



# A dynamic multiscale magnifying tool for exploring large sparse graphs

Pak Chung Wong<sup>1</sup>  
Harlan Foote<sup>1</sup>  
Patrick Mackey<sup>1</sup>  
George Chin<sup>1</sup>  
Heidi Sofia<sup>1</sup>  
Jim Thomas<sup>1</sup>

<sup>1</sup>Pacific Northwest National Laboratory,  
Richland, WA, U.S.A.

Correspondence:  
Pak Chung Wong, P.O. Box 999, K7-28,  
Richland, WA 99352, U.S.A.  
Tel: +1 509 372 4764;  
Fax: +1 509 375 3641;  
E-mail: pak.wong@pnl.gov

## Abstract

We present an information visualization tool, known as GreenMax, to visually explore large small-world graphs with up to a million graph nodes on a desktop computer. A major motivation for scanning a small-world graph in such a dynamic fashion is the demanding goal of identifying not just the well-known features but also the *unknown-known* and *unknown-unknown* features of the graph. GreenMax uses a highly effective multilevel graph drawing approach to pre-process a large graph by generating a hierarchy of increasingly coarse layouts that later support the dynamic zooming of the graph. This paper describes the graph visualization challenges, elaborates our solution, and evaluates the contributions of GreenMax in the larger context of visual analytics on large small-world graphs. We report the results of two case studies using GreenMax and the results support our claim that we can use GreenMax to locate unexpected features or structures behind a graph. *Information Visualization* (2008) 7, 105–117. doi:10.1057/palgrave.ivs.9500177

**Keywords:** Graph analytics; multiscale visualization; very large sparse graphs

## Introduction

We are interested in visually exploring large small-world graphs<sup>1,2</sup> with up to a million graph nodes on a desktop computer. Because drawing large graphs is a time-consuming task<sup>3,4</sup> and the number of pixels of a desktop monitor is limited, our general design philosophy on graph analytics<sup>5</sup> has always been to analyze large graphs without explicitly drawing the graphs. In other words, we want to aggregate the graphs, simplify their structures, or summarize their features before visualizing the results. Nevertheless, ‘seeing the whole’ before ‘drilling the parts’ is still a dominant daily practice for most of our analysts. This paper describes a customized visualization tool requested by our analysts to interactively browse a large small-world graph in its entirety smoothly and continuously on a desktop computer.

A major motivation for scanning a small-world graph in such a dynamic fashion is the demanding goal<sup>6,7</sup> of identifying not just the well-known features but also the *unknown-known* (you don’t know that you know) and *unknown-unknown* (you don’t know that you don’t know) features of the graph. This critical and somewhat elusive requirement often challenges traditional database aggregations that often look for outliers, norms, etc. of the underlying data. These unknown or undefined features can best be described as the kind of knowledge that ‘you know when you see it.’ Thus our goal is to provide a friendly, high-performance tool for the analysts to freely examine large graphs and look for subtle clues.

As it turns out, the above general user approach to browsing a large small-world graph has created a number of significant design challenges as the sizes of the underlying graphs grow to about a million nodes.

While there are graph drawing tools such as Pajek<sup>8</sup> and Tom Sawyer Software<sup>9</sup> that claim to draw a graph of comparable size, there is not yet proven technology or a working tool that can browse the same graph and explore its details interactively on a modest single-core desktop computer. The other option to browse a graph of that size is to represent it as a matrix,<sup>10</sup> which is not particularly effective at showing important graph features such as a ‘closed loop’ or ‘paths between two nodes.’

This paper presents a practical and working visualization tool, known as GreenMax, to browse the structural details of a large small-world graph adaptively and interactively. The design concept of GreenMax is similar to the ‘magnifier’<sup>11</sup> tool provided by Windows XP, except that GreenMax progressively generates finer views with greater detail on the fly as display resolution increases. Similar zooming capability is also found among geospatial applications such as Google Earth.<sup>12</sup> Our problem, however, is unique because of the nature and construction of a large small-world graph that requires sophisticated heuristics to overcome a number of computation and complexity hurdles in order to meet the response time requirement. This paper describes these challenges, introduces our solutions, and evaluates the contributions of GreenMax in the larger picture of visual analytics<sup>7</sup> on large small-world graphs. In the second half of the paper, we discuss the results of two case studies using the GreenMax technology that involve a number of real-life data sets.

GreenMax is not meant to be a standalone visualization tool. It is a part of our Have Green<sup>5</sup> architectural framework, which includes other components such as Greenland,<sup>13</sup> GreenSketch,<sup>14</sup> and GreenArrow<sup>15</sup> recently developed for different government and industry sectors. Together these components form a graph analytical platform that allows developers to customize visual analytics tools for different graph applications.

## Related work

GreenMax uses a multiresolution approach extensively to visualize large graphs, which would otherwise be too big to show on screen in detail. This section describes previous work on a number of topics that are related to the development of GreenMax.

### Small-world graphs

GreenMax’s design is primarily targeted at larger small-world<sup>1,2</sup> graphs that have high degrees of clustering and small average path lengths relative to their number of nodes. A classic example of a small-world graph is a social network where people generally organize and link to one another through short chains of associations or acquaintances. Beyond social networks, small-world graphs also occur in many other real-world models such as gene regulatory networks and internet network traffic. They are considered a class of random graphs that have been extensively studied in network theory. This paper later will show

how we take advantage of a small-world graph’s structural properties to develop a multiscale browsing algorithm that supports interactive response time.

### Graph drawing and visualization

For decades, the graph drawing community has led the studies in most of the graph drawing and layout issues. The two textbooks by Di Battista *et al.*<sup>3</sup> and Sugiyama<sup>4</sup> summarize most of the major graph drawing algorithms and their applications. The proceedings of the annual Graph Drawing Symposia,<sup>16</sup> now in its 16th year, provide a wealth of information on the cutting-edge technology.

Visualizing graphs and hierarchies has been a major research topic within the data visualization community since its conception in the early 1990s. The two textbooks by Card *et al.*<sup>17</sup> and Chen<sup>18</sup> cover much of the major research and applications surrounding graph and hierarchical visualization. The survey paper by Herman *et al.*<sup>19</sup> represents the most complete literature review up to 2000. The annual IEEE Symposium on Information Visualization<sup>20</sup> continues to produce new results on various topics of graph visualization.

A major difference between the graph visualization and graph drawing communities is that the former almost always involves some sort of interaction, whereas the latter focuses heavily on algorithmic developments. The latest challenge, however, is to integrate the best of the two communities and form a new environment of graph analytics.

Layout-wise, both Di Battista *et al.*<sup>3</sup> and Sugiyama<sup>4</sup> present a number of layout classes for a graph. Detailed discussion of individual layouts is out of scope for this paper. While a node-link layout still represents the most popular approach to visualize a very large graph, some have used a matrix-based layout<sup>10</sup> to accomplish different objectives of a very large graph visualization. Elsewhere, Shneiderman<sup>21</sup> and Aris *et al.*<sup>22</sup> introduce a ‘semantic substrates’ approach, which partitions graphs into smaller, more comprehensible ones that align to specific user tasks.

### Very large graph drawing

Drawing a graph nicely on screen is computationally expensive.<sup>3,4</sup> There is an ongoing community effort to speed up the drawing process by developing adaptive algorithms with complexity of  $O(n^2)$  and beyond. Notable work in this area has recently been presented by Harel *et al.*<sup>23</sup> and Walshaw.<sup>24</sup> In addition to the cutting-edge drawing algorithms, the two papers provide resourceful clues and ideas for further improvements on their designs. Their references sections also present a wealth of information covering topics from graph partitioning to Laplacian Eigenvector computation.<sup>25</sup> The design of GreenMax is based partly on Walshaw’s algorithm with multiple enhancements to address the unique

challenges of drawing a large small-world graph progressively in interactive time.

Some literature specifically defines the terms ‘drawing’ to express the sense of computer graphics and ‘layout’ to convey the concept of computation. For example, Tulip<sup>26</sup> speeds up the physical ‘drawing’ of a graph by eliminating its non-visible elements using an OpenGL-based rendering engine. The same tool uses the term ‘layout’ to specify the computation process that places the graph nodes and links in certain formats and styles. Our discussion follows the more traditional convention of using ‘drawing’ to mean ‘laying out’ a graph, as suggested by Di Battista *et al.*<sup>3</sup> and Sugiyama.<sup>4</sup>

### Digital magnifying glass

The concept of a digital magnifying glass, which zooms in on a selected area of a display for the purpose of clarity or understanding, can be found in many computer-user interfaces today. Some of them, like the one provided by Microsoft Windows XP, involve only pixel enlargement; others show continuous distorted views<sup>27</sup> like a fisheye lens; and others can see through layers of hidden details like a Magic Lens.<sup>28,29</sup> Most of these magnifying tools rely on some sort of spatial filtering that operates on Cartesian grids to generate multiple resolution views. This rather straightforward zooming approach, however, cannot be applied to graph data sets because of the lack of physical coordinates of the graph entities. Some of today’s graph analysis software, such as Analyst’s Notebook,<sup>30</sup> do provide a basic zooming capability by redrawing portions of the same graph again in the magnified window. GreenMax, on the other hand, uses an adaptive approach to cluster the neighboring graph nodes progressively so as to support a clutter-free visualization in every resolution.

Our discussion on the zooming lens concept has so far focused on bringing out the local details of a low-resolution visualization. Others have attempted the opposite by reducing the local resolution of a high-resolution display so that the users can see the details underneath the clutter. Notable visualization results in this direction are Ellis and Dix<sup>31</sup> and Wong *et al.*<sup>32</sup> The two approaches represent the classic top-down vs bottom-up visualization designs that are often found in large data analytics.

### Multilevel graph drawing

GreenMax uses a highly effective multilevel graph drawing approach to pre-process a large graph by generating a hierarchy of increasingly coarse layouts that later support the dynamic zooming of the graph. Fine details between the hierarchy levels are then generated on the fly during the zooming operation to accomplish the response time requirement. This section discusses the general requirements of drawing a large sparse graph, presents the concept of a multilevel graph drawing algorithm, describes the Walshaw multilevel algorithm, explains why the Walshaw approach falls short of meeting our

requirements, and finally introduces the new features provided by GreenMax that address the shortcomings. Bear in mind that this is an important pre-processing step required to support the zooming operation of a magnifying tool discussed later in the next section.

### Major requirements and challenges

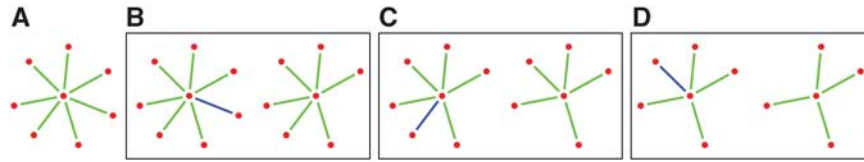
For GreenMax to succeed, it must have an acceptable interactive response time when users browse the graphs. In other words, when a graph is panned or zoomed, it will move accurately and promptly without hesitation. This requirement poses a serious challenge to most of the traditional force-directed graph layout techniques that generally require  $O(n^3)$  time to re-compute. Fortunately, the multilevel approaches suggested by Harel and Koren<sup>23</sup> and Walshaw<sup>24</sup> provide a relatively effective way to draw a large graph. The underlying ‘multilevel’ concept also provides a means to cluster fine local details when a computer screen does not have enough pixels to depict all the fine details. However, neither of these two algorithms fully meets the requirements for GreenMax. The Harel approach,<sup>23</sup> as suggested by Walshaw,<sup>24</sup> contains a component with complexity of  $O(n^2)$ , which will become a computational bottleneck as the graph grows. The Walshaw<sup>24</sup> approach, which was demonstrated using mostly mesh-like graphs, fails to perform effectively on sparse graphs that exhibit small-world properties. GreenMax has been created to address all these problems.

### A general multilevel layout design

Instead of drawing an entire graph all at once like a force-directed algorithm does, a multilevel algorithm first goes through a sequence of coarsening steps to create a hierarchy of increasingly coarse graphs. Links are progressively removed from the graphs at each level, and nodes tied to the removed links are merged with their neighbors. This coarsening step is repeated until a predefined graph size is reached. Note that the process so far does not involve any node placement operations, that is, no costly drawing has been done yet.

The second step is to generate an initial layout for the coarsest graph in the hierarchy. This initial layout serves as a seed to generate the layouts of the other graphs in the hierarchy in the final (uncoarsening) step.

The uncoarsening step refines the above initial layout and then extends it progressively through the hierarchy all the way to the original graph. Nodes that were removed in the coarsening step are now brought back to refine the graph layouts at the corresponding levels. Because the layout of a coarse graph provides critical clues towards the global structure of the next finer one (and thus speeds up the layout process at that level), the entire multilevel layout can be completed in an order of magnitude faster than any conventional force-directed algorithms. More performance discussion can be found in the section Computational performance.



**Figure 1** (A) A starburst. (B) The matching process selects one of the seven links (in blue) and merges its two connected nodes in the first iteration. (C) The same coarsening step is repeated in the second iteration. (D) A  $\sim 50\%$  size reduction is accomplished after the third iteration.

### The Walshaw algorithm

We briefly describe the multilevel graph layout technique presented by Walshaw.<sup>24</sup> The description follows our previous discussion on a general three-step multilevel layout design approach to coarsen, lay out, and then uncoarsen a graph to a final layout. Interested readers are directed to Walshaw<sup>24</sup> for fine implementation details.

**Graph coarsening** The coarsening step generates a hierarchy of increasingly coarse graphs while retaining the most important structural features at each level for rapid refinements in the uncoarsening step. Reducing too many nodes at a time (and thus a shallower hierarchy) may lose too much detail to support an effective recovery later. Reducing too few (and thus a deeper hierarchy), on the other hand, may lead to unnecessary computation for many fairly similar graphs in the hierarchy. Walshaw suggests a coarsening approach known as *matching* that maintains an approximately 50% reduction rate at each level. This is done by pairing each node with at most one neighbor to form clusters (of at most two nodes) and then collapsing these clusters to form a coarse graph. The new coarser graph is guaranteed to have no less than half the graph nodes of the previous (finer) one.

**Initial layout** The layout step creates an initial layout for the coarsest graph in the hierarchy. It is not hard to see that the *matching* approach can generate a coarse graph with as little as two nodes if the original graph is connected (i.e., no isolated nodes or subgraphs). In this case, the initial layout of the coarsest graph can indeed be substituted with any random graph with two nodes.

**Graph uncoarsening** The uncoarsening step brings back the previously removed details level by level to the coarse graphs and refines them all the way up to the finest level in the hierarchy. The process uses a force-directed algorithm developed by Fruchterman and Reingold<sup>33</sup> that uses an attractive force to tune the local details and a repulsive force to balance the global structure of a graph layout. To further speed up the node placement process, the drawing plane is divided into grid cells and only nodes in the adjacent cells (i.e., not the entire graphs) are used to tune the final positions of the nodes in the center cell. This shortcut

works well because a coarse layout is already a very good approximation to the layout of the next finer one in the same hierarchy.

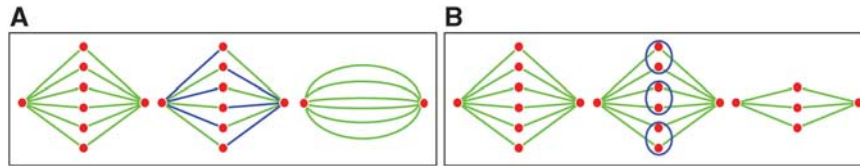
**Algorithm complexity** It is proved<sup>24</sup> that the total complexity of the Walshaw algorithm when applied to a sparse graph  $G(N, L)$  where  $N$  = number of nodes and  $L$  = number of links is  $O(|N_l| + |L_l|)$  at each hierarchy level  $l$ . Because the number of levels is a constant ( $\log_2 N$ ), the complexity of the algorithm remains in the order of  $O(n)$ .

### Design shortcomings and new solutions

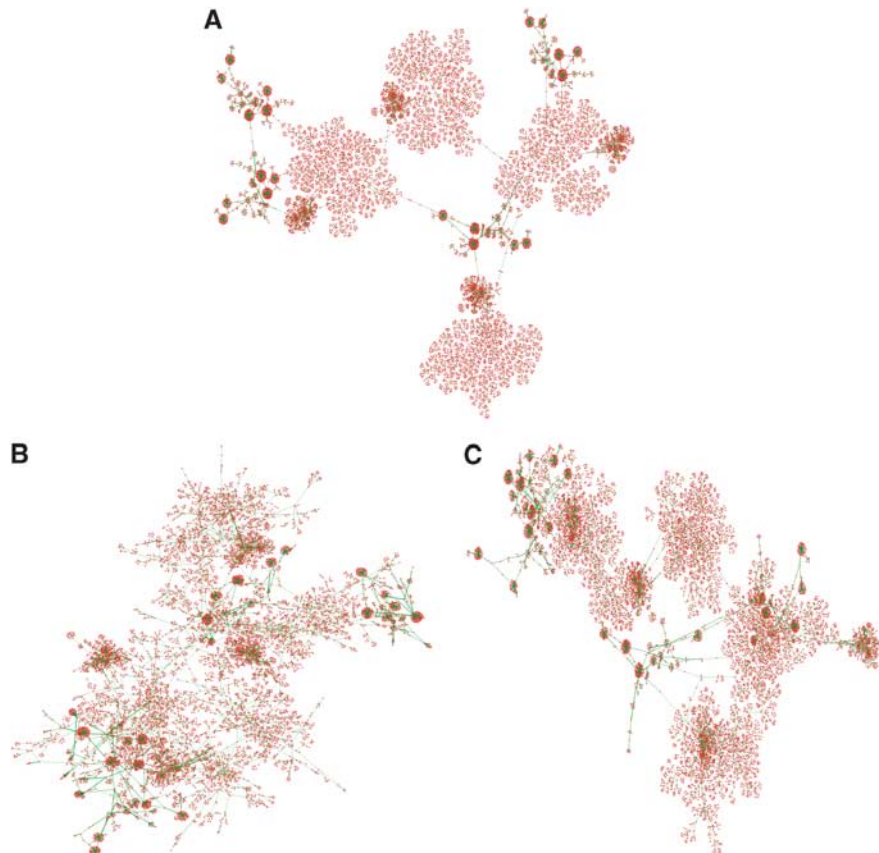
We identify a few shortcomings of the Walshaw algorithm when applied to a large small-world graph. New solutions and enhancements based on our implementation of GreenMax are presented and the performance results are discussed.

**Graph coarsening** Walshaw uses the *matching* concept to reduce the size of a graph in the coarsening step of the algorithm. Only one of the two nodes within a matching cluster can be removed in each iteration. This approach works well on mesh-like graphs with few or no node-clustering or link crossings. However, a hallmark characteristic of a small-world graph is the starburst-like clusters that spread out sporadically throughout the graph. The *matching* approach substantially restricts the number of nodes that can be removed and thus generates a deeper than necessary hierarchy. For example, given a starburst structure in Figure 1(A), the matching approach can only remove one link in an iteration in Figure 1(B) because all the other (six) nodes are connected to the center node that has already been marked. It will take two more iterations in Figures 1(C) and (D) to eventually accomplish the  $\sim 50\%$  reduction goal.

GreenMax relaxes the one-link-per-iteration restriction when a small-world feature such as a starburst is detected. Multiple nodes will be removed even though they are connected to a node that has already been matched (e.g., the center node of a starburst). For example, GreenMax removes all three blue links shown in Figures 1(B–D) in a single iteration and potentially forms a shallower hierarchy. Unfortunately, this approach alone does not solve all of our coarsening problems.



**Figure 2** (A) Undesirable coarsening result. (B) In order to preserve the shape of the coarsened graph, nodes that are indirectly connected can be merged.



**Figure 3** Layout of the same graph using (A) GreenMax after 11 s, (B) Walshaw after 10 s, and (C) Walshaw after 25 s.

A more complicated situation arises when multiple starbursts are connected together. As shown in Figure 2(A), even though we allow multiple links to be removed in the same iteration, the coarsened graph may not look like the original. The distorted feature, if not addressed early on, will continue to deteriorate throughout the hierarchy and cause more computation time later on. A more desirable coarsening approach for the same graph structure is shown in Figure 2(B), where pairs of nodes are merged (shown in blue) even though they are not connected directly. Together these two heuristic enhancements substantially reduce the depth of the coarsening hierarchy when the cluster features are abundant in the graph and preserve the desirable 50% reduction factor.

**Initial layout** Our experiments indicate that the Walshaw algorithm is heavily dominated by the last few finer graphs in the hierarchy. For example, consider a hierarchy with a 50% reduction rate in each level, the three finest ones represent a majority (i.e.,  $\frac{1}{2} + \frac{1}{4} + \frac{1}{8} = \frac{7}{8} = 87.5\%$ ) of the computations required to generate the final (finest) layout. What that means is we can afford to apply more complicated heuristics to improve the layout of the coarser graphs without causing major delay to the overall performance. Note that the establishment of a quality global structure at the coarse level early on can benefit all the later refinements in the hierarchy.

Instead of coarsening the graph all the way down to two nodes, GreenMax stops the coarsening process with about 100–400 nodes (depending on the size of a graph) and

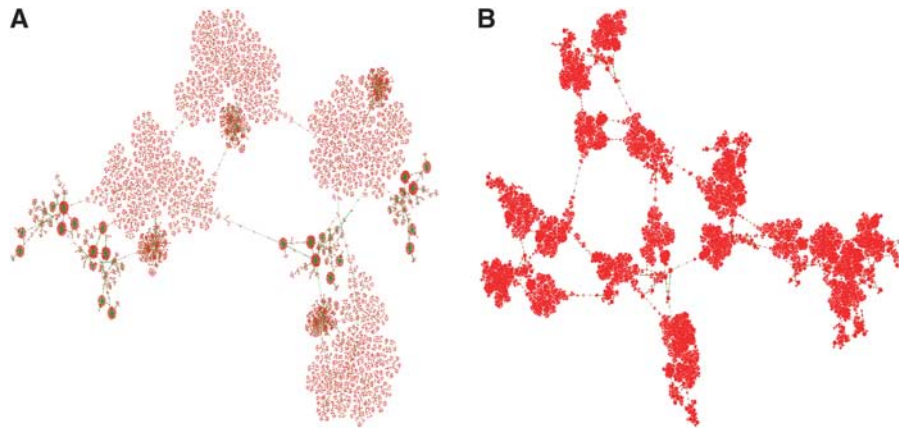


Figure 4 Layouts of graphs with (A) 24,310 and (B) 121,560 nodes and links.

then uses a very accurate but also expensive  $O(n^3)$  force-direct algorithm such as Kamada and Kawai<sup>34</sup> to generate a high-quality initial layout. While there is a slight delay in getting the initial layout because of the  $O(n^3)$  complexity, our experiments show that we can obtain a much better final layout in a shorter period of time by using the new approach.

Figure 3 illustrates a comparison between the GreenMax and Walshaw approaches. Figure 3(A) shows the final layout of a graph with 9296 nodes using the GreenMax approach (that starts with a 152-node coarse graph). The layout, which takes 11 s to complete, clearly separates four major clusters in the graph. Figure 3(B) shows the layout of the same graph using the Walshaw approach (that starts with a two-node coarse graph). It takes 10 s (i.e., one less than the GreenMax approach), but the quality of the layout suffers greatly. All four clusters are mingled together into a chaotic mess. Figure 3(C) repeats the same process in Figure 3(B) but the tuning time is extended from 10 to 25 s. While the layout quality of Figure 3C improves substantially over Figure 3(B), the former still lags behind the one in Figure 3(A).

**Graph uncoarsening** The Walshaw algorithm uses a fixed number of iterations to tune the final node placement in the uncoarsening step. For a very large graph, our experiments show that we can reduce the number of iteration steps when the graphs are getting bigger and still obtain the same drawing quality in the final layout. In other words, we use a dynamic scheme to continuously decrease the number of tuning steps as the size of the graph grows during uncoarsening. For example, GreenMax takes 11 and 38 s correspondingly to compute the two layouts in Figures 4(A) and (B) using varying number of iterations in the uncoarsening step. The same two layouts will take 16 and 73 s if constant numbers of iterations are used in the uncoarsening step as suggested by Walshaw.

Table 1 Computation performance of GreenMax to layout large small-world graphs

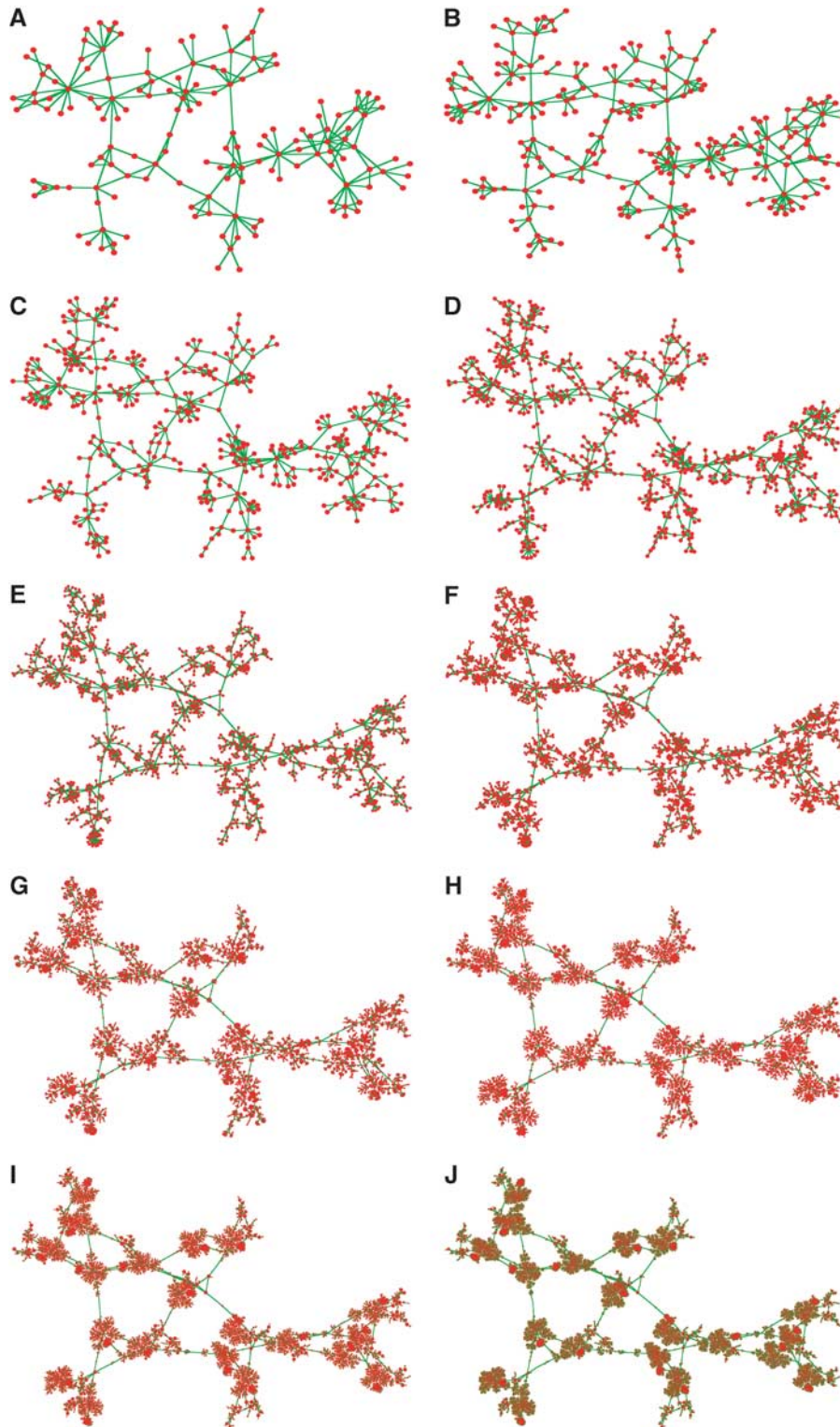
#Nodes	#Links	Wall clock seconds	# Nodes/second
450	1000	2	225
904	1546	2	452
1808	3723	3	602
3616	7449	3	723
9296	15,014	3	3099
18,592	30,031	6	3099
46,480	75,080	16	2905
102,256	165,178	40	2556
204,512	213,847	90	2272
409,024	660,721	197	2076
613,536	874,571	300	2045
1,022,560	1,535,295	541	1890

### Computational performance

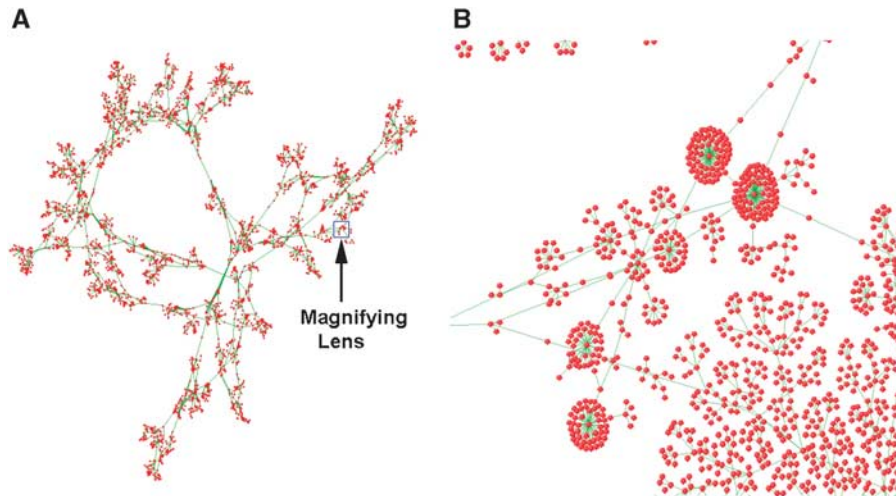
We create a dozen synthetic graphs with embedded small-world features to demonstrate the performance of GreenMax in drawing small-world graphs with sizes ranging from 450 to 1.2 million nodes (1000 to 1.5 million links correspondingly). More examples using real-life data sets are demonstrated in the case studies in the fifth section. The compiled C++ code is running on a 2.8 GHz Dell Precision 670 with 4 GB of memory. Table 1 shows the computational time to generate layouts for the graphs using only one processor.

GreenMax generally performs well in comparison to the results reported by Walshaw. For example, the Walshaw algorithm requires 5–7 min to draw a graph with 225,000 nodes.<sup>24</sup> In Table 1, GreenMax takes about 1.5 min to draw a graph with 204,512 nodes and about 3.28 min to draw one with 409,024 nodes. The improvement is roughly about 300% in terms of time spent.

Note that the average processing rate (i.e., number of nodes per second) in Table 1 becomes stabilized (and also



**Figure 5** Coarse layouts within a hierarchy with (A) 152, (B) 247, (C) 431, (D) 766, (E) 1,418, (F) 2,726, (G) 5,471, (H) 11,060, (I) 22,917, and (J) 46,480 nodes.



**Figure 6** (A) The navigation window of GreenMax. (B) The corresponding focus window that depicts the magnified view of the selected area (marked by a blue rectangle) on the navigation window.

most effective) once the number of nodes reaches  $\sim 9$  K. The near-linear processing rate shows that GreenMax's performance will only degrade slightly as the size of the graph grows, which is a very desirable quality possessed by algorithms that deal with very large data sets. Figure 5 demonstrates an uncoarsening sequence of a graph (shown in Table 1 with 46,480 nodes and 75,080 links) from a coarse version with 152 nodes in Figure 5(A) to the finest version with 46,480 nodes in Figure 5(J). Notice that the global structure of the finest graph has already been well established in the first (coarsest) layout in Figure 5(A). As shown in Table 1, the entire graph takes only 16 wall-clock seconds to compute.

### Dynamic multiscale magnifying tool

Once the multilevel layout hierarchy of a large graph (as described in the previous section) is established, GreenMax provides a friendly front-end for analysts to browse the multiscale graph details using morphing animation.

GreenMax's user-interface contains two primary views—a *navigation* window (Figure 6(A)) that selects the zooming area and a *focus* window (Figure 6(B)) that shows the corresponding magnified view at various resolutions. To maintain a clutter-free visualization, the navigation window always limits the size of the layout to be displayed. Graphs that are bigger than the limit will be replaced by a coarser version within the multilevel hierarchy that meets the requirement.

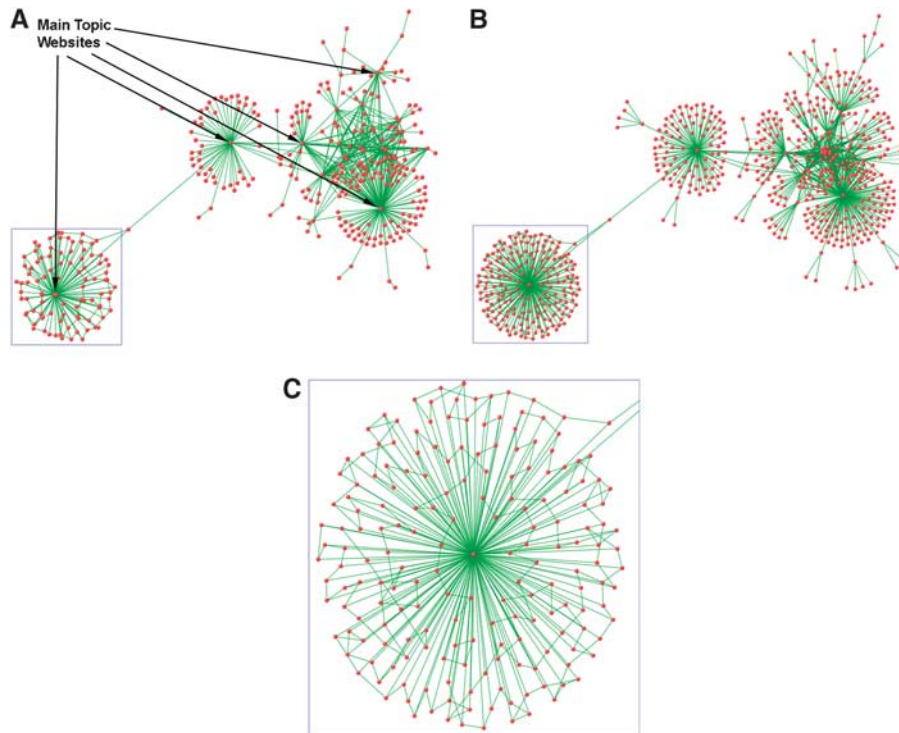
Similarly, the focus window also tries to maintain a clear and organized visualization by selecting the right coarseness level to clearly show the localized details. However, unlike the navigation window, the analyst can continuously zoom in the details on the focus window until the finest resolution is reached. During zooming, GreenMax

looks to see which coarsened graph is closest to having the right coarseness for the amount of area taken up by the focus window. If the current coarsened graph needs to change to meet this, a dynamic animation sequence occurs. If the graph becomes less coarse, new nodes are shown expanding from the previously coarsened nodes. If the graph becomes more coarse, nodes are contracted back to form new, coarser nodes. While there are graph analysis software tools such as Analyst's Notebook<sup>30</sup> that suggest similar zooming capability, none of them (both commercial and public domain) have reported the same performance as GreenMax.

The GreenMax interface was developed entirely in C# using Managed DirectX for the graphics library. The algorithmic backend was implemented in C++ as a dynamic-link library (DLL). While GreenMax was developed independently and can perform as a standalone tool, it has been successfully integrated into our Have Green<sup>5</sup> suite that includes a number of graph analytics applications from visualizing small-world graphs,<sup>15</sup> querying their heterogeneous semantics,<sup>13</sup> to analyzing their topological features.<sup>14</sup>

### Case studies

To study the use of GreenMax, we sought to apply GreenMax to real-world problems carried out by real-world analysts and scientists. In two separate case studies, we provided the GreenMax tool to a cybersecurity analyst and a bioinformaticist to apply to authentic problems and data on two different PNNL projects. The findings we present are uses and discoveries made by real users. Both case studies were exploratory in nature.<sup>35</sup> Their intent was not to conduct a rigorous evaluation of the GreenMax tool, but rather to discover and explore its potential applications and uses.



**Figure 7** Layout of graphs showing (A) full web crawler network with main topic websites identified (high coarsening), (B) full web crawler network (light coarsening), and (C) close-up view of web crawler subnetwork containing cross-links. The subnetwork is the section of the full network shown bounded by a blue box in both Figure 7(A) and (B).

In the two case studies, users applied GreenMax to investigate small-world networks that were typical of their domains. One aspect we wished to evaluate was GreenMax’s general applicability to different kinds of small-world network problems. Are users able to discover critical patterns and insights in their network data regardless of its domain-specific contents and properties?

### Web crawler data

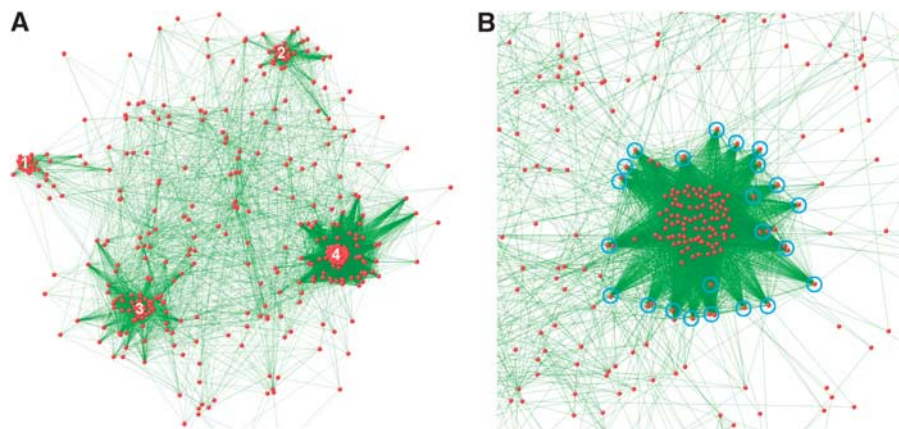
Using GreenMax, a cybersecurity analyst investigated network information collected from a web crawler. The cybersecurity analyst was a PNNL employee who had worked in the cybersecurity field for the past 10 years. She was a seasoned user of the Analyst Notebook<sup>30</sup> and possessed some experience with various PNNL-developed visualization tools. GreenMax was the analyst’s first experience using a graph analytics visualization tool outside of the Analyst Notebook. The size of link analysis graphs she customarily employed contained dozens of nodes, but she often restricted their sizes to what she could manage and comprehend. She received an hour of GreenMax training before she applied the tool to a specific cybersecurity problem.

Web crawlers are programs or software agents that browse the internet in a methodical and automated fashion. From an initial set of Universal Resource

Locators (URLs) known as seeds, web crawlers traverse the internet by following hyperlinks and collect information along the way. In intelligence analysis, the paths or networks that are revealed by a web crawler may be useful in identifying and tracking such things as the source of a particular piece of information, the evolution of a specific website, or the contents and authors that are similar or common among multiple websites.

In the case study, the web crawler was initially seeded with a specific website that was of particular interest to the analyst. Figure 7 shows the network induced by the web crawler at three different GreenMax resolutions. At the coarsest level (Figure 7(A)), the analyst discovered several ‘starbursts’ in the network, whose centers identified home or main topic websites and spokes conformed to hyperlinks within those main websites. As this view was uncoarsened in GreenMax (Figure 7(B)), additional spokes were added to each starburst to reveal more details of the contents of each main topic website and to expose more URLs. Through the uncoarsening of the network, the analyst saw that the basic shape of the overall graph remained the same while its starbursts grew fuller.

As shown in Figure 7(C), the analyst found that one of the starbursts in the network was distinct from the others in having cross-links among some of the spokes. When examining the URLs of this specific starburst, the analyst discovered that the cross-links identified advertisements



**Figure 8** Layout of graphs showing (A) full view of genome network with identified protein classes, and (B) close-up view of ABC transporter ATPase protein class along with satellite proteins (highlighted in blue) that are distantly related.

that would pop up in succession from this particular website. To promote sponsors, this website subjected visitors to a number of advertisements regardless of the point of entry into the website or the hyperlinks selected from within. As illustrated across the views of Figure 7, the cross-links in the described starburst were visible throughout the coarsened views. In this case, GreenMax had the desirable attribute of being able to preserve critical and identifying features or structures of a network as it coarsened.

Beyond the simple example of Figure 7, networks generated from web crawlers may easily reach millions of nodes. GreenMax's coarsening feature allows users to efficiently and interactively zoom into and out of large networks by providing coarsened summary views at different resolutions. An important consideration in optimizing multiscale viewing performance in GreenMax is to also be able to maintain and propagate global shapes, patterns, and cues in the processed views that would be generally evident in the fully elaborated drawing of the network.

The cross-linked starburst in the network visualization was an unexpected result for the analyst. Upon seeing normal, non-intersecting starbursts in the GreenMax visualization, the analyst was able to infer that the starburst patterns represented main websites and their enclosed hyperlinks. The cross-linked starburst, however, encouraged the analyst to investigate more deeply into the websites and URLs. As with most internet users, the analyst was familiar with websites that would pop up constant streams of advertisements, but she had not initially considered this kind of behavior to be in the web crawler data prior to conducting analysis using GreenMax. In essence, the GreenMax visualization generated a concept that the analyst had to comprehend and confirm as opposed to the analyst having a preconceived concept that would be confirmed using GreenMax. This amounts to an *unknown-unknown* type of discovery as

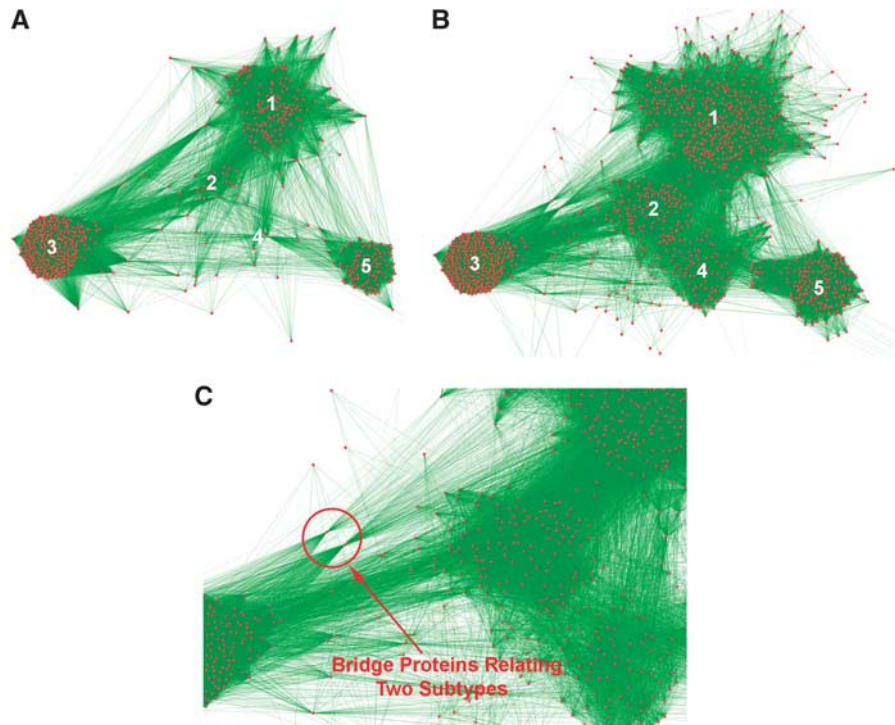
the analyst constructs and learns new concepts through GreenMax.

#### Bioinformatics data

Networks provide a fundamental way to represent biological systems.<sup>36</sup> A few examples of important biological networks include metabolic pathways, signal transduction events, protein-protein interaction networks, and protein homology networks. Network analysis has become especially relevant in biological research with technological advances that are producing large amounts of data.

For the first biology example, a bioinformaticist applied GreenMax to visualize a complete genome network for a microorganism. The bioinformaticist had been employed at PNNL for the past eight years. She was familiar with well-known graph-based bioinformatics tools such as Cytoscape<sup>37</sup> and Tom Sawyer<sup>9</sup> and had been involved in the development of similar tools at PNNL. In her research, the bioinformaticist typically worked with and analyzed biological networks consisting of hundreds to thousands of nodes. In certain cases, the bioinformaticist also modeled biological networks consisting of hundreds of thousands to millions of nodes. The bioinformaticist received an hour of GreenMax training before she applied the tool to a genome network problem.

Using GreenMax, the bioinformaticist constructed a genome-level protein network from an entire set of 4242 proteins encoded by the purple, nonsulfur bacterium *Rhodobacter sphaeroides* was used to build a genome-level protein network. Figure 8 shows the genome network in GreenMax at two different levels of resolution. Using the coarsest level visualization shown in Figure 8(A), the bioinformaticist visually discovered four clusters of proteins throughout the GreenMax visualization. One cluster of particular interest showed up in red (lower right of Figure 8(A)), which indicated a large collection of well-connected proteins. This cluster appeared as a



**Figure 9** Layout of graphs showing (A) protein homology network with identified protein classes (high coarsening), (B) protein homology network with identified protein subtypes (light coarsening), and (C) close-up view of two bridge proteins connecting or relating two subtypes.

star-like shape with a dense center, which continued to persist in higher-resolution views through various levels of uncoarsening.

When zooming down into a close-up view of the dense red cluster of Figure 8(B), the bioinformaticist was able to select and distinguish the individual proteins of the cluster. Upon further analysis, the bioinformaticist found that this cluster contained a class of proteins that power the transport of material across the cellular membranes using the energy molecule ATP. These ABC transporter ATPase proteins come in many different types, depending on whether it is a sugar, metal, lipid, or other substrate that is being transported. A prominent visual effect present in Figure 8(B) was the streaking of edges onto the cluster that emanated from specific external nodes. The bioinformaticist saw that these isolated proteins surrounded and constrained the central cluster of ABC transporter ATPase proteins. Upon further examining these isolated proteins, the bioinformaticist discovered that these satellite proteins belonged to the same ABC transporter ATPase class as those in the cluster but were more distantly related. The other protein clusters in the network identified other protein classes and carried properties similar to the cluster of ABC transporter ATPase proteins.

In a second biology example, the same bioinformaticist applied GreenMax to examine a superfamily protein homology network and to decipher the combinatoric

complexity of the regulatory switches that control the production of the cellular machinery.<sup>38</sup> The on/off switches for gene expression are controlled by proteins that bind DNA at specific sites in the chromosome. Sigma factors are a class of control protein typically found with an opposing anti-sigma factor. The exact response in the cell depends on which subtype of sigma/antisigma factor is operating.

Figure 9 shows visualizations of an antisigma factor homology network as produced by a bioinformaticist using GreenMax. From the most coarsened view of Figure 9(A), the bioinformaticist noticed five different protein clusters that emerged from the visualization. These clusters identified five major subtypes among the 1272 antisigma factor proteins. As the bioinformaticist uncoarsened the view of the homology network (Figure 9(B)), she found that the five clusters continued to persist but also gained definition as larger numbers of proteins packed more closely together within clusters. Furthermore, the full network views of Figures 9(A) and (B) both show strong connections among the five subtypes as conveyed by the high density of edges among the clusters. As the bioinformaticist zoomed more closely into the edges using GreenMax, she found that a small number of isolated proteins seemed to serve as conduits among subtypes. For example, Figure 9(C) shows a close-up view of the edges between the two subtypes shown at the top of the

GreenMax visualization. Interestingly, the bioinformaticist found that all the edges between the two subtypes in the close-up view traversed through one of two 'bridge' proteins. The bioinformaticist referred to these conduit proteins as 'bridge proteins,' which exhibit biological features that conform to multiple protein classes. The bioinformaticist was interested in bridge proteins because they conveyed how discrete protein subtypes are related to one another. Furthermore, the identification of bridge proteins extended and improved the analysis – leading the bioinformaticist to query protein databases using search tools such as Basic Local Alignment Search Tool (BLAST) to identify members belonging to the multiple subtypes associated with the bridge proteins. In contrast, a protein search performed from within a single subtype is likely to miss important relationships that exist across subtypes.

In the biology examples above, the bioinformaticist followed a natural path of investigation of looking for global patterns or features and then zooming down into the details of those features. To best support this kind of natural investigation, visualization tools need to provide real-time interactivity in panning, zooming, and other transformations. Of course, such interactivity is difficult to attain when drawing large networks or graphs. In our two examples, the genome network contained 4242 nodes and 33,824 edges, while the homology network contained 1272 nodes and 196,121 edges. Biological networks such as these may easily grow to hundred of thousands nodes and millions of edges. Thus, GreenMax's ability to interactively view and navigate large graphs while also conserving and maintaining global and local patterns and features in its visualizations is critical in the analysis of large biological networks.

Bioinformatic often involves grouping, classifying, and sorting genes or proteins. In the two biology examples, we applied GreenMax for this very purpose. For example, in the visualization of the genome network, we discovered a cluster containing a class of ABC transporter ATPase proteins. Upon zooming into the cluster, we found additional satellite proteins that were distantly related to the class. In the visualization of the homology network, we discovered the five subtypes or groups of the antisigma factor proteins. The GreenMax visualizations provide the visual groupings, patterns, and cues upon which the bioinformaticist investigates how the genes or proteins within an unknown collection hold together or relate to one another. In such cases, GreenMax supports *unknown-known* kinds of discovery where the bioinformaticist is able to build knowledge and attach meaning to unknown features of a graph or visualization. The homology network example also yielded an *unknown-unknown* type of discovery in the finding of isolated proteins that act as conduits among the subtypes. In this case, we did not anticipate the existence of such connecting proteins, whereas we did expect protein groups of unknown contents to emerge in the GreenMax visualizations.

## Strengths and weaknesses discussion

GreenMax has a number of strengths that cannot be found in many other graph drawing tools. First of all, the near-linear multiscale algorithm supported by GreenMax can handle very large sparse graphs with up to 1 million nodes on a modest desktop computer. Once a multilevel layout hierarchy (discussed in the third section) of a graph is established, it can be reused over and over again to interactively support zooming a graph or weighing it with different parameterizations for competing scenario analysis.

The same multilevel layout hierarchy can also be used to improve the graphics performance of GreenMax. Normally when a selected area of a graph is drawn, all the nodes at that level (which may be the entire graph) have to be reiterated to see which ones are in view to be displayed. However, with the multilevel hierarchy at our disposal, it is possible to iterate only through the nodes at a coarser level to find which ones are near the focus window, and then only iterate through the sub-nodes of those nodes to find which ones should be displayed.

Besides supporting GreenMax, the above multilevel layout hierarchy has been shared with a number of other multi-grid graph analytics algorithms within the Have Green framework including the finding of the minimum energy state Fiedler vector<sup>22</sup> for graph partitions.

The case studies in the previous section demonstrate only a rather limited aspect of the GreenMax application. When applied together with the other Have Green<sup>5</sup> tools, GreenMax can be used to cross-interrogate the graphs with the other Have Green tools and cross-validate their analyses and results. In other words, the sum of the parts is always bigger than the whole.

While the multilevel layout hierarchy is very powerful and valuable to the entire Have Green framework, it is also a source of weakness. The 50% reduction rate promised by the coarsening algorithm guarantees to double the data size (because  $1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \dots \cong 2$ ), which certainly is not the most desirable quality for any large data set algorithms.

Another less desirable and yet unavoidable weakness of a multilevel design is layout distortion. When a graph gets coarsened, the structure of it can become distorted. For instance, the center node of a starburst structure may be merged with one of its outlying nodes, causing its center to be moved to one side.

## Conclusion and future work

We improve a conventional static graph drawing technique and turn it into a dynamic multiscale magnifying tool, GreenMax, to interactively explore very large sparse graphs that exhibit small-world properties. Multiple graphs with sizes ranging from a few hundred to 1 million nodes are used to demonstrate the validity of our approach and the results are presented in the paper. We reported the results of two case studies using

GreenMax and the results support our claim that we can use GreenMax to locate totally unexpected features or structures behind a graph.

As for future plans, one potential GreenMax enhancement is to implement the Magic Lens<sup>28,29</sup> concept that allows an analyst to see through the graph nodes and links and access the metadata behind them.

While we do not foresee the need for high-performance computing in our application, we do want to take advantage of the multi-core processor technology that has become increasingly available in the market to further speed up the coarsening process of GreenMax.

## Acknowledgements

This work has been sponsored by the National Visualization and Analytics Center<sup>TM</sup> (NVAC<sup>TM</sup>) located at the Pacific Northwest National Laboratory in Richland, WA. The Pacific Northwest National Laboratory is managed for the U.S. Department of Energy by Battelle Memorial Institute under Contract DE-AC05-76RL01830.

## References

- Milgram S. The small world problem. *Psychology Today* 1967; **2**: 60–67.
- Watts DJ, Strogatz SH. Collective dynamics of small-world networks. *Nature* 1998; **393**: 440–442.
- Di Battista G, Eades P, Tamassia R, Tollis IG. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice-Hall: Englewood Cliffs, NJ, 1999.
- Sugiyama K. *Graph Drawing and Applications*. World Scientific Publishing: Singapore, 2002.
- Wong PC, Chin Jr G, Foote H, Mackey P, Thomas J. Have green—a visual analytics framework for large semantic graphs. *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology (VAST) 2006* (Baltimore, MD), IEEE Computer Society: Piscataway, NJ, 2006; 67–74.
- Kolda T, Brown D, Corones J, Critchlow T, Eliassi-Rad T, Getoor L, Hendrickson B, Kumar V, Lambert D, Matarazzo C, McCurley K, Merrill M, Samatova N, Speck D, Srikant R, Thomas J, Wertheimer M, Wong PC. *Data Sciences Technology for Homeland Security Information Management and Knowledge Discovery*. Report of the DHS Workshop on Data Sciences. Jointly released by Sandia National Laboratories and Lawrence Livermore National Laboratory, Alexandria, VA, 22–23 September 2004.
- Thomas JJ and Cook KA (Eds). *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. IEEE CS Press: Los Alamitos, CA, 2005.
- Pajek, <http://vlado.fmf.uni-lj.si/pub/networks/pajek/> (accessed 29 February 2008).
- Tom Sawyer Software, <http://www.tomsawyer.com/home/index.php> (accessed 29 February 2008).
- Abello J, van Ham F. Matrix Zoom: a visual interface to semi-external graphs. *Proceedings IEEE Symposium on Information Visualization 2004* (Austin, TX), IEEE Computer Society: Los Alamitos, CA, 2004; 180–190.
- Microsoft Magnifier, <http://www.microsoft.com/enable/training/windowsxp/magnifierturnon.aspx> (accessed 29 February 2008).
- Google Earth, <http://earth.google.com> (accessed 29 February 2008).
- Wong PC, Foote H, Chin Jr. G, Mackey P, Perrine K. Graph signatures for visual analytics. *IEEE Transactions on Visualization and Computer Graphics* 2006; **12**: 1399–1413.
- Wong PC, Foote H, Mackey P, Perrine K, Chin Jr. G. Generating graphs for visual analytics through interactive sketching. *IEEE Transactions on Visualization and Computer Graphics* 2006; **12**: 1386–1398.
- Wong PC, Mackey P, Perrine K, Eagan J, Foote H, Thomas J. Dynamic visualization of graphs with extended labels. *Proceedings of the IEEE Symposium on Information Visualization 2005* (Minneapolis, MN), IEEE Computer Society: Washington, DC, 2005; 73–80.
- GD2008. *16th International Symposium on Graph Drawing 2008*, <http://www.gd2008.org/> (accessed 29 February 2008).
- Card SK, Mackinlay JD, Shneiderman B. *Readings in Information Visualization, Using Vision to Think*. Morgan Kaufmann: San Mateo, CA, 1999.
- Chen C. *Information Visualization Beyond the Horizon*. 2nd edn Springer-Verlag: London, 2004.
- Herman I, Melancon G, Marshall MS. Graph visualization and navigation in information visualization: a survey. *IEEE Transactions on Visualization and Computer Graphics* 2000; **6**: 24–43.
- InfoVis 2008. *IEEE Information Visualization Conference 2008*, <http://vis.computer.org/VisWeek2008/infovis/index.html> (accessed 29 February 2008).
- Shneiderman B. Network visualization by semantic substrates. *IEEE Transactions on Visualization and Computer Graphics* 2006; **12**: 733–740.
- Aris A, Shneiderman B. Designing semantic substrates for visual network exploration. *Information Visualization* 2007; **4**: 1–20.
- Harel D, Koren Y. A fast multi-scale algorithm for drawing large graphs. *Journal of Graph Algorithms and Applications* 2003; **6**: 179–202.
- Walshaw C. A multilevel algorithm for forced-directed graph-drawing. *Journal of Graph Algorithms and Applications* 2003; **7**: 253–285.
- Barnard ST, Simon HD. A fast multilevel implementation of recursive spectral bisection for partitioning unstructured problems. *Concurrency: Practice and Experience* 1994; **6**: 101–107.
- Tulip, Tulip-Software.org, <http://www.labri.fr/perso/auber/projects/tulip/> (accessed 29 February 2008).
- Keahey A, Robertson EL. Nonlinear magnification fields. *Proceedings IEEE Symposium on Information Visualization 98*, 1998 (Research Triangle Park, NC), IEEE CS Press: Los Alamitos, CA, 1998; 51–58.
- Bier EA, Stone MC, Pier K, Buxton W, DeRose TD. Toolglass and magic lenses: the see-through interface. *Proceedings of Siggraph '93, 1993 (Anaheim, August)*, *Computer Graphics Annual Conference Series*, ACM: New York, 1993; 73–80.
- Stone MC, Fishkin K, Bier EA. The movable filter as a user interface tool. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '94) 1994* (Boston, MA), ACM Press: New York, NY, 1994; 306–312.
- Analyst's Notebook, <http://www.i2inc.com> (accessed 29 February 2008).
- Ellis G, Dix A. The Plot, the Clutter, the Sampling and its Lens: Occlusion Measures for Automatic Clutter Reduction. *AVI '06, 2006* (Venice), ACM: New York, 2006; 266–269.
- Wong N, Carpendale MST, Greenberg S. EdgeLens: an interactive method for managing congestion in graphs. *Proceedings IEEE Symposium on Information Visualization 2003* (Seattle, WA), IEEE Computer Society Press: Los Alamitos, CA, 2003; 51–58.
- Fruchterman TMJ, Reingold EM. Graph drawing by force-directed placement, software. *Practice & Experience* 1991; **21**: 1129–1164.
- Kamada T, Kawai S. An algorithm for drawing general undirected graphs. *Information Processing Letters*, 1989; **31**: 7–15.
- Yin R. *Application of Case Study Research*. Sage Publishing: Newbury Park, CA, 1993.
- Lesne A. Complex network: from graph theory to biology. *Letters in Mathematical Physics* 2006; **78**: 235–262.
- Shannon P, Markiel A, Ozier O, Baliga NS, Wang JT, Ramage D, Amin N, Schwikowski B, Ideker T. Cytoscape: a software environment for integrated models in biomolecular interaction networks. *Genome Research* 2003; **13**: 2498–2504.
- Medini D, Covacci A, Donati C. Protein homology network families reveal step-wise diversification of type III and type IV secretion systems. *PLOS Computational Biology* 2006; **2**: 1543–1551.