Conceptualizing Bandwidth Allocation in Network Management

George Melissargos
Ergonomics of Intelligent Systems & Design
ISR / DMT
Swiss Federal Institute of Technology
CH-1015 Lausanne, Switzerland
+41-21-6933081
George.Melissargos@epfl.ch

Pearl Pu
Ergonomics of Intelligent Systems & Design
ISR / DMT
Swiss Federal Institute of Technology
CH-1015 Lausanne, Switzerland
+41-21-6936081
Pearl.Pu@epfl.ch

ABSTRACT

In our work, we focus on how resource allocation can benefit from visualization. In this paper, we present the main visualization techniques used in a bandwidth allocation system for the management of circuit-switched networks.

Keywords

Resource allocation, interactive visualization, network management.

1. INTRODUCTION

Resource allocation problems arise in many different forms in everyday life operations. Characteristic examples are found in the application areas of personnel allocation, flight scheduling, room reservation, train scheduling, production planning, freight transportation and finally the focus of our interest, network configuration management. Managing a network involves such functions as routing, definition, monitoring and planning. This paper concentrates on bandwidth allocation, a resource allocation problem we meet in all these functions. An interactive visualization tool, called optiNet, has been implemented for aiding network operators in their bandwidth allocation tasks.

2. Modeling the Problem

At first, a model defining the network is required. According to the model, a network consists of nodes, links and demands. The nodes are identified by unique names. Each node has a geographical position defined by a pair of coordinates. Although the current implementation of the system does not use nodes of different types, the model supports such ability. The links are bi-directional and like the nodes, uniquely identified. A link is defined by the two end nodes it connects. The unique identification is necessary because more than one links may connect the same two nodes. A link has a certain transport capacity. Some of the capacity can be assigned for carrying out transmission tasks, thus leaving the rest free. The third type of network element is the demand. A demand is a transport task to be performed by the network. It is a request for ensuring a path of some bandwidth between two nodes of the network. The path may consist of any links creating a connection between the two nodes and it must be acyclic. All links along the path must guarantee the allocation of a portion of their capacity equal to the demanded bandwidth. It is important to note that the demands are indivisible. A demand is a task that has to be carried out in one piece. No time division or resource division is permitted. There are two kinds of demands, the allocated and unallocated ones. An allocated demand has an additional characteristic, the path of links that is routed through. So the basic network elements and their attributes are:

- Node: name, position(x,y), type
- Link: name, node A, node B, total capacity, load
- Unallocated Demand: name, node I, node J, bandwidth
- Allocated Demand : name, node I, node J, bandwidth, connection path.

After modeling the network elements, the problem is expressed:

Given a network of nodes, links and demands, the goal is to find routes for all the demands, observing the capacity limitations of the links and within a common time frame.

So, in its simplest form, the bandwidth allocation problem is a routing task. Effective routing is the center of all operations in network management. If an operation does not handle or affect routing directly, it affects other characteristics that influence the network routing policies. For example, the great majority of today's network management systems concentrate on network monitoring. During real time monitoring, the human operator is notified of hardware failures that may affect the overall network performance. Problematic network behaviors lead to unstable conditions that endanger the actual or any future routing requests. So, at the bottom of any network operation there is one objective: assure the satisfaction of the demands.

3. Conceptualization Model

Problem solving in resource allocation is divided in three stages, namely preprocessing, solving and postprocessing, [1]. In bandwidth allocation the three stages correspond to three distinct modes of operation we call "Edit", "Solve" and "Inspect". Each

mode is characterized by a reference model, similar to the one presented at [2]. In its turn, each reference model comprises of a task list, a set of user interactions and a visualization pipeline. The user has, with the help of the system, to understand the network state, formulate the allocation problem and finally solve it. He can freely switch between any of the modes until he succeeds solving the allocation problem in hand. The overall model, depicted in figure 1, is called the conceptualization model. It help us describe the process of using interactive visualization to perform resource allocation tasks.

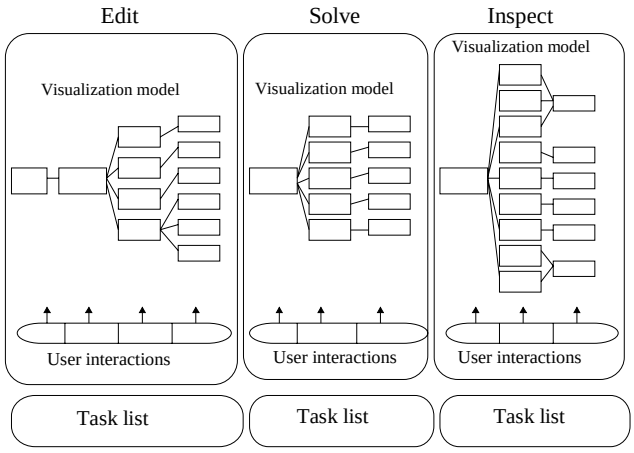

Figure 1 : Conceptualization Model

3.1 Edit Mode

The Edit mode is used to define and build the network. Each network element is read by a database or entered directly by the user. The system is able to convert the database contents into meaningful visual representations and vice versa, to codify the user's visual based interactions in computer stored information.

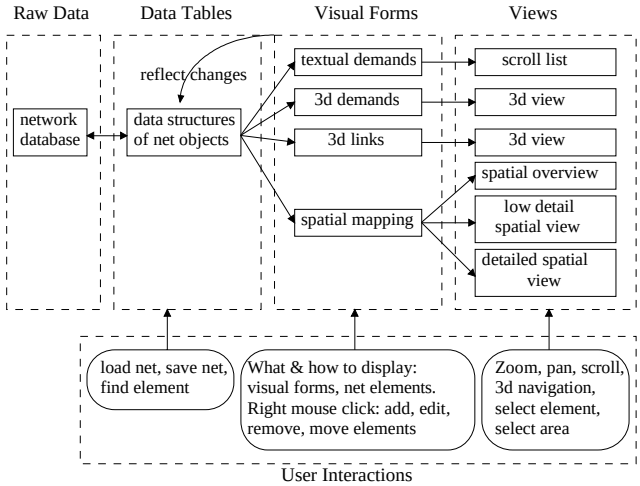

Figure 2 : Edit Mode Model

The main tasks to be executed during this mode of operation are:

- Visualize the network layout
- Add, remove nodes, links and demands
- Edit characteristics of nodes, links and demands (name, position, type, capacity, bandwidth)
- Load, save a network configuration

The diagram in figure 2 illustrates the details of the conceptualization model for the "Edit" mode.

3.1.1 Spatial Visualization

The most natural representation of a network is based on the spatial layout. A map of the geographical area covered by the network is first drawn in the background. The map is of very low detail, just a simple outline. Then the nodes are placed according to their geographic coordinates. It is important to select appropriate shapes to depict the network elements. There are two guidelines to be followed. The first is the relevant standard Z.361 set by the "Telecommunication Standardization Sector of the International Telecommunication Union" [3]. The second guideline is to keep the design of the system within the already established industry norms. A series of commercial network management tools we examined constituted a valuable source of input for the design decisions we made.

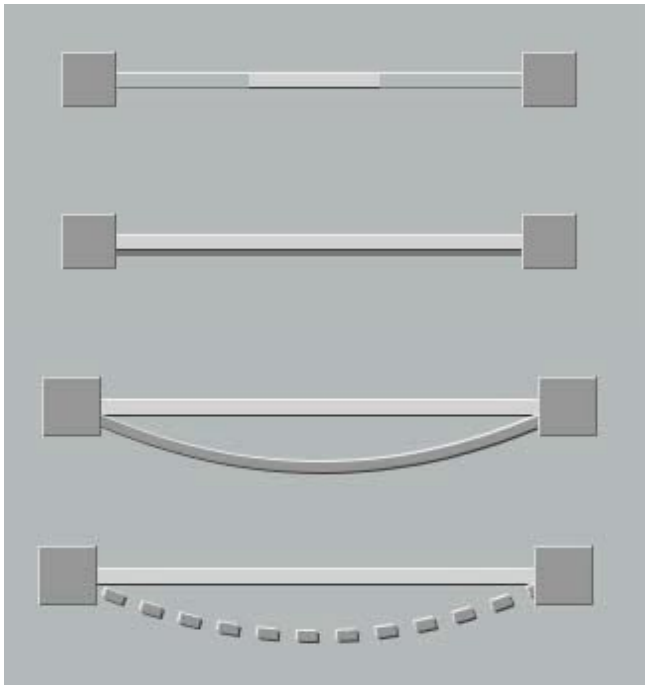

Figure 3 : Representation of Basic Network Elements

The generic representation of the node is a square with 3D-like raised relief. If the node is of a specific type, for example an exchange, then the appropriate graphic is drawn within the rectangular area. All the graphic symbols representing the network elements have a raised relief so that they give out the effect of a real life object. The user is invited to interact with a real like object. This way, when problem solving, he can better employ the perception and cognition skills he is normally using in his interactions in the real world. A link is depicted by a straight line of certain width and of raised relief. It is colored in light gray. There are cases that two nodes are connected by multiple links. To avoid screen clutter, only one link is drawn but with an additional shadow. The shadow implies that more links are stack underneath.

This invites the user to click the mouse on the link. Then, a pop up menu lists in textual form all the links. The one selected comes in the top of the stack. There is also the option to visualize the current load (allocated, free capacity) by coloring the link as shown in figure 3. A dark gray color fills the link proportionally to the amount of the total capacity allocated. So, the portion left in light gray indicates the remaining capacity of the link. The coloring starts simultaneously from both sides so that it doesn't give to the link the look of any particular direction. The links have no specific direction, thus they should not give the impression of one. The demands are depicted as arcs connecting two nodes. There is again a raised relief and an additional shadow in case of multiple demands between the same two nodes. The unallocated demands are drawn by dashed arcs. This gives immediately the impression of incompleteness. The unallocated demands are incomplete in the sense that they are not yet routed. The user must be fully aware of this fact when browsing the demands. Figure 3 depicts the graphic representation of the basic network elements while figure 4 shows an example network as visualized in optiNet.

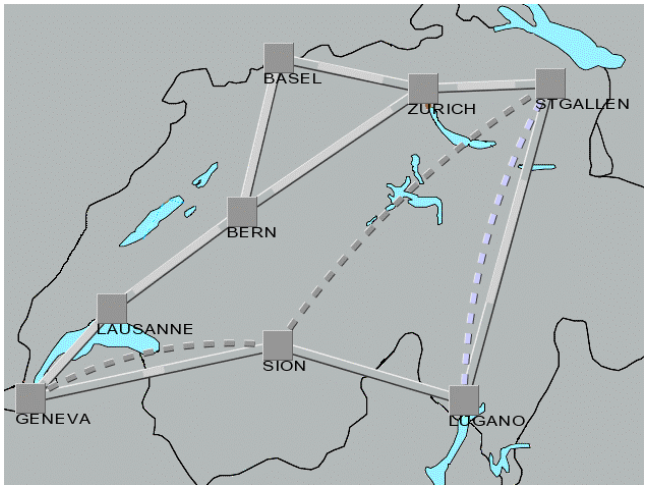

Figure 4 : Spatial Visualization of a Network

3.1.2 Ergonomics of interactions

The graphical user interface of the system, as seen in figure 5, is separated in five parts. The upper part hosts the menu bar. The bottom strip is used as a message banner. On the left side there is the toolbar. The rest of the GUI, split in four areas, is the visualization display space. The fifth element of the GUI is not shown in this figure, it is a virtual element. It groups together the pop-up windows of the application. The user can interact with any of the menubar, toolbar or the graphical objects found in the visualization space. All three areas are context sensitive. The interactions available with the menubar and the toolbar depend on the mode of operation. In "Edit" mode for example, the menu option "Edit" is available. In "Inspect" and "Solve" modes it is disabled, thus the user can not add, remove or change any of the network elements. The graphical objects in the visualization space can be selected by clicking the left mouse button on them. The right mouse button is reserved for object based functions. Every graphical object has a set of functions associated with it. The functions become available through corresponding pop-up menus and they depend on the current mode of operation. The pop-up menus are invoked by pressing the right mouse button on the graphical object. For example, as shown in figure 5, if the right mouse button is pressed upon a node, a pop-up menu appears. The options of the menu are for moving, deleting, renaming and changing the type of the node. These are actions directly affecting the state of the node. The two other options available, adding a demand and a link, don't manipulate the node. They use the node as the starting point for defining a new demand or link. Direct manipulation and querying of visual objects and forms, [4], is centric to the design of OptiNet.

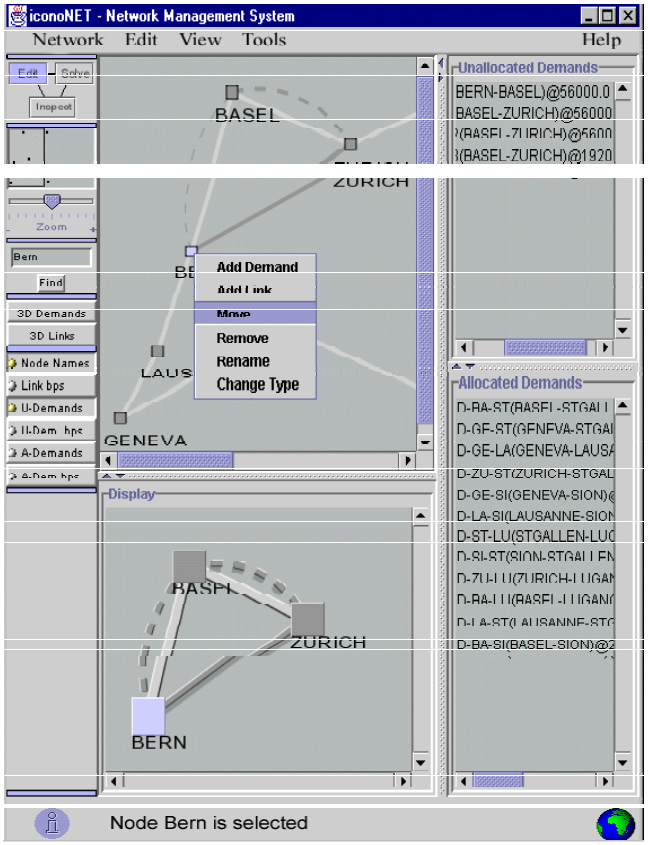

Figure 5 : GUI of the OptiNet tool

3.1.3 Level-of-Detail

The spatial visualization method works very well for small networks. Rendering the above mentioned objects in large numbers becomes painfully slow. For more than 15000 network elements the application becomes practically unusable. Even if a comfortable rendering speed of so many elements could be achieved, it would be impossible to display them in the limited size of a screen. The end result would be a confusing mass of graphical objects. A way to overcome this situation, is to offer multiple network views of varying detail. The network management tool we constructed offers an integrated "level of detail" visualization trio. First there is the small panning area in the toolbar (figure 6, part 1). It is a condensed overview of the network topology. The rectangle in that area represents a viewing window into the network. Dragging around the small rectangle causes the network main view, on the immediate right hand side, to move across the network.

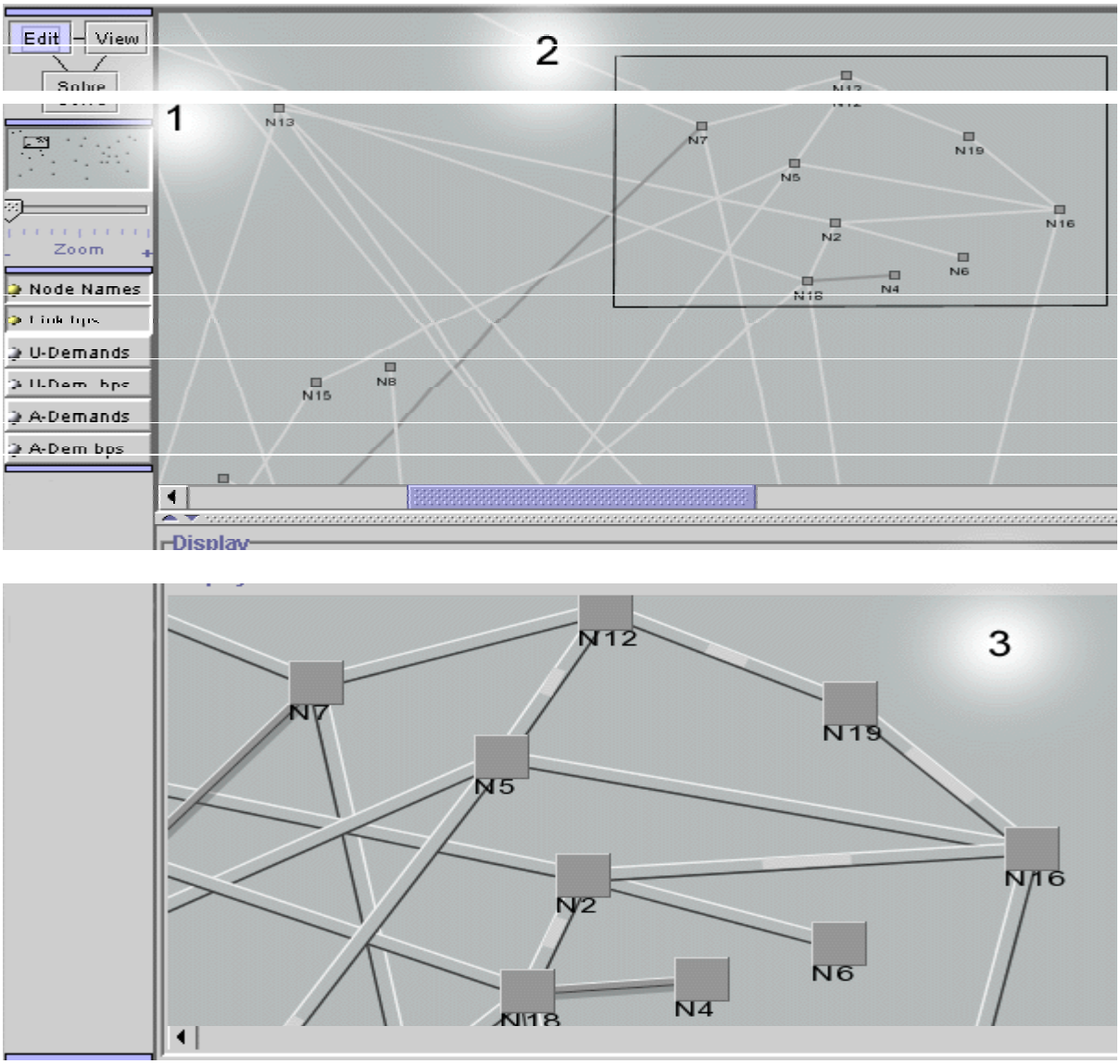

Figure 6 : Level of Detail View

The main view (figure 6, part 2) utilizes a simple visualization method for rendering the network. By dragging the left mouse, a corresponding rectangular area is marked and drawn. Whichever portion of the network is within this marked area appears on the bottom display area (figure 6, part 3). In this display, a more sophisticated method is used for rendering the network elements in the highest possible detail.

Another way to reduce screen clutter is to avoid rendering the demands, especially when knowing that they usually constitute more than 80% of the total number of network objects. Instead, there are two textual scroll lists that can be used for viewing and selecting the demands. If the spatial congestion persists, special layout algorithms can be called upon to reorder the nodes. This technique though, has two disadvantages. The real world geographical mapping is lost and the automatic layout algorithms are computationally expensive and don't perform satisfactorily for all kinds of layouts. As a last resort the layout can be adjusted manually by dragging the nodes around.

3.1.4 Matrix Visualization

In the previous sections, it became apparent that the spatial based visualization of the network has some limitations. In certain cases, it can lead to non uniformly or too densely populated displays; screens that are cluttered with difficult to perceive and understand information.

By observing the network model, the first remark that can be made is that the nodes define the rest of the network elements. At the same time, for the majority of networks, the number of nodes is much smaller to the number of links or demands. Conceptually, both the links and the demands can be expressed as connections between two nodes. Given n nodes, there are n2 combinations of nodes. Since the links and the demands have no direction, a [nodeA, nodeB] combination is equivalent to a [nodeB, nodeA]

one. Finally, loop connections [nodeA, nodeA] are not permitted. So, the number of valid combinations of nodes reduces to $(n^2 - n)/2$. Next step is to create a table that can express these combinations in a compact way. Lets us assume that there are 8 nodes called A, B, C, D, E, F, G, H. The matrix in figures 7, 8, 9 and 10 is sufficient for expressing all of their valid combinations. The matrix cells may host either links or demands. The columns are labeled by listing the names of the nodes, except the last one. Currently, there is no order in the list. An interesting ordering would be spatial proximity of the nodes. The rows of the matrix are labeled with the node names but this time, in exact reverse order of that of the column labels. The matrix builds on a similar

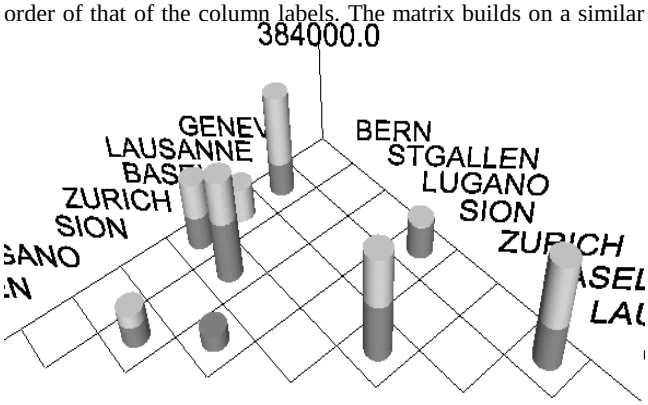

Figure 7 : 3D Matrix of Links

There are several alternative ways for visualizing the links and demands matrices. The 3D representations, explained here, are implemented in the OptiNet tool. Figure 7 shows the 3D visualization of links. The cylinders represent the links. The height of the cylinder is proportional to the capacity of the link. The load ratio of the link is encoded in the picture by the use of color. The darker portion of the cylinder corresponds to the amount of capacity already allocated to demands. The lighter part indicates how much capacity is still free. In case of multiple links between two nodes, the cylinders are simply placed on top of each other in the appropriate cell. The number in figure 7 is the maximum link capacity and corresponds to the higher cylinder. It gives to the user a notion of the overall scale in terms of capacity.

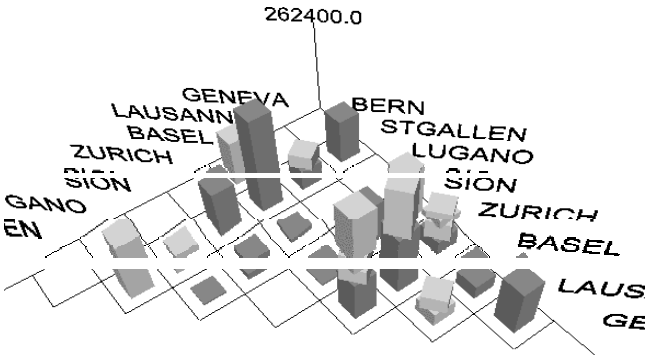

Figure 8 : 3D Matrix of Demands

The 3D visualization of demands is very similar to the one of links. The geometrical object chosen to represent a demand is the parallelepiped. Although the cylinder could have been sufficient for representing the demands, it was rejected for clearly distinguishing the demands from the links. The goal is to have an association of network objects to physical geometrical objects as distinct as possible. This way, there is no confusion about what is represented by what. The mental mapping of visual forms and their geometrical characteristics back to the abstract data they represent is automatic. Working with cubic objects is working with demands, working with cylinders is working with links. Figure 8 shows the 3D matrix of demands. The dark colored parallelepiped symbolize the already allocated demands. The lighter in color objects stand for the unrouted demands. The height of the demand-object is equivalent to the bandwidth requirement of the demand. In case of multiple demands between two nodes, the demand-objects are placed one on top of each other. For reasons of easier identification and selection, every other demand of the heap is rotated by 45 degrees.

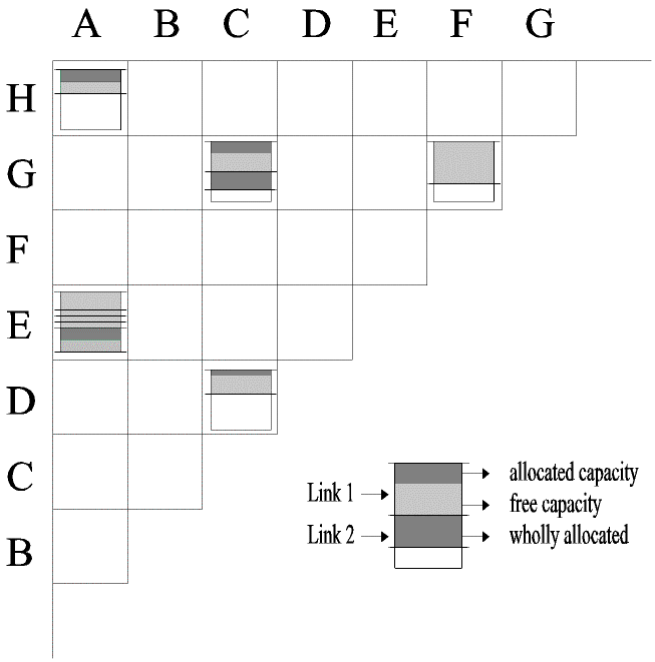

Figure 9 : 2D Matrix of Links

The same matrices can also be visualized in two dimensions. In figure 9, the links are represented by rectangles. The lines extending horizontally out of the rectangles mark the beginning and end of each link. The dark and bright colors indicate, respectively, the amount of allocated and free capacity of the link. For example, the cell [F,G] has only one link which is completely free for allocation. The cell [C,G] hosts two links, the one on the top semi allocated, the one on the bottom wholly allocated. The matrix cell that hosts the link(s) with the maximum overall capacity is the one defining the size scale of the drawn rectangles. The link-rectangles in that cell, [A,E] in the example figure, must fill up the whole available area within the cell. Figure 10 illustrates the 2D visualization method for the matrix of demands. The demands are represented by pie shapes. A pie sector in light color symbolizes an unrouted demand. A sector in dark color maps out a satisfied demand. The angle of the radii defining the sector is proportional to the demand's bandwidth. The circle that is totally filled up with demands is the one defining the size scaling of all the demand-sectors.

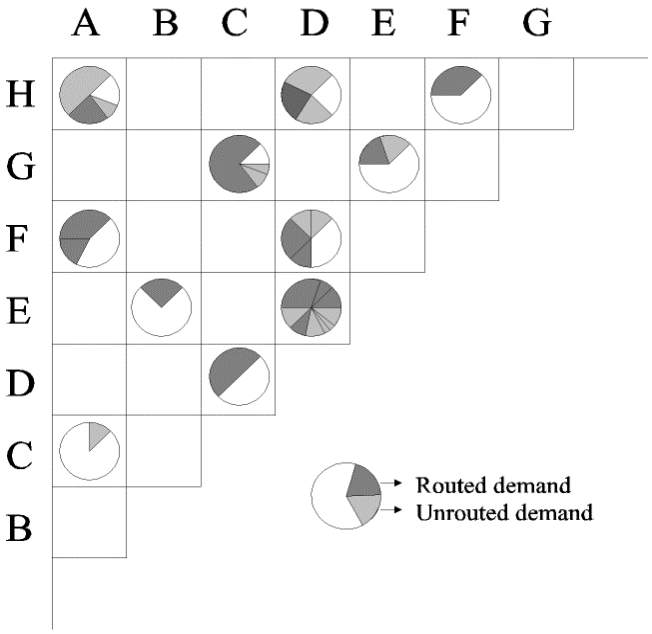

Figure 10 : 2D Matrix of Demands

3.2 Solve Mode

In the "Solve" mode the user tries to solve the allocation problem in hand. The conceptualization reference model of this mode is described in figure 11. The tasks that can be performed are:

- Visualize the network elements and their attributes
- Route one or multiple demands by hand
- Route one or multiple demands automatically
- Select routing algorithm
- Deallocate one or multiple demands

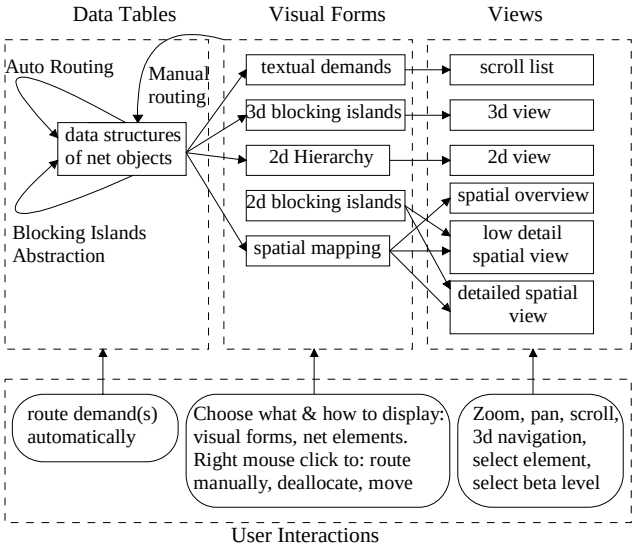

Figure 11 : Solve Mode

It is out of the scope of this paper to fully describe the functionality and the respective visualizations and interactions of this mode of operation. Instead, we will concentrate on the Blocking Islands (BI) abstraction, presented at [6]. The BI abstraction is based on a novel algorithm that segments the network in clusters of connectivity. Given a bandwidth, let us say of 64KB, the algorithm breaks the network in blocking segments of 64KB. There is no way to establish a connection of 64KB, or bigger, between two nodes belonging in two different segments. On the contrary, leaving any node we can reach any other node within the same segment, through a path of links of at least 64KB free capacity. The algorithm calculates the blocking islands for several critical bandwidth levels.

So, if one tries to manually route a demand of certain bandwidth between two nodes, he first has to check the BI segmentation at that bandwidth level. If the two nodes do not belong in the same segment, there is no route guaranteeing the demanded bandwidth. The biggest challenge is to visualize the BI abstraction. For the 2D spatial representation of the network, the solution is obvious. Each blocking island is defined by a group of nodes. The tool draws the outline of the area enclosing all these nodes. In the current implementation, we use the "Graham Scan" algorithm for calculating the convex hull of the nodes. In the case of one or two node islands, ellipses suffice as enclosing areas. The result can been seen in figure 12.

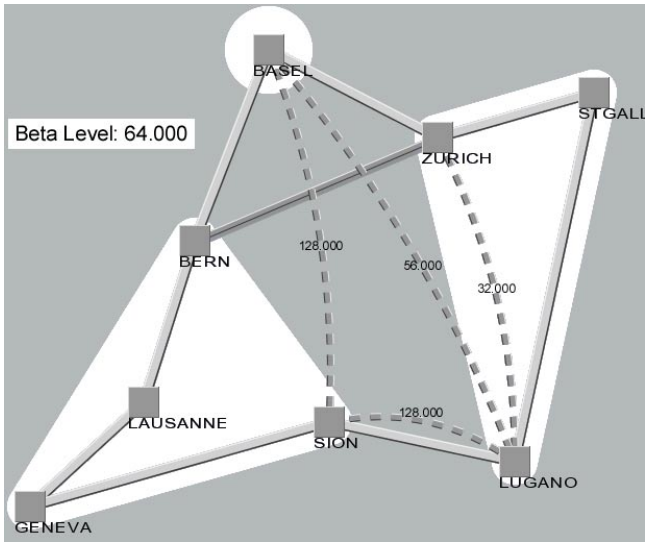

Figure 12 : Spatial Representation of the Blocking Islands

There is a weakness with this approach. If the network is densely populated with nodes, we may end up with polygons overlapping each other. The same may happen even with just two blocking islands. If one of the nodes of the first island is surrounded by the nodes of the second island, then their respective polygons overlap. There are a few remedies for this kind of problem. For example, the color of the polygons can be transparent. So, when parts of segments collide, it becomes immediately apparent. If the user wishes to remove the collision, he can move some of the nodes around with the mouse. As the nodes move, the blocking islands are continuously redrawn. The Graham-Scan algorithm is fast enough to support the continuous display renderings. Another way is to try to diminish the surface of the segments. A good candidate is a heuristic based on the use of minimum spanning trees. The heuristic first calculates the minimum spanning tree connecting the nodes of a blocking island. It then walks along the

calculated path, drawing a thick line that encloses the nodes. The final result looks like a snake line. This technique reduces collisions but does not totally eliminate them. There are more complex layout algorithms from Graph Drawing theory but it is out of the scope of this work to review them all. Besides, any of these algorithms will fail for extreme network configurations.

3.3 Inspect Mode

The outcome of the "Solve" mode can be either a successful bandwidth allocation or a routing deadlock. In both cases, the user will enter the "Inspect" mode. If the allocation problem was not solved, he will attempt to identify possible network bottlenecks. If, on the other hand, there was a solution, he may want to check the status of the network after the new allocations took place and make sure that the network is configured in a balanced manner. So, the "Inspect" mode of operation can be used for finding current network anomalies or for identifying load patterns that may create problems in the future. The second function is a basic ingredient of network planning tasks. By using the "Edit' and "Inspect" mode the network planner can decide on configurations that accommodate future load patterns. He can also ameliorate the fault tolerance of the network against element failures.

The main tasks of the "Inspect" mode are:

- Analyze the current state of network elements (links: load ratio, demands routed through; demands: allocation paths, bandwidth)
- Visualize overall state of the network (statistics, blocking islands hierarchy)
- Identify any connectivity bottlenecks
- Project the network status if links or nodes breakdown
- Explain reasons for routing failures of "Solve" mode

Here, we extend our discussion on the Blocking Islands abstraction. We review its use as an aid for balancing the network, according to the current or projected load patterns. The BI algorithm partitions the network in blocks of connectivity depending on certain predefined bandwidth levels. For each of these levels, called thereafter beta levels, there is a new pattern of segmentation. At level 0 all the network nodes belong into the same big block. As the beta level increases, the network is broken in a bigger number of smaller segments. After a certain bandwidth, that of the link with the largest free capacity, the network will have been partitioned down to node-blocks.

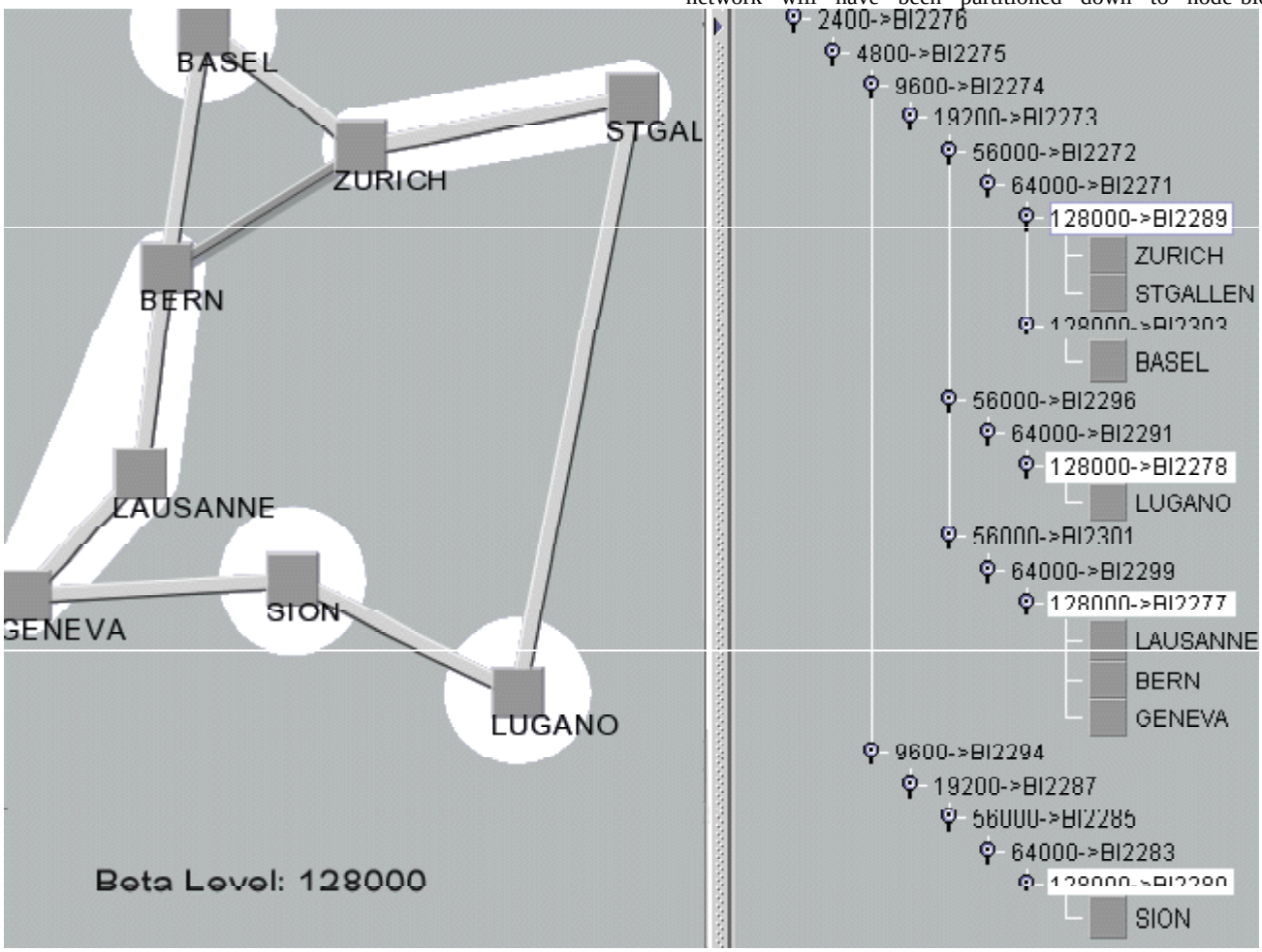

Figure 13 : Spatial Representation of the Blocking Islands Hierarchy

An important characteristic of the overall segmentation, across all beta levels, is the fact that it translates into a tree structure. Starting with the root at 0, it continues branching out as a tree. The parents separate into one or more children blocks. The leafs of the tree match the nodes of the network. A natural depiction of this hierarchical structure, called the Blocking Islands Hierarchy (BIH), is that of a 2D drawing of a tree. Each level of the tree corresponds to a beta level. Figure 13 illustrates the 2D representation of the BIH. When the user clicks the mouse on a level of the tree, the blocking islands in that level will be drawn into the display area hosting the spatial representation of the network. By navigating up and down the tree, the user can visualize the state, as far as the connectivity is concerned, of the network. In this manner, the human operator can quickly tell at which beta level and at which part of the topology, the network becomes dangerously congested. When the bottleneck concerns high traffic links, immediate action has to be taken. If on the other hand two nodes, which usually have low traffic patterns, belong in different blocking islands and the beta level is high, the operator can safely ignore it and take no action.

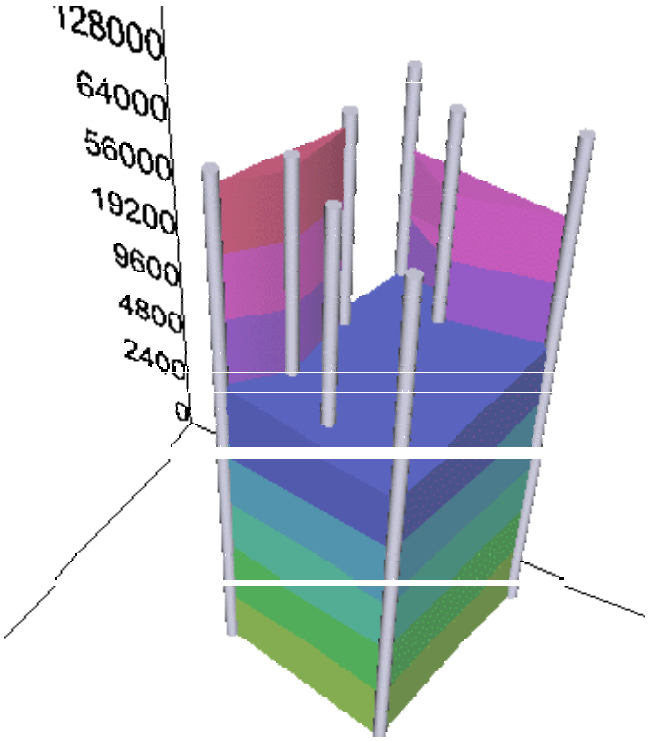

Figure 14 : Visualizing the BIH as a Solid Object

The BIH can be also visualized in three dimensions. Figure 14 depicts the results of a simple method for turning the BIH abstraction into the visual form of a solid. We first render the blocking islands of each beta level like with the spatial representation. Each beta level occupies a plane parallel to the XZ plane in space. The beta level 0 actually sits on the XZ plane. At each plane the blocking islands are rendered as polygons (corresponding to convex hulls) and the nodes as circles. At the end, every level is connected with its neighbors by projecting the polygons to the respective planes. The same is done for the nodes; thus they look like cylinders extending from the first to the last plane. The main advantage of this approach is the fact that there is no need for displaying the tree of the BIH. Experience has shown us that quite often people find it difficult to "read" a tree structure. This is especially true for people that do not have a formal engineering education, something common among operators. The main problem of this visualization is that it inherits the limitations of the spatial representation of blocking islands, described earlier. To overcome this difficulty we can use the 3D visualization of the links matrix, at figure 7. We first choose one of the two label axis of the matrix, where the node names are put. There, we create a new row of cells, the BI cells Then, we order the node names according to their position as leafs in the BIH. This way, nodes that belong to the same island will always be adjacent, no matter the beta level. The next step is to start rendering cubes inside the cells. The cubes that belong to the same blocking islands are extended sideways until they meet each other. Since the first row represents the beta level 0, it always touches the XZ plane and it has cubes that are all connected to each other. For each beta level we add a new row of cubes parallel to the first one and in the Y direction. The final result looks like an incomplete wall of bricks. The wall visualization doesn't carry any of the problems of the spatial representations but it does have a drawback. The row of node names, thus the whole matrix, has to be reordered each time the BIH changes. Rearranging the matrix very often results in unnecessary information flow from the computer to the user, thus, it may lessen the effectiveness of the visualization.

4. REFERENCES

[1] Melissargos, G. and Pearl, P. Interactive Visualization for Resource Allocation Systems. Visual Data Exploration and Analysis IV Conference at the IS&T/SPIE Symposium on Electronic Imaging '97. San Jose, California, February 1997.

[2] Card, S. and Mackinlay, J. and Shneiderman B. Readings in Information Visualization, Using Vision to Think. Morgan Kaufmann Publishers Inc, San Fransisco CA, 1999.

[3] Telecommunication Standardization Sector of the International Telecommunication Union (ITU-T). Recommendation Z.361, Design guidelines for Human-Computer Interfaces (HCI) for the management of telecommunications networks, February 1999.

[4] Kumar, H. and Plaisant, C. and Teittinen, M. and Shneiderman, B. Visual Information Management for Network Configuration. Technical Report CS-TR-3288, University of Maryland, College Park, June 1994.

[5] Becker, R. A. and Eick, S. G. and Wilks, A. R. Visualizing Network Data. IEEE Transactions on Visualization and Computer Graphics, 1995. (1) 16-28.

[6] Frei, C. and Faltings, B. A Dynamic Hierarchy of Intelligent Agents for Network Management. Proceedings of the Second International Workshop on Intelligent Agents for Telecom Applications IATA'98. Paris, France, 1998.