GVis: A Scalable Visualization Framework for Genomic Data

Jin Hong[1], Dong Hyun Jeong[2], Chris D Shaw[1], William Ribarsky[3], Mark Borodovsky[1], and Chang Song[2]
[1]Georgia Institute of Technology, [2]Hallym University, and [3]UNC Charlotte

Abstract

This paper describes a framework we have developed for the visual analysis of large-scale phylogeny hierarchies populated with the genomic data of various organisms. This framework allows the user to quickly browse the phylogeny hierarchy of organisms from the highest level down to the level of an individual genome for the desired organism of interest. Based on this framework, the user can initiate gene-finding and gene-matching analyses and view the resulting annotated coding potential graphs in the same multi-scale visualization framework, permitting correlative analysis and further investigation. This paper introduces our framework and describes the data structures and algorithms that support it.

1 INTRODUCTION

The DNA sequencing revolution initiated the exponential growth of DNA and protein sequence data that are changing the paradigm of biological research. Thus, Biology has entered the realm of Big Data, where large-scale investments in technology, machinery, and infrastructure are needed to generate new scientific knowledge. The new challenge is to build tools that enable interactive analysis, exploration of complex data, and data-driven discovery.

Currently, the process of annotating a genome (finding genes and estimating their function) in genomic sequences is increasingly labor intensive. Generally, about one half of a genome can be annotated automatically, one half of what is left requires twice as much effort. If functions of genes have to be identified, one half of what is left takes again twice as much time, and so on. Gene function prediction involves multiple steps, access to multiple tools, parsing and reconciling the outputs and inferences of relationships between genetic data using multiple pieces of evidence.

In this paper we present a new framework for highly interactive visual analysis of comprehensive genomic data that supports the above analysis process and can be scaled to any number of genomes. The framework uses a zoomable, multiresolution approach that can handle genomes of any size from millions of nucleotides (eukaryote) to a few thousand (viral). Structures on these widely different scales can be efficiently accessed and visualized together. The navigable visual environment provides overviews and detailed views in a rich, dynamic context. Gene analysis tools, such as BLAST or GeneMark [Bor93] (which finds likely genes), can be launched from within the framework and can efficiently access the data stored there. The results from such analyses are immediately available for visualization and comparative study. This visualization framework significantly changes the genome analysis process by speeding it up greatly, removing tedious steps, and displaying unforeseen avenues of analysis.

This framework permits a scientist to interactively explore, browse, compare and analyze many types of biological information. It helps researchers identify new information within DNA sequences about genes and encoded proteins and to establish relationships between pieces of genetic information across species. Our approach is to develop data structures and systems to create a spatial embedding for genomic data sets that can be accessed by a visualization procedure that employs level-of-detail techniques to rapidly display the data that the user is interested in. Its ability to combine visual exploration with efficient access to comprehensive data and embedded analysis tools is novel.

The framework, called *GVis*, has the following properties, which will be described further in Sections 3 through 7:

- A multiscale structure based on the Pad++ "infinite pan and zoom" paradigm,
- A general tree structure capable of providing quick access to many thousands of genomes of any size,
- A layout scheme for arranging genomic information, including annotations, at different scales that supports iterative analysis and comparison,
- Level of detail techniques to continuously reveal increasing amounts of detail as one zooms in,
- Multiple interaction techniques for presenting "details on demand" through direct interaction with the visual presentation,
- Integrated analysis tools that can be launched within GVis and whose results can then be compared.

We have first applied GVis to a database of 1330 complete virus, archaea, bacterial, and eukaryote genomes. The viral portion of this database was recently reannotated by Borodovsky and colleagues and is a valuable resource for investigating viral function and behavior. Since then we have added several thousand additional genomes to the database.

2 Prior Art

Given the wealth of data generated by the Human Genome project for human and other species, there is a well-recognized need [Ste02] for powerful integration & visualization tools to explore massive genomic data, structures and gene networks.

A number of visualization tools have been created to allow users to view genome-sized data in an integrative manner. These have been constructed such that the information can be scaled to the level of interest. One such tool is Ensembl, www.ensembl.org, which has a "Mapquest-like" interface (i.e., discrete jumps between viewpoints rather than the continuous method we employ), and contains a wealth of information about a particular genome down to the nucleotide sequence level. Another visualization program is GenDB [Goe03], which makes use of static databases to produce a dynamic interface, which the user can peruse. However, neither these nor other tools make use of highly interactive exploration (in particular continuous zooming and panning through multiple levels of detail), real time analyses and comparative genome analysis. Instead, they merely display data that has already been analyzed and compiled into a database for future viewing. In addition, while many such tools are made for use with large-scale data, their actual visualization is primitive and not easily scalable, which often causes navigation through the data to be slow and views narrow.

In the visualization community, tools for comprehensively handling large scale genomic data are rare. Chi et. al. [Chi95] developed a graphical representation for multivariate sequence similarity search results. A 3D viewer compared sequences obtained via BLAST, including frame numbers and frame shifts for protein encoding. Several results can be compared in this viewer, but there is no possibility of showing contextual data; only the results from the BLAST hits are displayed.

There has also been some preliminary work on genome function analysis in an immersive virtual environment [Kan02]. Here a CAVE-like environment with 5 screen walls is used for pair-wise comparison of cluster sets from different gene expression datasets. These comparisons are visualized as a set of 3D histograms and overlapping clusters that surround the viewer.

There has been work on genome visualization using a desktop "virtual environment" [Ada02]. This study uses a desktop environment similar to the one used here and has the capability to zoom in from overviews of chromosomes to individual genes and their associated proteins. The study supports our premise that a multiresolution genome visualization is useful and can be supported on off-the-shelf desktop computers (with 3D graphics cards), but it doesn't have the complete scalable hierarchy and large range of scales and resolutions supported in our framework. Finally, the challenges for interactive visualization posed by very large amounts of data and tasks such as comparing different genomes, studying variations between individuals, interpreting protein expression data, and other tasks has been emphasized [Tur01]. The authors promote interdisciplinary collaborations and level of detail approaches, both of which are part of the framework presented here.

Our framework is based on the "Pad" [Per93] metaphor and its extension, Pad++ [Bed94, Fur95]. In this metaphor, the information space is considered as an infinite 2D plane, which can be stretched by orders of magnitude at any point to investigate details. This is an important capability because we have found, in other highly scalable interfaces [War99], that one often discovers new things to investigate while navigating somewhere else. Pad++ has been mostly explored as a highly interactive, zoomable alternative to windows and icons interfaces and in applications such as navigable Web interfaces.

The Pad++ software has been used for genomic visualization [Lor02]. Here the zoomable capability is put to good use as a browser to investigate human genome data at different scales showing overall gene structure for some part of the genome and then gene marking, protein, sequence, and other annotations at different scales. Our approach has all of these capabilities but generalizes to multiple genomes, comparative analysis, and support for very large databases.

3 Enhanced Gene Finding and Analysis Process

We are applying a human-centered design process for the interactive visualization aspects of GVis [Dix98,Nor02]. In brief, the process we have followed is to iterate over the phases of Task Analysis, Design, and Rapid Prototping. Part of our findings from Task Analysis (developed in discussions with bioinformaticists) is that gene-finding is a multi-step process fraught with delays and that overviews plus fast access to genomes both in the vicinity of a target genome and at other locations in the database is quite useful. In addition, launching analyses and seeing their results from within the visualization are quite important. We have therefore focused on these aspects in the development of the GVis framework.

The task analysis indicates a work flow shown in Figure 1. Here a researcher zooms in from the universe of genomes stored in the system (level 1) to a particular genome (level 2), to a gene neighborhood (level 3), to an individual gene and its structure (level 4) to its nucleotide sequence (level 5). In following sections, we explain how this rapid traversal in scale and context is accomplished. For example, the actual nucleotide sequence of a stop codon can only be checked by looking at level 5, while the presence of many possible start codons (marked by icons) may only be visible at level 4. Level 3 may reveal an unusually high G+C content that would otherwise be hard to detect at more detailed levels. Level 2 allows viewing of a whole geneome's structure, and level 1 reveals possible similar genetic structure with related nearby organisms.

Figure 1 shows protein sequence information in the center column and domain architecture in the right column. For simplicity, we have shown these as being at finer levels of scale. Although the user may initially follow a particular path through the genome universe, there will then typically be much branching, jumping across the genome space, or returning to higher levels to follow different paths. Detailed and often complex analyses occur at levels 4 and 5, which can result in recursive links and jumps to other levels. This will often be the result of analyses carried out as the user follows the path. In particular, tools such as BLAST establish connections to other genomes (often not in the neighborhood of the original genome). It is a main goal of

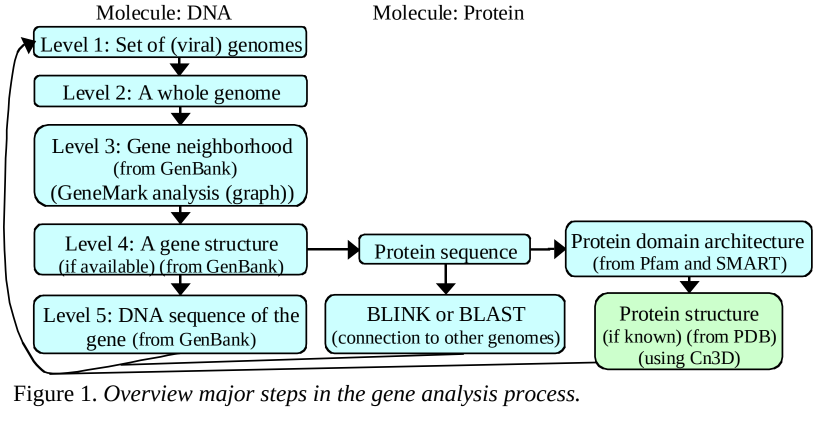

Molecule: DNA
Molecule: Protein
Level 1: Set of (viral) genomes
Level 2: A whole genome
Level 3: Gene neighborhood (from GenBank) (GeneMark analysis (graph))
Level 4: A gene structure (if available) (from GenBank)
Level 5: DNA sequence of the gene (from GenBank)
Protein sequence
BLINK or BLAST (connection to other genomes)
Protein domain architecture (from Pfam and SMART)
Protein structure (if known) (from PDB) (using Cn3D)

Figure 1. *Overview major steps in the gene analysis process.*

our GVis framework to effectively support this gene-finding, analysis process, and resulting correlative analysis among genomes. The framework we present in this paper so far supports levels 1, 2, 3 and 5 of the genetic analysis represented in the left column of Figure 1, plus some analyses represented in the middle column.

4 Multiscale Approach

To support this general process, we require an interactive approach that encompasses both space and scale. We consider a "2D + scale" space that has two spatial dimensions and a scale dimension. The basic pan and zoom navigation modes are easy to use and understand, yet the 2D + scale space is able to handle very much more data than a non-scalable 2D visualization. In addition the reduction to two spatial dimensions makes existing 2D layouts straightforward to use and dynamic models for clustering and general layout control simple.

In our vision, the GVis framework allows the scientist to freely interact with data describing the genetic structure of any known organism, and trace the relationships between similar sequences, performing analysis and comparison at key points. As one zooms in on a particular region of the data, for example, the system brings forth and displays new detail for that region. Such a model is especially useful for exploration and discovery in large data, which has multiple scales and many different features.

The exploratory visualization encompasses two types of scalability: scalability in the number and diversity of genomes, and scalability in being able to navigate freely from an overview to the level of detail (LOD) of the individual nucleotide. Interactivity must be maintained for both types of scalability and when comparing and analyzing genomes, which is the main use for our methods. We are applying multiscale, multiresolution methods to maintain interactivity while revealing necessary detail at each level. Our methods retain contextual information as one navigates the genomic space. Contextual information, including overviews so that one knows where one is in the overall genomic space and detailed views that show relevant information in the vicinity of one's navigation path, has been shown to be quite important in exploratory analysis, where one does not always know in advance what one is looking for [Fau00]. Most analyses of large and complex data have an exploratory component. This principle has been encapsulated in the dictum, 'Focus + Context' from information visualization [Pir01].

For the gene finding task, maintaining scalability, multiresolution data, and interactivity in a visual interface requires a special type of data organization. We consider this data organization next and discuss how it can be coupled with existing genomic data structures. An important aspect of this data organization is a *universal model* that permits multiple data servers and users who have individual copies. Because individual users can build their own parts of the universal data model, it also supports a new form of annotation that captures the process of exploration and analysis, rather than just individual notes.

5 Managing Big and Complex Data

Since the total size of the GVis gene database will grow significantly as users start acquiring more data of interest to view, we have developed data management techniques that apply a level of abstraction approach across the genomes in the database. Clearly, there is no point in drawing millions of nucleotides per genome when we are viewing all of them at the broadest level of overview, since each nucleotide would project to a small fraction of a pixel.

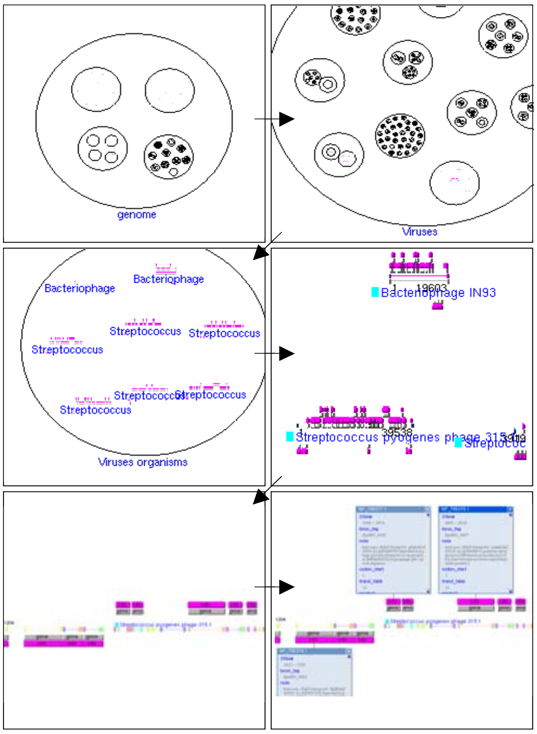


Figure 2. *Zoom in operation from root (all genomes) to a specific genome (Human herpesvirus 7).*

For efficient navigation and exploration, data should be stored in a spatial hierarchy for incremental access and multiresolution display. The Pad authors [Per93] mention that the 2D + scale layout is geographical, but they do not fully capitalize on this fact. In this work we have used the geographical layout in a specific way. We lay out this structure in such a way that it satisfies, for the universal genome knowledge organization described below, the criterion that the contents of children be totally contained spatially in their parents and not overlap each other. This makes the domain-specific structure especially efficient to traverse and query spatially. The domain-specific structure encompasses both the 2D spatial extent and the scale dimension (through use of tree depth). It lends itself to interactive navigation of the scalable data space, with incremental updates from out-of-core data storage.

5.1 Taxonomy Layout and Data Structure

We want to build a structure that is based on a universal model containing comprehensive domain knowledge appropriate for genomic investigations. A universal structure is quite important because it can be used everywhere, on servers or by individual users collecting and annotating data, and is known apriori by everyone. This means that everyone knows where to put a new genomic annotation or find an existing piece of data. A universal structure means that genomic data can be highly distributed

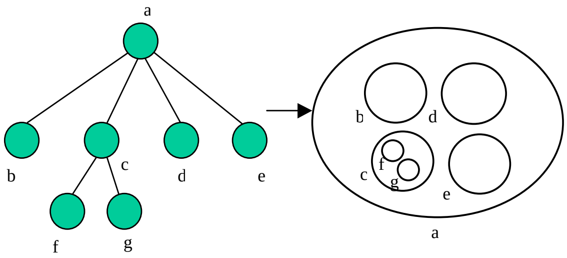


Figure 3: *Translation from tree structure to Venn diagram*

and easily shared. Everybody knows the structure but nobody possesses all the data. However, parts of the structure can easily be compared between users or servers to coordinate their contents. We chose the comprehensive phylogeny tree for all organisms as the universal structure. This tree has a place for everything in its branches, including archaea, bacteria, viruses, and eukaryotes. The phylogeny tree is widely recognized by users. It tends to group genomes with similar characteristics in size and functional structures on nearby branches. The phylogeny classification structure embodies how biologists tend to think about organisms. Nevertheless, it is also true that no structure, even if comprehensive, can encompass all the relationships that genomic scientists discover and investigate. Indeed, it is increasingly true that scientist compare quite disparate species (e.g., viruses and bacteria) to discover common mechanisms. Finally, it is important to note that the data layout and organization mechanism does not depend on the details of the structure used. If a universal structure other than a phylogeny tree were deemed more useful, the data could be efficiently organized for this structure.

The embodiment of the phylogeny structure is a taxonomy tree based on the taxonomy and directory structure defined in NCBI. Each directory is considered as a node in the tree, and each genome is considered as a leaf node. (A tree structure appropriate for navigating within a genome is then attached to these leaf nodes.) The storage scheme on disk simply uses the file system tree to create categories and subcategories, so that data for new genomes can be added with little effort.

In GVis, each node is represented by an ellipse and each leaf node is represented by a genome structure as shown in Figure 2. GVis adopts a containment relationship between parent and children nodes, which is similar to a Venn Diagram. Figure 3 illustrates the conventional node link tree structure in this representation. The location and size of child nodes inside the parent node depends on the number of children nodes, with no child nodes overlapping. First, locations are assigned randomly around the origin and then each of the locations is considered as a particle that pushes others nearby. To prevent particles from moving too far away, we add a counter-force that attract particles towards the origin (and keeps them within the parent node). These two complementary forces move particles in equilibrium positions eventually, and we then store them.

When GVis loads the tree structure, depending on the number of children, it reads the location table and assigns this spatial coordinate into the node. In this way, it accelerates the loading process because the system does not need to compute the location of the children and users can freely add or remove genomes from a directory without worrying about changing the spatial information.

Because of the containment relationships between parent and child nodes, the tree can be efficiently traversed. When the root node is put into the visual query, GVis will decide if the root node needs to be drawn by checking whether the view window contains or intersects it. If so, GVis traverses to the next level of the tree and applies the same test recursively. At some level, the size of node on the screen will be too small compared to a screen-space threshold, and this is the point when the traversal stops. In this way, the tree can be traversed very quickly.

5.2 Multiscale Objects in Scale-Space

Interactive visualization within the scale-space depends on two types of multiscale methods. One, already discussed above, is semantic zooming [Per93], which is purely multiscale and the other is view-dependent geometry and image simplification, which is both multiscale and multiresolution. The central idea in semantic zooming is to allow the user to examine areas of the data space in greater detail by smoothly allocating more and more screen area to it. While this is happening, data outside the current area of interest are shrunk and represented more and more abstractly. View-dependent simplification involves continuous reduction (or increase) of geometric and image detail based on screen space errors that are updated as the viewpoint changes. We invoke semantic zooming when we wish to relate levels of abstraction without regard to a detailed relation between geometry or appearance at different levels. We invoke simplification when it is effective to simplify the detailed geometric representation. For example in our system, the various views of gene sequence structure in Figure 2 are connected by semantic zooming, while the similarity graphs in Figure 10 below use view-dependent multiresolution simplification.

6 Overview Windows

One of the drawbacks of a zooming interface is that it is easy to get lost as we navigate, especially with a rich information space with a wide range of scales. To resolve this problem, GVis provides two overview windows. One contains the tree structure by representing the hierarchy as a circular node link tree as shown in Figure 4, and the other overview uses the genome structure in the main window (Figure 2). Thus changes of view on the main window are displayed in this second overview. For large genomes, more overviews could be constructed.

Two forms of the first overview are provided. In the circular node link overview, the child nodes of the root are distributed evenly around the root node. To prevent overlapping between node c and d in Figure 5, the length of the link between parents (a and b) and children nodes (c and d) are defined as less than the half of the distance between parent nodes (a and b). The remaining locations of child nodes can be obtained recursively and if the length of link between the nodes is very small, the algorithm stops drawing the following children and stops traversing the tree. By applying this algorithm, the upper two pictures in Figure 4 can be generated, which correspond to zoom-in operation as shown in Figure 2. As the current node in the main window changes while zooming in, the path of the

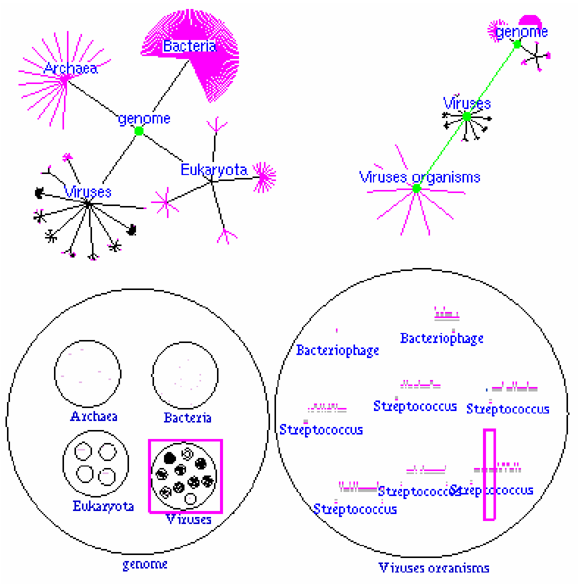


Figure 4. *Upper Left: Node link tree of our genome database containing 500Mbytes of data and 1500 genomes. Lower Left: Corresponding and overview image of zoomable phylogeny tree. Upper Right: Node link tree with a virus highlighted in green Lower Right: Corresponding zoomed-in view of unclassified bacteriophages. The pink rectangle represents the view being displayed in the main window.*

traversed node is shown. The pink links indicate leaf nodes, representing genomes.

The circular node link tree is good for providing the path information of traversed nodes, but it is a completely different representation from that of main window. GVis therefore provides another overview window, which contains the same representation as that of the main window. Instead of actually zooming in, this second overview displays the viewed region of the main window, which shows where the user is located in the taxonomy hierarchy. In the lower two images in Figure 4, the pink rectangle denotes the main window's view area. (See the color plate.) When zooming and panning, the size and location of this rectangle change. To avoid shrinking the pink rectangle to invisibility, GVis also zooms in (or out) the overview by a certain amount when the pink rectangle size is too small (or too large) for the current overview. The lower right picture in Figure 4 corresponds to the last one in Figure 2. By using Figure 4, the user can clearly see what the path is of the genome in the hierarchy and where she is in the information space. This structure provides a contextual and understandable set of views that go all the way from a general overview of all genomes to individual nucleotides.

GVis handles navigation between widely separated genomes by providing interaction between main and overview windows. For example, in the circular tree in Figure 4, users can click any node, and this action will cause the main window to jump directly into that information space or genome. In addition, for overview 2 in Figure 4, users can drag the desired viewing area on the overview window, and the main window will pan and zoom to that desired view. Through these interactions, users do not need to zoom-in, zoom-out, pan, and zoom-in again to go to different information spaces.

6.1 GVis in Action

The following example shows the highly interactive data navigation possible with GVis. The user starts off with an image of the collection of genomes in the database Our initial test database has some thousands of genomes and multiple Gigabytes of data. Note that because of the hierarchical structure and the multiscale views, described further next, the database can be scaled to any size. For any size database, the only data that need be rendered from active memory are those for a particular view and scale. These particular data can be quite efficiently retrieved from the multiscale spatial layout using a mechanism similar to one used for large scale geospatial data [Dav99].

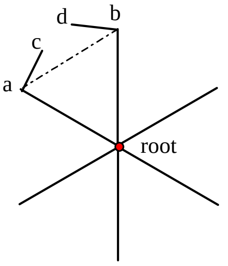


Figure 5. *Node link circular tree*

This is represented as a hierarchical layout of clusters of genomes, divided by taxonomy. As shown in the top left image of Figure 2, a large oval encapsulates the different classes of organisms; Archaea, Bacteria, Eukaryotes, and viruses. From here, the user can immediately zoom in interactively to the class, family, sub-family, or genus for the organism of interest. As the user zooms in, more and more semantic detail is revealed, starting with the segment of DNA that holds a gene, the gene name, its nucleotide index number, and so on.

The image sequence from the top left to lower right in Figure 2 shows screenshots of our prototype viewer running on a 2.4GHz Pentium 4 computer with an nVidia GeForce 4 graphics card. Each succeeding image was taken at 500ms intervals over the 3 seconds it took to zoom in from the overview to the single virus genome in the lower left small image. Note, because the zoom is logarithmic in scale [Bed94], even a database containing all known genomes could be navigated in seconds (presuming that one had fast access to the scalable data structure in Sec. 5).

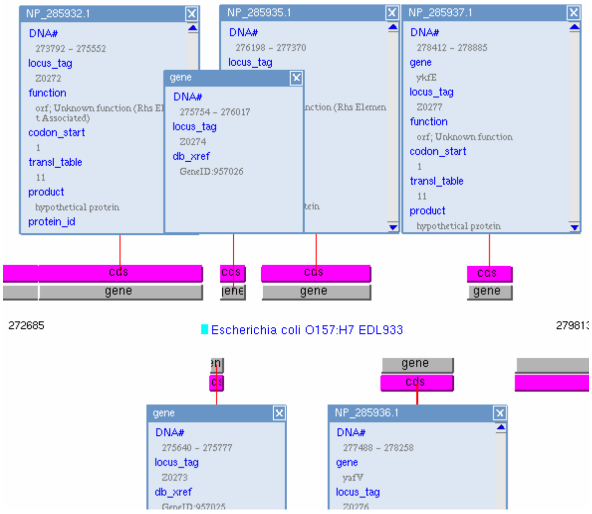
For the user, the effect of this interface is that one is smoothly flying in, as in an overhead view of a landscape, and increased detail is unfolding. (The reverse effect occurs as one flies out.) The entire *E.coli* genome can be navigated in a few seconds, permitting one to investigate any level of detail at any location in the genome. The entire genome appears to be a coherent and accessible whole that can be navigated in context. This is contrasted with the usual Web query structure, where each query produces a result after a (sometimes lengthy) pause and the genome appears to be composed of a set of unconnected images.

6.2 DNA Strand Spatial Organization

The fundamental unit of spatial layout for our visualization is the horizontal DNA strand. This is a well-understood spatial organization, appearing in gene viewing websites, such as Ensembl, the sequence alignment output of BLAST, and in many other displays of genomic data. GVis allows immediate navigation of the DNA strand to the left or right by simply dragging the displayed scene left or right with the mouse. For simple display of genomes, this organization would have the disadvantage of requiring horizontal motion to see more data while the rest of the screen lay unoccupied. However, in GVis the screen space above and below the DNA strand is used for the unified and scaled display of multiple annotations of the DNA strand.

Currently, each individual gene (Figure 6) is represented by a colored box (or button) that is simply an icon for the underlying sequence. Each color represents a different type of data. For example, a gray button is for *gene* and a magenta button is for *CDS* (coding sequence). Clicking on a box (Gene *Button*) will immediately pop up a little window with the text record for the item, including the name, the sequence start and end, the gene's function, the protein it creates, and so on. Users can pan the text inside any pop-up window by dragging the text with the mouse. The buttons above the organism name represent the positive DNA coding strand and those below the organism name are for negative strand. The annotation windows align themselves dynamically in the 2D space, near the section of the genome they represent, so that overlap is minimized as several boxes are opened.

We apply several guiding design principles. First, we locate related items as closely as possible to one another. Second, we seek to minimize direct screen space rearrangement by the user (mainly done to make occluded items visible) by dynamic rearrangement and by dynamic scaling (items in a collection next to the DNA sequence can be brought forward individually while others fall back to minimize overlap). All this permits the user to quickly get the maximum amount of information while not losing context. Third, we make a mouse selection of a feature in one type of display (sequence, say) result in the other related analyses or objects being highlighted in a brushing and linking interaction.



NP_285932.1

DNA#
273792 – 275552
locus_tag
Z0272
function
orf; Unknown function (Rhs El
t Associated)
codon_start
1
transl_table
11
product
hypothetical protein
protein_id

NP_285935.1

DNA#
276198 – 277370
locus_tag
nction (Rhs Elemen
tein

gene

DNA#
275754 – 276017
locus_tag
Z0274
db_xref
GeneID:957026

NP_285937.1

DNA#
278412 – 278885
gene
ykfE
locus_tag
Z0277
function
orf; Unknown function
codon_start
1
transl_table
11
product
hypothetical protein

gene

DNA#
275640 – 275777
locus_tag
Z0273
db_xref
GeneID:957025

NP_285936.1

DNA#
277488 – 278258
gene
yafV
locus_tag
Z0276

Figure 6. *Representation of a small segment of the E. Coli genome.*

6.3 Representing Gene Annotations

GVis uses information on GenBank files to visualize genomes. Based on gene annotations by NCBI (National Center for Biotechnology Information, http://www.ncbi.nlm.nih.gov). The following is a small fragment of the GenBank file for *E.coli.*

CDS 2046677..2048227
/gene="ansP"
/function="transport; Transport of small molecules: Amino acids, amines"
/note="Residues 1 to 516 of 516 are 100.00 pct identical to residues 1 to 516 of 516 from Escherichia coli K-12
Strain MG1655: B1453"
/codon_start=1

This fragment of text refers to one of the coding sequences in *E.coli.* All other genes or misc_features, etc. are represented in this way. To represent this text file visually, the key words such as *CDS*, *gene*, etc. are represented by the gene buttons shown across the horizontal centerline in Figure 6. Other genomes (e.g., viruses or eukaryotes) are represented in a similar way.

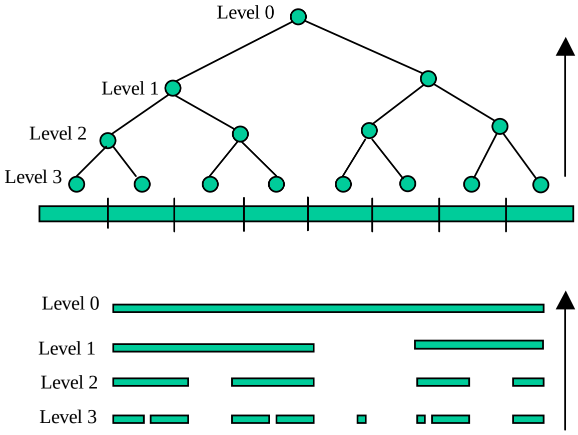


Figure 7. *Building a binary tree from bottom to top for a DNA sequence. As we go up the tree, gene buttons that are close merge into one box.*

6.4 Data Management for One DNA Sequence

Currently, the DNA sequences in our database range in size from 50Kbytes to 40Mbytes, and DNA sequences range in lengths from a few thousand to 20 million. Loading all of this data at once will exceed the memory size of many desktop machines and will not admit scalability. We therefore manage the data in a way that allows GVis to load and display only the data being viewed at the moment.

To manage data for a single DNA sequence, we have built a data structure that manages the spatial detail according to the projected size of the information to be displayed. This data structure provides semantic zooming capabilities, which uses levels of abstraction rather than levels of detail, the goal being to transmit as much meaning as possible at each scale. Thus upon zooming in, the user sees not only more detail but also more types of information coordinated in a meaningful way.

The data for each DNA sequence is preprocessed into a binary tree structure, and each level of this tree has an aggregate representation of the contents in its child nodes. Figure 7 illustrates this multi-resolution genome tree. The

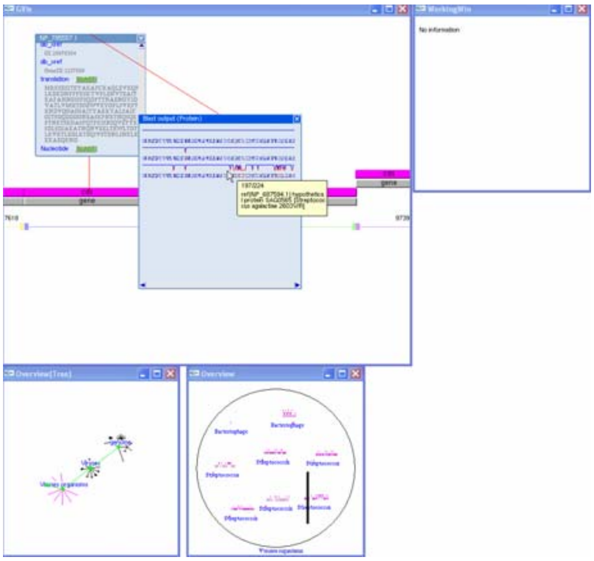


Figure 8. *Selection from genome and resulting BLAST output window.*

domain of the tree is the DNA sequence; with each node in a tree representing a range of DNA base pairs.

At the leaf node level (Level 3 in Figure 7), each node represents a contiguous range of 64K basepairs. Each node also contains a list of visually salient items, so leaf nodes thus contain the highest level of detail. Thus, each leaf node has all the information of each gene in its range, such as start and stop sequence, direction, name, protein coding sequence, etc.

When two nodes are combined at Level 2, those buttons (genes, CDS, RNA, etc) that are close together are clustered and are represented as one button, and those smaller than one pixel will disappear. This structure is intended to support a viewing program in which at least 1 or 2 nodes are visible at one of the levels in the tree. By comparing the view window size with the genome sequence length inside the view window, the level and node are determined and this is when the information of the node to be displayed is retrieved from disk. By representing a genome with several levels of abstraction, it can be explored efficiently, and by retrieving data only when it is needed, the system can lessen the memory load when handling large amount of data. This structure is entirely scalable to collections of genomes of any size with individual genomes of any length.

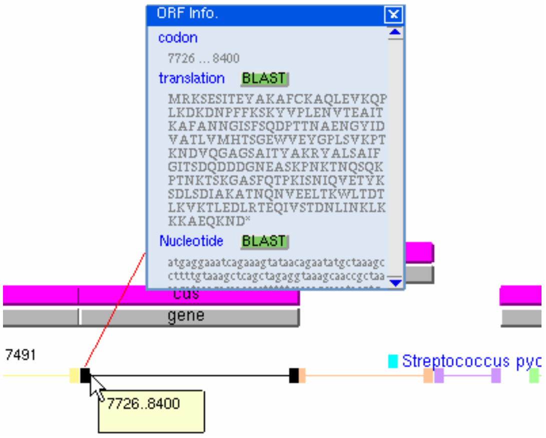
We measured the real-time disk access for GVis as the user zooms in to a single large DNA sequence. Retrievals of 10KB are typical during the zoom when the projected size of the entire sequence is between a few pixels and 500 pixels. As the DNA sequence stretches across the entire width of the display, retrievals are 80KB on average. On average, there are about 10 retrievals per second, each resulting from a traversal from parent to child. This retrieval rate is governed largely by drawing speed of the retrieved data, and retrieval causes no pauses in the drawing. In fact, on newer desktop PCs, the speed of retrieval and drawing is so fast that pauses must be inserted to keep navigation at a reasonable speed, which can be adjusted according to user preference.

7 Analysis Within GVis

One of the most powerful aspects of GVis is to analyze data found as a result of fast, interactive exploration and display the results in context for correlative analysis. All the data needed, including annotations, can be collected in one database and thus quickly retrieved and displayed. Here we demonstrate with BLAST results, but a variety of other gene or protein analysis tools could be inserted.

Since BLAST and like tools find related nucleotide or protein sequences that might belong to widely different organisms, it is best to display results using a series of secondary windows, as shown in Figure 8. The secondary windows can all be panned and zoomed. In addition, the user has the option of selecting in one window and having the appropriate genome (including the relevant part of the genome) highlighted in the main window.

The user, guided by the annotations, directly selects a nucleotide sequence of interest (Figure 9). The selection bar defaults to open reading frames (ORFs), likely sites for gene expression, but can also be interactively placed and sized. The selected area is sent off to BLAST. In the example shown in Figure 10, the nucleotide sequence is automatically expressed into possible proteins and BLAST then provides a list, in its own window, of hits in descending order of interest, with the original protein sequence on top. When the user passes the cursor over the hits, names and other information pop up (Figure 10 (left)). The curves on top of each sequence in the hit list show the local similarity to the original sequence. Thus peaks and especially plateaus are of interest and can quickly be found. Since hits from different places on the list may be of interest, the user can select from this list and an analysis window pops up that contains the user's selections in the order selected. Detailed comparisons will be made from the working (analysis) window (Figure 10 (right)), so any annotations should be displayed there. For example, when BLAST is run to find related nucleotide sequences, the analysis window contains gene, CDS, and other annotations as shown in Figure 6. Multiple BLAST windows can be set up and selections made into one or more working windows. Furthermore, BLAST can be run again from selections within the working window. This is the iterative process, indicated in Figure 1, supporting deeper analysis and



codon

7726 ... 8400

translation

MRKSESITEYAKAFCKAQLEVKQP
LKDKDNPFFKSKYVPLENVTEAIT
KAFANNGISFSQDPTTNAENGYID
VATLVMHTSGEWVEYGPLSVKPT
KNDVQGAGSAITYAKRYALSAIF
GITSDQDDDGNEASKPNKTNQSQK
PTNKTSKGASFQTPKISNIQVETYK
SDLSDIAKATNQNVEELTKWLTDT
LKVKTLEDLRTEQIVSTDNLINKLK
KKAEQKND*

Nucleotide

atgaggaaatcagaaagtataacagaatatgctaaagc
cttttgtaaagctcagctagaggtaaagcaaccgctaa

Figure 9. *Selection interface. The black barbell at the bottom is the selection and the colored barbells indicate ORFs.*

understanding.

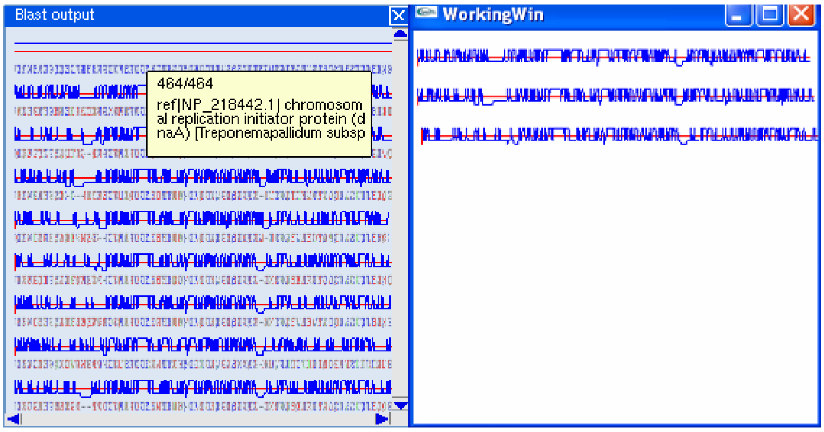
This collection of fast, intuitive, and interconnected analysis tools greatly increase the capabilities of bioinformatics investigators. We have put these tools into the hands of these investigators both at Georgia Tech and UNC Charlotte. Their feedback has already led to a round of additions and refinement.

8 Conclusions and Future Work

The goals of this work have been to support the new gene finding and analysis process illustrated in Figure 1. The current GVis framework supports the viewing and analysis of genomic data by allowing the user to investigate, and launch analyses on, any of the genomes in the database at any level of detail. These levels of detail range from taxonomy or phylogeny to the individual nucleotides.

The GVis framework improves significantly on the tools currently available to many biologists in the international public databases such as Ensembl and GeneDB. These web interfaces provide only a fragmented view of a single piece of data at a time, and offer interaction times ranging from seconds to minutes when accessing data. By contrast, the GVis framework provides a comprehensive taxonomical view of the genetic database that the user may have, and can view that data at any level of desired detail. Moreover, the GVis framework provides very fast access to these data at any level of detail. Rapid access is provided by hierarchical structures that speed up both the graphical display of data and the incremental retrieval of data from disk.

Our future work is to extend the GVis framework to support more analysis tools, and to display correlative analyses of genomic data in a more comprehensive and effective way. We also plan to add a system for users to enter their own annotations to the database. Finally, we are planning thorough evaluations and comparative testing.



464/464
ref|NP_218442.1| chromosomal replication initiator protein (dnaA) [Treponemapallidum subsp

Figure 10. *Close-up of BLAST output window (left) and working (analysis) window (right).*

References

[Ada02] R.M. Adams, B. Stancampiano, M. McKenna, and D. Small. Case Study: A Virtual Environment for Genomic Data Visualization. *IEEE Visualization 2002*, pp. 513-516 (2002).

[Bar94] Bartram, L., R. Ovans, J. Dill, M. Dyck, A. Ho, and W.S. Havens. "Contextual Assistance in User Interfaces to Complex, Time-Critical Systems: The Intelligent Zoom". In *Graphics Interface 1994*, Banff, AB, Canada, May 1994.

[Bed94] Bederson, B.B., and Hollan, J.D., "Pad++: A Zooming Graphical Interface for Exploring Alternate Interface Physics," *Proc. ACM UIST '94*, Marina del Rey, CA,Nov. 1994, pp. 17-26.

[Bel00] Bell, B. and Feiner S, "Dynamic space management for user interface", *Proceedings of the 13th annual ACM symposium on User interface software and technology*, 2000, Sandiego, California, pp. 239-248.

[Bor93] Borodovsky, M. and J. McIninch. GeneMark: Parallel Gene Recognition for both DNA Strands. *Computers & Chemistry* 17: 123-133, (1993)

[Chi95] Chi, E.H.-H.; Barry, P.; Shoop, E.; Carlis, J.V.; Retzel, E.; Riedl, J. Visualization of biological sequence similarity search results. *IEEE Visualization '95*, pp. 44-51 (1995).

[Dav99] Davis, D., W. Ribarsky, T.Y. Jiang, N. Faust, and S. Ho. "Real-Time Visualization of Scalably Large Collections of Heterogeneous Objects," *IEEE Visualization '99*, pp. 437-440.

[Fau00] N. Faust, W. Ribarsky, T.Y. Jiang, and T. Wasilewski. Real-Time Global Data Model for the Digital Earth. *Proc. Intl Conf.on Discrete Global Grids (2000).*

[Fur95] G. Furnas and B.B. Bederson. Scale Space Diagrams: Understanding Multiscale Interfaces. *CHI'95*, pp. 234-241 (1995).

[Goe03] A. Goesmann et al. Building a BRIDGE for the integration of heterogeneous data from functional genomics into a platform for systems biology. J Biotechnol **106**(2-3): 157-67 (2003).

[Joh91] Johnson, Brian, and Shneiderman, B. "Treemaps: A space Filling Approach to the Visualization of Hierarchical Information Structures", *Proceedings of the IEEE Visualization*, 1991, San Diego, CA, pp.284-291

[Kan02] M. Kano, S. Tsutsumi, and K. Nishimura. Visualization for Genome Function Analysis Using Immersive Projection Technology. *IEEE Virtual Reality '02*, pp. 224-231

[Lor02] Ann Loraine and Gregg Helt. Visualizing the Genome: Techniques for Presenting Human Genome Data and Annotations. *BMC Bioinformtics* 3(19) (2002).

[Per93] Perlin, K., and Fox, D., "Pad: An Alternative Approach to the Computer Interface," *Proc. ACM SIGGRAPH '93*, Anaheim, CA, Aug. 1993, pp. 57-64.

[Pir01] P. Pirolli, S. Card, and M. Van Der Wege. Visual Information Foraging in a Focus + Context Visualization. *Proceedings of CHI 2001*, pp. 506-513 (2001).

Rib03 William Ribarsky. Virtual Geographic Information Systems. To be published. The Visualization Handbook, C. Hanson and C. Johnson, eds (Academic Press, New York, 2003).

[Sha02] C. Shaw, W. Ribarsky, Z. Wartell, and N. Faust. Building the Visual Earth. Vol. 4744B, SPIE 16th Int. Conf. on Aerospace/Defense Sensing, Simulation, and Controls (2002).

[Ste02] Stein L., Creating a Bioinformatics Nation. A Web Services Model Will Allow Biological Data to be Fully Exploited, *Nature*, 2002, 417, pp.119-120

[Tur01] R.J. Turner et. al. Visualization Challenges for a New Cyber-Pharmaceutical Computing Paradigm. *Proc. IEEE 2001 Symposium on Parallel and Large Data Visualization*

[War99] Wartell, Z., Ribarsky, W., and Hodges, L.. Third Person Navigation of Whole-Planet Terrain in a Head-tracked Stereoscopic Environment. *IEEE Virtual Reality 99*, pp. 141-149.

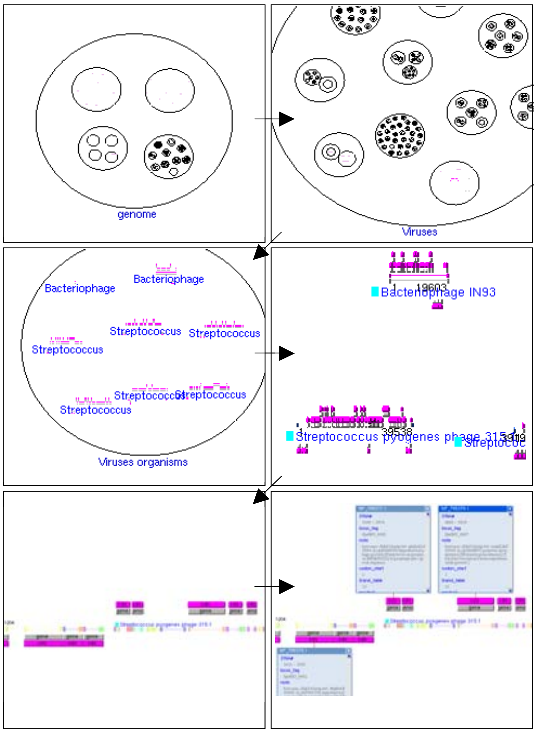


Figure 2. *Zoom in operation from root (all genomes) to a specific genome (Human herpesvirus 7).*

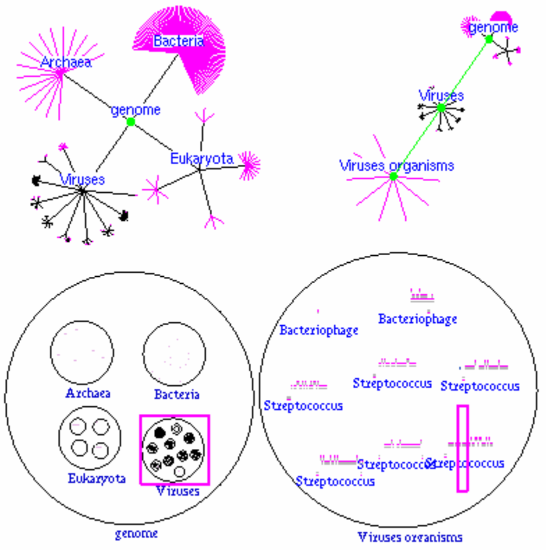


Figure 4. *Upper Left: Node link tree of our genome database containing 500Mbytes of data and 1500 genomes. Lower Left: Corresponding and overview image of zoomable phylogeny tree. Upper Right: Node link tree with a virus highlighted in green Lower Right: Corresponding zoomed-in view of unclassified bacteriophages.*

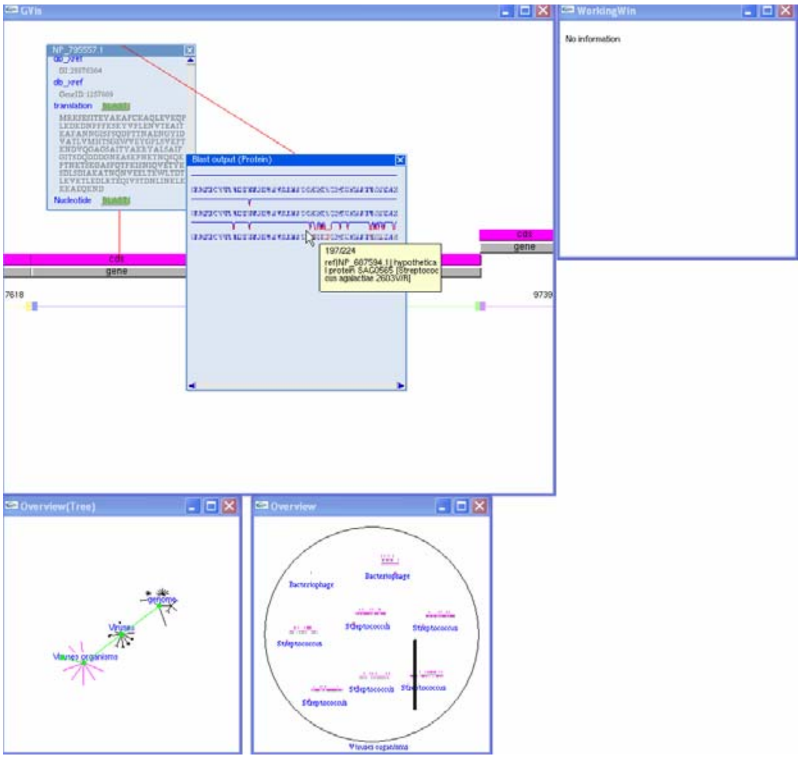


Figure 8. *Selection from genome and resulting BLAST output window.*