BIOINFORMATICS APPLICATIONS NOTE Vol. 28 no. 17 2012, pages 2267–2269
doi:10.1093/bioinformatics/bts368

Genome analysis Advance Access publication June 28, 2012

VAT: a computational framework to functionally annotate variants in personal genomes within a cloud-computing environment

Lukas Habegger[1,*,†], Suganthi Balasubramanian[1,2,†], David Z. Chen[3,†], Ekta Khurana[1,2], Andrea Sboner[4,5], Arif Harmanci[1,2], Joel Rozowsky[1,2], Declan Clarke[6], Michael Snyder[7] and Mark Gerstein[1,2,3,*]

[1]Program in Computational Biology and Bioinformatics, [2]Department of Molecular Biophysics and Biochemistry, [3]Department of Computer Science, Yale University, New Haven, CT 06520, USA, [4]Department of Pathology and Laboratory Medicine, Weill Cornell Medical College, New York, NY 10065, [5]Institute for Computational Biomedicine, Weill Cornell Medical College, New York, NY 10021, [6]Department of Chemistry, Yale University, New Haven, CT 06520 and [7]Department of Genetics, Stanford University School of Medicine, Stanford, CA 94305, USA

Associate Editor: Michael Brudno

ABSTRACT

Summary: The functional annotation of variants obtained through sequencing projects is generally assumed to be a simple intersection of genomic coordinates with genomic features. However, complexities arise for several reasons, including the differential effects of a variant on alternatively spliced transcripts, as well as the difficulty in assessing the impact of small insertions/deletions and large structural variants. Taking these factors into consideration, we developed the Variant Annotation Tool (VAT) to functionally annotate variants from multiple personal genomes at the transcript level as well as obtain summary statistics across genes and individuals. VAT also allows visualization of the effects of different variants, integrates allele frequencies and genotype data from the underlying individuals and facilitates comparative analysis between different groups of individuals. VAT can either be run through a command-line interface or as a web application. Finally, in order to enable on-demand access and to minimize unnecessary transfers of large data files, VAT can be run as a virtual machine in a cloud-computing environment.

Availability and Implementation: VAT is implemented in C and PHP. The VAT web service, Amazon Machine Image, source code and detailed documentation are available at vat.gersteinlab.org.

Contact: lukas.habegger@yale.edu or mark.gerstein@yale.edu

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

Received on January 6, 2012; revised on May 25, 2012; accepted on June 24, 2012

1 INTRODUCTION

Recent technological advances have significantly reduced the cost of DNA sequencing and have made it possible to sequence complete genomes on a large scale. Currently, a number of efforts aim to sequence and genotype large numbers of individual genomes (The 1000 Genomes Project Consortium, 2010). These studies have already revealed many novel single nucleotide polymorphisms (SNPs), multi-nucleotide polymorphisms (MNPs), small insertions and deletions (indels) and structural variants (SVs). In order to assess the functional impact of identified variants, a key objective is to determine whether those variants intersect with annotated elements. However, the intersection of variants with a gene annotation set is non-trivial (Balasubramanian *et al.*, 2011). First, a variant may affect only a subset of the possible transcript isoforms of a given gene or it may have different effects on alternatively spliced transcripts. For example, a variant can affect the coding region of one transcript and overlap the canonical splice site of another. In addition, for cases in which neighboring SNPs (i.e. MNPs) lie within the same codon, one must assess both SNPs simultaneously to evaluate the resultant codon change, as considering each independently could give rise to erroneous codon changes. Second, indels in coding regions can either preserve the frame or introduce frameshifts. They can also partially overlap coding exons, thereby impairing splice sites as well as coding regions. Assessing the functional impact in such cases is especially challenging. Lastly, large SVs can have drastic effects on the structure of a gene if exons are removed in whole or in part. As a result, it can be difficult to assess the overall functional impact of different types of variants on gene structures without having visual representations (Supplementary Fig. 1).

To address these challenges, we have developed the Variant Annotation Tool (VAT). Like VAT, other tools have been implemented to assess the functional impact of variants (Ng and Henikoff, 2003; Ramensky *et al.*, 2002; Wang *et al.*, 2010). One issue with these tools is that they are not cloud enabled.

Cloud-computing provides immense storage capacity and scalable compute resources as well as the ability to share data and perform collaborative analyses. Given the increasing rate of data production, many foresee that sequencing reads will be stored on the cloud. In addition, the importance of software residing in the same space as the data on which it operates requires that the analysis pipelines processing these reads migrate to the cloud as well. Thus, as VAT will constitute an integral part of such pipelines, having it reside on the cloud will be necessary.

Thus, we provide VAT as a virtual machine (VM) that can be run within a cloud-computing environment (including that operated by Amazon) to take advantage of the scalability and unlimited storage

*To whom correspondence should be addressed.
†The authors wish it to be known that, in their opinion, the first three authors should be regarded as joint First Authors.

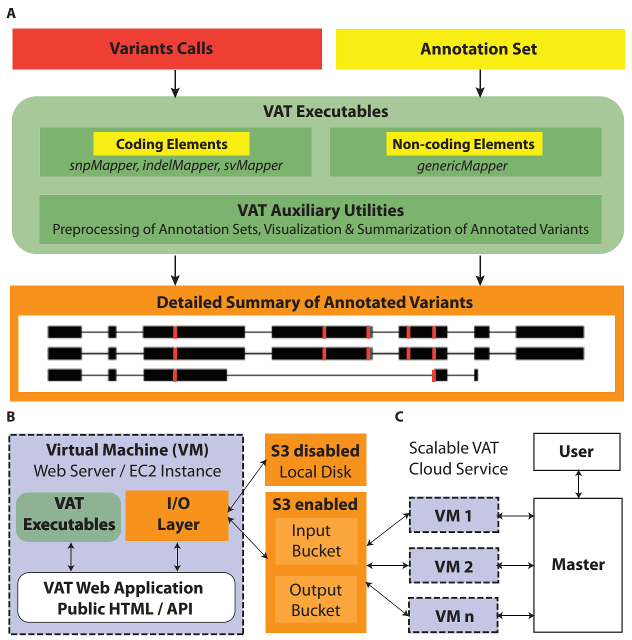

Fig. 1. (**A**) VAT comprises a number of modules that relate variants to both protein-coding genes and non-coding elements. These modules use a set of variants and an annotation set as inputs to generate annotated VCFs. (**B**) Architecture of the VAT web application. The web application may be accessed through the browser or a JSON-based interface. The I/O layer of VAT takes advantage of the Amazon S3 service and stores all data in S3 buckets or, if S3 support is disabled, simply writes to a local disk. This architecture may also be easily scaled to use more sophisticated storage schemes, such as hashing across multiple input and output buckets. (**C**) The VAT EC2 cloud service is implemented in a service-oriented architecture consisting of a master node and a number of worker nodes. The master node hosts the user-facing interface and delegates tasks on behalf of the user to the worker nodes

capacity offered by this framework. The utility of VAT has been demonstrated by its extensive use in annotating the loss-of-function variants obtained as part of the 1000 Genomes Project (MacArthur *et al*., 2012).

2 FEATURES AND METHODS

VAT is implemented in C for efficiency, and consists of a number of modules to pre-process gene annotation sets, intersect variants from multiple individuals with both coding and non-coding genes, generate summary statistics across these individuals and at the single gene level and provide clear visualization summarizing the functional impact of the annotated variants. The overall workflow is depicted in Figure 1A.

A number of modules in VAT relate variants to protein-coding genes (*snpMapper*, *indelMapper* and *svMapper*) and non-coding elements (*genericMapper*). These four core modules use an annotation set and a set of variants from multiple individuals as inputs. The variants are typically represented using the Variant Call Format (VCF; Danecek *et al*., 2011). A key feature of VAT is that the annotation is performed at the transcript level to determine whether all or only a subset of the transcript isoforms of a gene is affected. Therefore, the output explicitly shows which isoforms are affected by each variant and provides detailed information about the location of a given variant within a transcript as well as the variant's effect on its coding potential. In addition, a principal advantage of VAT lies in its ability to annotate MNPs. Moreover, VAT can be executed using gene annotation sets and genome builds beyond human, such as *Arabidopsis thaliana.*

VAT contains a number of utilities for the downstream analysis of annotated variants. For instance, an auxiliary module generates detailed summaries of annotated variants across multiple individuals as well as at the level of single genes. For variants intersecting protein-coding genes, VAT includes a module for generating an image for each gene to give a clear overview. This schematic representation displays the various transcript isoforms of a gene, which are superimposed with the annotated variants (Fig. 1A).

As shown in Figure 1B, VAT uses the Amazon web services cloud-computing platform. Each instance comprises a command-line executable of the VAT pipeline and a PHP web application, which serves as the user interface and driver for the pipeline. The VAT I/O abstraction layer may be customized using the configuration file to take advantage of Amazon's simple storage service (S3). With S3 support enabled, VAT reads input from a bucket storing raw VCF files and stores output in another bucket. Otherwise, VAT reads and writes locally.

The VAT cloud service uses the Amazon Elastic Compute Cloud (EC2) platform, and it is implemented in a service-oriented architecture consisting of a master node and a number of worker nodes. Each node consists of a VAT installation running on an EC2 VM (Fig. 1C). The master node hosts user-facing web components and serves as a load balancer for the worker nodes. A user action is forwarded by the master node as a request to one of the worker nodes. Each worker node communicates with the S3 buckets and reports updates to the master node asynchronously. VAT also uses the Amazon EC2 API to allow the master node to dynamically create worker instances. Intensive batch requests may thus be parallelized and handled efficiently. The S3 buckets' growing data are then available for further analyses.

3 CONCLUSIONS

In summary, VAT offers a combination of unique advantages in variant annotation. First, VAT operates as a VM in a cloud-computing environment, which is likely to serve as the future framework for the collaborative analysis of rapidly growing datasets. Second, VAT provides a novel means of clearly visualizing the functional impact of variants across different transcript isoforms of a given gene. Third, VAT can be used to functionally annotate MNPs, which has often been challenging. Fourth, VAT provides output files in VCF format. This readily facilitates the post-processing of output files with tools that are already widely used by the community, such as those used for variant filtering. Given that VAT has been an integral part of many analyses conducted as part of the 1000 Genomes Project, we believe that it will be of broad utility in other research contexts.

ACKNOWLEDGEMENT

We thank Raymond Auerbach for critical reading of this article.

Funding: National Institutes of Health.

Conflict of Interest: none declared.

REFERENCES

Balasubramanian,S. *et al*. (2011) Gene inactivation and its implications for annotation in the era of personal genomics. *Genes Dev.*, **25**, 1–10.

Danecek,P. *et al*. (2011) The variant call format and VCFtools. *Bioinformatics*, **27**, 2156–2158.

MacArthur,D.G. *et al*. (2012) A systematic survey of loss-of-function variants in human protein-coding genes. *Science*, **335**, 823–828.

Ng,P.C. and Henikoff,S. (2003) SIFT: predicting amino acid changes that affect protein function. *Nucleic Acids Res.*, **31**, 3812–3814.

Ramensky,V. *et al*. (2002) Human non-synonymous SNPs: server and survey. *Nucleic Acids Res.*, **30**, 3894–3900.

The 1000 Genomes Project Consortium. (2010) A map of human genome variation from population-scale sequencing. *Nature*, **467**, 1061–1073.

Wang,K. *et al*. (2010) ANNOVAR: functional annotation of genetic variants from high-throughput sequencing data. *Nucleic Acids Res.*, **38**, e164.