Visualization Enabled Mixed Initiative Systems

Pearl Pu

Human Computer Interaction Group
Institute of Core Computing Science
Swiss Institute of Technology (EPFL)
IN-Ecublens, 1015 Lausanne
Switzerland
pearl.pu@epfl.ch

Abstract

Mixed-initiative systems (MIS) are interactive problem solvers where human and machine intelligence are combined for their respective superiority. A flexible model of interaction is employed and each problem-solving agent (human or machine) contributes to the tasks what it does best, in the best opportune task context. Several interaction models have been proposed, such as natural language dialogs, direct manipulation interfaces, and interactive intelligent agents. However, there has been little work on MIS using visual representations as an interaction medium, contrarily to how ubiquitously humans use and benefit from diagrams while solving problems. We outline in this paper the challenges facing the design of visual representations in such MIS and provide a case study in resource management tasks. Our main conclusion is that to achieve effective visual representations so that users can both easily perform tasks and perceive the results of intelligent agents, analyses of the underlying reasoning tasks must be carefully carried out to identify and map cognitive tasks to perceptual tasks, and user studies are needed to find the optimal visual rendering of information needed for perceptual tasks. In scaling the visual representation to support more complex user tasks, we observe that users switch to a different mode of problem solving by relying more on capabilities of intelligent agents. Different visualizations are thus required for the effective support of these two different task scenarios.

Keywords

Mixed initiative interaction, visualization, visual representation, resource management, scheduling, resource allocation

1. Introduction

Mixed-initiative systems (MIS) are interactive problem solvers where human and machine intelligence are combined for their respective superiority [6]. The concept has been successfully applied to a number of domains: planning [5], interactive search [2], configuration design [12], and etc. MIS (or sometimes called human-involved intelligent systems) are becoming popular as an alternative to automatic intelligent systems for the following reasons:

1. Human strategy: when a MIS externalizes the reasoning process interactively, human problem solvers are able to influence the reasoning strategy in case they do not agree with that being currently employed by the machine.
2. Human criteria: the criteria for selecting optimal solutions are generally under-specified and impossible to model in the system. Humans are able to detect the changing context and are good at finding the right context to apply optimization criteria.
3. Human choices: in situations where humans not only want to say yes or no to a proposition of solution, but also desire to solve a part or an entire problem instance, they are able to intervene and override the machines' solutions.
4. Sense of control: last but not the least reason why many prefer MIS is that humans like to be in control of making decisions and choices of problem solving paths.

A flexible model of interaction, that is also convenient to human problem solving styles, is key to integrating results from each problem-solving agent (human or machine) for the tasks that it does best, in the best opportune task context. The earliest interaction models are based on natural language dialogs [5,6], using visual representations only to display results and update problem solving status. Alternatives to dialogs are direct manipulation interfaces, which have been found to be more comprehensible, predictable and controllable [15], and are less costly to construct than dialog systems. Some even propose that that the concept of mixed initiative systems, which advocates for a computational environment where human and machines augment each other's intelligence, should serve as future models for human computer interaction [8].

We observe that humans ubiquitouslyuse diagrams when solving problems, either alone or collaboratively. When solving physics or geometry problems, for example, we use diagrams as external representations, marking and labeling relevant information on the diagram before attacking the problem. Sometimes an addition of a line in a diagram could even make the solution apparent. However, there has been relatively little work on the utility of visual representation as an interaction medium for MIS besides using it as output devices. The human visual system is known for its remarkable ability in perceiving large quantities of data, detecting trends, and spotting evidences. Over many years of research, scientists from information visualization have developed powerful visualization techniques useful for representing data and their relationships [3]. While there is no doubt visualization provides a high bandwidth for human machine communication, and is capable of representing meaning relationships between data, we are interested in knowing whether it can also make human problem solving (reasoning) easier? If so, such visualization enabled mixed initiative systems hold significant promises for future intelligent systems because they provide perceptual and reasoning advantages, as well as direct manipulation interface benefits.

According to cognitive science studies on the role of diagrams in problem solving [9], diagrams help us group information needed for problem solving in the same area, thus making data retrieval efficient during the solving phase. Furthermore, when problem solving requires several steps, the order of data retrieval in diagrams corresponds to the reasoning steps. Later on, scientists were able to show that certain externalizations are not only temporary storages for reasoning, but also intrinsic to human thinking [18]. Unfortunately, previous works on design and automation of graphical presentations indicate that not all diagrams are effective for problem solving [9,10]. In fact, according to Casner [4] and Tweedie [17], different user tasks require different visual representations, and diagram efficiency depends highly on a mapping between visual representation and the tasks that it supports. This has left the burden of constructing effective visual representations to the MIS designers. In particular, for a given domain of problems, important analytical work is necessary to understand user tasks (task analysis), experiment with several mappings from tasks to visual representations, and establish design principles.

We provide a case study of a suite of visualization tools for resource management systems based on mixed-initiative interaction. We first identify the set of resource types, analyze user resource management tasks, discuss effective visual models capable of reducing problem solving complexities, and describe two applications that we built based on these visual models.

2. Resource Types and Management Tasks

A resource is a supply of equipment, fund, personnel, or information to satisfy a demand. The consumers of a resource can be a person, an organization, or an operation that will fulfill its functional requirement by the utilization of the provided resources. Four types of resources based on the DARPA/Rome laboratory report [1] are distinguished:

1. Consumable - some resource types (e.g., money, fuel) are gone once they are allocated to consumers, and resource capacity can only be increased through subsequent production of new capacity, or releasing (e.g., occupied digital space becomes free).
2. Reusable/non-sharable (schedulable resources) -- those that are unavailable only while being used by a particular consumer and are otherwise available. These resources are non-divisible and their capacity must be wholly allocated to a single consumer, e.g., an airplane can be assigned to one flight route. Because the only way to share these resources is through a scheduling effort, we call them schedulable resources.
3. Reusable/synchronized-sharable (allocable resources) --- these resources can be simultaneously allocated to multiple consumers but only in a temporally synchronized manner, e.g., communication bandwidths. Since allocation is a way to share these resources, we call them allocable resources in this paper.
4. Reusable/independently sharable --- these resources can be simultaneously allocated to multiple consumers and used without consumer synchronization. Most information resources fall in this category.

A review of effective visualization techniques for representing and managing information resources can be found in [14]. We are mainly concerned with the first three types of resources, although our notion of consumable resources is about their temporal occupancy rather than consumption. Some resources such as aircraft, trains, or public facilities (e.g., rooms, stadiums, and cinemas) can be modeled at different levels of detail. An aircraft, for example, to a flight route is a schedulable resource, but an allocable resource to individual passengers. Similarly, a meeting room is schedulable for groups of people, but allocable to individuals.

Examples of resource management tasks include scheduling airplanes to designated travel routes in an airline company, the allocation of personnel to a set of tasks in a hospital, or managing computer equipment in our working environment. Poorly managed resources will lead to critical consequences, such as an emergency room unattended in a hospital, bottlenecks of overly used computer equipment, or the inability to find an aircraft for a booked flight. Monitoring resource capacities is crucial to both scheduling and allocation of resources, as well as scheduling and allocation tasks. Further, when resources break down or dynamic needs arise, alternative schedules or equivalent allocation schemes are required to maintain the system reliability. This task is called resource exchangeability analysis. We disucss four management tasks in this paper:

1. Resource assessment – viewing and monitoring resource capacity, distribution of resources, temporal evolution, and problematic areas such as bottlenecks, and under-utilized resources
2. Resource scheduling – assigning reusable resources to demands with temporal durations
3. Resource allocation – allocating sharable resources to multiple consumers
4. Resource exchangeability – identifying resources that can replace one no longer capable of serving assigned tasks

We first present visualization techniques for assessment tasks. A unique viewing task in this domain, that of visually distinguishing between resource availability and occupancy, will be analyzed. Then we will take the task analytic approach to explore visual representations suitable for performing scheduling and allocation tasks. The visualizations used for resource exchangeability will be discussed in each of the applications presented since exchangeability is very application dependent.

3. Assessment tasks

The resource assessment task is concerned with the viewing and monitoring of the capacity, availability, distribution and temporal evolution of resources. Several requirements on the viewing of this task are required: views of overall resource distribution, views of proportional data, contrasting views of resource availability and consumption, and views of the change of resources over time. While each viewing task can be implemented by different visualization techniques such as pie charts, scatter plots, histograms, a fundamental issue relevant in all resource assessment tasks is how to achieve an effective visual contrast for resource availability and consumption. It is fundamental for the construction of other views. A look at a very common graphical display of resources can illustrate the importance of this point. Figure 1 is Microsoft's Window95 depiction of disk space allocation. According to the legend, the slice of the pie chart in red corresponds to free disk space, while blue is for used space. This color-coding is rather awkward because red is often associated with inhibition, thus more intuitively representing used space. Our empirical study showed that more than 90% of users prefer using gray levels rather than colors to represent the contrast between resource availability and consumption. In particular, gray (dark gray) is most commonly perceived as used resources, and light gray as available ones. Figure 2 is a visual rendering of available and allocated network bandwidth respectively in light and dark grays on the link. If color coding is desired, more than half of our users prefer using pairs of opposite colors, of which warm colors should be used for occupied resources, and cool ones for available resources.

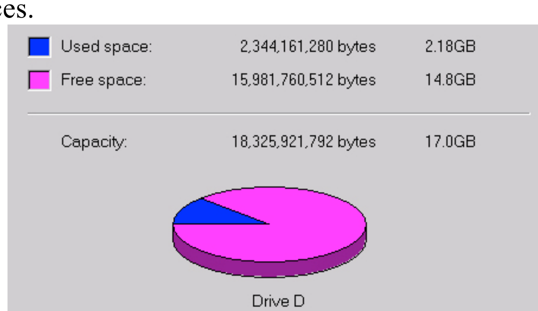

Figure1: Pie chart for disk space allocation in Window system

Figure 2: Dark and light grays contrast between allocated and available resources

Distribution and Proportional data based on contrast

Overview techniques (Figure 3) intent to capture overall distribution of resources, and at the same time provide proportions of data. The height of each cylinder represents the total bandwidth capacity between two nodes of a network. As in the network link structure, consumption and availability can be contrasted using shades of grey. For example, high capacity networks exist between (Geneva, Lausanne), (Basel, Zurich), (Geneva, Bern) and (Zurich, St-Gallen), but not for others. A relatively resource weak area, between Sion and Lugano, can also be identified. Currently all resources have been allocated on that link. It can be a potential bottleneck depending on whether there will be more demands for that type of resources.

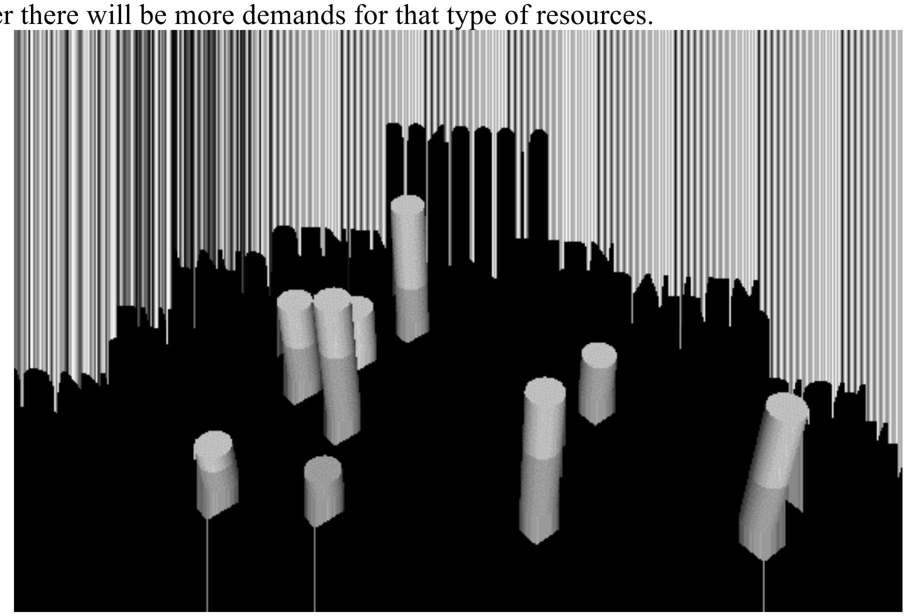

Figure 3: network resource distribution of cities in Switzerland

Temporal view based on contrast

By combining views of history and contrast, we obtain the view for resource consumption and availability in temporal direction. Figure 4 visualizes the availability of an airplane over a certain period of time. Contrast is achieved by using two opposite colors and filled glyphs, yellow tiles for availability and filled orange tiles for occupancy. The additional coding by filled glyphs makes visual mapping less ambiguous as to which color is used for occupancy.

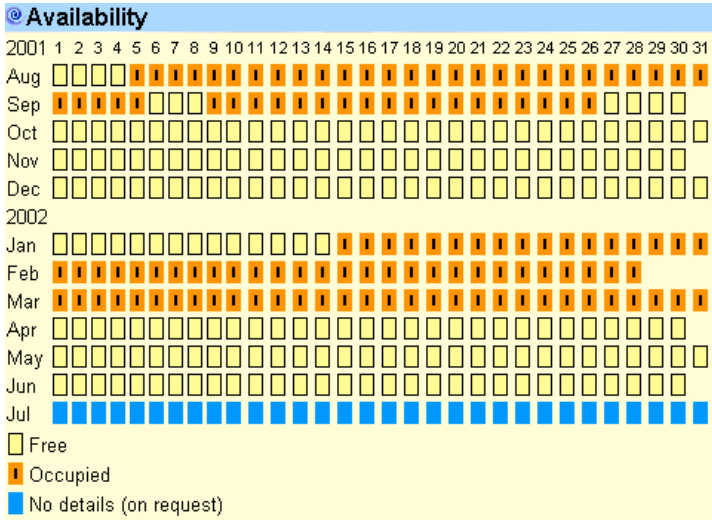

Figure 4: temporal view of resource availability

4. Scheduling task

4.1. Task analysis

The general problem of scheduling can be described as given a set of m reusable resources and n set of tasks, assign as much as possible all n tasks to resources for the specified duration while making sure that no single resource is used for more than one task at any given time. When a task has been accomplished, the resource is freed and can be assigned to another task. Scheduling problem is often done over a particular period of time. There are many variations of this problem. The simplest is a verification problem where a problem solver is to verify whether it is possible for a given task to be served by a desired resource for a specific time interval. A more general scheduling problem is to consider several tasks at the same time, while trying to maximally satisfy those tasks.

4.2.The verification task

Consider an example of a scheduling problem where seven tasks have been assigned to a total of four resources as shown in Table 1. Each assigned task is specified by

$$T_i = (R_j, t_{i\text{-start}}, t_{i\text{-end}})$$

where $0 < i < n$, $0 < j < m$, and $t_{i\text{-start}}$ and $t_{i\text{-end}}$ specify the time interval required to accomplish task T_i using resourcej.

Table 1: Specifying assigned tasks in a list

T_1=(R_1, 3,7)
T_2=(R_4, 6,10)
T_3=(R_3, 16,22)

T_4=(R_2, 6,13)
T_5=(R_4, 14,19)
T_6=(R_4, 21,29)
T_7=(R_3,4,9)

The verification problem can be specified by T_k = (R_x, $t_{k\text{-start}}$, $t_{k\text{-end}}$). That is, we are verifying whether the interval required by Tk of resource Rx has not been assigned to another task. Let us assume T_k = (R_3, 12, 16). If the problem solver is given textual representation similar to Table 1, he must scan the entire list of task assignments for at least once to verify that question. Each time he finds a Ti where R_i=R_3, he must also check if interval ($t_{i\text{-start}}$, $t_{i\text{-end}}$) and (12,16) intersect. The interval intersection costs two comparisons of two end points. The total amount of work he has to do is around n, times a constant. Therefore the time complexity of the verification task is 0(n).

4.3. Support Reasoning with High Level Perceptual Tasks

Users have to perform costly perceptual tasks (scanning and interval comparison) using textual representation such as Table 1. A crucial reduction can be gained by replacing them by more efficient perceptual tasks using visual representation. The dimensions of such a representation came directly from the task analysis: the resource being used, and the time interval for which the resource has been assigned to a task. These two variables also appear in the verification problem. This suggests a two dimensional representation where each dimension represents one of the variables. Since the number of resources is often limited while the temporal dimension is not, we use the x-axis for resources while using the y-axis for time as shown in Figure 5. A perceptual task, which is to access an area on the representation using a direct index, is all that is needed for users to check whether a resource is available for a particular interval to serve the new task. Furthermore, when resource availability has been contrasted from occupancy, the interval comparison has become a task of comparing one-dimensional geometrical objects (lines).

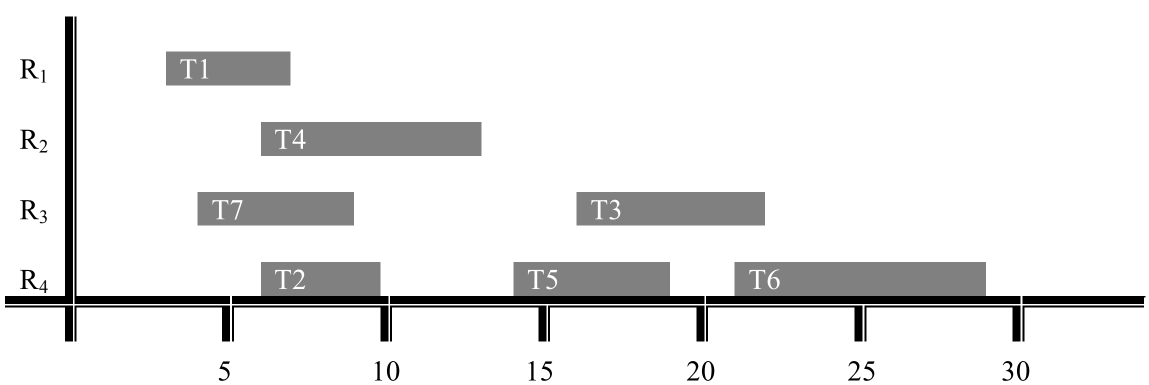

Figure 5: a visual representation of the scheduling problem

Given the new task, T_k = (R_3, 12, 16), the problem solver looks up the position on the 2D diagram, using R3 and 12 as an index. From that location, he verifies if any gray rectangles (lines) lie between (12, 16). Thus, in constant time, the problem solver is able to obtain an answer for the verification problem.

Figure 5 is more commonly known as a Gantt chart, and has been a popular visualization for scheduling tasks. Our analysis has showed why Gantt charts are advantageous: the visual representation transforms costly perceptual tasks to efficient ones. As we consider more complex scheduling problems, however, we observe that more visual representation is needed to maintain the reasoning advantage.

4.4.A General Scheduling Problem

Since scheduling multiple tasks requires backtracking on previously assigned resources, we relax the constraint that a task must be assigned to a specific time interval. That is, we consider the problem of finding resources for a given task without constraints on the starting, nor the ending time. It is specified by T_k = (R_x, interval_k), and for a more concrete example, we assume that interval_k = 10.

Using the data structure specified as in Table 1, a problem solver must scan the list of current task assignments. For each item scanned, of the format T_i={R_j, $t_{i\text{-start}}$, $t_{i\text{-end}}$}, two interval intersection calculations must be made to determine whether T_k can be potentially assigned to R_j, the interval preceding $t_{i\text{-start}}$, and the interval succeeding $t_{i\text{-end}}$. If none of these pairs of intervals intersect, then T_i is a potential solution. He must verify further down the list whether R_j has been assigned to another task whose interval intersects with ($t_{i\text{-start}}$, $t_{i\text{-end}}$). If none such task can be found, then T_i can be assigned to R_j.

For example, we start with T_1={R_1, 3,7} and compare this interval with the interval preceding 3 and succeeding 7. (0, 3) is too small for interval-k, although (7, 30) is big enough. Therefore, we have a potential assignment of T_k. We now scan down the rest of the list and verify that R1 is not assigned to any other tasks. We thus conclude that R_m=R_1.

In a similar manner, after numerous comparisons, we came to the conclusion that R_m=R_2, where interval_x can be anywhere between (13, 30). Therefore, solving for R_X requires scanning the list in Table 1 in n square time. Using the diagram as shown in Figure 5 (Gantt chart), a problem solver will now scan line by line on the Gantt chart for gaps that are sufficiently large for interval_k. This requires maximally two comparisons of end points for each of the gaps found. In this example, there are two results, R1 and R2. The problem solver is thus able to solve the general scheduling problem in O(m) time, where m is the total number of resources and is much smaller than n.

4.5.Additional Perceptual Operators

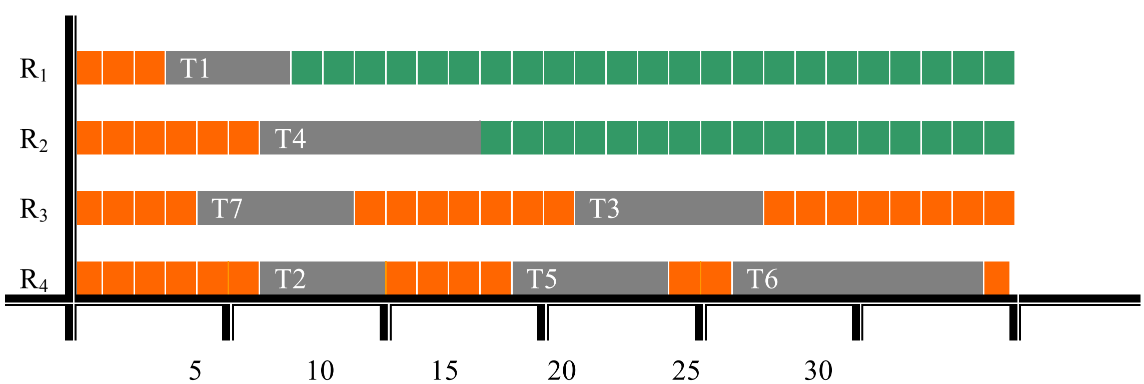

Figure 6: diagram with all admissible regions

Finding intervals of certain length is not the simplest perceptual task, especially when a row is far away from the x-axis. To render additional information perceptually, we will now mark intervals equal or greater than the new task interval in green, and shorter intervals in orange. Now the scheduler's job is to simply verify if there are any green intervals from the diagram. A Gantt chart coupled with dynamic display of color regions can therefore transform a $O(n^2)$ time problem into a constant-time one.

4.6. Scheduling Multiple Tasks with Backtracking

General schedulers try to strike a good balance between respecting temporal constraints of tasks and maximal utilization of resources. In reality, only those tasks with *soft* temporal constraints can be reassigned, meaning the constraint of the time interval can be relaxed (e.g., a patient can be operated on Tuesday, or Wednesday). We assume that the problem solver tries to minimize rescheduling and would only backtrack when a solution for a new task failed. The time complexity of performing this with the linear data structure in the worst case is NP-complete (equivalent to list coloring problem).

Using the simple Gantt chart (Figure 5), users are able to visually spot areas that can be potentially rearranged so to fit new tasks. With the region coloring technique, it is easy to verify how new tasks and their intervals can fit. Since R4 already has three tasks assigned and is getting quite cluttered, he will go directly to R1 and R2 where there is more potential solve the problem. The maximum amount of time he will spend doing this is m times the number of tasks to be assigned. The important difference is the fact that when a failure occurs, the scheduler performs a perceptual backtracking (choose resources where there are more gaps than cluttered ones), rather than doing chronological backtracking as he did when manipulating a list structure.

4.7.Implementing Gantt chart in ICARUS

We have implemented a scheduling system based on mixed-initiative interaction, called ICARUS [11], for scheduling aircraft. A Gantt chart based visualization (Figure 7) is employed, where each row represents an aircraft, and the x-axis marks the calendar time. The length of the smallest rectangle renders the total time of a flight. A set of related flights (most often outbound and inbound flights) are grouped by an outer rectangle and is defined as the smallest task unit. The ground time of an aircraft, the idle time of a resource, is the gap between rectangles, revealing information about resource over- or under-utilization. In addition to facilitating users to perform scheduling, Gantt chart also enables a user to obtain the following information:

1) contrast of availability and consumption of resource, thus temporal constraints of each resource
2) perception of proportional data between heavily deployed and under deployed resources

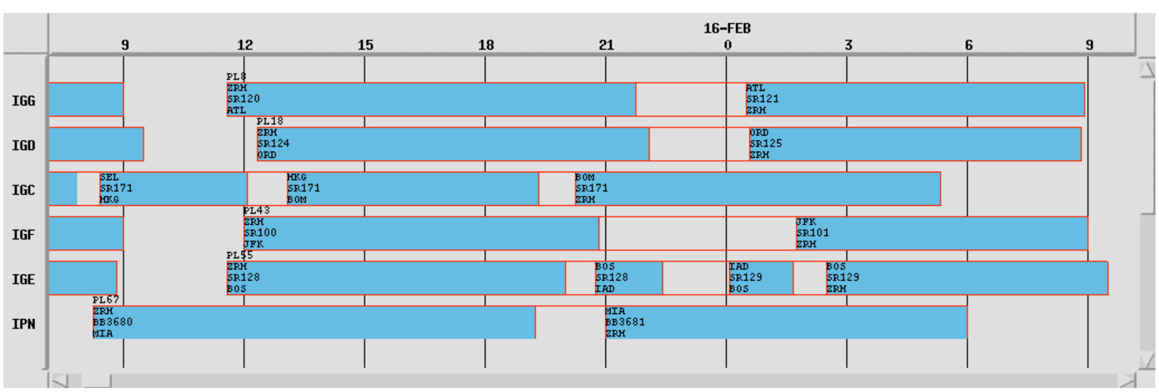

Figure 7: Gantt chart for scheduled flights

Furthermore, when users point to the intervals where resources have not been assigned, a detail-on-demand display will show the exact length of that interval.

4.8. Resource Exchangeability, Visual Scalability

In many situations, original assigned resources become inadequate for tasks or unavailable and must be exchanged with others or simply replaced to satisfy shifting needs. For example, due to irregularities caused by maintenance, weather, or promotion of certain flights, airlines often reschedule airplanes.

Consider the example of exchanging a small airplane with a larger one using the Gantt chart as shown in Figure 7, a case we studied in ICARUS when suddenly more passengers have booked on a particular route. The goal is to identify other aircraft capable of serving the routes of the aircraft being replaced, which must satisfy a certain seating capacity at the same time. This requires considering two resources having similar temporal activities over a period of time, while respecting scheduling constraints. For example, two aircraft can be exchanged given that they originate and return to the same airport, and the time interval does not interfere with other scheduled tasks. As shown in Figure 7, IGG and IGD are exchangeable between 11am on February 15 and 9am the next day for a route of two segments. If seating capacity satisfies the new requirement, then these two airplanes can be swapped without perturbing other scheduled routes. When the aircraft are not located next to each other, users can identify exchangeability by rearranging the rows in the Gantt chart. In reality, however, most of the possible exchangeable aircraft do not appear as in our example, but much more in a scenario of involving the exchange of more than two resources.

A dilemma thus exists in aircraft rescheduling tasks. For the convenience of manual rescheduling, users are more inclined to consider resource exchangeability over a short period of time. However, it is hard to find solutions this way. On the other hand, if temporal constraints were relaxed, there can be many solutions involving non-simple exchanges and requires scrolling along the y-axis. In this case, the task has become both cognitively and perceptually complex. Under such circumstances, humans resort to intelligent agents and only want to evaluate solutions. Users have demonstrated capabilities, complementary to machines, by introducing many dynamic criteria such as fuel efficiency and maintenance requirements.

An overview map (Figure 8) of all solutions was employed to enable users to compare and choose appropriate solutions inn ICARUS. Each block is a solution along three criteria: the time interval during which an altered schedule is used (total time span until settlement), the total number of aircraft, and total flight segments involved in the rescheduling. By default, the smallest block in all dimensions is the best solution. However, in most cases, users perform tradeoff analysis such as favoring the least number of aircraft involved over the total time span. Users can select any block (shown in yellow) and examine the criteria values in detail. [12] describes several other techniques, such as treemaps and decision by exclusion, to perform similar decision tasks.

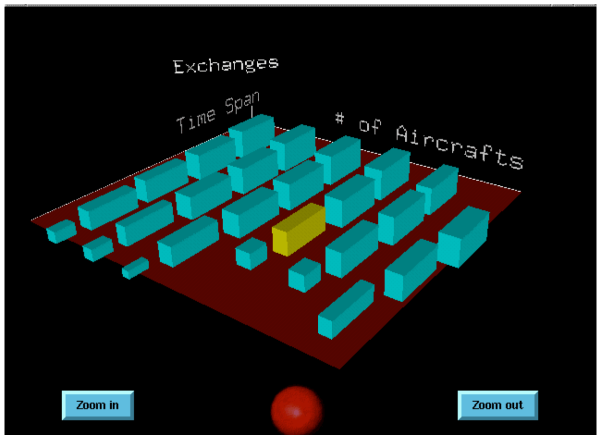

Figure 8: Overview of airplane reallocation solutions

5. Resource allocation tasks

5.1. Task Analysis

Network resource allocation is a problem of dividing available bandwidth into smaller quantities and satisfying resource consumers' demands. Demands are specified as bandwidth requirement between a pair of nodes in the network. A typical example is establishing a private virtual network between two branches of an organization, physically located in two cities. To satisfy such a demand, it involves identifying a set of connected paths in the network capable of satisfying the bandwidth requirement specified in the demand. Multiple paths or none can be found depending on network resource characteristics. Several objective functions during allocation can be considered, such as maximum demand satisfiability, quality of service, and load balancing. Maximum satisfiability refers to the fact that the more demands satisfied, the better the solution is. Quality of service can be defined as a guarantee of services for certain demands, such as the case when an allocated resource goes down (e.g., network link goes down), alternative resources are readily available to replace it. Load balancing refers to an even distribution of resource utilization.

Consider a complete network whose resources have been specified as shown in Table 3. ($node_i$, $node_j$, bandwidth) is used to specify the available resources between a pair of nodes in a network.

Table 3: Resources are defined as tuples between pairs of nodes.

(A,B, 56)
(A,C, 34)
(B,C, 12)
(C,D, 50)
(A,G, 7)
(E,G, 8)
(E,F, 36)
(D,E, 8)
(D,F, 11)

Consider satisfying a demand specified by (A,E, 6), that is, finding a path of link capacity of 6 units between node A and E. One straight-forward way to do this is to start with node A, search for a turple whose starting node is A with link (A,B, 56), search for the next turple whose starting node is B with link (B, C, 12), and continue with this procedure (chaining) until either no tuple is found, or the end node E is reached. While doing this chaining, we will mark all turples traversed, and check each time if the capacity is equal or greater than the amount specified by the demand. If not, then the chaining has failed. When more than two turples are found to match the search, we will take one and continue with the chaining, while keeping other possibilities in a temporary storage. When a failure has been encountered, we take nodes from the temporary storage and continue with chaining. In this example, a possible solution is a connected path between (A,B) (B, C) (C,D) and (D,E).

The user is obviously performing depth-first search with backtracking, thus the task complexity is exponential in terms of the number of already assigned tasks. Users also have to mark traversed nodes and keep a temporary storage for branches encountered in the chaining. Consider an improved visual representation (Table 4) which is a matrix representation containing both nodes and their connectivity. Now, the resource allocation problem requires traversing from the starting node, A, scanning across that row and picking a node say B, whose link capacity is sufficient for the task, moving to row B, and continuing with the same process until node E is reached or failure is encountered. This is faster since a user no longer has to examine the entire list to find reacheable nodes. However, there is still no notion of path distance between nodes, nor it is easy to move eyes in the matrix from one position to another.

Table 4: Defining network resource capacities in a matrix

	A	B	C	D	E	F	G
A		56	34				7
B			12				
C				50			
D					8	11	
E						36	8
F							
G							

A topo-geographical model, which is most commonly used to visualize network resources, turns out to be most superior for the purpose of resource allocation (Figure 9). It compactly displays the connectivity of nodes, the resource distribution in the network, and more importantly enables users to allocate network resource tasks in a much more direct way. The problem solver starts with node A, traverses along the connected link from A to one of the reacheable nodes, say C, and continues with the traversal until E is reached, without taking his eyes off the identified path. The visual representation succeeded in matching directly the problem solving paths. That is, the data needed for reasoning (connectivity, transition to the next pair of nodes, link capacity) has been grouped and ordered compactly in the graph, thus facilitating the problem resolution steps. Furthermore, the geographical model of the underlying network allows perceiving short and more direct paths (eyes' ability to compare distances).

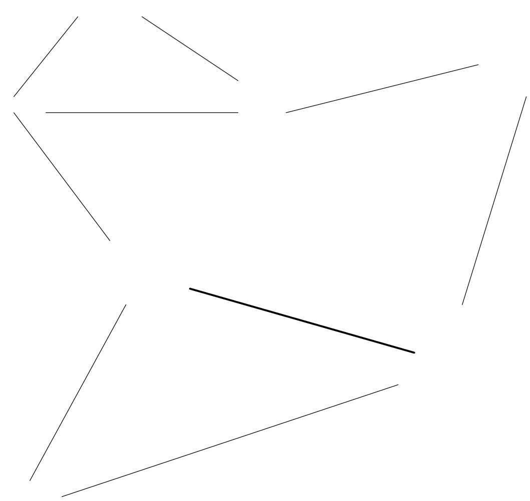

Figure 9: Defining network resource capacities in a topological model

In our informal empirical user study, we found that humans willingly and easily perform resource allocation tasks while the number of nodes does not exceed 20, and they often pick a solution that to them is the most intuitive. On the other hand, they are not extremely concerned with resource allocation optimality, such as maximum satisfiability, quality of services, and network load balance. Once the underlying problem becomes complex or optimization criteria become sophisticated, users are much more likely to rely on intelligent agents to perform these tasks. The pattern has been that users would perform allocation tasks manually before the network reaches a state where further allocation requires backtracking. For example, once a bottleneck has been detected, users often perform an optimization cleanup, that is invoking intelligent agents to reallocate network resources in order to allow new demands to be satisfied.

5.2. Implementing Topo-geographical Visual Representation in OptiNet

We have implemented a network resource allocation system based on mixed initiative interaction, called OptiNET [11]. Users interact with a topological representation as shown in Figure 10. It compactly displays information about resource distribution, allocation, and unsatisfied demands in the same data map. Each link models the bandwidth capacity, whose gray and light portions represent allocated and available resources respectively. The dashed lines show unsatisfied demands. The graph has been super-imposed on top of a geographical map of Switzerland. Consider the example of allocating network bandwidth demands between Sion and St-Gallen. Using this graph, a problem solver will trace a route from Sion, Lugano, and St-Gallen, or by a route from Sion, Geneva, Lausanne, Bern, (Basel or Zurich), and St-Gallen. However, if the network capacity stands as it is, the first route, even though geographically shorter, is impossible due to the lack of resource between Sion and Lugano. Users can also invoke agents to perform an allocation, optimize for maximum satisfiability, quality of services, and load balancing, or any combinations of these operations.

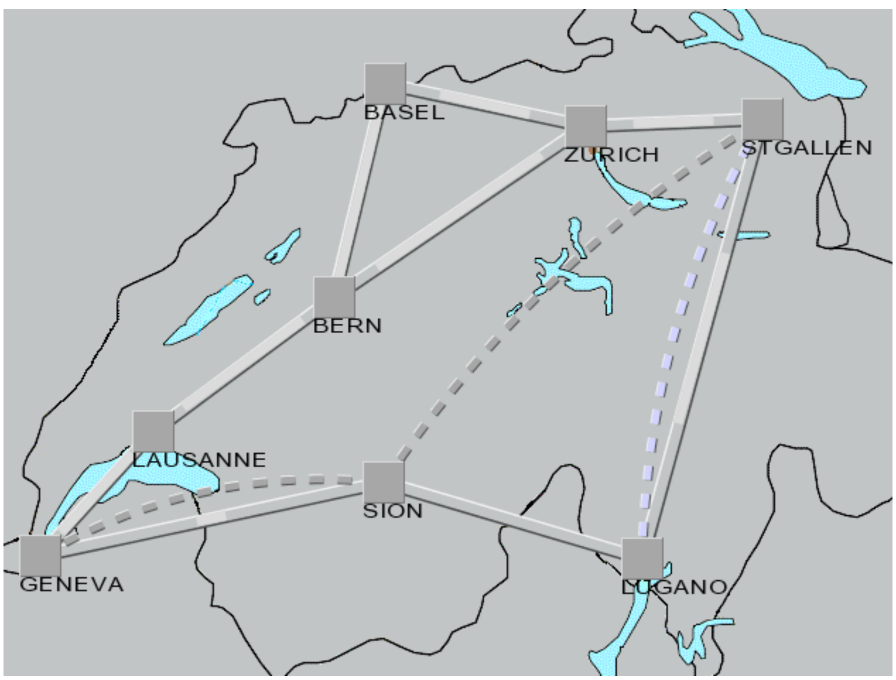

Figure 10 : resource distribution in a data map

To summarize, OptiNet allows users to perform several tasks using the graphs shown in Figure 3 and 10:

1) viewing of network capacities and distribution (Figure 3);
2) identifying areas of bottleneck (low cylinders in Figure 3, or gray filled areas in Figure 10);

3) allocating resources (Figure 10);
4) automatically allocating resources by invoking intelligent agents, viewing results, and selecting a solution (Figure 10 and menu item corresponding to resource allocation agents).

5.3. Resource Equivalence

When a link breaks down in a communication network, alternative paths must be found to service the original demands. This requires reallocating the original demands using other resources, possibly involving backtracking. Ideally, however, resource equivalence should be pre-calculated so that when a link breaks down, solutions can be quickly found. In many situations, resource exchangeability calculation can also indicate whether networks are robust or brittle.

OptiNet relies on a set of algorithms based on constraint satisfaction problem (CSP) solving techniques [16] to break a network into clusters of nodes considered resource equivalent. The results are visually rendered by grouping network nodes of the same cluster into regions as shown in Figure 11. In this example, nodes of the same cluster are reacheable from one another as long as the resource capacity required does not exceed 64K. That is, bandwidth allocation of up to 64K between anyone node to any other node within the same cluster is possible. From viewing this graph, it is clear that the current network resource capacities cannot satisfy the demands between Sion and Lugano (128K) and between Sion and Basel (128) because they are located on separate clusters. It is also clear from this graph that the current network is rather brittle. A breakdown of any of the links will decapitate the communication between nodes connected by the link. Adding a link with 64K bandwidth between Bern and Sion will augment the robustness of this network, allowing alternative paths in case a link between Lausanne and Bern, for example, breaks down. Therefore, the visualization has transformed a cognitive hard problem (reallocate network bandwidth) into relatively simple perceptual tasks (deciding whether nodes are bounded in the same cluster).

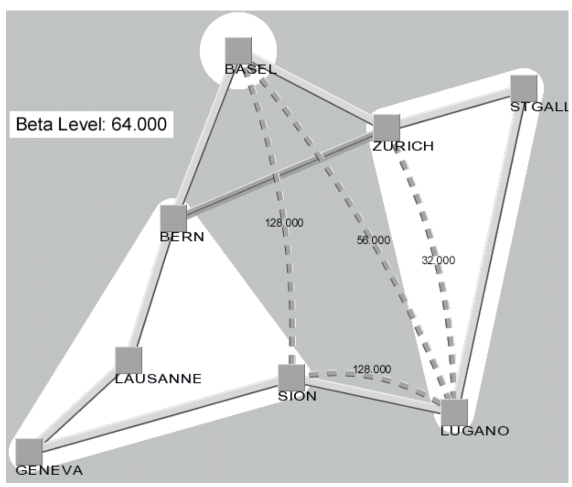

Figure 11: Network nodes within blocking islands are equivalent in terms of routing capacity up to 64k

6. Conclusion

We have argued for visual representation as an interaction medium between human and intelligent agents in mixed initiative systems. Visualization provides perceptual cues (visual affordance) that prompt humans to assimilate their knowledge and decision criteria into machine models which may not have been specified originally in the problem and cannot be exploited otherwise. Visualization compactly and intuitively displays results computed by intelligent agents in the problem context, achieving the goal of combining machine intelligence with human knowledge from the external world. To design effective visual representations that correspond users tasks, we provided a case study in resource management, namely scheduling and allocation of reusable and sharable resources. For each visual representation that was used, we studied the reduction in complexity by replacing cognitive tasks by perceptual ones. These analyses have given a fundamental ground to the Gantt chart and Topo-geographical representations for the respective case studies. However, as tasks become cognitively and visually more complex, users rely more on intelligent agents for problem solving. Rather than switching to a fully automatic system, we showed that visual representations help users evaluate decisions and perform tradeoff analysis in order to achieve a consistent framework and maintain the visual advantage.

7. References

1. Allen, J., Lehrer, N., Boddy, M., Breese, J., Burstein, M., Carciofini, J., Desimone, R., Hammond, C., Lowrance, J., MacGregor, R., Russ, T., Schrag, B., Smith, S., Tate, A., Wellman, M., and Wilkins, D., *Technical Report Version 2.0, Knowledge representation specification language*. 1992, DARPA / Rome Laboratory Planning and Scheduling Initiative.
2. Anderson, D., Anderson, E., Lesh, N., Marks, J., Mirtich, B., Ratajczak, D., and Ryall., K., *Human-guided simple search*, in *Proceedings of the National Conference on Artificial Intelligence*. 2000, AAAI Press. p. 209-216.
3. Card, S.K., Mackinlay, J.D., and Shneiderman, B., *Readings in information visualization : using vision to think*. Morgan Kaufmann series in interactive technologies. 1999, San Francisco, Calif.: Morgan Kaufmann Publishers.
4. Casner, S.M., A Task-Analytic Approach to the Automated Design of Graphic Presentations. ACM Transactions on Graphics, 1991. 10(2): p. 111-151.
5. Ferguson, G., Allen, J., and Miller, B. TRAINS-95: Towards a mixed-initiative planning assistant. in Proceedings of the 3rd Conference on AI Planning Systems. 1996: AAAI Press.
6. Ferguson, G., Allen, J., and Miller, B., TRIPS: An Integrated Intelligent Problem-Solving Assistant, in Proceedings of the National Conference on AI (AAAI-98). 1998, AAAI Press.
7. Hearst, M., *Mixed-Initiative Interaction.* IEEE Intelligent Systems & their applications, September/October, 1999. **14**(5).
8. Horvitz, E., Principles of mixed-initiative user interfaces, in The ACM SIGCHI Conference on Human Factors in Computing Systems. 1999, ACM Press. p. 159-166.
9. Larkin, J.H. and Simon, H.A., *Why a Diagram is (Sometimes) Worth Ten Thousand Words.* Cognitive Science, 1987. 11: p. 65-99.
10. Mackinlay, J.D., Automating the Design of Graphical Presentations of Relational Information. ACM Transactions on Graphics, 1986. **5**(2): p. 110-141.

11. Melissargos, G., *Interactive Visualization for Resource Allocation Tasks.* 2000, Swiss Federal Institute of Technology (EPFL), Lausanne (Switzerland).
12. Pu, P. and Melissargos, G., *Visualizing Resource Allocation Tasks.* IEEE Computer Graphics and Applications, 1997. 17(4).
13. Pu, P. and Lalanne, D., Design Visual Thinking Tools for Mixed Initiative Systems, in International Conference on Intelligent User Interfaces. 2002, ACM Press.
14. Pu, P., *Visualizing Resources and Resource Management*. 2001, Human Computer Interaction Group, Swiss Federal Institute of Technology: Lausanne.
15. Schneiderman, B., *Direct Manipulation Versus Agents: paths to Predictable, Controllable, and Comprehensive Interfaces*, in *Software Agents*, Bradshaw, J.M., Editor. 1997, MIT Press. p. 317-345.
16. Tsang, E., *Foundations of Constraint Satisfaction*. 1993: Academic Press.
17. Tweedie, L., *Characterizing Interactive Externalizations*, in *CHI97*. 1997. p. 375-382.
18. Zhang, J., The Nature of External Representations in Problem Solving. Cognitive Science.