systemPipeR: a generic workflow environment federates R with command-line software

Daniela Cassol, Le Zhang, Ponmathi Ramasamy, Gordon Mosher & Thomas Girke

Institute for Integrative Genome Biology, University of California, Riverside, California 92521

Introduction

The *systemPipeR* project provides a suite of R/Bioconductor packages for designing, building and running end-to-end analysis workflows on local machines, HPC clusters and cloud systems, while generating at the same time publication quality analysis reports. Pre-configured workflow templates are provided for a wide range of omics applications. The package allows to choose for each workflow step the optimal R or command-line software, customize workflows, and design entirely new ones while taking advantage of central community S4 classes of the Bioconductor ecosystem. Thus, it serves as a federation hub for integrating both R and command-line based software, that makes efficient use of existing software without limiting users to tools from a single community project. The most substantial new features we have recently added to *systemPipeR* include the following enhancements. First, a graphical workflow management interface has been added allowing users to: visualize workflow designs in different graphical layouts; execute workflows step-wise or entirely; monitor their run status while tracking all metadata associated with a project, and subsequently generate both scientific and technical status reports. Second, a workflow control class (S4) has been added allowing users to execute single or any number of complex workflow steps with a single R command, such as *runWF(sys[1:3])*, or using pipes (*%>%*). Third, a Shiny interface has been created to support interactive graphics in workflow reports. This includes interactive visualization of various statistical results and workflow topologies. Finally, we have made many improvements to *systemPipeR*'s command-line interface based on CWL (Common Workflow Language). In contrast to other CWL-based interfaces, *systemPipeR* is unique by implementing most of CWL's functionalities in R rather than just providing a wrapper tool for them. This way users can render, debug, and run all command-line definitions within efficiently designed S4 workflow container classes.

systemPipeR's Functionalities and Graphical Features

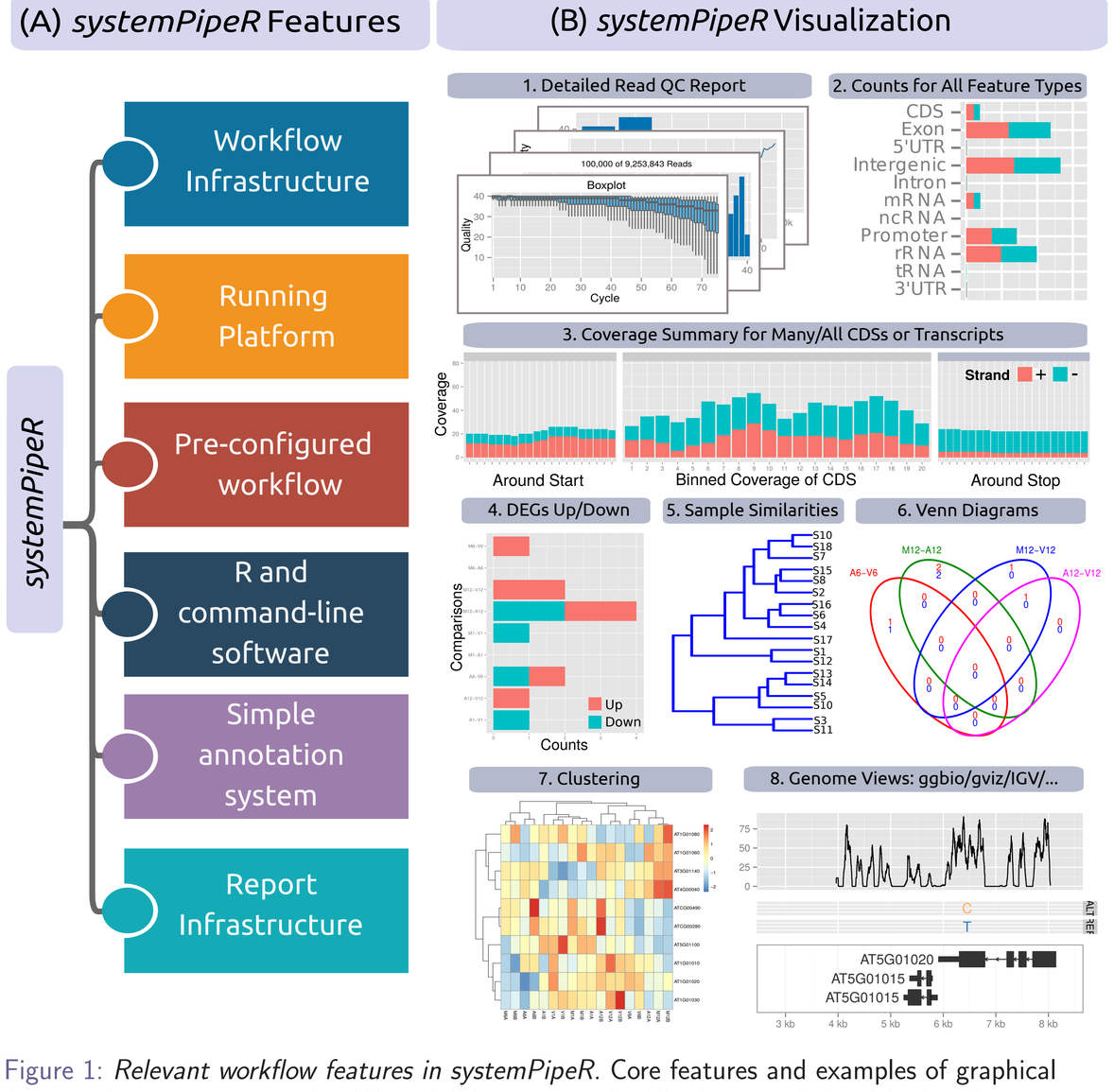

Figure 1: *Relevant workflow features in systemPipeR*. Core features and examples of graphical functionalities are given under (A) and (B), respectively. The latter include: (1) plots for summarizing the quality and diversity of short reads; (2) strand-specific read count summaries for all feature types provided by a genome annotation; (3) summary plots of read depth coverage for any number of transcripts, as well as binned coverage for their coding regions; (4) enumeration of up- and down-regulated DEGs for user defined sample comparisons; (5) similarity clustering of sample profiles; (6) 2-5-way Venn diagrams for DEGs, peak and variant sets; (7) gene-wise clustering with a wide range of algorithms; and (8) support for plotting read pileups and variants in the context of genome annotations along with genome browser support.

Acknowledgement

We acknowledge the Bioconductor core team and community for providing valuable input for developing *systemPipeR*. Funding: This work was supported by grants from the National Science Foundation (PGRP-1546879, ABI-1661152), the National Institutes of Health (U24AG051129, U19AG023122, R01-AI36959) and the National Institute of Food and Agriculture (2011-68004-30154).

References

Backman TW, Girke T (2016) systemPipeR: NGS workflow and report generation environment. BMC Bioinformatics: 17, 1-8.

Huber W, ..., Morgan M (2015) Orchestrating high-throughput genomic analysis with Bioconductor. Nature Methods: 12, 115-121.

systemPipeR Workflow Management Solutions

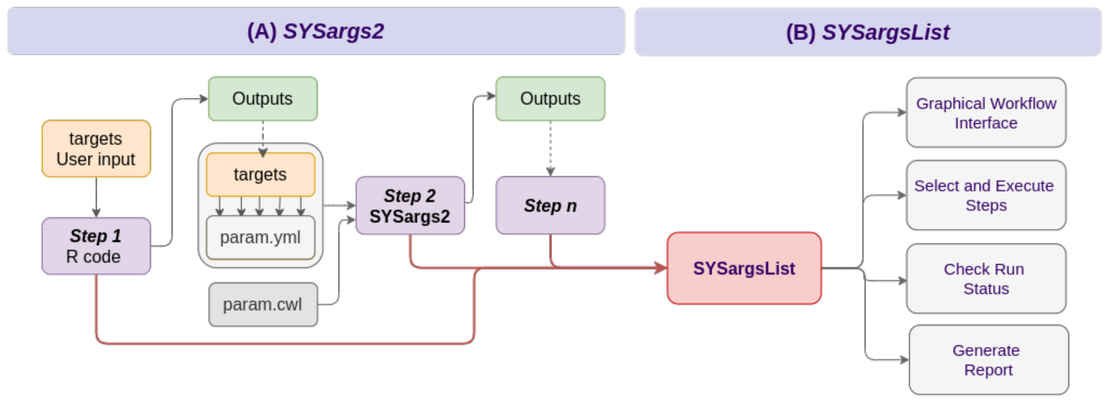

Figure 2: A central concept for designing workflows in *systemPipeR* is the use of workflow management containers (S4 class). (A) Workflow steps with input/output file operations are controlled by *SYSargs2* objects, where each instance is constructed from a *targets* and a set of workflow definition files (here *param.cwl* and *param.yml*). The only input required from the user is the initial *targets* file. Subsequent instances are created automatically. Any number of predefined or custom workflow steps is supported. (B) *SYSargsList* objects organize one or many *SYSargs2* containers in a single compound object capturing all information required to run, control and monitor complex workflows from start to finish.

Important Features

Workflow infrastructure: *systemPipeR* offers many utilities to build, control, and execute workflows entirely from R. The environment takes advantage of central community S4 classes of the Bioconductor ecosystem. Workflows are managed by generic workflow management containers supporting both analysis routines implemented in R code and/or command-line software. A layered monitoring infrastructure is provided to design, control and debug each step in a workflow. The run environment allows to execute workflows entirely or with a intuitive to use step-wise execution syntax using R's standard subsetting syntax (*runWF(sys[1:3])*) or pipes (*%>%*).

Command-line software support: An important feature of *systemPipeR* is support for running command-line software by adopting the Common Workflow Language (CWL). The latter is a widely adopted community standard for describing analysis workflows. This design offers several advantages such as: (i) seamless integration of most command-line software, (ii) support to run *systemPipeR* workflows from R or many other popular computer languages, and (iii) efficient sharing of workflows across different workflow environments.

Parallel evaluation: The processing time of workflows can be greatly reduced by making use of parallel evaluations across several CPU cores on single machines, or multiple nodes of computer clusters and cloud-based systems. *systemPipeR* simplifies these parallelization tasks without creating any limitations for users who do not have access to high-performance computer resources.

Visual, Scientific and Technical Reports: *systemPipeR's* reporting infrastructure includes three types of interconnected reports each serving a different purpose: (i) a scientific report, based on R Markdown, contains all scientifically relevant results; (ii) a technical report captures all technical information important for each workflow step, including parameter settings, software versions, and warning/error messages, etc.; and (iii) a visual report depicts the entire workflow topology including its run status in form of a workflow graph.

Shiny Web Interface: Recently, the *systemPipeShiny* package has been added that allows users to design workflows in an interactive graphical user interface (GUI). In addition to designing workflows, this new interface allows users to run and to monitor workflows in an intuitive manner without the need of knowing R.

Workflow Templates: A rich set of end-to-end workflow templates is provided by this project for a wide range omics applications. In addition, users can contribute and share their workflows with the community by submitting them to a central GitHub repository.

Containerization: Workflow templates are also distributed as Singularity containers.

Overview of Important Functions

Function Name	Description
`loadWF/renderWF`	Constructs and populates *SYSargs2* object from *CWL param* and *targets* files
`runCommandline`	Executes command-line software on samples and parameters specified in `SYSargs`
`clusterRun`	Runs command-line software in parallel mode on a computer cluster
`initWF`	Constructs *SYSargsList* workflow control module from R Markdown file
`configWF`	Controls which steps of a workflow will be run and generates new R Markdown
`runWF`	Runs all steps defined in an R Markdown file or a subset, e.g. runWF(sys[1:3])
`renderReport`	Renders scientific report based on R Markdown
`plotWF`	Plots visual workflow designs and topologies with different graphical layouts
`statusWF`	Returns status overview of workflow steps
`createWF`	Creates from command-line syntax *param.cwl* and *param.yml* files automatically from R
`config.param`	Custom configuration of the *param.cwl* files directly from R
`genWorkenvir`	Generates workflow templates provided by *systemPipeRdata* helper package
`preprocessReads`	Filtering and/or trimming of short reads using predefined or custom parameters
`seeFASTQ/seeFASTQplot`	Generates quality reports for any number of FASTQ files
`alignStats`	Generates alignment statistics, such as total number of reads and alignment frequency
`run_edgeR/run_DESeq2`	Runs *edgeR* or *DESeq2* for any number of pairwise sample comparisons
`filterDEGs`	Filters and plots DEG results based on user-defined parameters
`overLapper/vennPlot`	Computation of Venn intersects for 2-20 or more samples and 2-5 way Venn diagrams
`GOCluster_Report`	GO term enrichment analysis for large numbers of gene sets
`variantReport`	Generates a variant report containing genomic annotations and confidence statistics
`predORF`	Prediction of short open reading frames in DNA sequences
`featuretypeCounts`	Computes and plots read distribution for many feature types at once
`featureCoverage`	Computes and plots read depth coverage from many transcripts

Table 1: The table lists a subset of over 50 methods and functions defined by *systemPipeR*. Usage instructions are provided in the corresponding help pages and vignettes of the package.

Availability

systemPipeR is freely available for all common operating systems from Bioconductor.

danielac@ucr.edu