Subject Section

Anduril 2: Upgraded large-scale data integration framework

Alejandra Cervera[1], Ville Rantanen[1], Kristian Ovaska[1], Marko Laakso[1], Javier Nuñez-Fontarnau[2], Amjad Alkodsi[1], Julia Casado[1], Chiara Facciotto[1], Antti Häkkinen[1], Riku Louhimo[3], Sirkku Karinen[1], Kaiyang Zhang[1], Kari Lavikka[1], Lauri Lyly[1], Maninder Pal Singh[1], and Sampsa Hautaniemi[1,*]

[1]Research program in Systems Oncology, Faculty of Medicine, University of Helsinki, Helsinki 00014, Finland, [2]Institute for Molecular Medicine Finland, University of Helsinki, Helsinki 00014, Finland, [3]Finnish Institute of Occupational Health, Helsinki 00032, Finland

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Summary: Anduril is an analysis and integration framework that facilitates the design, use, parallelization and reproducibility of bioinformatics workflows. Anduril has been upgraded to use Scala for pipeline construction, which simplifies software maintenance, and facilitates design of complex pipelines. Additionally, Anduril's bioinformatics repository has been expanded with multiple components, and tutorial pipelines, for next-generation sequencing data analysis.
Availability: Freely available at http://anduril.org.
Contact: sampsa.hautaniemi@helsinki.fi
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Measurement technologies, such as next-generation sequencing, proteomics and automated imaging, are able to produce enormous amounts of data, which have transformed medical research into a data-rich field. While producing data from biological samples is cost efficient and easy, data analysis and interpretation has become a bottleneck. Computational frameworks that allow systematic, parallel and flexible pipeline design are indispensable for the reproducibility, maintenance and execution of large-scale analyses.

The design of current frameworks is heavily influenced by the expected end-user. Some frameworks like Galaxy (Goecks et al., 2010), Taverna (Wolstencroft et al., 2013), and GenePattern (Reich et al., 2006), offer easy-to-run capabilities of existing pipelines with a graphical user interface (GUI), while frameworks like Anduril (Ovaska et al., 2010), Snakemake (Koster and Rahmann, 2012), Ruffus (Goodstadt, 2010), and Nextflow (Di Tommaso et al., 2017), offer more flexibility in pipeline construction and integration of tools for users with at least some level of programming skills. Currently no single framework caters to all users and the differing demands of all data analysis projects (Leipzig, 2017). Here we present an updated version of the Anduril data analysis and integration framework, designed for bioinformaticians and ideal for laboratories working with few in-house samples or considerably larger datasets, for example from The Cancer Genome Atlas (TCGA) (Bell et al., 2011), which may require integration of several layers, such as clinical data and outputs of various high-throughput technologies.

The two major improvements in Anduril 2 are 1) the change from a custom-made scripting language to Scala (Odersky et al., 2004) which grants more freedom and flexibility in pipeline construction, and 2) the expansion of Anduril's bioinformatics resource bundles. These resources confer built-in support to pipelines for analysis of central technologies in biomedicine, such as high-throughput imaging (Rantanen et al., 2014), exome or whole genome, and micro- and mRNA data analysis (Icay et al., 2016). Other recent additions to the Anduril framework include both components for specific analysis such as methylation extraction and decomposition based on tumor purity (Häkkinen et al., 2018), as well as components that facilitate general data analysis through a quick interface to R library dplyr (Wickham et al., 2018) or Python Data Analysis library (pandas) (Mckinney, 2011). Anduril 2 comes with extensive documentation, which shows not only how to get started, build new pipelines and make use of parallelization, but also how to best exploit the

available components, and start processing and analyzing high-throughput data sets. Several worked examples are available in https://bitbucket.org/anduril-dev/sequencing/wiki. Anduril 2 is freely extendable and is distributed and licensed as open source software. An overview of the framework is depicted in Fig. 1.

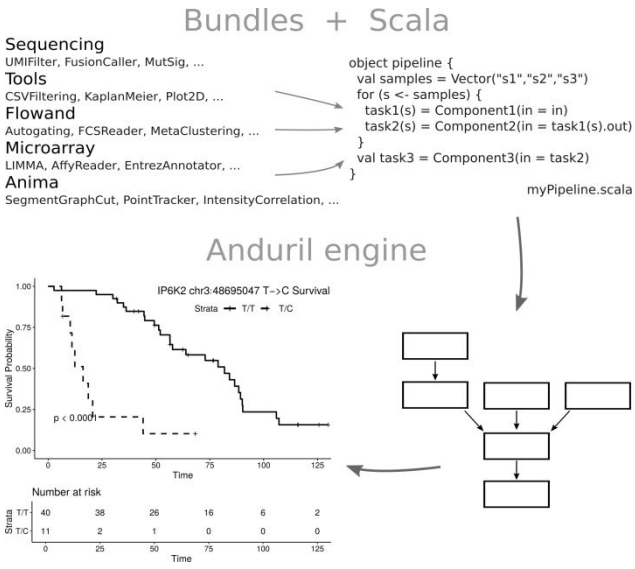

Fig. 1. Anduril 2 overview. Many common pre-processing and downstream analysis steps have been encapsulated in components coded in a variety of supported languages (R, Matlab, Python, Java). Components are organized in bundles dedicated to specific datasets (anima - image processing, flowand - flow cytometry, sequencing - omics data, tools - general, microarray) and are combined into pipelines in a Scala program run by the Anduril engine. Anduril constructs a graph and handles execution of the pipeline tasks in parallel while keeping track of the changes and status (pass/fail) of different steps to ensure reentrancy. The Kaplan-Meier curve shown here is a direct output of the case study and shows the survival difference in the TCGA ovarian patients with a T/C genotype in IP6K2 gene (chr3:48695047, p-value 0.0001).

2 Software Description

Most popular bioinformatics frameworks, including Anduril 2, handle both serial and parallel steps, complex dependencies, varied software and data file types, user-defined parameters and deliverables. Below we describe additional features and advantages of Anduril 2.

Automatic parallelization: Anduril models the component dependencies as a graph and parallelizes independent parts. The generalized prefixing of the processes enables flexible use of SLURM (Jette et al., 2002) and Sun Grid Engine (qsub).

Reentrancy: Resuming execution at the point of interruption is extremely useful when executing long-running complex pipelines on big datasets, as it spares the user from having to identify from which point onward to re-execute or which samples have been already analyzed. It is possible to update the component or their parameters and to add samples into the workflow without triggering execution of the completed independent steps. This results in significant improvements on both computing and programming time.

Dependency support: An update, such as a change in parameter, on a given step will cause re-execution of all dependent downstream processes. Components can be annotated to create synthetic dependencies between them when their input-outputs are not linked. For example, a component may not produce an output but can modify its environment, such as a database entry, and trigger downstream execution of a component marked as its dependent.

Bioinformatics resources: More than 400 components and functions for performing common tasks for diverse bioinformatics analyses are available and fully documented (see Fig. 1). Installers, for most third-party software supported by Anduril components, are included, which simplifies the task of installing the myriad of software packages used in standard bioinformatics analysis. Anduril 2 can use its own optional installation or a user-defined one. Furthermore, any component can be run outside Anduril 2 with the same parameters and inputs derived from the pipeline since the effective configuration of each component is stored in a bash script facilitating testing and providing reproducibility.

Ease of integration of new tools or custom analysis: Integrating additional tools into a pipeline is extremely simple since own or third-party code can be embedded in eval-based components. Adding a new tool to the repository of components, for private or community use, requires only defining inputs, parameters and outputs through an XML file, ideally with appropriate test cases. Tools like Taverna require third-party software to implement plug-ins to be used in the pipelines. Both Galaxy and Anduril 2 offer an easy way to build wrappers, but Anduril also supports immediate integration of custom analysis and software in any pipeline (see Supplementary for an example).

3 Results and Conclusions

3.1 Case study

To illustrate Anduril 2 in data analysis, we studied RNA-seq data from good and poor responding high-grade serous ovarian cancer (HGSOC) patients from level 1 TCGA data. We hypothesized that comparing the 10% patients with the longest response (n=26) to the 10% with the shortest response (n=24) would reveal genes and genetic variants that are associated with treatment resistance and disease progression.

An interesting finding emerged by combining the variants with survival analysis. The polymorphism (T->C in chr3:48695047) in *IP6K2* showed the most significant association to survival ($p<0.0001$) as shown in Fig. 1. *IP6K2* activity has been linked to therapy response in ovarian cancer (Morrison et al., 2002), but the mechanism on how *IP6K2* mediates apoptosis is still unclear (Nagata et al., 2005). Figures and reports produced for this case-study, as well as the pipelines for processing and analyzing the data, are available in http://csbi.ltdk.helsinki.fi/p/anduril2.

3.2 Conclusions

With many frameworks for data analysis available, the choice of which to adopt needs to take into consideration the skills and backgrounds of the user, as well as the needs of the projects. Compared with Galaxy and Taverna, Anduril 2 offers ease of integration of new tools and custom analysis as well as batch processing. Frameworks like Luigi (https://github.com/spotify/luigi) handle efficient execution of pipelines, but do not provide any bioinformatic-related components. Cromwell + WDL (https://software.broadinstitute.org/wdl/), a workflow management can mimic Anduril 2 dynamic for-loops with a scattering control flow, although nested scattering and parametric data typing are not supported. A comparison of Anduril 2 to other frameworks is shown in

Supplementary. Current and future work on Anduril 2 include integration of new tools and data types for single-cell transcriptomics and proteomics data analysis, as well as extended support for docker-based components and kubernetes integration.

Acknowledgements

We thank CSC – IT Center for Science Ltd. for computing resources as well as the current and past Hautaniemi lab members for their contributions to components and documentation. The results published here are in part based upon data generated by TCGA managed by the National Cancer Institute (NCI) and the National Human Genome Research Institute (NHGRI). Information about TCGA can be found at http://cancergenome.nih.gov.

Funding

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 66740, the Academy of Finland (project 305087), the Sigrid Jusélius Foundation, and Finnish Cancer Associations.

Conflict of Interest: none declared.

References

Bell,D. et al. (2011a) Integrated genomic analyses of ovarian carcinoma. Nature, 474, 609–615.

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.

Goodstadt,L. (2010) Ruffus: a lightweight Python library for computational pipelines. Bioinformatics, 26, 2778–2779.

Häkkinen,A. et al. (2018) Identifying differentially methylated sites in samples with varying tumor purity. Bioinformatics, 34, 3078–3085.

Icay,K. et al. (2016) SePIA: RNA and small RNA sequence processing, integration, and analysis. BioData Min., 9, 20.

Jette,M.A. et al. (2002) SLURM: Simple Linux Utility for Resource Management. Lect. NOTES Comput. Sci. Proc. JOB Sched. Strateg. PARALLEL Process. 2003, 2862, 44--60.

Koster,J. and Rahmann,S. (2012) Snakemake--a scalable bioinformatics workflow engine. Bioinformatics, 28, 2520–2522.

Leipzig,J. (2017) A review of bioinformatic pipeline frameworks. Brief. Bioinform., 18, 530–536.

Mckinney,W. (2011) pandas: a Foundational Python Library for Data Analysis and Statistics.

Morrison,B.H. et al. (2002) Inositol hexakisphosphate kinase 2 sensitizes ovarian carcinoma cells to multiple cancer therapeutics. Oncogene, 21, 1882–1889.

Nagata,E. et al. (2005) Inositol hexakisphosphate kinase-2, a physiologic mediator of cell death. J. Biol. Chem., 280, 1634–40.

Odersky,M. et al. (2004) An overview of the Scala programming language. EPFL Lausanne, Switz.

Ovaska,K. et al. (2010) Large-scale data integration framework provides a comprehensive view on glioblastoma multiforme. Genome Med., 2, 65.

Rantanen,V. et al. (2014) Anima: modular workflow system for comprehensive image data analysis. Front. Bioeng. Biotechnol., 2, 25.

Reich,M. et al. (2006) GenePattern 2.0. Nat. Genet., 38, 500–501.

Di Tommaso,P. et al. (2017) Nextflow enables reproducible computational workflows. Nat. Biotechnol., 35, 316–319.

Wickham,H. et al. (2018) A Grammar of Data Manipulation.Wolstencroft,K. et al. (2013) The Taverna workflow suite: designing and executing workflows of Web Services on the desktop, web or in the cloud. Nucleic Acids Res., 41, W557-61.