tromp, Gerard. "Node-Oriented Workflow (NOW): A Command Template Workflow Management Tool for High Throughput Data Analysis Pipelines." *Journal of Data Mining in Genomics & Proteomics* (2014) .

Journal of Data Mining in Genomics & Proteomics

Lipsky et al., J Data Mining Genomics Proteomics 2014, 5:2
http://dx.doi.org/10.4172/2153-0602.1000159

Short Communication Open Access

Node-Oriented Workflow (NOW): A Command Template Workflow Management Tool for High Throughput Data Analysis Pipelines

Eric B. Lipsky[1], **Brian R. King**[2], **Gerard Tromp**[1*]

[1]*Sigfried and Janet Weis Center for Research, Geisinger Health System, 100 North Academy Ave., Danville, PA 17822, USA*

[2]*Dept. of Computer Science, Bucknell University, 1 Dent Drive, Lewisburg, PA 17837, USA*

[*]**Corresponding author:** Gerard Tromp, Sigfried and Janet Weis Center for Research, Geisinger Health System, 100 North Academy Ave., Danville, PA 17822, USA, Tel: 1-570-271-5592; Fax: 1-570-214-3074; E-mail: gctromp@geisinger.edu

Received date: Jun 06, 2014, **Accepted date:** June 28, 2014, **Published date:** June 30, 2014

Abstract

Next generation sequencing (NGS) systems produce vast quantities of data that require substantial computational resources for typical analysis tasks. In addition, data that are generated by different NGS systems are not homogeneous. Moreover, there are an overwhelming number of tools available for performing typical tasks. Managing NGS workflows involves writing custom scripts that quickly grow in complexity, often resulting in unwieldy workflows that underutilize typical high performance compute resources, and increase the demands of the staff managing these workflows. We present Node-Oriented Workflow (NOW), a dynamic command template workflow engine for high performance distributed computing (HPC) systems. Our system provides a simple-to-use browser-based front end for designing and managing complex workflows. Workflows are configured using a simple browser interface, and are managed by the integrated job engine, which initializes nodes, monitors node status, and processes results of individual jobs across nodes in an HPC configuration. We reduce excessive messaging across nodes by placing the burden on nodes to start tasks in a workflow when dependencies are met, i.e., node oriented workflow. Our system was designed for NGS processing in the clinical research setting, emphasizing user simplicity, tool scalability, minimization of redundancy in workflows, while maximizing throughput in an HPC environment. Furthermore, NOW is not restricted to NGS pipeline management, but can used to manage any computational pipeline.

Keywords: High performance computing; Computational pipeline management; Next generation sequencing; Workload distribution

Introduction

In 2005, the first massively parallel pyrosequencing platform was made commercially available, opening the door to a new means of conducting genetic analysis. These systems, dubbed next-generation sequencing (NGS) systems, have made DNA sequencing more efficient while reducing costs through automation and massive parallelization of the sequencing processes. Most experiments that are run on NGS systems produce enormous quantities of data. One typical whole-genome sequencing (WGS) experiment can generate millions of reads, or short snippets of sequenced DNA, resulting in gigabytes or even terabytes of data, presenting enormous challenges for the analysis and interpretation of these data. To ease the computational complexity of WGS analysis, many researchers adopt whole-exome sequencing (WES) for their experiments. The exome refers to ~1% of the genome that contains the protein coding regions, i.e. the regions that are expressed in mRNAs, which are then transcribed to proteins [1]. While the sheer volume of the data produced from a single WES experiment is smaller, multiple experiments are often executed simultaneously (i.e. high throughput), particularly in studies attempting to perform some association with disease through subsequent association studies, either genome-wide association studies (GWAS) for common variants or some form of burden test for rare variants. In these studies, the primary aim is to identify common genetic variants that are associated with a specific disease or other phenotype of interest. Accurate identification of variants is a difficult since it is a function of read depth, read quality, mapping quality and allele representation among the reads. The challenge of accurate variant identification increases dramatically for rare variant detection. Accurate variant detection often requires replication of sequencing experiments at different levels of coverage and performing multiple analyses with different variant calling tools. Greater depth of sequencing and more uniform distribution of reads increases the overall accuracy, but also the computational demand. The tools themselves usually require repeated executions to verify the ideal tool parameters in order to improve the probability of making the correct call [2]. This results in even more data, placing an even greater burden on existing compute and storage resources.

Despite their challenges, GWAS and burden test studies are improving our understanding of the role of genetics in human disease. Significant variants are regularly published and deposited in public databases, adding valuable information that geneticists have at their disposal. For example, ClinVar, which is NCBI's public archive of variants, contains over 62,421 genetic variations (as of January 2014) that have been accumulated from a large number of respected research laboratories worldwide [3]. These valuable resources are being used by geneticists in medical research to improve the likelihood of pinpointing underlying genetic risks for common and rare diseases in humans, including cancer and other genetic abnormalities [4,5].

Given its success in the medical research setting, NGS systems are beginning to move beyond the lab and pure research settings, with substantial effort being put forth to help genomic sequencing become a part of clinical diagnosis. Here, it is increasingly being viewed as a viable tool by the clinician to assist in diagnosis and treatment selection for a select number of diseases that have well studied genetic variants identified [6]. Not only are we using these tools to identify disease risk alleles, but it is also becoming common to use sequencing as a proactive screening tool in high-risk cases. For example, it is becoming more common for women with a history of breast or ovarian cancer in one or more of their ancestors, or those with ancestral heritage with known predisposition toward cancer (such as Ashkenazi Jews), to proactively screen their own DNA for well-known BRCA1 and BRCA2 mutations, allowing the clinician and patient to assess their own risk toward developing breast and ovarian cancer [7]. In addition, the public perception toward screening for such diseases is becoming increasingly positive, placing further demand on the use of various forms of genetic screening in the clinical setting [8]. Though we have many significant challenges to conquer to improve the accuracy and reliability of variant calling, most agree that its use in clinical diagnosis and treatment identification will only continue to increase in the coming years [9,10].

The challenges for a clinical diagnosis pipeline

The general workflow required for individual variant calling might seem simplistic: sequence the DNA, analyze the data to identify variants, perform verification, and compare them against a trusted database of known variants. The difficulty of this process does not lie in initial step of sequencing DNA, and the cost of sequencing instrumentation continues to drop, allowing whole-exome sequencing to be conducted below $1,000 [11]. NGS pipelines require large, high speed, redundant arrays of disks that are in the tens or even hundreds of terabytes in total size, making storage and management of the data a challenging task. There are up-front costs with the selection and installation of these systems, but usually require comparatively minimal user intervention once installed and configured properly. However, these challenges pale in comparison to the challenges that arise in the analysis of sequence data generated; these costs are growing, with most expecting these costs to rise through the foreseeable future [12].

NGS analysis cost is multifaceted. A portion of the cost is the personnel skills and labor required to configure and manage these pipelines. This is particularly true of pipelines for variant calling, which is an immensely complex task that requires numerous individual tasks to be performed. A typical WES workflow consists of five primary steps: (i) quality assessment, which consists of cleaning, filtering, and trimming of the raw data; (ii) alignment of the reads to a reference genome; (iii) identification and classification variants; (iv) annotation of identified variants to establish potential significance with respect to the observed phenotype; and (v) use of tools to allow ease of validation and verification of identified variants, including tools for visualization. We refer the reader to the thorough survey by Pabinger et al. [13], which presents an overview of the enormous number of tools available for WES and variant calling. Each of these individual steps has a large number of tools available, each with their own strengths and weaknesses. The tools used in most settings are in the public domain. They are complex, have a large parameter set, and often generate individual results that are heterogeneous with other steps in the workflow. Commercial workflows may be considered, but they are expensive, complex, and lack the infrastructure for practical use in the clinical setting [14]. The clinical setting would require these workflows to be stable, and executed in a high throughput fashion, with a rapid turnaround of results. The burden that is placed on bioinformatics and IT personnel is evident – the task of managing WES workflows is an immensely complex endeavor.

From an informatics viewpoint, hardware costs can be enormous due to the compute resources required for high throughput variant calling pipelines. Software costs are often falsely viewed as cheap because many are freely available in the public domain. The fact is that identifying, installing, and learning how to effectively use and manage NGS tools for the workflow are time-consuming endeavors. Assuming you have the proper storage facilities, and adequate compute resources, and have all tools installed properly, the most significant challenge lies in making the most effective use of these resources on a daily basis in a high throughput setting.

High-performance compute servers and clusters are becoming commonplace for NGS pipelines, requiring frameworks that are able to manage complex workflows on these systems. Perhaps the most comprehensive framework available is Galaxy [15]; a web-based platform that strives to wrap an enormous number of independent computational biomedical research tools into highly configurable workflows configured and controlled with a browser-based user interface. Galaxy is powerful, and designed for the research setting, making its complexity unsuitable for the simplified, streamlined high throughput needs required for the clinical setting. Other systems, such as SIMPLEX, have made an effort to move the compute resources for NGS processing to the cloud [16]. Off-site, service-oriented compute resources such as those managed in cloud environments may offset start-up costs, but the lack of security and privacy controls may not be suitable for the clinical setting. Numerous other frameworks have been released in recent years [17-20], but all have limitations that make them unsuitable for the clinical setting, including, (i) offering a complex command-line interface that is designed for general cluster workflows in research settings, (ii) they are designed for a specific NGS platform or a limited set of tools, (ii) they lack a streamlined, simple, platform-independent interface to manage workflows and monitor execution, (iv) they lack scalability or functionality, or both, that make them unsuitable for use on a high performance compute platform, (v) or they fail to provide an audit log trail facility to record workflow parameters.

Our own experience has shown that proper selection, installation, and management of the numerous tools and computational pipelines that are required for NGS analyses involves numerous challenges, requiring a dedicated staff of bioinformatics experts. Most NGS tasks require many tools working in sequence across numerous large data files to achieve the desired result, with each individual task requiring its own set of parameters depending on the analysis being performed. Making the most efficient use of existing computational resources can be difficult, as most job control software in a grid environment can be quite complex. These tools, most of which are in the public domain, are difficult to configure and manage. Once jobs are spawned they often lack the appropriate infrastructure for appropriate job monitoring. Finally, recent studies have noted that existing tools that are designed to accomplish the same task often generate dissimilar results, stressing the need to run simultaneous experiments using different parameters, or entirely different tools altogether [21].

We present Node-Oriented Workflow (NOW), a scalable, extensible framework for the management of complex NGS workflows in high performance distributed systems. NOW was designed with an emphasis on scalability, simplicity, and reliability, without sacrificing power and flexibility. With respect to an HPC configuration utilizing multiple nodes over a network, NOW was designed with the aim of automatic distribution of workload among nodes, while minimizing the network traffic between nodes as jobs are run, by placing the burden on individual nodes to be responsible for their work. NOW is freely available under the GPL license and can be downloaded from https://github.com/eblipsky.

Implementation

NOW utilizes a dynamic command template workflow engine that provides a browser-based interface designed for ease of pipeline configuration and monitoring. At the top-most level, the user configures a batch, which represents the work to be performed on a set of files. More specifically, a batch contains an ordered list of pipelines and a list of files that are required to start the first pipeline in the batch. A pipeline is a sequential list of atomic tasks which are designed to carry out a major job in the batch, such as mapping reads from a WES experiment to a reference genome, or variant calling on mapped reads. Each pipeline has one required start task, and optional subsequent tasks that are executed in order. A task is the most important abstraction in NOW; it wraps the information needed to perform a single step of actual work in a pipeline. Individual tasks can be configured by referring to a command template in the template library, or by directly entering a command in the task window. As pipelines are executed, task parameters are stored and associated with its results, providing a repository of useful historical information for each pipeline executed. On successful completion of a task, the designated next task will execute, or its error task will start if there is a failure. Advanced task configurations provide for optional preprocessing or postprocessing scripts to be executed before or after, or both before and after, the task executes. Figure 1 depicts the hierarchical relationship between these important user-level components in NOW.

NOW was designed to work in a high performance computing (HPC) environment, or any platform where multiprocessing can be utilized. Many tasks in NGS pipelines can be automated and parallelized to fully utilize the power of modern HPC systems. NOW provides a simple mechanism to maximize throughput with little human interaction. Each task has optional multiprocessing hints, which provide NOW with information on how to distribute tasks among all nodes and processors currently available at the time a task is scheduled to run.

The user interface to NOW is entirely browser-based, written in standard HTML5 ensuring platform independence. The entire configuration of a pipeline is performed through the browser, including management of command templates (Figure 2). Once a workflow is configured, a graph-based visualization of the workflow can be rendered, aiding in configuration and verification of the desired logical flow of the tasks for each pipeline. NOW is able to visualize the workflow templates and perform asynchronous updates of page content to allow monitoring of pipeline completion in real time (Figure 3).

NOW is comprised of several publicly available off-the-shelf components. We made an effort to incorporate proven, stable, publicly available components with default installation procedures. The critical data management backbone of the system uses Redis and CouchDB, with the web-based user interface implemented using the Laravel PHP framework managed with Apache as the web server. Python is used to handle the executions of tasks within NOW, with JSON selected as the main data sharing protocol throughout the system. The browser-based point-and-click interface is developed with HTML following standard AJAX technologies, with the visualization of job configuration and management developed using jQuery and jsplumb. An audit log has been implemented that stores the precise configuration and completion statistics of every task that is run within NOW. The audit log is stored and managed with CouchDB. (Figure 4 for a diagram depicting the system configuration of NOW).

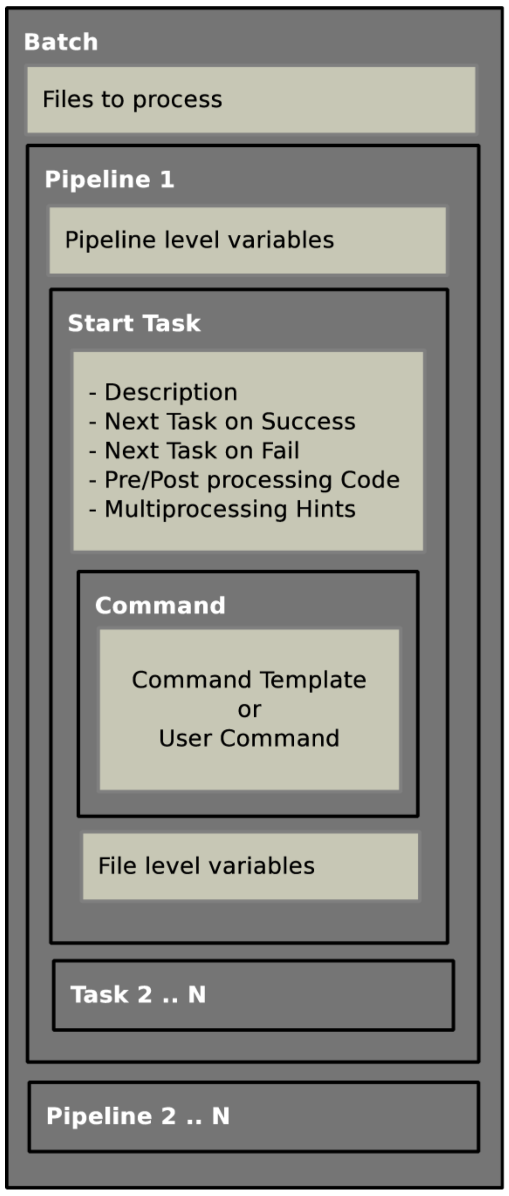
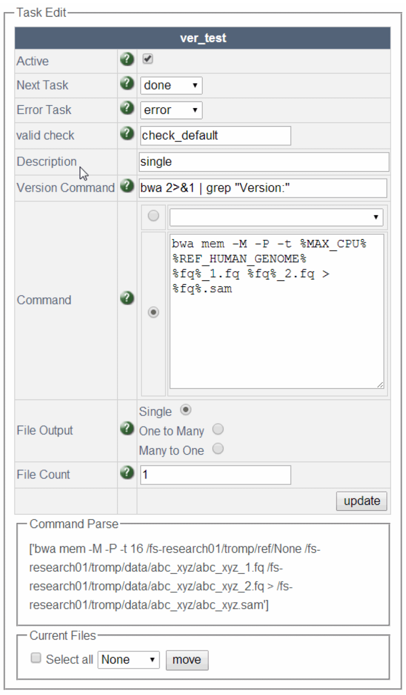


Figure 1: Hierarchy of NOW workflow components – The above diagram shows the different components in NOW. At the highest level, the user configures a batch of pipelines to execute on a set of files. Pipelines encapsulate an ordered list of tasks to be carried out.



Task Edit

ver_test		
Active	?	✓
Next Task	?	done ▾
Error Task	?	error ▾
valid check	?	check_default
Description		single
Version Command	?	bwa 2>&1 | grep "Version:"
Command	?	○ ▾ / ● bwa mem -M -P -t %MAX_CPU% %REF_HUMAN_GENOME% %fq%_1.fq %fq%_2.fq > %fq%.sam
File Output	?	Single ● / One to Many ○ / Many to One ○
File Count	?	1
		update

Command Parse

['bwa mem -M -P -t 16 /fs-research01/tromp/ref/None /fs-research01/tromp/data/abc_xyz/abc_xyz_1.fq /fs-research01/tromp/data/abc_xyz/abc_xyz_2.fq > /fs-research01/tromp/data/abc_xyz/abc_xyz.sam']

Current Files

☐ Select all None ▾ move

Figure 2: Task edit window – This window shows the task editor for configuring command templates or for directly entering a task in the workflow. In this picture, a task is configured using a user-defined command.

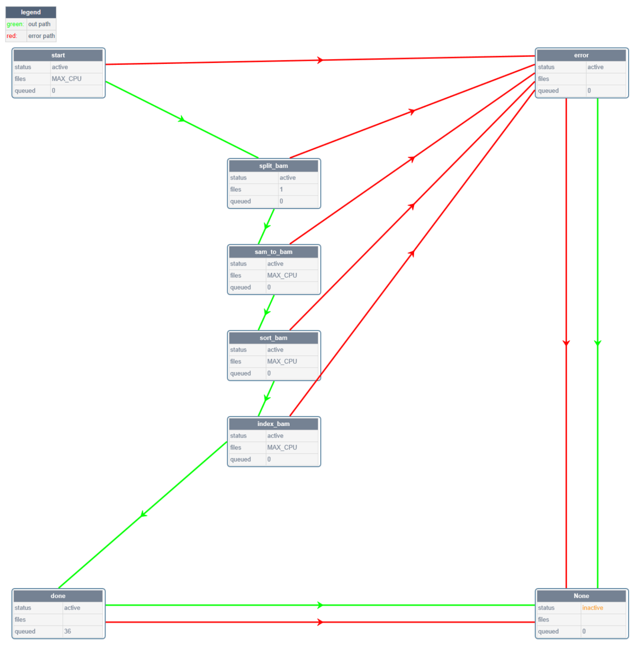


Figure 3: Pipeline visualization – This screen visualizes a pipeline that completes an alignment of reads to a reference. The pipeline is implemented in four distinct tasks.

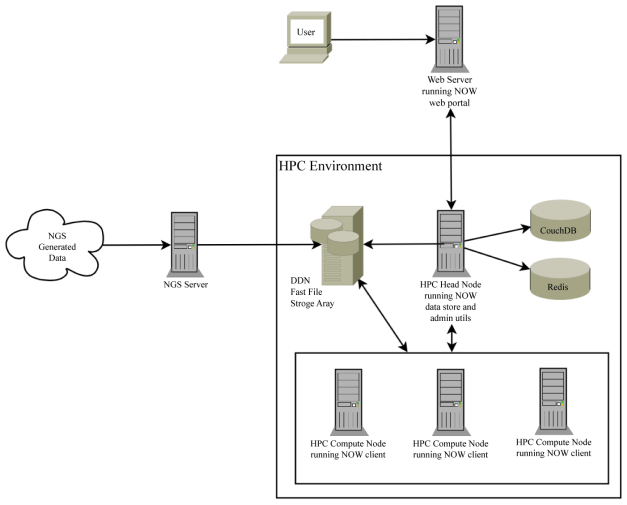

User
Web Server running NOW web portal
HPC Environment
NGS Generated Data
NGS Server
DDN Fast File Stroge Aray
HPC Head Node running NOW data store and admin utils
CouchDB
Redis
HPC Compute Node running NOW client
HPC Compute Node running NOW client
HPC Compute Node running NOW client

Figure 4: System Configuration of NOW – The interaction and dependencies between important components of NOW are displayed in this figure. In general, data obtained from an NGS system are copied over to the HPC file storage facilities. NOW administrative tools are used to register these files in the framework. The web server provides the front end to the system, allowing users to create pipelines and start processing of registered files.

Results

NOW has been thoroughly tested and is now regularly used on our own system here at Geisinger Health Systems, which consists of a 25 node HP Unified Cluster, with one node configured to be the head node. There are 228 TB of attached storage from DataDirect Networks available in the system. Each node has dual Intel Xeon processors running at 2.60 GHz, providing 16 cores, and 128 GB of RAM per node. In addition, nodes have NVIDIA Tesla M2090 PCIe graphics processor cards for GPU computing.

To test the efficacy of NOW, we configured two pipelines in a batch, designed to perform variant calling on a set of targeted resequencing data (Figure 5). The resequencing platform was designed for the Pharmacogenomics Research Network (PGRN) [22] and comprises the exonic sequences from 84 genes that are known to have substantial pharmacological impact in human drug metabolism, as is named the PGx platform. The data represent PGx sequences generated for 96 individuals. One PGx resequencing dataset contains approximately 12.8 GB paired-end reads sequenced at 500 fold coverage, yielding a total of 500 million bases.

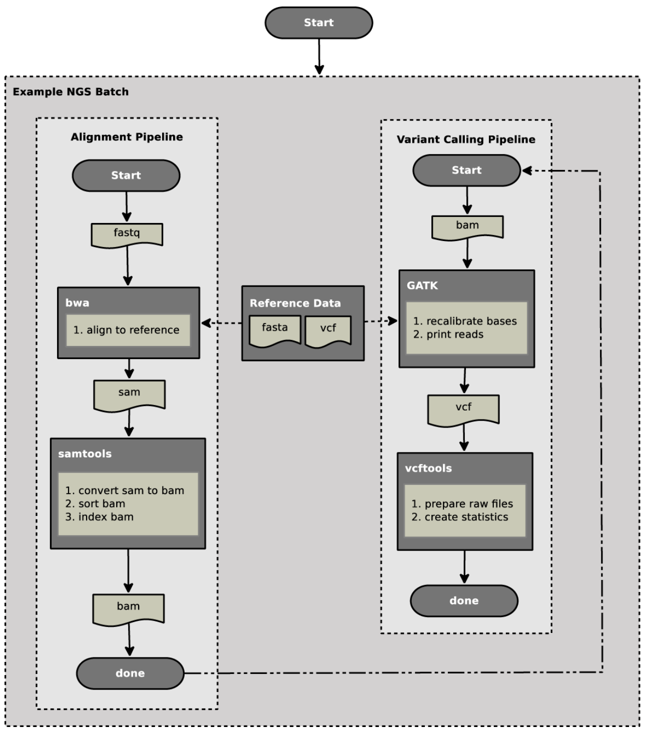


Figure 5: Batch configuration for variant calling – A single batch consisting of two pipelines for variant calling from whole exome sequence data are shown. The first pipeline performs the mapping of the reads to the reference genome, and the second performs variant calling from the mapped data.

The first pipeline in the batch performed the mapping of all of the paired-end reads to the human genome, reference GRCh37. BWA [23] was used to perform the alignment of the reads to the reference. The resulting SAM file was sorted and indexed using samtools [24], yielding a BAM file of aligned reads. These data were then sent to the second pipeline in the batch for variant calling. In this pipeline, GATK was used to recalibrate base quality scores and to perform the actual variant calling using the latest reference of human variation from NCBI dbSNP's repository [25]. Finally, the resulting VCF files were then postprocessed using VCFtools to generate reports and perform statistical analyses [26].

As a baseline for comparison purposes, we executed the batch job for a single exome dataset from a manual script in a strict sequential fashion, restricting all computational resources requested from the program to one node. No other jobs were executing on the node, and no other nodes were utilized. The entire batch took approximately two hours and 10 minutes to complete on this system configuration. Specifically, the alignment pipeline took one hour to complete, and the variant calling pipeline completed in an hour and 10 minutes. If this batch was performed serially on sequence data from 96 HapMap reference DNA specimens without using NOW, it would have taken over 8 days to complete. We reran the same experiment, utilizing a total of 10 nodes. Under this configuration, using NOW, all 96 files were processed in 6 hours. If no multiprocessing enhancements in NOW were utilized, and all 96 batches were allowed to run in sequential fashion over 10 nodes, this would have still required approximately 20 hours to complete. NOW's multiprocessing improvements allowed the same work to be completed in only 30% of the time, maximizing CPU utilization and overall data throughput in both pipelines.

Discussion

There are several important criteria that influenced the design of NOW. Our first goal was to provide a scalable platform not tied to any specific toolset or sequencing instrumentation. There is no dearth of tools available for NGS processing. Moreover, the field is nascent, and continues change rapidly. We needed a tool that was able to utilize a wide range of NGS tools available, and be free of any tie to a specific NGS platform, data format, or tool set. NOW provides this flexibility through the task abstraction. A NOW task represents any general command, analogous to a command that would execute from the command line or from within a custom script. This is ideal for majority of our users; most bioinformatics tools we use, particularly those for NGS processing, are command-line driven. This flexibility opens NOW to be used for a wide range of computational tasks, not just those dominated by NGS systems.

Another important design goal was to minimize dedicated staff for NGS workflows management. As a clinical diagnostic lab, with numerous tasks to handle with minimal bioinformatics support staff, we could not afford dedicated staff that constantly monitors job submission, status, and completion to maximize job throughput. Moreover, many individual jobs are highly similar, requiring only minimal parameter changes from one job to the next. By creating a template-based system, pipeline similarities are captured through the command templates, allowing the user to focus only on parameter alterations needed for each job by eliminating the complexity of writing and managing custom scripts. Moreover, multiple batches can be submitted to NOW. NOW is autonomous; each node monitors its own state and resource availability, maximizing throughput around the clock. In the event of errors in individual tasks, the job engine notifies the user via e-mail, and continues to process other jobs ensuring maximal throughput.

A third important design goal was to provide scalability and flexibility with respect to the underlying commands and tools used in any given pipeline. As a clinical medical research group, our responsibilities with NGS processing can be varied, and conflicting. We evaluated several existing pipeline / workflow frameworks; however, none of them were found to meet our needs. We found existing systems to be either too advanced, meaning, they were designed solely for the pure research setting, or too restrictive, designed to run with a specific toolset or platform. Some systems expected the user to follow its built-in workflow. We needed a simple, easy to use, lightweight tool to manage a wide range of NGS workflows. Nevertheless, our experience has shown that there is nothing simple about NGS workflows, especially in a HPC system. NOW offers the flexibility needed without sacrificing throughput.

Our final design goal was the incorporation of an audit log system. NOW incorporates a logging system within the same CouchDB and JSON framework that is used throughout the system. NOW will log the precise command line that was executed for every task within every workflow managed by the framework. Additionally, it logs important execution statistics, including the node that the task was executed on, start and stop times, CPU usage, task output, and successful completion status. The audit logger provides the mechanism to record how a workflow is executed, easing the burden of identifying and fixing individual task or node failures.

An important distinction between NOW and other similar workflow systems lies in the placement of the responsibility of task execution. A typical setup in an HPC system places responsibility on one head node to constantly monitor the state of all tasks executing on all nodes in the cluster, and executes new tasks as resources become available. NOW follows a different approach, making each node responsible for monitoring its own state and running tasks that are appropriate given its current load. This approach substantially decreases messaging overhead between one head node and the rest of the nodes in the system.

We anticipate that future improvements will be made to the NOW framework through our own internal efforts, as well as the efforts of the open source community. One possible improvement to the existing framework would be to provide some restricted capabilities for end users that are not part of the bioinformatics staff to utilize NOW for their own processing needs. To this end, NOW could be modified to create an abstraction layer over each task that hides much of the configuration options that are normally available to the bioinformatics staff. This would provide access to power and flexibility of the computational resources that NOW manages, while only exposing those parameters that are appropriate for end users, such as the location of their data files.

Conclusion

Next generation sequencing has brought forth a plethora of methods used to identify interpretable mutations that are associated with disease and other clinical phenotypes with high confidence. Coincidentally, interest in genetic screening for predisposition and diagnostic explanation of various phenotypes has surged at both clinical and research levels. The computational pipelines required to perform these services are complex, requiring substantial compute and storage resources, in addition to dedicated bioinformatics support staff. These costs can be prohibitive, necessitating a streamlined framework that can maximize throughput while minimizing continual support from dedicated staff. In this paper, we presented NOW, a framework for managing high-throughput pipelines required to perform complex computational analyses. NOW creates a web-based user interface to create and manage pipelines having a wide range of complexity, providing numerous abstractions that hide the intricacy of the tools used to perform these analyses. The system can run in either a standalone or HPC environment, utilizing as many nodes and processors as possible to maximize throughput. NOW was developed based on the needs of our own clinical and research facilities to execute whole genome and exome experiments with rapid turnaround times. In addition, the framework offers the flexibility to be used on any computational pipeline where HPC systems are available.

Acknowledgement

Part of this work was funded through the Bucknell / Geisinger Research Initiative awarded to BRK, and NHGRI eMERGE network funds to Geisinger Clinic (U01HG006382).

References

1. Ng SB, Turner EH, Robertson PD, Flygare SD, Bigham AW, et al. (2009) Targeted capture and massively parallel sequencing of 12 human exomes. Nature 461: 272-276.
2. Robasky K, Lewis NE2, Church GM3 (2014) The role of replicates for error mitigation in next-generation sequencing. Nat Rev Genet 15: 56-62.
3. Landrum MJ, Lee JM, Riley GR, Jang W, Rubinstein WS, et al. (2014) ClinVar: public archive of relationships among sequence variation and human phenotype. Nucleic Acids Res 42: D980-985.
4. Choi M, Scholl UI, Ji W, Liu T, Tikhonova IR, et al. (2009) Genetic diagnosis by whole exome capture and massively parallel DNA sequencing. Proc Natl Acad Sci U S A 106: 19096-19101.
5. Ng SB, Buckingham KJ, Lee C, Bigham AW, Tabor HK, et al. (2010) Exome sequencing identifies the cause of a mendelian disorder. Nat Genet 42: 30-35.
6. Berg JS, Khoury MJ, Evans JP (2011) Deploying whole genome sequencing in clinical practice and public health: meeting the challenge one bin at a time. Genet Med 13: 499-504.
7. Mavaddat N, Peock S, Frost D, Ellis S, Platte R, et al. (2013) Cancer risks for BRCA1 and BRCA2 mutation carriers: results from prospective analysis of EMBRACE. J Natl Cancer Inst 105: 812-822.
8. Shkedi-Rafid S, Gabai-Kapara E, Grinshpun-Cohen J, Levy-Lahad E (2012) BRCA genetic testing of individuals from families with low prevalence of cancer: experiences of carriers and implications for population screening. Genet Med 14: 688-694.
9. Koboldt DC, Steinberg KM, Larson DE, Wilson RK, Mardis ER (2013) The next-generation sequencing revolution and its impact on genomics. Cell 155: 27-38.
10. Caskey CT, Gonzalez-Garay ML, Pereira S, McGuire AL (2014) Adult genetic risk screening. Annu Rev Med 65: 1-17.
11. Dondorp WJ, de Wert GM (2013) The 'thousand-dollar genome': an ethical exploration. Eur J Hum Genet 21 Suppl 1: S6-26.
12. Sboner A, Mu XJ, Greenbaum D, Auerbach RK, Gerstein MB (2011) The real cost of sequencing: higher than you think! Genome Biol 12: 125.
13. Pabinger S, Dander A, Fischer M, Snajder R, Sperk M, et al. (2014) A survey of tools for variant analysis of next-generation genome sequencing data. Brief Bioinform 15: 256-278.
14. Desai AN, Jere A (2012) Next-generation sequencing: ready for the clinics? Clin Genet 81: 503-510.
15. Goecks J, Nekrutenko A, Taylor J; Galaxy Team (2010) Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. Genome Biol 11: R86.
16. Fischer M, Snajder R, Pabinger S, Dander A, Schossig A, et al. (2012) SIMPLEX: cloud-enabled pipeline for the comprehensive analysis of exome sequencing data. PLoS One 7: e41948.
17. Lam HY, Pan C, Clark MJ, Lacroute P, Chen R, et al. (2012) Detecting and annotating genetic variations using the HugeSeq pipeline. Nat Biotechnol 30: 226-229.
18. Ji HP (2012) Improving bioinformatic pipelines for exome variant calling. Genome Med 4: 7.
19. Rubio-Camarillo M, Gómez-López G, Fernández JM, Valencia A, Pisano DG (2013) RUbioSeq: a suite of parallelized pipelines to automate exome variation and bisulfite-seq analyses. Bioinformatics 29: 1687-1689.
20. Oinn T, Addis M, Ferris J, Marvin D, Senger M, et al. (2004) Taverna: a tool for the composition and enactment of bioinformatics workflows. Bioinformatics 20: 3045-3054.
21. O'Rawe J, Jiang T2, Sun G2, Wu Y, Wang W3, et al. (2013) Low concordance of multiple variant-calling pipelines: practical implications for exome and genome sequencing. Genome Med 5: 28.
22. Giacomini KM, Brett CM, Altman RB, Benowitz NL, Dolan ME, et al. (2007) The pharmacogenetics research network: from SNP discovery to clinical drug response. Clin Pharmacol Ther 81: 328-345.
23. Li H, Durbin R (2009) Fast and accurate short read alignment with Burrows-Wheeler transform. Bioinformatics 25: 1754-1760.
24. Li H, Handsaker B, Wysoker A, Fennell T, Ruan J, et al. (2009) The Sequence Alignment/Map format and SAMtools. Bioinformatics 25: 2078-2079.

25. Sherry ST, Ward MH, Kholodov M, Baker J, Phan L, et al. (2001) dbSNP: the NCBI database of genetic variation. Nucleic Acids Res 29: 308-311.

26. Danecek P, Auton A, Abecasis G, Albers CA, Banks E, et al. (2011) The variant call format and VCFtools. Bioinformatics 27: 2156-2158.