GEMINI: The Genomic Search Engine

A Major Qualifying Project

Submitted to the Faculty of

Worcester Polytechnic Institute

in partial fulfillment of the requirements

for the Degrees of Bachelor of Science

in

Bioinformatics & Computational Biology

and

Computer Science

by

Timothy DeFreitas

May 2015

Approved by.....

Patrick Flaherty

Assistant Professor, Biomedical Engineering Department

Approved by.....

Carolina Ruiz

Associate Professor, Computer Science Department

Abstract

Recent large-scale genomics projects have made genomic data for thousands of research samples publicly available to answer a diverse range of questions. Traditional search paradigms are based on string matching in the title or description, which can be slow and error-prone. We have developed GEMINI, a search engine that uses the data itself as the query object and a vantage-point tree to organize profiles. We show that GEMINI accurately identifies nearest-neighbor samples when applied to breast and ovarian cancer gene expression data from The Cancer Genome Atlas project in $\mathcal{O}(\log n)$ time.

Contents

List of Figures

Chapter 1

Introduction

The collection and curation of large genomic databases has grown exponentially since the first human genome was sequenced in the early 2000s. The presence of publicly available sample data have enabled the successful translation of basic research into therapeutics by allowing unprecedented aggregation and re-use of data. For example, type 2 diabetes was linked with the CD44 receptor in adipose tissue by re-using 1,175 gene expression microarrays. And there are dozens of large public databases currently available, supporting a wide variety of data types. The GEO database for microarray data currently contains more than 800,000 samples [2], the International HapMap3 Project contains 16 million common small nucleotide polymorphisms (SNPs) [9], and the Cancer Genome Atlas Project hold more than a petabyte of genomic data on 20 types of cancer [34].

Unfortunately, as the size of these repositories grow, the ability to find relevant data by browsing or keyword search diminishes unless a mechanism is provided to extract and compare useful data. With the ever-decreasing cost of next-generation sequencing technologies, the ability to meaningfully browse these datasets will only deteriorate. One solution is to generate a high-quality search engine that links related resources in the repositories, like the PageRank algorithm organizes access to the internet [38]. These algorithms rely on intelligent data structures to efficiently organize related information.

This paper describes GEMINI [13], a website and search tool built with python to

support nearest-neighbor search for bioinformatic data. GEMINI uses a vantage-point tree (vp-tree) data structure to store genomic data records [35, 45, 13]. The vantage-point tree is a special case of a binary search tree where the left subtree of a node contains records that are closer than some distance, μ, and the right subtree contains records farther than μ. Thus at each node the feature space is partitioned from the vantage point of a node. GEMINI allows users to query databases of gene-expression from the Cancer Genome Atlas, enabling researchers to find relevant resources quickly and easily.

Chapter 2

Background

We are living in a golden age of bioinformatics analysis. With sequencing and data-generation technologies growing at an unprecedented rate, there are thousands of opportunities for discovery, even using solely public data from web databases. New visualization and aggregation tools, like mapping the differences of human gene expression across cell types are just now becoming possible [30]. With GEMINI, we hope to further facilitate novel research methodologies by enabling even more sophisticated nearest-neighbor searching.

2.1 Growth of Biology Data

Computer technology increases famously quickly –according to Moore's law [41], the number of transistors per chip (and therefore approximate computing power) doubles every 18 months. This trend, though largely a marketting goal by manufacturers and not truly a law, has held roughly true for nearly 40 years.

But sequencing technology has grown even faster in the last two decades. Since 2001, when the complete human genome was first sequenced [29], the cost per genome has dropped from billions to just a few thousand dollars [31]. As shown in the figure below, next-generation sequencing technologies dropped the total cost dramatically. Similar techniques for generation of other types of -omics and bioinformatic data such as microarrays [11] have let to generous quantities of data available for analysis.

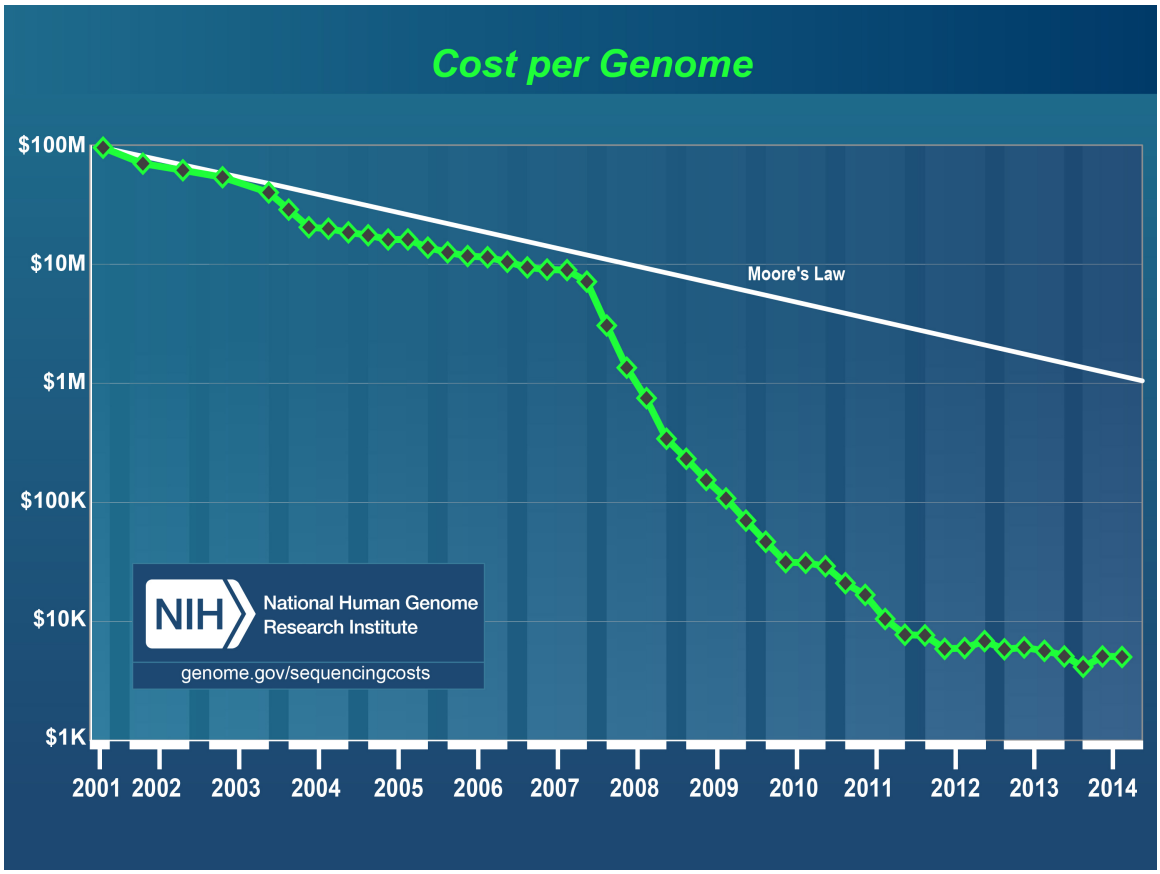

Figure 2-1: Total cost to sequence one human genome since 2001 [21].

2.2 Types of Data

Biology is a diverse field, with each sub-field producing different types of data with varying analytical challenges and approaches. Some of the most common data types for genomic analysis that GEMINI could potentially service are listed here.

2.2.1 Nucleotide and Protein Sequences

Biologists have known about the presence of nucleotide bases for more than 50 years, but recent developments in rapid sequencing technologies have produced a great deal of sequence data for analysis. By 1966, scientists had established the trinucleotide sequences that encoded each of the 20 amino acids present in proteins [36]. In computer science, a sequence is a string, an ordered list of characters with a specific alphabet. All words and sentences in English are strings with an alphabet of A-Z plus punctu-

ation. Chromosomes are strings with an alphabet of the 4 nucleotide bases A, G, C, & T, and each protein can be represented by the string of amino acids that created it, a direct translation of the mRNA sequence of the encoding gene. As described in the next section, biologists can therefore use algorithms for string comparison such as edit distance to evaluate similarities and deduce evolutionary history. Additionally, the presence of certain features in these sequences can be markers for disease.

2.2.2 RNA/Gene Expression

Every cell in an organism contains the same set of DNA, yet clearly each cell differs in function. This is a result of differential protein expression across each cell type. Typically, regulation of protein synthesis is done through promoter and silencer proteins which augment or inhibit the production of mRNA transcripts. If the mRNA for a sequence is present in large amounts, there will be a large amount of that protein present. Using microarray technology, scientists can measure the levels of specific sequences of mRNA from a given sample, or even the quantities of proteins themselves. Thus scientists can compare the expressed proteins from different samples. Thus a scientist could construct an n by m matrix where n is the number of samples and m is the number of genes being measured, and each entry is a sample's expression level for a particular gene.

2.2.3 Copy Number Variation

Mendelian genetics proposes that each parent donates sets of alleles to their offspring, and the combination of received alleles specifies a specific physical outcome, a phenotype. However, research has shown that this model is overly simple, and that cells show variation in the number of copies of each gene received [39]. Copy number variation could be expressed in a similar data structure as gene-expression, keeping track of the relative copy number for each gene, but also of the variation in non-coding regions. A patient with breast cancer might show an increase in the quantity of oncogene copies, or even in non-coding trinucleotide repeats (NTRs) [17]. Comparing the

copy number variation between patients could lead to diagnostic tools for disease.

2.2.4 Methylation

In a similar vein, chromosome structure has been shown to affect the expression of proteins [6]. Methyl groups added to the DNA backbone by methyl-transferase enzymes can prevent RNA synthesis, and methylation varies between individuals and cells. In addition to the nucleotide sequence, biologists can analyze the positions of methyl groups to infer diagnostic information. This data can be represented as a vector of chromosome positions with a methyl group, and multiple samples could be compared with a sparse matrix.

2.2.5 Clinical Information

With the digitalization of health records, additional patient information can be linked to any of the more sophisticated techniques. This enables text-based data mining of prior medical history and confounding factors that would otherwise not be possible at scale. In some cases, as with FDA case report submissions [43], there is a data standard for recording specific information, most commonly a variant of XML. In other cases, especially research contexts, experiment-specific data is recorded in a tabular format such as excel. Associating pieces of medical history with more specific sequencing data could enable highly sophisticated diagnostics.

2.3 Challenges of -omics Data

Genomics data, and many other biological data sets, have analytical challenges associated with them. In many cases the data set is large and the underlying biological mechanisms are not yet understood. Unlike physics and math, for which most interactions can be modeled by relatively simple equations, biological data is very noisy and correctly controlling all effects on an experiment can be close to impossible. This section discusses the primary challenges that "-omics" datasets (genomics, proteomics,

etc.) suffer from.

2.3.1 Dimensionality

Cells are incredibly complex, especially in the case of multicellular organisms like humans. Though humans don't have the longest genome, each one is still roughly 3 Gigabases long and has around 20,000 genes [44]. Thus any attempt to compare whole-genomes to one another suffers from the curse of dimensionality. Common organizational structures perform poorly with that many features, and common algorithms such as sequence alignment are computationally expensive. Additonally, the datasets can become hard to store and manipulate in memory. Large matrices of gene expression can exceed 4Gb per file, requiring enormous computing and storage resources to work with.

2.3.2 Complex Networks of Unknowns

A related challenge that arises from the dimensionality of biological data is that there are many systems about which our knowledge is incomplete. For example, there are some protein pathways and interaction networks we understand very well, like the Krebs cycle, but certain diseases remain poorly understood. As such determining the cause of one's set of symptoms based on their genomic fingerprint can be difficult. There are interactions that are not yet known, and the sample populations for many research studies are too small to discover all causes of a particular disease. Cancer, for instance, isn't just one disease [40, 7], and cell growth is regulated by hundreds of different proteins. Each individual may have a different cause for the same end result, and because some of these interactions are not well studied, we cannot make informed judgments based solely on similarity or differences between samples.

2.3.3 Noise

Noise is not a unique problem to genomics research, but it exists on an enormous scale. Organisms are diverse, both within and across species, having polymorphisms

in gene sequences, modifications to chromosome structure, differing environmental factors like diet & sun exposure, and many others. This results in variation in many experiments because it is very hard to control for genomic differences in non-model organisms.

2.4 Algorithms and Data Structures

As described above, the classic types of genomic data are nucleotide and protein sequences generated from an organisms chromosomes. This data, an ordered sequence of characters, is known to computer scientists as strings [28], and are commonly stored in arrays, allowing random access to elements and efficient splicing operations. When stored as strings, many common biological analyses become similar to string operations in other contexts.

Consider the problem of comparing similar words as you might encounter in a spell-checking application. Assuming one mistypes a word, the most likely replacement word to suggest would be the word with the smallest edit distance – it is a natural assumption to assume the user makes few errors and got close to the intended word. Biologists make similar assumptions about related proteins in an evolutionary history. Nucleotide sequences with relatively few insertions, deletions, and mismatches are considered closely related. Using algorithms for strings, such as the Needleman-Wunsch algorithm [33], biologists can compute optimal alignments.

For other common data types, like RNA expression matrices, a variety of linear algebra and statistical algorithms are available. Hierarchical [10], k-means [24], and density-based approaches like DBSCAN [14] can be applied to a variety of biological data sources to identify groups with similar characteristics. These groups can map naturally to known subtypes or can show novel distinctions between subgroups for a particular sample set.

2.4.1 Graphs and Trees

Biological interactions, such as those between proteins in a cell, are often best modeled using graphs. Graphs are formally defined as a set of nodes V and a set of edges E where an edge can be either undirected or directed and connect exactly two nodes [5]. With this model, a protein could be considered a node, and if two proteins interact, there is an edge between them. A variety of interesting protein network questions can then be posed in the form of common graph problems, like finding a minimum cut or a maximum flow [27].

Trees are a special subset of directed graphs which have a root node and contain no cycles. Thus algorithms on graphs will also work on trees, but trees are commonly used to improve search time for complex data. Since no cycles exist, if you can prove your search target is not in a subtree, your algorithm does not need to search it. The simplest such tree is a binary-search tree, but the multidimensional equivalent KD-tree [4] has been successfully applied to biological data [25]. Numerous related tree structures are in use as well, including R*-trees [3] and SR-trees [26].

2.4.2 Dimensionality Reduction Techniques

The inherent size of biological data still poses a problem for many common algorithms, especially trees involving metric space partitions like kd and vp-trees. For datasets with many dimensions, such as gene-expression, dividing across one dimension becomes less significant, and there is little performance improvement to the search techniques. Because of this "curse of dimensionality", techniques have been developed to attempt to isolate relevant features and remove noise before using the above algorithms. One such technique is principal component analysis [22], which seeks to find a projection of the data that maximizes the explained variance in the first few dimensions. From a linear algebra perspective, this is similar to eigenvalue decomposition, and it simplifies repeated matrix operations to improve performance. Other techniques involve trying to factor an input matrix to identify subtypes. Oncologists have been able to identify subtypes of lung cancer using non-negative matrix

factorization [19].

2.5 Vantage-point Trees

GEMINI uses a vantage-point tree structure to improve the speed of searching for nearest neighbors in a database [13]. The vantage-point tree is a special case of a binary search tree where the left subtree of a node contains records that are closer than some distance, μ, and the right subtree contains records that are further than μ. The tree gets its name because the subtree nodes are partitioned from the vantage point of the current node. The advantage of the vp-tree in genomic search applications lies in the fact that is does not impose a particular coordinate structure on the data and instead employs a user-definable metric to measure distance. The construction and search algorithms used by GEMINI are described below.

2.5.1 VP-tree Construction

Construction of the vp-tree takes $\mathcal{O}(n \log n)$ time for records with constant dimension where n is the number of records in the dataset. We briefly summarize the simplest version of the recursive construction algorithm here and refer to the original article for further details and extensions [45].

function `MakeVPTree(`$\mathcal{S}$`)`**:**

Data: a set of records, $\mathcal{S}$

Result: a pointer to the root of the vp-tree

if $\mathcal{S} = \emptyset$ **then return** $\emptyset$;

node $\leftarrow$ a pointer to a new node;

node.p $\leftarrow$ random element of $\mathcal{S}$;

node.mu $\leftarrow$ median $d(\text{p}, s)$ over all $s \in \mathcal{S}$;

L $\leftarrow \{s \in \mathcal{S} - \{p\} | d(\text{p}, s) < \text{mu}\}$;

R $\leftarrow \{s \in \mathcal{S} - \{p\} | d(\text{p}, s) \geq \text{mu}\}$;

node.left $\leftarrow$ `MakeVPTree(`L`)`;

node.right $\leftarrow$ `MakeVPTree(`R`)`;

return node;

Algorithm 1: Vantage-point tree construction algorithm [13]

This binary search tree construction works by taking a set $\mathcal{S}$ of records. If the $\mathcal{S}$ is not empty, we create a new node and store a random element, p, in the node. We store the median distance between p and all the other elements in $\mathcal{S}$ in mu in the node using any distance metric that satisfies the triangle inequality. We partition the set $\mathcal{S}$ into two roughly equal size sets L and R, where L contains all of the elements of $\mathcal{S}$ that are closer to p than the median distance, mu and R contains all of the elements of $\mathcal{S}$ that are further than mu. The function recurses by calling itself with arguments L and R for the left (closer) and right (further) subtrees The recursion ends when the subtree sets are empty and the algorithm returns the pointer to the root node. Clearly, because the size of the set in each subtree is half the original set, due to the use of the median distance, the time to construct the tree is $\mathcal{O}(n \log n)$.

2.5.2 VP-tree Search

Search in the vantage-point tree proceeds by recursive depth-first search. The left subtree of a node contains records that are closer than μ from the vantage point of the current node's records. Symmetrically, the right subtree contains records further than μ.

If we have a query profile, q and a vantage-point node, p, by symmetry and the triangle inequality of a distance metric $d(\cdot, \cdot)$, we have

$$d(q, s) \geq |d(q, p) - d(p, s)| = d_p(q, s), \tag{2.1}$$

where s is any other record in the database and $d_p(\cdot, \cdot)$ is defined as the vantage-point distance. Since the vantage-point distance shrinks the true distance between q and s, if $d_p(q, s) \geq \tau$, then $d(q, s) \geq \tau$ [45, 13].

Suppose that we have found a record at distance τ from the query and we are at vantage-point node p in the tree. If $d(p, q) \geq \tau + \mu$, then the nearest-neighbor is not closer than μ and we can fathom (remove from further consideration) the left subtree as shown in Figure 2-2A. Conversely, if $d(p, q) + \tau \leq \mu$, then the nearest-neighbor is certainly closer than μ and we can fathom the right subtree (Figure 2-2B). Thus, the vantage-point tree data structure allows us to exclude records from examination and we achieve super-linear search time. As shown by Yianilos, the average-case querying time scales as $\mathcal{O}(\log n)$ when the data is low-dimensional [45].

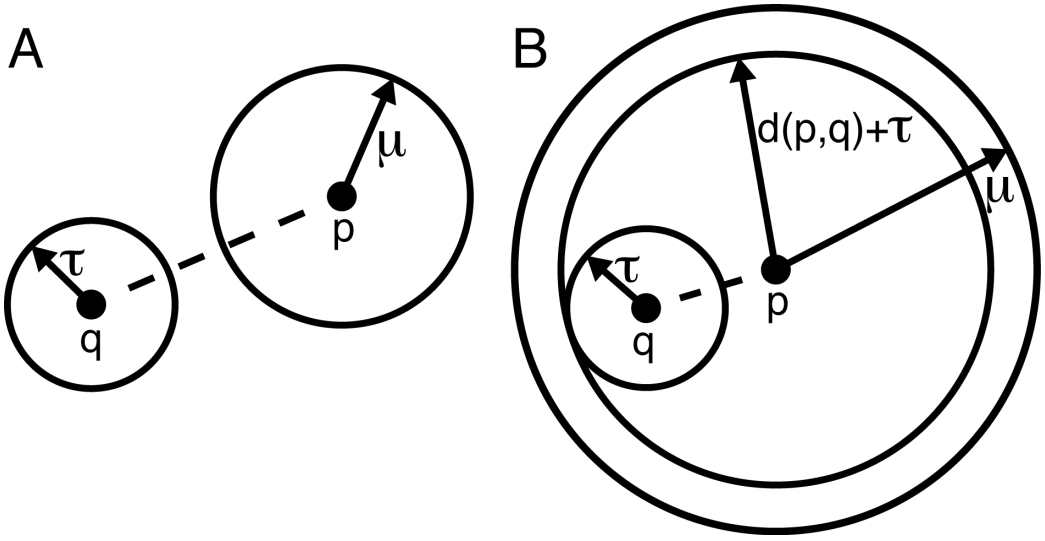

Figure 2-2: Vantage-point tree structure allows search algorithm to fathom subtrees. [13]

The search algorithm can be written as a recursive depth-first search algorithm as described previously [45]. The algorithm holds the node of the nearest neighbor

in the global variable `best` and is initialized with $\tau \leftarrow 0$. If d(q, node) $< \mu + \tau$, only the left subtree is traversed and if d(q, node) $> \mu - \tau$ then only the right subtree is traversed. If $\mu - \tau \leq$ d(q, node) $\geq \mu + \tau$ then both subtrees are traversed.

function `SearchVPTree(`*`node`*`):`
  Data: a vantage point tree root node, `root`
  Result: a pointer to the root of the vp-tree
  if *node* $= \emptyset$ **then return**;
  if *d(q, node)* $< \tau$ **then**
    $\tau \leftarrow d(q,$ `node`$)$;
    `best` $\leftarrow$ `node`
  end
  if *d(q, node)* $< \mu + \tau$ **then** `SearchVPTree(`*node.left*`)`;
  if *d(q, node)* $> \mu - \tau$ **then** `SearchVPTree(`*node.right*`)`;
return

Algorithm 2: Vantage-point tree search algorithm [13]

Chapter 3

System Design

The GEMINI system envisioned requires significant scalability and flexibility, needing to provide a flexible enough data structure to support a variety of resources and queries. However, the interface itself needed to be simple. Many modern bioinformatics tools suffer from feature overload, making simple exploratory research difficult. This section describes the software tools and design paradigms used to enable the extensibility of GEMINI.

3.1 System Goals

GEMINI was designed to support quick and easy exploratory analysis of genomic data. The idea here is that although genomic tests themselves are becoming cheap and fast to perform, it is difficult to use this data directly. A couple of core use-cases describe the ideal features of the system.

An Oncologist at a research hospital sees dozens of patients a year for a handful of common cancer types, such as skin, breast, and ovarian cancer. Though each patient is unique, the Physician is aware that some patients respond well to similar treatmens such as Avastin or Herceptin while others do not. He maintains extensive medical histories on his patients, in addition to some standard genomic analyses and hopes to improve the outcomes of future patients. Now, a new patient comes in and receives a battery of genomic testing. The Physician uploads the results to GEMINI,

and is instantly notified of patients with similar features, and is connected directly to relevant clinical information. By evaluating treatment options on similar patients, the Physician improves the success rate of his chosen treatments and the lives of his patients.

GEMINI was also designed with researchers in mind. A bioinformatics researcher is studying new disease, and receives a sample and little additional information. The new disease is relatively unknown, so the researcher does a broad search using the genomic data from the sample to see similar sets of samples. He looks at the results and sees that the RNA expression pattern matches a group of patients who suffered a viral infection. The researcher hypothesizes that the new disease is similar to a virus and leads to further research. He adjusts to a more specific type of search on a particular dataset, in an attempt to identify potential modes of action.

GEMINI must clearly manage a large wealth of data, but in the first use case the dataset searched is fairly small. The Physician knows the patient has cancer, and can limit his search to a particular set of samples. In the second, the researcher wants to cast a broad net. To handle both cases, GEMINI must scale effectively, and be simple to use. Complexity should be minimized up front, but support refinement.

3.2 Python

Python is very high-level programming language in widespread commercial and academic use, popular in the bioinformatics field for it's easy-to-learn syntax and its wide array of libraries [8]. A variety of scientific packages such as NumPy [37], SciPy [23], and Pandas [32] support numerous cutting edge algorithms and data analysis methods for a variety of different data types. Python also supports both functional and object-oriented programming paradigms, and can interface directly with optimized C code for efficient execution. This makes python an easy choice for exploratory research and prototyping. Additionally, python supports rapid and efficient deployments through the use of virtual environments, so that software dependencies can be minimized and release code shared easily.

3.3 Django and Bootstrap

Interfacing between web sites and analytical frameworks is challenging, but possible by using a variety of web scripting technologies. Though theoretically possible using entirely javascript, we chose the Django [16] web framework to build our site. Django provides an html template language for embedding python objects into web pages, and provides built in security and automated testing features. Django simplifies the database development by optimizing for our own models, such as search queries and trees. To provide stylistic improvements, as none of us are familiar with direct CSS development, we used the Twitter Bootstrap for javascript and CSS provided by MaxCDN [1].

3.4 System Organization

Python provides the backend development for all of the search algorithms and data classes used by Gemini. Currently expression matrices are stored in the HDF5 [15] file format. These support both the collection and organization of the data, as well as interfacing with Django. A Django application is deployed on an Apache server, providing the web functionality and serving the HTML pages created for the project. Cascading Style Sheets (CSS) are loaded externally from a CDN and the combined webpage is presented to the user.

Chapter 4

Experimental Results

Prior to implementation of the GEMINI website, tests were performed to evaluate the ability of the vantage-point tree structure and dimensionality reduction techniques to successfully facilitate efficient nearest-neighbor searching of genomic data. For our experiments, we used a combination of Level 3 gene-expression data from the Cancer Genome Atlas project and randomly-generated databases to test scalability. We first compared a vantage-point tree structure to brute force search times and a similar tree method, then evaluated the ability for vantage-point trees to elucidate interesting data from real cancer datasets.

4.1 Comparison to Other Search Methods

We compare the vp-tree to the related KD-Tree as well as a brute-force approach in Figure 4-1. For this experiment, randomly generated databases of gene-expression profiles were created and searched using a randomly selected query sample. Because tree structures that use space metrics suffer from performance losses in completely random high-dimensional datasets [20], 4 of the "genes" in each sample were given large variance ($\sigma^2 = 1.0$), while the remaining 2996 genes were distributed with lower variance ($\sigma^2 = 0.1$)[13]. As discussed in the next chapter, various dimensionality reduction techniques can emulate this procedure.

The brute force algorithm simply compares the query to every record in the

database. As expected, the brute force (BF) approach scales linearly in the size of the database. However, the tree structure approaches scale as the log of the size of the database because of the savings achieved by being able to exclude distance samples from consideration based on their position in the vp-tree.

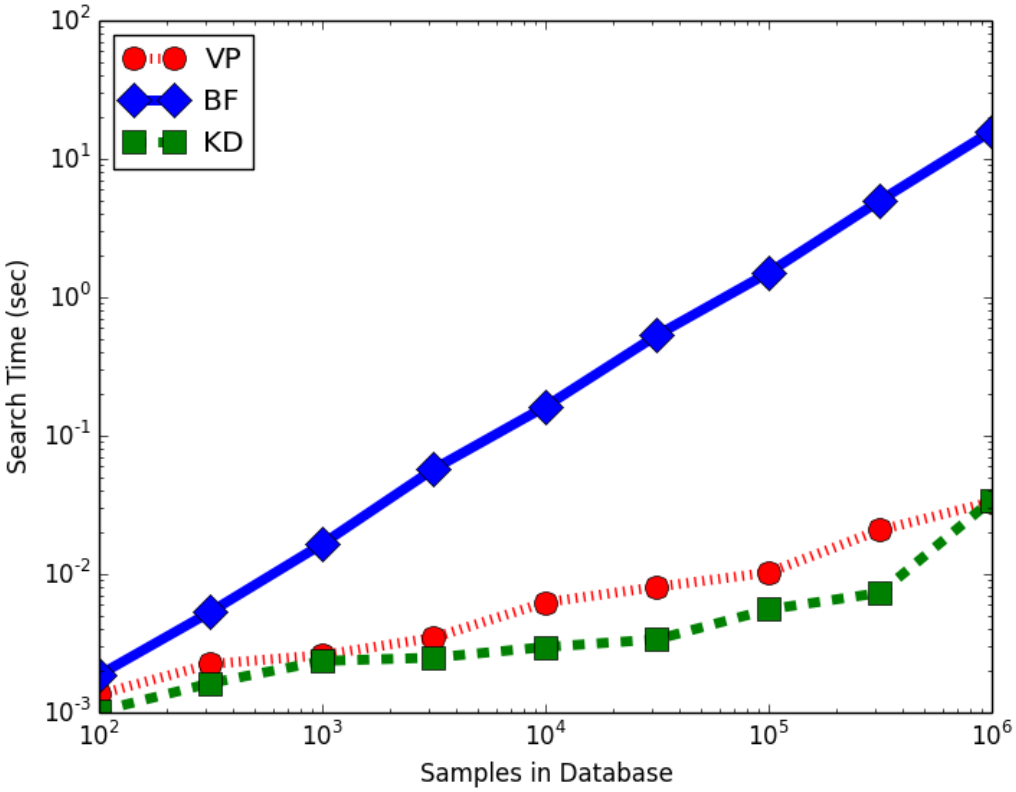

Figure 4-1: Timing comparison of search methods.

Though both of the tree based query algorithms scale similarly with the log of the database size in the average case for low-dimensional data, and their end structures are related, they differ in their construction algorithms and use of distance metrics. KD-Trees use non-leaf nodes to divide the dataset using a hyperplane whose normal vector is equivalent to one of the dimensions of the data [25]. Splits continue recursively until the number of instances in each node is smaller than some threshold. Although both tree methods can vary in branching factor, each was limited to binary splits and unique leaves. Optimizations achieved by changing these factors were not explored in this project.

Both kd and vp-trees are constructed in $\mathcal{O}(n \log n)$ with a linear time median-finding algorithm [13]. In our tests, tree construction took up to 5 minutes for the largest databases, though this must only be performed once. Other trees achieve similar complexity and differ in the use of split heuristics and amount of reinsertion during construction.

4.2 Search in Cancer Genome Atlas Database

We next sought to observe whether the vantage-point tree structures could effectively organize real data with an appropriate dimensionality technique. Principal component analysis is a common technique for matrices which projects the data onto new dimensions to preserve variance. Using a database of gene expression data from the cancer genome atlas comprised of 559 ovarian (OV) and 599 breast cancer (BRCA) samples, we projected the probe-level data onto the first 10 principal components. This reduced the dimensionality from 17,813 genes to 10, and preserved 25% of the variance. To see if this projection retains biological significance, we plotted the first two principal components for the BRCA and OV samples, shown in Figure 4-2.

Clearly, the two cancer types differ in their gene expression patterns and form observable clusters. However, there are two ovarian samples that do not cluster with the rest. One falls within a group of BRCA samples and the other falls outside of either cluster. We then tested GEMINI's nearest-neighbor search using these points as well as two prototypical samples from each cancer type. These were determined by taking the average values for gene-expression for each cancer type.

This resulted in four unique queries (shown circled in Figure 4-2): (A) a prototypical BRCA sample, (B) a prototypical OV sample, (C) a BRCA-like OV sample, and (D) an outlier OV sample. The top 10 hits by similarity to the prototypical OV sample are all OV samples and the top 10 hits for the prototypical BRCA are all BRCA samples as expected (Figure 4-3). The BRCA-like OV sample (59-2349) has 4 BRCA samples and 5 OV samples in the top 9 hits. This result indicates that the BRCA-like OV sample is genomically similar to both types of cancer. The OV outlier,

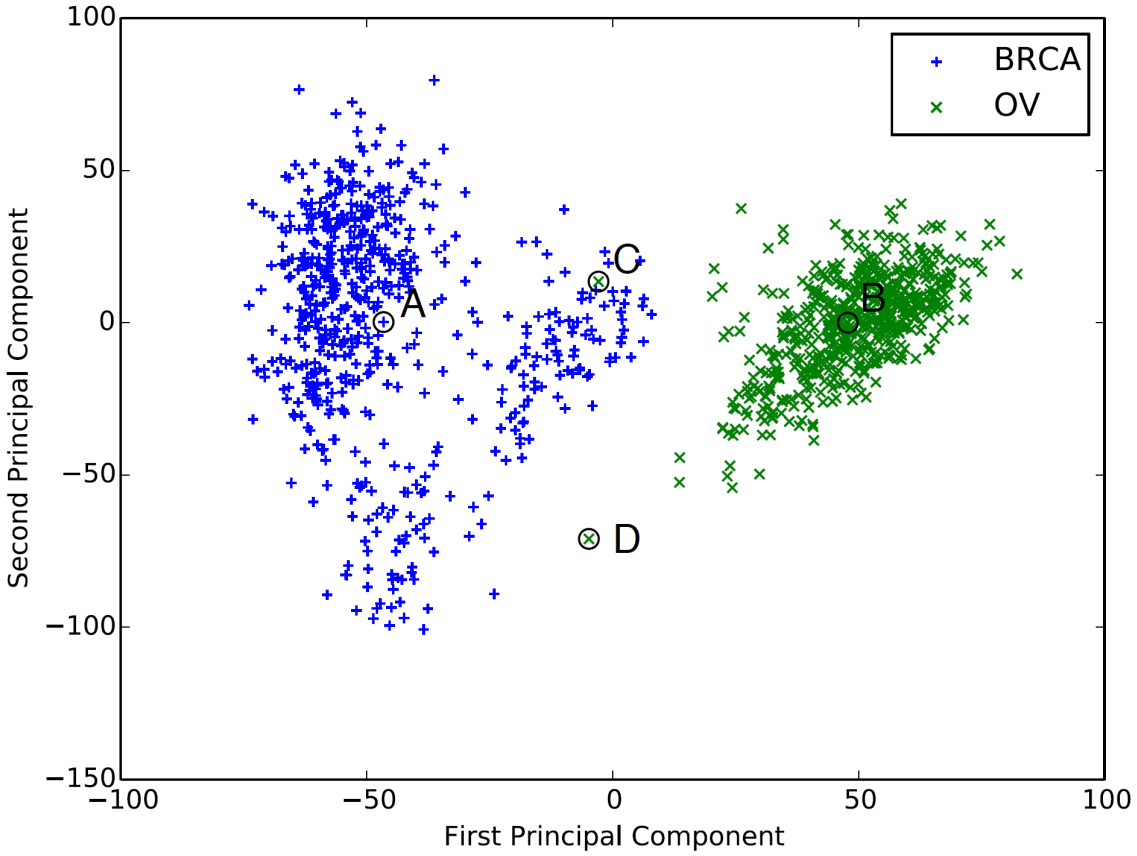

Figure 4-2: Differential gene expression for ovarian and breast cancer samples from TCGA.

surprisingly, shows the most similarity to 9 BRCA samples. This result indicates that though the sample was isolated as an ovarian type cancer, it appear to most resemble breast cancer. Indeed, the genomic similarity between ovarian and breast cancer has been noted, with clear therapeutic implications [42].

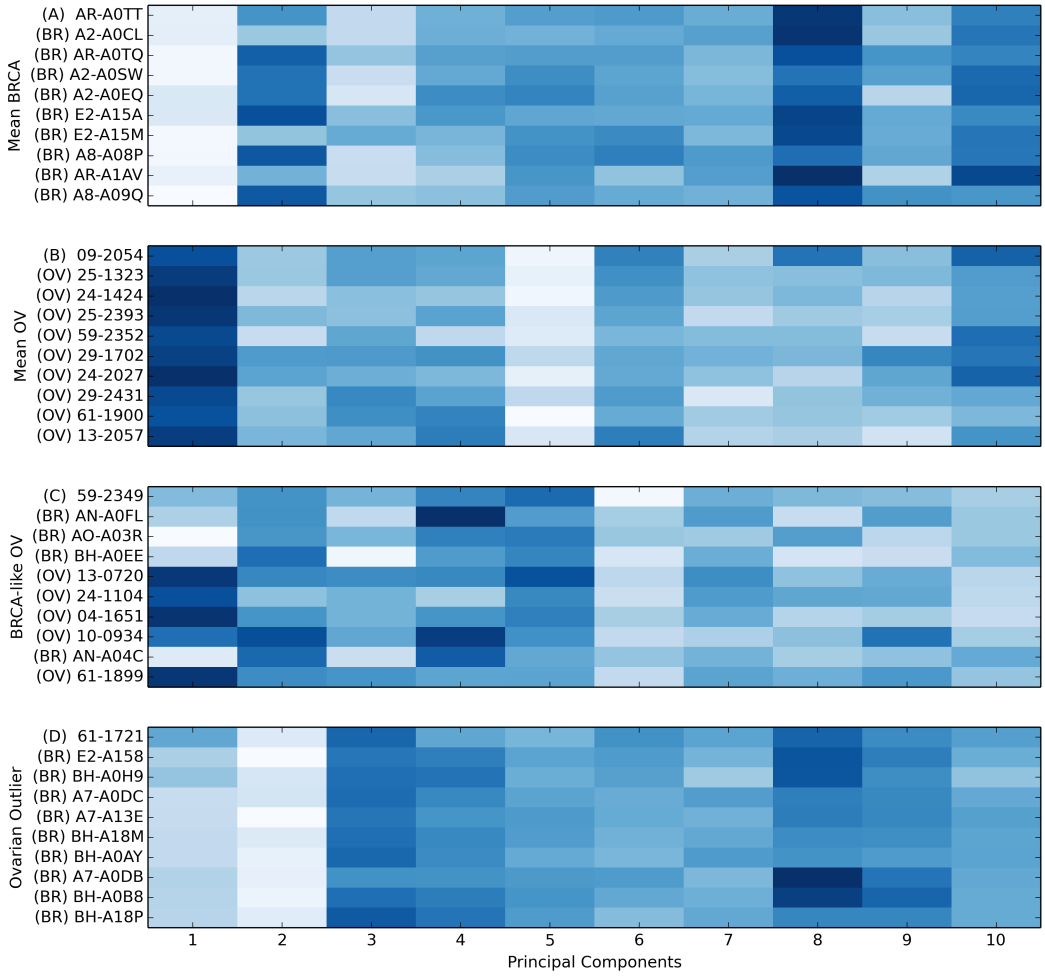

Figure 4-3: GEMINI heat map results showing 9 nearest neighbors to the query (top row) for four samples.

Chapter 5

Discussion

GEMINI forms the basis of an open-source platform for machine learning optimization of search result relevance in genomic data repositories. A clinician may be interested in different types of profiles than a patient or basic researcher, or indeed any other motivated user. By observing the click-through behavior of an individual or group of users, the platform may learn and re-rank results based on individualized probabilistic assessments of relevance. In much the way a user's browser is personalized, so too can GEMINI provide individualized results.

GEMINI's organizational structure shows promise and tremendous flexibility. Because vantage-point tree structures are ignorant of the underlying data source, we can develop distance metrics for arbitrary data, an enormous benefit as there are few data standards for the storage of bioinformatic data. Stochastic algorithms could compare and organize hundreds of existing data sources and allow GEMINI to facilitate research. Methods and techniques for comparison can now be experimented with and used on the platform.

This search engine fits in the context of a large database of profiles that are centrally located as well as with distributed databases. While it may be impossible to store all of the public genomic data in one repository, autonomous software that crawls the web identifying genomic data resources can temporarily store the profile long enough to identify the insertion location in the tree. Only the url of the root source of the data would then be needed in the vp-tree. Then, if the record is

identified as a near-neighbor, the profile can be retrieved on-demand. This approach provides scalability to address the enormous bandwidth needs that would be otherwise necessary to search.

Our capability to generate genomic data is outpacing our capability to analyze and re-use that data. Current genomic data search engines use text-based queries to search for numerical (e.g. gene expression) genomic data profiles. But this paradigm represents a mismatch between the subject and object of the query. A fast, accurate search engine for genomic data may enable researchers to make discoveries using community-collected data more effectively. GEMINI uses a vp-tree to enable us to make effective use of the massive genomic data repositories that we have created. As our experiments demonstrate, even simple assumptions about similarity metrics lead to efficient retrieval and meaningful biological results. With further development of similarity functions for various disease and the aggregation of existing data, GEMINI can facilitate an enormous wealth of biological analyses.

Chapter 6

GEMINI Website

In addition to the command line scripts GEMINI was built as a standalone website [12]. The initial version of the site allows users to upload their own data in HDF5 format and search against available databases, such as breast or ovarian cancer. As mentioned in the system design chapter, the website was built with simplicity in mind, allowing relatively few initial features, but contains flexibility to add more in the future. Below is a screenshot of the current homepage for the site.

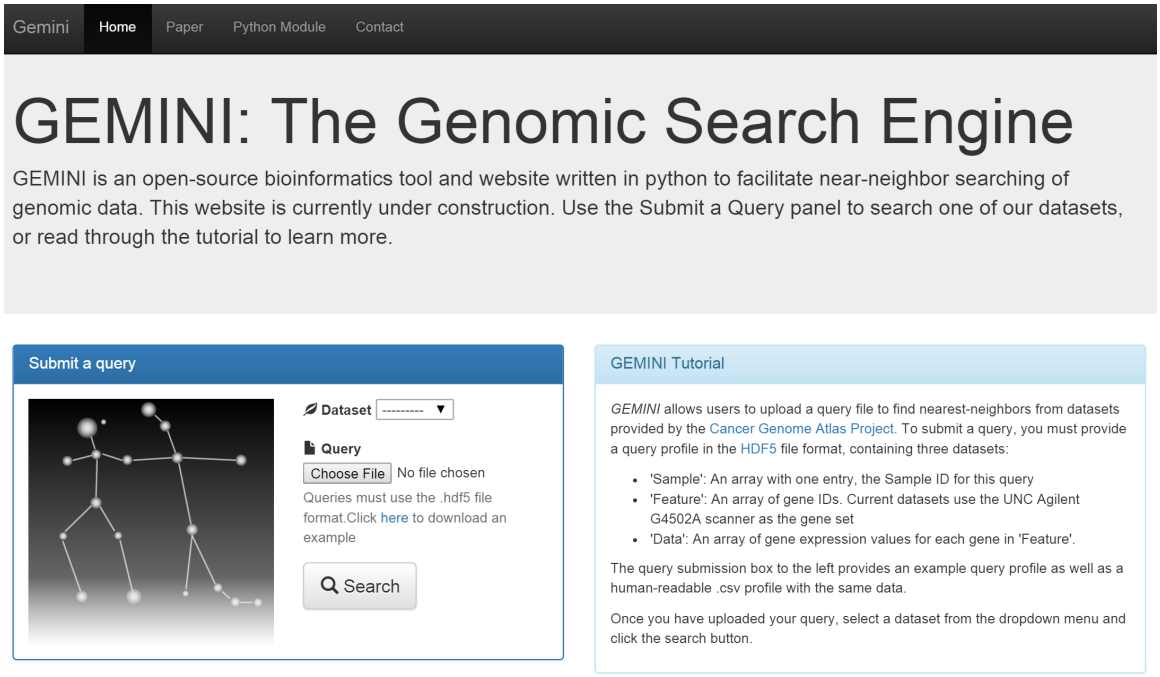

Figure 6-1: GEMINI homepage, with search functionality and instructions

Query Results

Results for *ov_q_0.hdf5* in **OV**

#	Sample	Distance
1	TCGA-59-2349-01A-01R-0711-07	0.000
2	TCGA-24-1551-01A-01R-0504-07	48.628
3	TCGA-04-1361-01A-01R-0456-07	48.986
4	TCGA-61-1899-01A-01R-0584-07	51.115
5	TCGA-61-2097-01A-02R-0670-07	52.516
6	TCGA-24-1104-01A-01R-0436-07	53.124
7	TCGA-24-1470-01A-01R-0504-07	53.534
8	TCGA-29-1707-02A-01R-0810-07	54.153
9	TCGA-04-1651-01A-01R-0584-07	55.841
10	TCGA-04-1516-01A-01R-1050-07	55.919

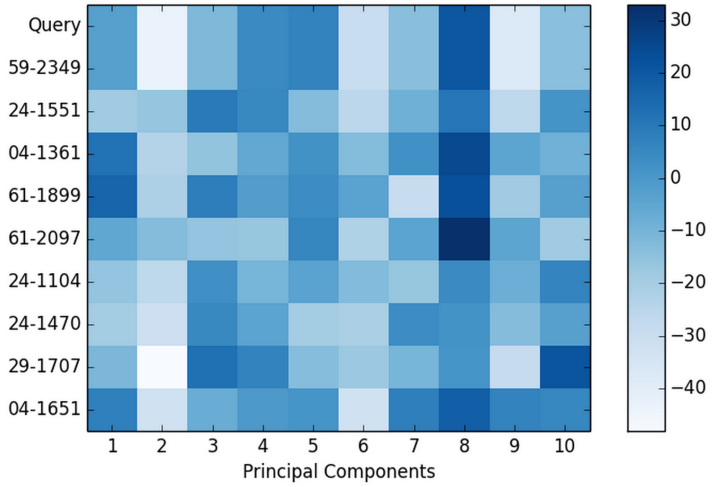

Figure 6-2: GEMINI example results, with the list of most similar samples and heatmap

The Twitter Bootstrap provides CSS classes for the key layout features, including the navigation bar and the organizational panels. The source code for the vp-tree implementation in python can be downloaded from the link provided, and there is also a .zip file containing an example HDF5 query. The user can upload a query into the field provided and select a tree from the dropdown menu. Clicking submit will give the results of their query on a separate page. Some sample results are shown below.

Currently GEMINI results come in two parts. First, the nearest 10 neighbors are returned, along with the distances themselves, in order. Additionally, a heatmap visualization of the similarities to the query appears to the right. Because of the principal component analysis, the dimensions themselves do not have a specific meaning, though more sophisticated reduction techniques could provide their own visualizations.

Chapter 7

Future Work

GEMINI has shown potential to facilitate fast nearest-neighbor searching for genomic profiles, but is far from complete. In addition to search functionality, we hope GEMINI can aggregate and synthesize public research data to provide meaningful analytics. There are a handful of areas for which future research and development would aid the website, and we describe some of the most valuable areas of future research here.

7.1 Data Collection: Web Crawler

Much of the publically available genomics and bioinformatic data is retrieved from internet-hosted repositories, with much of this data requiring no special credentials to access or download. Automated web crawlers have demonstrated the ability to aggregate large datasets such as these without the need for tedious human oversight [18]. Though the test datasets for GEMINI were uploaded manually, a web crawler that scraped large databases automatically and built search indices would expand the usefulness of GEMINI exponentially. However, more research must be done on conversion techniques between filetypes. Each data source has different rules and organization, and associates different resources. A web crawler that is able to parse, translate, and associate meaningful datasets with each other can expand the potential users for GEMINI.

7.2 Data Associations

Collecting and organizing a database of genomic information may be a comparatively minor challenge compared to the task of providing meaningful data associations between data types. Though "Big Data" is often used to refer to enormous data volume, the volume itself is not the major obstacle to diagnostic development. Instead, the challenge is the complexity of data comparisons and learning rules to analyze it. What could GEMINI learn by having both genomic expression data and clinical notes? New research is needed to determine what knowledge can be gained by searching across a variety of data types.

7.3 Visualization

Many of the analyses described here would involve complex comparisons and results, and would require specific skills and training to be able to interpret correctly. GEMINI's current data visualizations are fairly primitive, but developing creative methods for result presentation would greatly increase GEMINI's usability. Modern data visualization tools like d3.js [46] allow sophisticated drawing and presentation techniques, and are already compatible with the web framework we developed. Research to determine the best method for communicating results would result in an extremely helpful tool.

7.4 Search Optimizations

Currently, GEMINI uses only principal components analysis to reduce the dimensionality of the data. However, the vantage-point tree structure is very flexible, and could support a number of different compression techniques and distance functions. For example, performing a linear transformation of known oncogenes could lead to a more sensitive tool for comparing cancer samples. Similarly, machine learning algorithms such as neural networks could learn parameters for different diseases, and a distance function using the neural network could help quickly compare new samples.

This is a new area of research, and it is currently unknown what the best techniques for genomic data dimensionality reduction are.

Bibliography

[1] CDN by MaxCDN | Experts in Content Delivery Network Services.

[2] Tanya Barrett, Dennis B. Troup, Stephen E. Wilhite, Pierre Ledoux, Carlos Evangelista, Irene F. Kim, Maxim Tomashevsky, Kimberly A. Marshall, Katherine H. Phillippy, Patti M. Sherman, Rolf N. Muertter, Michelle Holko, Oluwabukunmi Ayanbule, Andrey Yefanov, and Alexandra Soboleva. NCBI GEO: archive for functional genomics data setsâĂŤ10 years on. *Nucl. Acids Res.*, 39(suppl 1):D1005–D1010, January 2011.

[3] Norbert Beckmann, Hans-Peter Kriegel, Ralf Schneider, and Bernhard Seeger. The r*-tree: An efficient and robust access method for points and rectangles. SIGMOD '90, pages 322–331, New York, NY, USA, 1990. ACM.

[4] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[5] Claude Berge and Edward Minieka. *Graphs and hypergraphs*, volume 7. North-Holland publishing company Amsterdam, 1973.

[6] Adrian P Bird. Cpg-rich islands and the function of dna methylation. *Nature*, 321(6067):209–213, 1985.

[7] Liang Cheng, Roxann M Neumann, Ajay Nehra, Bruce E Spotts, Amy L Weaver, and David G Bostwick. Cancer heterogeneity and its biologic implications in the grading of urothelial carcinoma. *Cancer*, 88(7):1663–1670, 2000.

[8] Peter JA Cock, Tiago Antao, Jeffrey T Chang, Brad A Chapman, Cymon J Cox, Andrew Dalke, Iddo Friedberg, Thomas Hamelryck, Frank Kauff, Bartek Wilczynski, et al. Biopython: freely available python tools for computational molecular biology and bioinformatics. *Bioinformatics*, 25(11):1422–1423, 2009.

[9] The International HapMap 3 Consortium. Integrating common and rare genetic variation in diverse human populations. *Nature*, 467(7311):52–58, September 2010.

[10] Florence Corpet. Multiple sequence alignment with hierarchical clustering. *Nucleic acids research*, 16(22):10881–10890, 1988.

[11] Marina Cretich, Francesco Damin, and Marcella Chiari. Protein microarray technology: how far off is routine diagnostics? *Analyst*, 139(3):528–542, 2014.

[12] Timothy DeFreitas. Gemini, the genomic search egine, 2015.

[13] Timothy DeFreitas and Patrick Flaherty. Gemini: A genomic search engine. *BMC Bioinformatics*, (in review).

[14] Damodar Reddy Edla, Prasanta K Jana, and IEEE Senior Member. A prototype-based modified dbscan for gene clustering. *Procedia Technology*, 6:485–492, 2012.

[15] Mike Folk, Gerd Heber, Quincey Koziol, Elena Pourmal, and Dana Robinson. An overview of the hdf5 technology suite and its applications. In *Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases*, pages 36–47. ACM, 2011.

[16] Django Software Foundation. Django, 2015.

[17] A Marquis Gacy, Geoffrey Goellner, Nenad Juranić, Slobodan Macura, and Cynthia T McMurray. Trinucleotide repeats that expand in human disease form hairpin structures in vitro. *Cell*, 81(4):533–540, 1995.

[18] Allan Heydon and Marc Najork. Mercator: A scalable, extensible web crawler. *World Wide Web*, 2(4):219–229, 1999.

[19] Kentaro Inamura, Takeshi Fujiwara, Yujin Hoshida, Takayuki Isagawa, Michael H Jones, Carl Virtanen, Miyuki Shimane, Yukitoshi Satoh, Sakae Okumura, Ken Nakagawa, et al. Two subclasses of lung squamous cell carcinoma with different gene expression profiles and prognosis identified by hierarchical clustering and non-negative matrix factorization. *Oncogene*, 24(47):7105–7113, 2005.

[20] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613. ACM, 1998.

[21] National Human Genome Research Institute. Cost per genome, 2014.

[22] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[23] Eric Jones, Travis Oliphant, and Pearu Peterson. {SciPy}: Open source scientific tools for {Python}. 2014.

[24] Tapas Kanungo, David M Mount, Nathan S Netanyahu, Christine D Piatko, Ruth Silverman, and Angela Y Wu. An efficient k-means clustering algorithm: Analysis and implementation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(7):881–892, 2002.

[25] Tapas Kanungo, D.M. Mount, N.S. Netanyahu, C.D. Piatko, R. Silverman, and A.Y. Wu. An efficient k-means clustering algorithm: analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):881–892, July 2002.

[26] Norio Katayama and Shinichi Satoh. The SR-tree an index structure for high-dimensional nearest neighbor queries. SIGMOD 97, pages 369–380, New York, NY, USA, 1997. ACM.

[27] Jon Kleinberg and Éva Tardos. *Algorithm design*. Pearson Education India, 2006.

[28] Donald E Knuth. Optimum binary search trees. *Acta Informatica*, 1(1):14–25, March 1971.

[29] Eric S Lander, Lauren M Linton, Bruce Birren, Chad Nusbaum, Michael C Zody, Jennifer Baldwin, Keri Devon, Ken Dewar, Michael Doyle, William FitzHugh, et al. Initial sequencing and analysis of the human genome. *Nature*, 409(6822):860–921, 2001.

[30] Margus Lukk, Misha Kapushesky, Janne NikkilÃď, Helen Parkinson, Angela Goncalves, Wolfgang Huber, Esko Ukkonen, and Alvis Brazma. A global map of human gene expression. *Nat Biotech*, 28(4):322–324, April 2010.

[31] Elaine R Mardis. A decade/'s perspective on dna sequencing technology. *Nature*, 470(7333):198–203, 2011.

[32] Wes McKinney. pandas: a foundational python library for data analysis and statistics. *Python for High Performance and Scientific Computing*, pages 1–9, 2011.

[33] Saul B Needleman and Christian D Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453, 1970.

[34] The Cancer Genome Atlas Network. Comprehensive molecular portraits of human breast tumours. *Nature*, 490(7418):61–70, October 2012.

[35] F. Nielsen, P. Piro, and M. Barlaud. Bregman vantage point trees for efficient nearest neighbor queries. pages 878–881, June 2009.

[36] Marshall Nirenberg, T Caskey, R Marshall, R Brimacombe, D Kellogg, B Doctor, D Hatfield, J Levin, F Rottman, S Pestka, et al. The rna code and protein synthesis. In *Cold Spring Harbor Symposia on Quantitative Biology*, volume 31, pages 11–24. Cold Spring Harbor Laboratory Press, 1966.

[37] Travis E Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

[38] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: Bringing order to the web., November 1999.

[39] Daniel Pinkel, Richard Segraves, Damir Sudar, Steven Clark, Ian Poole, David Kowbel, Colin Collins, Wen-Lin Kuo, Chira Chen, Ye Zhai, et al. High resolution analysis of dna copy number variation using comparative genomic hybridization to microarrays. *Nature genetics*, 20(2):207–211, 1998.

[40] Victor L Roggli, Robin T Vollmer, S Donald Greenberg, Malcolm H McGavran, Harlan J Spjut, and Raymond Yesner. Lung cancer heterogeneity: a blinded and randomized study of 100 consecutive cases. *Human pathology*, 16(6):569–579, 1985.

[41] Robert R Schaller. Moore's law: past, present and future. *Spectrum, IEEE*, 34(6):52–59, 1997.

[42] Dennis J Slamon, William Godolphin, Lovell A Jones, John A Holt, Steven G Wong, Duane E Keith, Wendy J Levin, Susan G Stuart, Judy Udove, Axel Ullrich, et al. Studies of the her-2/neu proto-oncogene in human breast and ovarian cancer. *Science*, 244(4905):707–712, 1989.

[43] United States Food & Drug Administration. *E2B Data Elements for Transmission of Individual Case Safety Reports*, 3 2005.

[44] J Craig Venter, Mark D Adams, Eugene W Myers, Peter W Li, Richard J Mural, Granger G Sutton, Hamilton O Smith, Mark Yandell, Cheryl A Evans, Robert A Holt, et al. The sequence of the human genome. *science*, 291(5507):1304–1351, 2001.

[45] Peter N. Yianilos. Data structures and algorithms for nearest neighbor search in general metric spaces. SODA '93, pages 311–321, Philadelphia, PA, USA, 1993. Society for Industrial and Applied Mathematics.

[46] Nick Qi Zhu. *Data Visualization with D3. js Cookbook*. Packt Publishing Ltd, 2013.