Querying the genome: the missing link of the evidence layer

Vineet Bafna* George Varghese

July 21, 2011

1 Executive summary

We consider the imminent future where individual (*donor*) sequences will be generated as chromosomal fragments. A reliable catalog of the variations in the donor genome relative to a standard reference will be the first step in any biomedical inference. In this short note, we propose two things

- first, the development of a layered abstraction of software modules that processe, map, compresses and queries the donor data for cataloging variations. The layering provides a context for much of the different software that is being generated.
- second, we propose the implementation of two specific layers. The first is a compression layer which will extend the ideas presented in Kozanitis []. The second is an *evidence layer (EL)*. The EL is a collection of APIs that efficiently retrieve *all* raw data relevant to inferring specific variations.

Our vision is that the EL is a critical missing link in making seamless queries and its development will spur the development of novel inference tools for visualizing and validating observed variations, as well as their use in biomedical research.

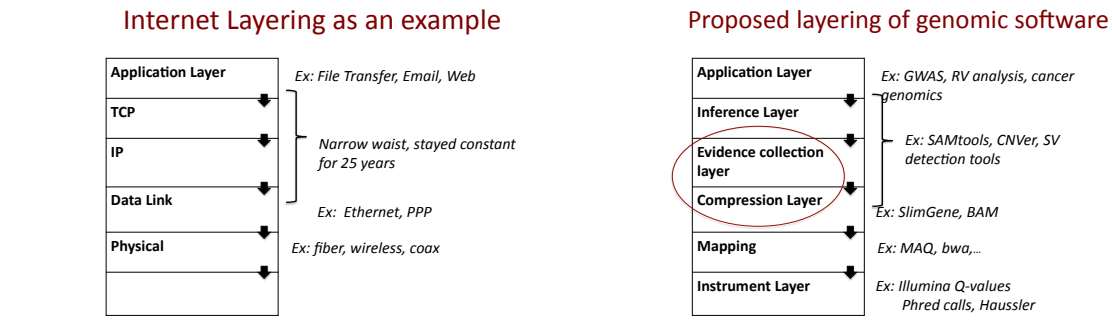
2 Rationale behind abstraction

Figure **??** provides a direct analogy between the development of internet traffic protocols and genome sequencin data. A thin 'waist' of TCP/IP protocols provide abstractions for a variety of applications above. The applications using TCP/IP are completely oblivious to the bottom layer, where internet traffic is routed seamlessly over different hardware (fibre, copper, wireless) even though each has its own error characteristics. On the one hand, this provides a loss of efficiency; on the other hand, the abstraction, and the ease of use is exactly what has revolutionized the development of applications using the internet.

Figure **??**b provides an analogy of the genomic software layers that are currently in play, either explictly or implicitly. We will describe the layers in some detail below, but we start by describing the assumptions that motivate this abstraction. In the near future, it is likely that

1. Genome sequences will continue to be fragmented (sub-chromosomal); no technology on the horizon comes close to sequencing entire chromosomes in one shot.
2. The most cost-effective sequencing will continue to be a randomized, redundant sequencing of either the entire genome or specified regions. The redundancy (often $\sim 30\times$ or higher) will add to the data bottleneck.

*



(a) (b)

Figure 1: aa

3. In sequencing members of the same species, the genomes are widely expected to be near-identical. It is the few variations that are important in mediating phenotypes. The preferred method for cataloging variations will involve comparing all sequences against a single *reference*.

The final assumption is mainly for computational efficiency. For n individuals (*donors*), this implies $\sim n$ genomic comparisons instead of the $\binom{n}{2}$. On the flip side, it does not provide an adequate catalog of *insertions*; sequence fragments present in the donor, but not in the donor the reference. More on that later.

With these assumptions, the layering in Figure **??**b becomes somewhat natural. *Instrument specific software* interprets raw instrument data (often, fluorescence signals) as nucleotide base-pairs of genomic fragments, often with error-profiles. These are *mapped* to a standard reference, typically the first human reference assembly that was derived as a composite sequence derived from anonymous donors []. The mapping reveals variations, which can form the basis of new compression schemes that store only the edits relative to the reference [], and also provide the basis for the variation catalog.

Our understanding of human variation is very incomplete, with major new sources of variation identified in the last few years, as described in the following section. The *inference of variations* is therefore an important software layer based on the mappings of donor sequences against a reference as well as knowledge of error-profiles to help distinguish true variations from sequencing and mapping errors. This layer is the motivation for much recent bioinformatics research.

We note that each specific variation typically needs only a fragment of the donor sequence and its mapping to the reference, and propose an *evidence layer*, that provides this evidence upon demand. As huge numbers of individuals are sequenced, most inference will be in the *query* mode, limited to a few genes, and a subset of the variations. Therefore, the explicit development of the evidence layer will help make inference more efficient. Finally, we have the top layer of applications (biomedical diagnostics, pharmacogenomics, etc.) that use the inferences to biomedical research.

So, what exactly is the analogy to computer networks? Note first the narrow waist analogy. While we have a huge variety of instrumentation producing sequences at the bottom layer, and likewise a suite of applications at the top that exploit the knowledge of genetic variation, there are ultimately only a handful of variation types (see Section **??**). Unlike networking however, the separation of software layers is not explicit in genomics. Thus, we have applications at the top that seek to exploit deep understanding of the instrument. Here, we suggest making the separation explicit, so that each layer communicates only with the layers above and below.

We reconize the heresy in the analogy. Unlike engineering, there is an inherent resistance in the sciences (especially Biology) to abstract away from the raw data. At the same time, excessive attention to the error characteristics, and other arcane details of sequencing hardware actually limits the availability of people to exploit the wealth of information.

3 The narrow waist of human genetic variation

VB to fill in.

3.1 Human genetic variation

1. small nucleotide changes: SNPs, SNVs, MNVs
2. Large structural variation
3. copy number changes
4. epigenetics

3.2 All against one, or all pairs

1. Pros and cons of comparing to a single reference, including fewer comparisons, compression. Problem with insertions
2. What makes a good reference?

3.3 Discovering variations versus querying for them

4 EL-API: an API for the evidence layer

We start with some definitions: a *read* is a DNA clone sampled from a donor. A *sub-read*, or a fragment is the part of the read that is sequenced. A paired-end read will have two sub-reads, but newer strobe-sequences can have multiple sub-reads.

We propose the following API. Our considerations in desgning the API included the following: the API must be (a) technology agnostic; (b) able to obtain the evidence for all inferences regarding human variation; (c) concise, and simply stated; and (d) allow for efficient implementation. The EL-API automatically assumes a collection of reads mapped to a reference. For simplicity, we assume that all reads are from a single donor sequence. However, the API extends unchanged to a population of individuals. See Figure **??**

1. Partner: Given a sub-read, or a read as input, identify all of the sub-reads that come from the same read.
2. BestMap: The input is a collection of reads or sub-reads. The output is the location of the optimal mapping of the sub-reads, along with an encoding of the alignment.
3. AllMaps: The input is a collection of reads or sub-reads. The output is the set of multiple locations where the sub-reads match up, along with an encoding of the alignments.
4. MapsTo: The input is a collection of intervals I. The output is a collection of all reads that whose bestmap alignments intersect with I.
5. Coverage: The input is a collection of intervals I. The output is the number of reads in MapsTo (**Derived from MapsTo**).

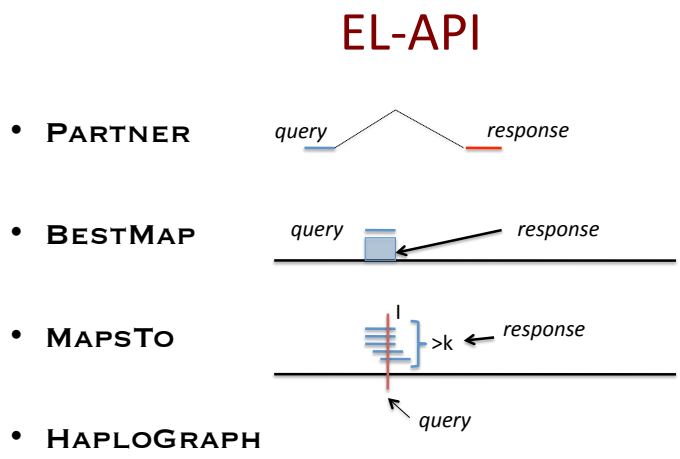


Figure 2: A cartoon depiction of the evidence layer API.

6. HAPLOGRAPH: The input is a collection of locations S corresponding to known SNVs. The output is a graph $G(S, E_S)$, where each edge $(u, v) \in E_S$ is labeled with reads whose sub-reads span both u and v. An edge exists only of the corresponding set of reads is non-empty.

5 Using EL-API for inferences

5.1 Calling SNPs:

In order to infer the alleles at a given site, we can use MAPSTO to identify all sub-reads that overlap, and their alignments.

5.2 Large deletions:

The typical questions in this case are:

Q1: Does the individual have a deletion in a gene (specified as a collection of intervals, I? **Ans.** Use MAPSTO to quickly retrieve all sub-reads that map to I. Return the subset of reads, whose partners have length-discordant mapping. The inference layer will also use ALLMAPS to varify that the discordant reads are accurate.

Q2: Identify all deletions that are supported by at least k reads? **Ans.** First retrieve all regions, where the COVERAGE is at least k. For each such region, check if the reads have length-discrepant partners. Return all length-discrepant reads (using MAPSTO) if the coverage is at least k.

5.3 Haplotypes:

Q1: Given two variant locations, a, b, and a collection of variant locations S, return the two haplotypes connecting a, b using variants in S. **Ans.** Use HAPLOGRAPH on S to get a graph. Prune the graph to get the sub-graph G' connecting a, b. Return G' and all reads labeling its edges.

Q2: Identify all deletions that are supported by at least k reads? **Ans.** First retrieve all regions, where the COVERAGE is at least k. For each such region, check if the reads have length-discrepant partners. Return all length-discrepant reads (using MAPSTO) if the coverage is at least k.