BIOINFORMATICS APPLICATIONS NOTE Vol. 27 no. 20 2011, pages 2907–2909
doi:10.1093/bioinformatics/btr478

Sequence analysis

Advance Access publication August 27, 2011

Extending KNIME for next-generation sequencing data analysis

Bernd Jagla[1,*], Bernd Wiswedel[2] and Jean-Yves Coppée[1]

[1]Departement Génomes et Génétique, Institut Pasteur, Plate-forme Transcriptome et Epigénome, 25 Rue du Docteur Roux, F-75015 Paris, France and [2]KNIME.com AG - Technoparkstr. 1 - 8005 Zürich, Switzerland

Associate Editor: Martin Bishop

ABSTRACT

Summary: KNIME (Konstanz Information Miner) is a user-friendly and comprehensive open-source data integration, processing, analysis and exploration platform. We present here new functionality and workflows that open the door to performing next-generation sequencing analysis using the KNIME framework.

Availability: All sources and compiled code are available via the KNIME update mechanism. Example workflows and descriptions are available through http://tech.knime.org/community/next-generation-sequencing.

Contact: bernd.jagla@pasteur.fr

Supplementary Information: Supplementary data are available at *Bioinformatics* online.

Received on May 16, 2011; revised on July 29, 2011; accepted on August 2, 2011

1 INTRODUCTION

KNIME (Konstanz Information Miner; Berthold *et al*., 2008) distinguishes itself from other workflow management systems like Mobyle (Néron *et al*., 2010), Galaxy (Goecks *et al*., 2010), Taverna (Hull *et al*., 2006), Kepler (Ludäscher *et al*., 2006), geWorkbench (Floratos *et al*., 2010), Conveyor (Linke *et al*., 2011) and many others by not being a domain-specific solution, but an integration backbone with strong data preprocessing and data analytics capabilities. It is mainly used in the customer relationship management and financial sector and, through a list of commercial and non-commercial vendors, in the cheminformatics area. (See participation of recent KNIME conferences.) The focus has been on providing functionality for building professional reports, statistical analysis, cheminformatics and very recently, high-throughput/high-content (HCS/HCA) image analysis and scripting using languages such as Perl, Python, Matlab, R and Java. Here, we introduce new nodes that allow next-generation sequencing (NGS) data analysis to be performed using KNIME. These nodes take advantage of some of KNIME's general features including memory management, allowing the handling of billions of rows on a standard desktop computer with only about 4 GB of RAM. The workflows can be executed from the command line where all variables can be manipulated if desired. This enables the administrator to easily incorporate workflows in web-based tools such as Mobyle or Galaxy. KNIME is based on Eclipse and JAVA 1.6; the workflows are stored in plain-text XML files and can be executed on basically any modern operating system, and also easily exchanged with or without data.

*To whom correspondence should be addressed.

2 DESIGN AND IMPLEMENTATION

The current version (2.3.4) of KNIME is based on JAVA 1.6 and Eclipse 3.6.2. The functionality presented here follows the general guidelines for implementing nodes within the KNIME framework and augments the KNIME workflow management system with specific nodes for the correct handling of NGS data. Detailed information and examples are available through the KNIME web site (http://tech.knime.org/community/next-generation-sequencing). There, we have also posted a collection of workflows with extensive descriptions and use cases. The purpose of these workflows is to provide new users with some examples of data handling and provide a good starting point for fast data generation. For more complicated workflows, we encourage the community to use the myexperiments.org web site (Goble *et al*., 2010) (http://www.myexperiment.org/search?query=KNIME). In the following description, we use italic font to indicate names of nodes.

The first set of nodes that we have released contains: *FastQReader*, *FastQWriter*, *SAMReader*, *AdapterRemovalAdv*, *CountSorted*, *OneString*, *GetRegions*, *PositionStr2Position*, *RegionOverlapp*, *Seq2PosIncidents*, *Bash*, *CmdWInput, BEDGraphWriter* and *JoinSorted*. Other nodes mentioned in the text below have been developed by KNIME developers and other community contributors.

There are nodes specific for NGS-related file types such as for reading and writing (compressed and uncompressed) FastQ- (*FastQReader*), reading SAM/BAM- (*SAMReader*) and writing BED files (*BEDGraphWriter*). (Reading BED files and writing SAM files can already be accomplished with standard nodes). NGS-specific tasks that can be executed through the KNIME environment include adapter removal and working with regions of interest (ROIs). In this context, we define a ROI as successive nucleotides that have a common property within a reference sequence, such as annotations and sequencing reads mapping to short regions.

One node that is related to ROIs is *GetRegions*, which identifies successive nucleotides with counts greater than zero in a sorted table where each row represents a position defined by a string identifier for the chromosome and an integer describing the position on the chromosome. This can be used to analyze pile-up files that hold information about how many reads align to a given sequence position. Other nodes such as *CountSorted*, *PositionStr2Position*, *Seq2PosIncidents* and *OneString* represent functionality with improved performance or tasks that otherwise would have taken more than one node to realize with out-of-the-box functionality. Detailed descriptions on these and all other nodes can be found in the help section within the KNIME environment or on the KNIME community pages.

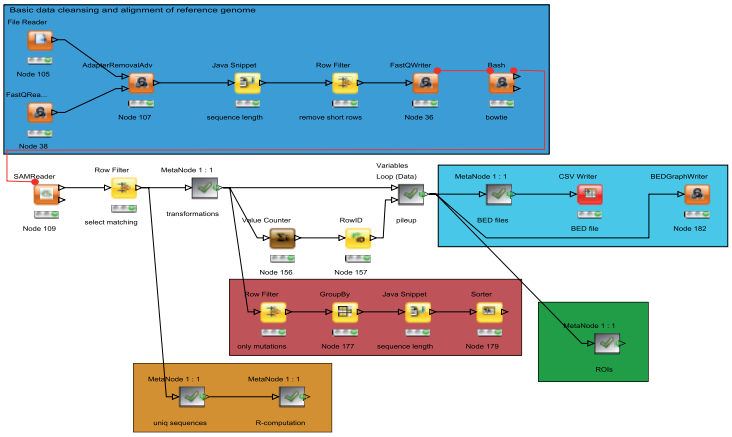

Fig. 1. KNIME sample workflow for NGS-related data analysis. Different work areas are distinguished by background color: data cleansing and alignment to the reference genome (dark blue); data preparation of aligned reads (white); creating BED files (light blue); mutation analysis (pink); ROI analysis (green); and identification of uniquely aligning sequences and integration with R (orange).

NGS reads that have already been mapped to a reference genome can be of varying lengths and counts per sequence positions cannot be obtained in a straightforward manner using general purpose tools such as KNIME. A naive way to deal with this problem is to convert individual sequences of length *n* into *n* individual entries, one per sequence position (*Seq2PosIncidents*). This list can then be sorted by chromosomal position (*Sorter*) and counts per position can be calculated (*CountSorted*). This is equivalent to the pileup file generated by samtools (Li *et al*., 2009). From these counts we can now identify ROIs where we combine successive positions with counts greater than zero into a single entry (ROI) in a table (*GetRegions*). Those ROIs can be compared with annotation that is read in from, e.g. gff formatted file (*File-Reader*). This functionality resembles, e.g. intersectBED from BEDTools (Quinlan and Hall, 2010).

When developing workflows it is usually impractical to work with a full dataset that might comprise many millions of reads. Thus, sample files have to be created, sometimes for each project. The burden of maintaining these files can be avoided when using KNIME. Almost all nodes that function as entry points in KNIME, i.e. read in data, can limit the number of rows they are reading. Thus, a workflow can be developed on a small subset of the data and once the workflow is validated it can be launched on a different, more powerful machine (through the export/import mechanism). This is also true for the *FastQReader* and *SamReader* nodes.

Thus, KNIME now enables the user to generate workflows for a wide range of tasks in the field of NGS analysis. For example, we can now create workflow that read in a FASTQ file from a sequencing machine (*FastQReader*), remove adapters (*AdapterRemovalAdv*), select sequences by length constraints (*JavaSnippet*, *Row Filter*), write out a FASTQ file (*FastQWriter*), execute an alignment program (*Bash*), read in the resulting data from a SAM/BAM file (*SAMReader*), select sequences that map to a unique position on the reference, create a pile up, select and analyze mutations, identify successive ROIs, create counts per gene (see above), sort counts (*Sorter*), communicate the results to R and then perform additional analysis and graphs in R or Matlab (See Fig. 1 and Supplementary Materials for a more in-depth description of this workflow). This workflow can be exported with or without data and shared with the community (http://www.myexperiment.org/workflows/2183.html). It can be integrated into Galaxy or Mobyle to allow others to use it without installing KNIME (see Supplementary Material for details). This gives just a small glimpse on the powerful features of this working environment.

It is not in the scope of this application note to compare KNIME with other tools such as Galaxy. Here, we are merely opening the door for KNIME to be used in the field of NGS. Thus, we would like to briefly discuss some of the points that helped us with this rather subjective decision. KNIME is more visually intuitive and enables us to better understand the flow of data. Some of the complexity can be hidden in 'meta-nodes' or sub-workflows; it is possible, for example, to construct loops for iterating through lists of files; together with if/else statements the basic elements of programming are available; missing basic functionality can usually be prototyped using the Java/Python/Perl/R snippets that allow the user to employ their favorite programming language. Those things are not possible at the moment in web-based tools such as Galaxy. A comparison with Taverna and other desktop-oriented workflow management tools is much more complex since it is mostly not the list of functionalities that makes the difference between being accepted by the user but rather an 'intuitive feeling' that is gained when installing and first launching the program. Missing functionality can most of the time be easily developed and in none of the tools that we have seen so far are all the things needed (or thought to be needed) readily available. Thus, we have decided on using KNIME because we believe in its potential and user-friendliness, which are very specific to our own scenario. The availability of cheminformatics, statistical, image processing algorithms and features for visualizing and interacting with data were also relevant. We hope that opening the door to KNIME will create further interest within the NGS community.

To further close the remaining gaps, we are actively working on enhancing KNIME by developing, among others, a Distributed Annotation System (DAS) client; handling protein and nucleotide sequence data; GBrowse (http://gmod.org/wiki/GBrowse),

Integrative Genomics Viewer (IGV) (Robinson *et al.*, 2011) and University of California Santa Cruz (UCSC) genome browser (Kent *et al.*, 2002) interaction; as well as parallelization for multi-core computers and cluster environments.

ACKNOWLEDGEMENTS

We thank Odile Sismeiro and Caroline Proux for helpful discussions and testing of the workflows. We thank Dr Kenneth Smith for critical reading of the manuscript. We also thank the KNIME team and community for timely and helpful support.

Conflict of Interest: none declared.

REFERENCES

Berthold,M.R. *et al*. (2008) KNIME: The Konstanz Information Miner. In Preisach,C. *et al*. (eds) *Data Analysis, Machine Learning and Applications: Studies in Classification, Data Analysis, and Knowledge Organization*, Vol. V, pp. 319–326.

Floratos,A. *et al*. (2010) geWorkbench: an open source platform for integrative genomics. *Bioinformatics*, **26**, 1779–1780.

Goble,C.A. *et al*. (2010) myExperiment: a repository and social network for the sharing of bioinformatics workflows. *Nucleic Acids Res*., **38** (Suppl. 2), W677–W682.

Goecks,J. *et al*. (2010) Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biol.*, **11**, R86.

Hull,D. *et al*. (2006) Taverna: a tool for building and running workflows of services. *Nucleic Acids Res*., **24**, 729–732.

Kent,W.J. *et al.* (2002) The human genome browser at UCSC. *Genome Res*., **12**, 996–1006.

Li,H. *et al*. (2009) The sequence alignment/map format and SAMtools. *Bioinformatics*, **25**, 2078–2079.

Linke,B. *et al*. (2011) Conveyor : a workflow engine for bioinformatic analyses. *Bioinformatics*, **27**, 903–911.

Ludäscher,B. *et al*. (2006) Scientific workflow management and the Kepler system. Grid systems. *Concurr. Comput. Pract. Exp.*, **18**, 1039–1065.

Néron,B. *et al*. (2010) Mobyle: a new full web bioinformatics framework. *Bioinformatics*, **25**, 3005–3011.

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

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