ISSN 1407-7493 COMPUTER SCIENCE
DATORZINĀTNE 2008-34

APPLIED COMPUTER SYSTEMS
LIETIŠĶĀS DATORSISTĒMAS

DEVELOPMENT AND IMPLEMENTATION OF PARTIAL HYBRID ALGORITHM FOR GRAPHS VISUALIZATION

DAĻĒJĀ HIBRĪDA ALGORITMA IZSTRĀDE UN REALIZĀCIJA GRAFU VIZUALIZĀCIJAI

Vitaly Zabiniako, *Riga Technical University,*
Meza 1/3, Riga, LV 1048, Latvia, Mg. sc. ing.,
Vitaly.Zabiniako@inbox.lv

Pavel Rusakov, *Riga Technical University,*
Meza 1/3, Riga, LV 1048, Latvia, Dr. sc. ing., as. prof.,
Pavels.Rusakovs@cs.rtu.lv

Graph, algorithm, visualization, integration, evaluation

1. Introduction

Considering that construction and analysis of mathematics conception of graph in the field of modern information technology is often performed by human, it is clearly seen that he/she needs effective comprehensive model for manipulation with elements of a graph. That is why a visual representation aspect in usage of graph is of particular importance. Due to increase of amount of the information being analyzed, the number of graph nodes and connecting edges increases inevitably. The total result is graph nodes and edges "overlap", which complicates a perception of the displayed information by the human. In this case, a possible solution for this non-trivial problem is a usage of algorithms for positioning of graph model [1].

There are many algorithms of such kind, which form different categories, but usage of these is narrowed as it depends on context of information being visualized. That is why the question about integrating of existing categories of algorithms into single *partial hybrid algorithm*, becomes significant. Such algorithm could inherit *multiple visualization advantages* and allows performing efficient visualization of the graph regardless its structure.

Another important concept is *mechanism of a graph description*, which resides in visualization software and converts information of graphs into easy traceable descriptions, which allow storing and exchanging of this information between multiple visualization tools.

Both of these concepts are parts of *visualization environment* that must be integrated. It is also important to evaluate performance of such environment considering both quantitative and qualitative properties. The last case also requires the development of objective criteria system, considering that perception properties of single individual are subjective.

Taking into consideration the aforesaid, it is possible to identify the following goal: *to develop partial hybrid algorithm for integrating it into single visualization system*. In order to reach this goal, four tasks being defined: 1) to improve existing algorithms by integrating their positive peculiarities; 2) to propose 3D visualization-based graph description language; 3) to integrate all mentioned concepts into single software system; 4) to evaluate quantitative and qualitative characteristics of this system using the criteria system has drown out.

2. Development of partial hybrid algorithm

Algorithms for spatial models could be classified conditionally on various categories (for example – force-based layout algorithms; orthogonal layout algorithms; tree layout algorithms etc.). We propose to combine force-based layout algorithms and orthogonal layout algorithms in this work. The first of them possesses a useful property to visualize a graph regardless its maximum allowed degree of a node, it is also capable to automatically distinguish symmetric structures, although a final performance in this case is hardly predictable due to iterative nature. On the contrary, a performance of the orthogonal layout algorithm is predictable (due to strictly defined steps), although a maximum allowed degree of a node in this case is equal to *6*.

Force-based layout algorithms act as follows: geometric properties of elements of a graph are being calculated using simulation of forces [2]. The graph is presented as a physical system, in which nodes are being replaced with metallic rings, but edges – with springs. At initial step rings are scattered randomly in space with deformation of springs.

Taking into consideration, that springs will tend to compensate the deformation, all systems will also tend to find a state with minimal potential energy. This equilibrium state is iteratively calculated using an algorithm of a higher abstraction degree, which is shown in Figure 1.

```
/*1*/ until average kinetic energy ( √(dx² + dy²) ) < ξ
/*2*/    for each node i
/*3*/       velocity(i) = velocity(i) * f;
/*4*/       for each other node j
/*5*/          velocity (i) += velocity to repulse from j;
/*6*/       end
/*7*/       for each node j connected to i
/*8*/          velocity (i) += velocity to attract to j;
/*9*/       end
/*10*/      position(i) += velocity (i);
/*11*/    end
/*12*/ end
```

Figure 1. Force-based layout algorithm pseudo code

Orthogonal layout algorithms, on the contrary, implement a concept of three-dimensional orthogonal grid, which is formed by many points in Cartesian coordinate system, each having (*X,Y,Z*) coordinated defined by whole numbers. A visualization of graph in this case requires associating set of graph's elements to set of points of orthogonal grid, applying a general rule, which possesses, that edges of a graph could have 90^o bends [3]. As it was aforementioned, this category of algorithms allows visualizing graphs with a maximum allowed degree of a node of *6*. This restriction has been defined considering peculiarities of three-dimensional orthogonal grid – each grid point, which refers to a node of a graph, is located at intersection of three perpendicular lines, each of these in turn can hold not more than two nodes connected with given node by straight line as it is shown in Figure 2.

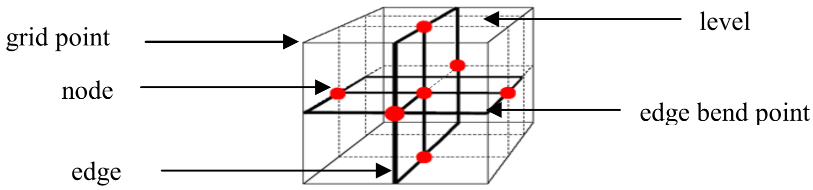


Figure 2. Orthogonal model of a graph

An empiric evaluation of implementation of orthogonal layout algorithms shows, that a possession of elements of a graph requires searching of the compromise between an overall three-dimensional model's volume and a number of edge bends.

A development of partial hybrid algorithm by itself is based on concepts of *primary* and *secondary* algorithms. After evaluation of both algorithms being integrated, it is possible to conclude, that orthogonal layout algorithms produce bends of edges, which can decrease a comprehension degree. That is why the force-based layout algorithm has chosen as primary, while adding certain concepts of the orthogonal layout algorithm.

An initial version of algorithm pseudo code was modified as it is shown in Figure 3.

```
/*1*/ for each node k
/*2*/    position(k) = point of orthogonal grid(k);
/*3*/ end
/*4*/ until average kinetic energy (sqrt(dx^2 + dy^2)) < ξ
/*5*/    for each node i
/*6*/       if average kinetic energy > previous average kinetic energy
/*7*/          for each node k
/*8*/             position(k) = random offset from position(k);
/*9*/          end
/*10*/      end
/*11*/      previous average kinetic energy = average kinetic energy;
/*12*/      velocity(i) = velocity(i) * f;
/*13*/      for each node j
/*14*/         velocity(i) += velocity to repulse from j;
/*15*/      end
/*16*/      for each node j connected to i
/*17*/         velocity(i) += velocity to attract to j;
/*18*/      end
/*19*/      position(i) += velocity(i);
/*20*/    end
/*21*/ end
/*22*/ for each node k
/*23*/    position(k) = approximation to orthogonal grid(k);
/*24*/    position(k) = position(k) - (min_poz + max_poz)/2;
/*25*/ end
```

Figure 3. Partial hybrid algorithm pseudo code

Potential improvements in this case are following:

- An initial random scattering of nodes is shown in unpredictable number of iterations for equilibrium state's search. *Solution*: to place nodes of a graph on predefined points of the orthogonal grid before the first iteration will start.
- In case if a value of average kinetic energy achieves local minimum, it takes an additional time to retrieve from this. *Solution*: to make a step with random offset from current position, bypassing "slow" iterative retrieving upon detection of local minimum.
- After achieving the equilibrium state, coordinates of nodes holds fractional parts increasing randomness of orientation of model of a graph. *Solution:* forcibly approximate nodes of a graph to points of orthogonal grid after achieving the equilibrium state.
- The equilibrium state is being formed with unpredictable offset from the origin of coordinate system. *Solution*: to place the model of graph near the origin of a coordinate system (similar to orthogonal model), using formula (1) for each dimension of nodes of a graph:

$$koord' = koord - \frac{\min\_koord + \max\_koord}{2} \quad (1)$$

Summary of evaluation of the developed algorithm is presented in chapter 5.

3. Three-dimensional visualization oriented graph description notation

Visualization of graphs using aforementioned algorithms is directly connected with definition of a graph information. Before it is converted to final image, it is necessary to precisely define procedures for storing and interpreting of this information.

Nowadays *XML* (*Extensible Markup Language*) is considered as universal solution for information determination. This specification becomes a base for many graph descriptive languages: *GraphML* [4], *GraphXML* [5], *XGMML* [6] etc. It is possible to lead the following brief comparison of these languages (Table 1):

Table 1. Comparison of graph descriptive languages

Criteria	GraphML	GraphXML	GML	XGMML	GXL
Field of application	General	General	General	General	Software designing and reengineering
Validation mechanisms	Schemes	DTD	Rules of language	DTD, rules of language	DTD, schemes
Additional checking for data consistency	No	No	Yes	Yes	No
Support for hierarchical graphs	Yes	Yes	Yes	Yes	Yes
Support for hyper graphs	Yes	No	No	No	Yes
Support for custom types of data	Yes	Yes	Yes	Yes	Yes
Support for specific 3D related data	No	Partial	No	No	No

With analyzing of this information, is possible to conclude, that all these languages were primarily meant for visualization of common two-dimensional graphs. Visualization procedures of three-dimensional graphs require an introduction of specific descriptive structures. First of all, it has applied to spatial positions of nodes of a graph. A description of each node must be enhanced with tag **pos**, which must contain **z** property along with traditional **x**, **y** properties. It is also important to introduce tag **shape**, which defines an appearance of nodes and edges, which is of great importance in visualization activities. Three aspects – **model**, **color** and **size** of a surface mainly define the appearance of the node. A description of the edge has the same properties except of **size**, which is replaced by withdrawal property **width**.

As a result, the possible description of elements for trivial 3D graph in Figure 4.

```
/*1*/  <node id="1" label="V1">
/*2*/    <pos x="-32.79" y="20.22" z="6.69"/>
/*3*/    <shape model="1" color="FF0000" size="3"/>
/*4*/  </node>
/*5*/  <node id="2" label="V2">
/*6*/    <pos x="-4.07" y="24.94" z="-24.17"/>
/*7*/    <shape model="1" color="FF0000" size="3"/>
/*8*/  </node>
/*9*/  <edge source="1" target="2" directed="1" label="E1">
/*10*/   <shape color="0000FF" width="5"/>
/*11*/ </edge>
```

Figure 4. Specification of trivial graph for 3D visualization

A position property can hold real numbers, and these can be both positive and negative (taking into consideration, that elements of model can be localized in different regions of the coordinate system in relation to its origin). There is one exception in a description of attribute of position – in case, if precise values are unknown or were not calculated previously, it will hold character "**?**". In this case a positioning of nodes of a graph is in concern of visualization system itself (for example, the original force-based layout algorithm could place nodes randomly, the partial hybrid algorithm could place these on predefined points of the orthogonal grid etc.).

A proposed specification has one disadvantage in relation to a size of an information being stored. In most cases, sizes, colors and withdrawals of elements of graph are the same. This enforced duplication of certain information fragments (for example, in case if there are *100* black edges with same withdraw equal to *5*, this will produce equal *100* tags <**shape color**="000000" **width**="5"/>), which negatively affects the size and the interpretation of the information being stored. *Solution:* to introduce two additional tags, one of which holds default information for nodes, and the other (with the same purpose) for edges. In this case, precise values will be included only in case, if these differ from default. A fragment of the description of default attributes is performed in Figure 5.

```
/*1*/ <nodedefault label="New node">
/*2*/       <pos x="?" y="?" z="?"/>
/*3*/       <shape model="1" color="FF0000" size="3"/>
/*4*/ </nodedefault>
/*5*/ <edgedefault directed="1" label="New edge">
/*6*/       <shape color="0000FF" width="5"/>
/*7*/ </edgedefault>
```

Figure 5. Specification of default attributes

It is also proposed to include a description of initial setup of interactive visualization software for visualization of graph in refined specification (for example, if it is known that the graph being described is localized near the origin of a coordinative system, it would be useful to shift a virtual camera along *Z* axis, before visualization will be initiated in order to see a whole body of the graph, when it will be shown for the first time). To implement this feature, a block of an additional information will be added to the description as separate **environment** tag. It holds an information about screen background color attribute **bgcolor**, a virtual camera tag **cam** with position tag **pos**, which holds attributes **x**, **y**, **z** for specification of the initial position of the virtual camera and rotation tag **rot**, which holds same attributes for specification of rotation of the virtual camera. The turned out description fragment, which can be included in final description, is performed in Figure 6.

```
/*1*/ <environment bgcolor="000000">
/*2*/   <cam>
/*3*/     <pos x="0.00" y="0.00" z="200.00"/>
/*4*/     <rot x="0.00" y="0.00" z="0.00"/>
/*5*/   </cam>
/*6*/ </environment>
```

Figure 6. Description of initial setup of visualization software

According to performed experiments with visualization of graphs, such refined description is useful for solving of all main tasks of visualization procedures.

4. Implementation of visualization software system

A software system "*3DIIVE*" (*Three–Dimensional Interactive Information Visualization Environment*), which includes numerous sub modules, was built as a part of this research visualization. It is necessary to point out that every graph regardless its complexity can be considered as a whole object. That is why an architecture of the aforementioned software was developed utilizing main principles of the object-oriented approach.

In this case the object, which represents a graph, contains:

- A set of attributes (which defines an information on basic graph elements):
  - For nodes (coordinates, color, size, labels etc.).
  - For edges (linked nodes, color, width, labels etc.).
- A set of methods (which allows to obtain an information on the graph, making transformations of a spatial model etc.):
  - For obtaining information about graph elements.
  - For adding and removing graph elements.
  - For dynamical change of the visual graph structure.
  - For other major processing purposes.

It is necessary to point out, that processing of a graph also matches with other basic concepts of the object-oriented approach, for example – *inheritance* and *aggregation* of a model of the graph [7].

A framework of drawing up of the developed software system is performed in Figure 7.

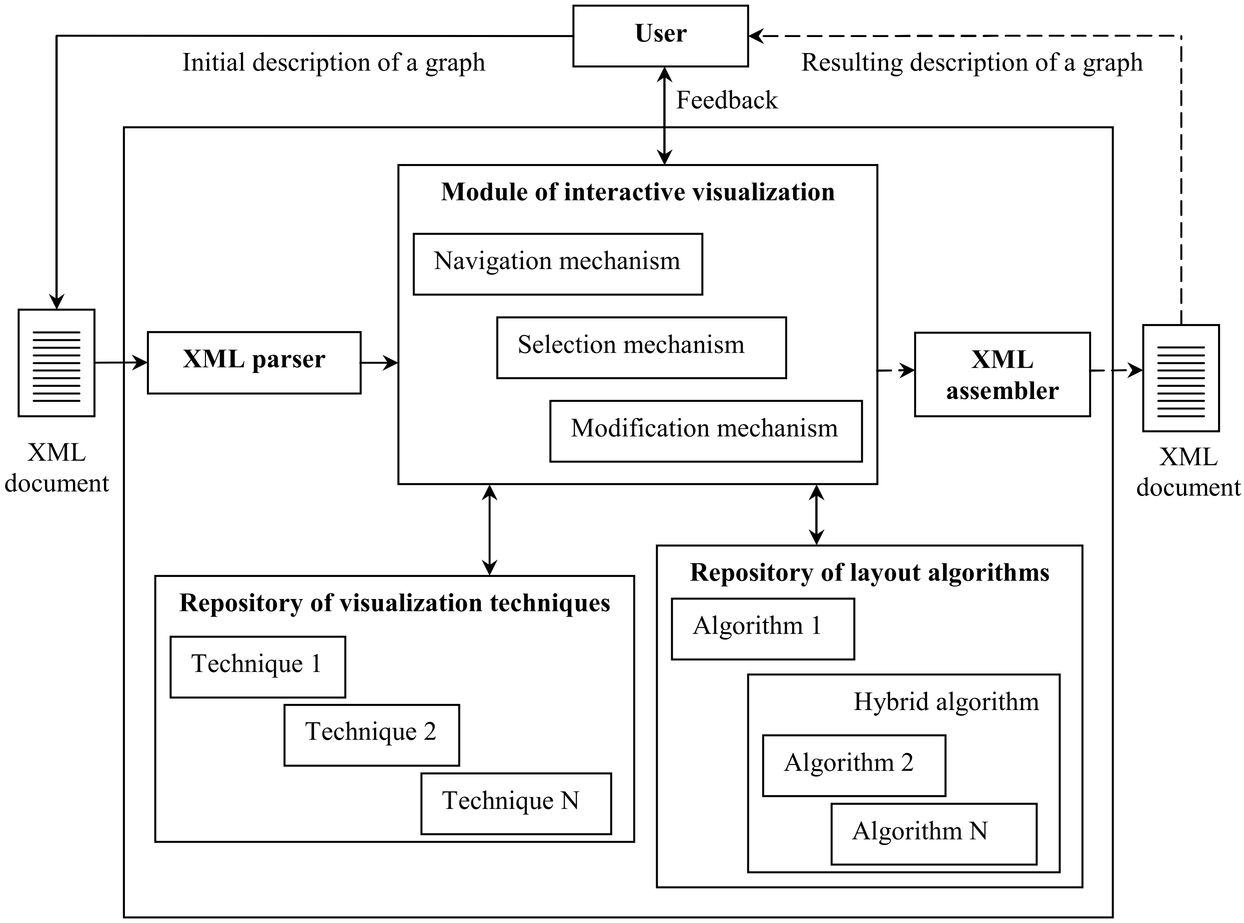


Figure 7. Architecture of visualization software system for graphs

The main parts of drawing up of this system are as follows: *XML parser*, *module of interactive visualization*, *repository of layout algorithms*, *repository of visualization techniques*, and *XML assembler*.

XML parser is represented by a software module, which parses XML document with a description of the graph provided by the user, and allows extracting necessary information about general topology of the graph and its' separate elements (for example – names of nodes and edges, color of these etc.) [8].

Module of interactive visualization is considered to be the main part of the system, which performs visualization of received information (by activating necessary layout algorithms from *repository of layout algorithms* and visual techniques for improved comprehension from *repository of visualization techniques*), offers the analysis and processing abilities (by using *navigation*, *selection* and *modification mechanisms*) [9] and passes processing results to *XML assembler* in case, if the user requires to save processed and modified data. Implementation of this module has been based on the object-oriented model: the graph is presented as a separate object with multiple attributes and methods. An information about topology of the graph derived from the parser module, is converted into the adjacency matrix, while a rest of the information about elements of the graph is loaded into multiple arrays with references to associated elements. Separate methods are used for positioning spatial model of a graph by calling certain layout algorithm and for triggering multiple visualization techniques. *Navigation mechanism* is implemented as sub module, which allows transforming visual appearance of 3D model by altering position of a virtual camera. *Selection mechanism* allows to distinct and to mark certain set of visible parts of a graph for further modification of these. *Modification mechanism* allows altering data or a structure of previously selected elements of a graph.

Repository of visualization techniques holds a set of multiple techniques for improvement of comprehension (for example – "magnifying glass", "fish-eye view", "Benediktine space" etc.), separate elements of which can be triggered within the *module of interactive visualization*.

Repository of layout algorithms holds a set of available layout algorithms for visualization of graphs. Implementation of this module relates to separate methods of the object of the graph. It can hold both representatives of existing categories of the algorithms and hybrid algorithms that unite them.

XML assembler is a software module, which acts similar to *XML parser* with only distinction that relates to a direction of information processing flow. It is necessary to point out, that usage of this module is optional (information flows are represented with dashed arrows) and is required only if user decides to save changes in original information, after data processing procedures were carried out.

Visual examples of output of this system are performed in Figure 8.

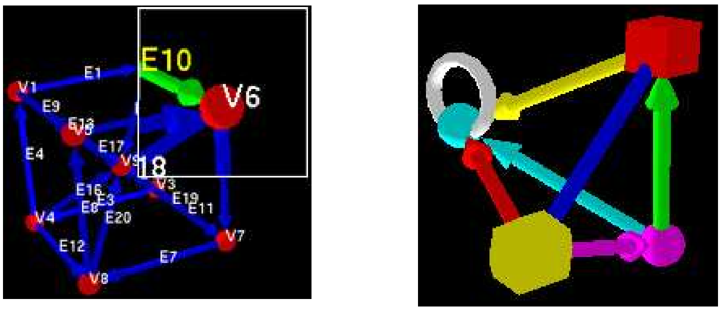
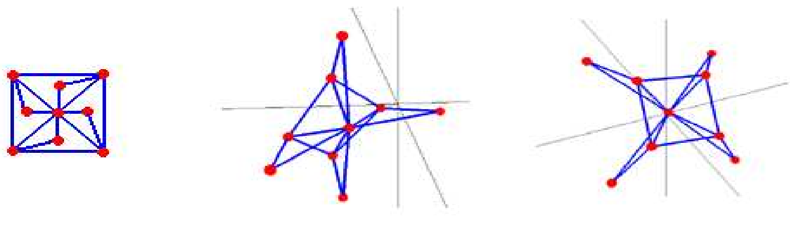


Figure 8. Screenshots of visualization outputs

5. Evaluation of partial hybrid algorithm and visualization system

A set of visualization experiments was carried out to demonstrate a performance of partial hybrid algorithm. One of these experiments is as follows: an input – graph with maximum degree of a node equal to *8* that is shown in Figure 9 (part *A*). Considering that orthogonal layout algorithms cannot visualize such types of graphs, we will evaluate application of the force-based layout algorithm (part *B*) and the partial hybrid algorithm (part *C*).



(A) **(B)** **(C)**

Figure 9. Results of visualization experiments

We point out a centered model of a graph as well as ability to visualize straight edges as the result of usage of the partial hybrid algorithm. We also evaluated performance of algorithms (performing *50* measurements starting with first iteration and ending with the moment, when an equilibrium state was reached) – Table 2.

Table 2. Quantitative evaluating of performance of algorithms

Force-based layout algorithm											
Time (s)	13.8	14	14.7	15.3	15.8	16.5	16.9	17.1	17.2	17.7	18.2
Count	2	3	6	7	9	7	6	4	3	2	1
Frequency	0.04	0.06	0.12	0.14	0.18	0.14	0.12	0.08	0.06	0.04	0.02

Partial hybrid algorithm				
Time (s)	13.4	13.5	13.6	13.7
Count	11	12	15	12
Frequency	0.22	0.24	0.30	0.24

The graph of frequency distribution is shown in Figure 10.

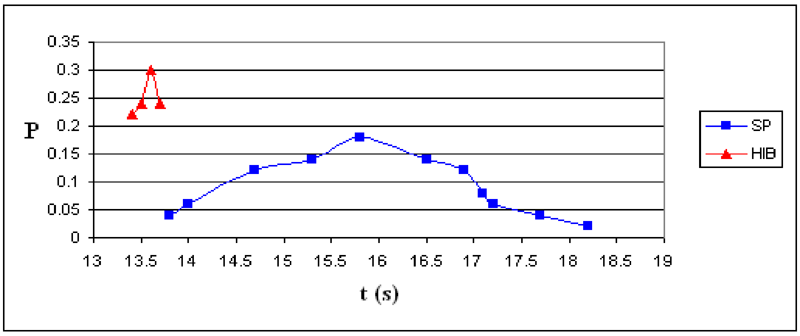


Figure 10. Frequency distribution graph

During statistical processing of visualization results, following metrics were acquired: mathematical expectation for force-based layout algorithm $M=15.95$, statistical dispersion $D=1.24$ and standard deviation $\sigma=1.11$. For the partial hybrid algorithm, these values are following: $M=13.56$, $D=0.012$, $\sigma=0.11$.

While performing of a qualitative evaluation of a developed software system, it is necessary to use an objective mechanism, which implies a set of multiple criteria. That is why visualization evaluation activities were supplied by querying of experts from different Latvian educational institutions (Riga Technical University, University of Latvia, Transport and Telecommunication Institute, Rezekne Higher Education Institution). Development of questionnaire and processing of received results was carried out according to *Delphi* method [10], with main task to define the set of objective criteria for evaluation of visualization. Both the set of criteria and the set of importance indexes for them were defined after summarizing of results. A fragment of querying results is presented in Table 3.

Table 3. Part of results of querying of experts

Proposed criteria	Number of answers	Percentage of answers
What are main properties of visualization software?		
Ability to work with multiple modern graph description languages	3	38%
Ability to implement repository with templates of structures of common graphs	4	50%
Ability to inherit and combine models of graphs	3	38%
Ability to implement animation for reaction to events triggered by user	3	38%
Ability to work with many graphs at a time	5	63%
Ability to work with many instances of same graph	3	38%
Ability to manually define parameters for visualization procedures	5	63%

An analysis of this data allowed us to design a criteria system for evaluation purposes, which in turn allows to define Quality of Visualization Environment using formula (2):

$$QVE = \frac{\mathrm{A} * \mathrm{k(A)} + \mathrm{T} * \mathrm{k(T)} + \mathrm{L} * \mathrm{k(L)}}{\mathrm{Q}VE_{\max}} * 100\% \tag{2}$$

where A, T and L corresponds to evaluation metrics of layout (A)lgorithms, visualization (T)echniques and description (L)anguages, according to set of criteria defined by experts; $k(A)$, $k(T)$ and $k(L)$ corresponds to according fractional normalization indexes within interval *[0..1],* and QVE_{max} corresponds to biggest possible evaluation value.

The evaluation metric for layout algorithms is calculated using the formula (3):

$$A = \sum_{i=1}^{N(A)} a_i * k(a_i) \tag{3}$$

where $N(A)$ corresponds to number of criteria for layout algorithms; a_i – evaluation of criteria i in interval [*0..10*] ("0" relates to maximum discrepancy with this certain criteria, while "10" relates to maximum conformity with this); $k(a_i)$ – importance indexes for criteria i according to results of the querying (these values were derived from a column "percentage of answers"). Also, to reduce subjectivity the following rule was applied: <u>only those criteria are included in final set of criteria that were pointed out by at least two experts</u>. The same principle was applied to visualization techniques and description languages.

After integration of all calculations the following *final formula* (4) for evaluation of a quality of visualization environment is proposed:

$$QVE = \frac{0.4\sum_{i=1}^{N(A)} a_i * k(a_i) + 0.4\sum_{i=1}^{N(T)} t_i * k(t_i) + 0.2\sum_{i=1}^{N(L)} l_i * k(l_i)}{70.34} * 100\% \tag{4}$$

Values for normalization metrics are being proposed by authors while estimating relative importance of visualization aspects based on querying of experts. During evaluation of "*3DIIVE*" software A, T and L metrics were calculated using according a_i, t_i and l_i values. A fragment of evaluation results is presented in Table 4.

Table 4. Part of results of evaluating of 3DIIVE" software

Criteria	Coefficient	Evaluation	Substantiation of evaluation
Ability to work with multiple modern graph description languages	0.38	5	This software uses custom refined language which is derived from majority of existing languages
Ability to implement repository with templates of structures of common graphs	0.5	10	Such kind of repository can be easily implemented introducing ".xml" files with predefined topology of graphs
Ability to inherit and combine models of graphs	0.38	5	There is ability to inherit models of graphs which can be achieved by adding new element of graphs
Ability to implement animation for reaction to events triggered by user	0.38	10	This ability is fully supported
Ability to work with many graphs at a time	0.63	10	This ability is fully supported
Ability to work with many instances of same graph	0.38	0	This ability is not supported in current version
Ability to manually define parameters for visualization procedures	0.63	10	This ability is fully supported

Upon evaluation completion of a quality of visualization environment "*3DIIVE*", a final value was calculated, which is equal to *51%*. Our opinion is that such result seems to be true (taking into consideration that presented software holds status of demo version, and it was designed during academic research without claiming for a status of a commercial solution), which confirms that this criteria system is enough reliable.

6. Conclusion

This research provides an analysis of current problems in visualization of graphs and advises of some possible solutions by utilizing achievements of modern information technologies. The following problems were pointed out in this research:

1) The usage of existing layout algorithms is narrowed as these heavily depend on context of information being visualized and achieve success only in case, if properties of this information (for example, graph's topology) satisfies requirements of this certain algorithm.

2) A majority of modern graph descriptive languages are oriented towards as description of common aspects of graphs. In this case, an information, which is required for full-fledged visualization activities, can be introduced into description only as custom user's data. In turn, it negatively affects a size of the information, being stored, and its interpretation time.

3) Evaluation of visualization of graphs is non-trivial, taking into consideration that comprehension of the result is mostly subjective.

In order to introduce possible solution for these problems, we designed, integrated and evaluated a partial hybrid algorithm, which inherits *multiple visualization advantages* both from force-based layout algorithms (an ability to visualize the graph regardless its maximum allowed degree of a node; an automatic distinguishing of symmetric structures) and orthogonal layout algorithms (an ability to center a body of the graph; an ability to visualize straight edges). Another improvement related to discrete positioning of nodes of a graph before performing the first iteration step allowed to *decrease* average time, which is necessary for reaching the equilibrium state for $(1 - 13.56/15.95) * 100 = 15\%$ while *reducing* its statistical dispersion.

We also propose a refined graph descriptive language, which is oriented towards a support of visualization activities. It could be used successfully with implemented hybrid algorithm. The demonstrative software system for visualization purposes was drawn out. It integrates all discussed concepts and can be used as a tool for the analysis of graphs.

A combination of this software with a developed criteria system can serve as a base for further research in a field of visualization of graphs. Following up of this research, it is planned to improve the further partial hybrid algorithm by integrating it with other categories of layout algorithms in accordance with improvements in the developed visualization system.

References

1. J. Rilling, J. Wang, S. P. Mudur "MetaViz – Issues in Software Visualizing Beyond 3D", 2003. Internet – http://www.cs.uvic.ca/~mstorey/vissoft2003/submissions/rilling2.pdf
2. X. Liu, B. Shizuki, J. Tanaka "Dynamic Parameter Spring Modeling Algorithm for Graph Drawing", 2001. Internet – http://citeseer.ist.psu.edu/574826.html
3. P. Eades, A. Symvonis, S. Whitesides "Two Algorithms for Three Dimensional Orthogonal Graph Drawing", 1997. Internet – http://www.cs.mcgill.ca/~sue/papers/eades-sym-whi.pdf
4. U. Brandes, M. Eiglsperger, I. Herman, M. Himsolt, S. Marshall "GraphML Progress Report", 2002. Internet – http://www.inf.uni-konstanz.de/algo/publications/behhm-gprsl-01.pdf
5. I. Herman, M.S. Marshall "GraphXML – An XML Based Graph Interchange Format", 2000. Internet – http://ftp.cwi.nl/CWIreports/INS/INS-R0009.pdf
6. J. Punin "XGMML 1.0 Draft Specification", 2000. Internet – http://www.cs.rpi.edu/~puninj/XGMML/draft-xgmml.html
7. V. Zabiniako, P. Rusakov "Analysis of Visualization Problems of Graphs and Models of Graphs". Scientific Proceedings of Riga Technical University, Computer Science, Applied Computer Systems, ser. 5, vol. 30, 2007, pp. 138 – 148.
8. U. Brandes, C. Pich "GraphML Transformation", 2004. Internet – http://www.inf.uni-konstanz.de/algo/publications/bp-gt-04.pdf
9. L. Deligiannidis, Amit P. Sheth, B. Aleman-Meza "Semantic Analytics Visualization", 2006. Internet – http://lsdis.cs.uga.edu/library/download/DSA06-ISI.pdf
10. T. J. Gordon, "The Delphi Method", 1994. Internet – http://www.futurovenezuela.org/_curso/5-delphi.pdf

Zabiņako V., Rusakovs P. Daļējā hibrīda algoritma izstrāde un realizācija grafu vizualizācijai

Šis raksts satur informāciju par oriģināliem risinājumiem grafu vizualizācijas problēmsfērā, izmantojot trīsdimensiju datorgrafiku. Šāda tipa vizualizācija ļauj atvieglot grafa informācijas uztveri ar cilvēku, salīdzinot ar "klasisko" plakano attēlošanu. Šī pētījuma galvenais mērķis ir konstruēt daļējo hibrīdo algoritmu ar nolūku uzlabot grafu vizualizāciju trīsdimensiju telpā. Divas eksistējošo algoritmu klases tika izmantotas par pamatu šī algoritma konstruēšanai: uz spēku iedarbību un uz ortogonalitāti pamatotu klašu pārstāvji. Tika intensīvi izmantota arī objektorientētā pieeja. Izveidotais daļējais hibrīdais algoritms tika veiksmīgi realizēts eksperimentālā sistēmā. Raksts satur informāciju par šīs sistēmas arhitektūru, citam tās fundamentālām koncepcijām, kā arī eksperimentu rezultātiem. Visi iegūti rezultāti tika novērtēti izmantojot matemātiskās statistikas paņēmienus. Ir piedāvāta kritēriju sistēma vizualizācijas efektivitātes novērtēšanai. Ir salīdzinātas populārās grafu aprakstīšanas valodas un ir piedāvātas dažas rekomendācijas to uzlabošanai. Šo rekomendāciju realizācija var būt veiksmīgi pielietota kopā ar izveidoto hibrīdo algoritmu. Ir izdarīti secinājumi par sasniegtiem rezultātiem. Ir arī sniegta informācija par šī pētījuma iespējamo turpinājumu.

Zabiniako V., Rusakov P. Development and Implementation of Partial Hybrid Algorithm for Graphs Visualization

This paper contains an information about some original solutions in graphs visualization problems, using three–dimensional computer graphics. Such visualization allows simplify the graph's comprehension by human in comparison with a "classical" planar visualization. The main goal of the research is to create a partial hybrid algorithm to improve a graph's visualization in 3-D space. Two categories of existing algorithms were used as the base for this algorithm: a force-based layout and orthogonal layout algorithms. An object-oriented approach was applied intensively too. The developed hybrid algorithm was successfully implemented in experimental system. The article contains an information about creation of this system, other fundamental concepts of the system and some results of experiments. All achieved results were evaluated, using elements of mathematical statistic. A conception of the criteria system to evaluate an efficiency of visualization, is advised. Popular graph descriptive languages are compared, and some recommendations to improve mentioned languages are proposed. These recommendations can be used successfully with an implemented hybrid algorithm. A conclusion about already achieved results is made. An information about future researches is presented.

Забиняко В., Русаков П. Разработка и реализация частичного гибридного алгоритма для визуализации графов

Данная статья содержит информацию о некоторых оригинальных решениях в сфере визуализации графов с использованием трехмерной компьютерной графики. Такой подход позволяет упростить восприятие графа человеком в сравнении с "классической" визуализацией на плоскости. Главной целью данного исследования является создание частичного гибридного алгоритма для улучшения отображения графов в трехмерном пространстве. Базой создаваемого алгоритма являются два класса существующих алгоритмов – ориентированные на имитацию физических сил и ориентированные на ортогональность. В процессе создания нового алгоритма интенсивно использовался объектно-ориентированный подход. Разработанный гибридный алгоритм был успешно реализован в экспериментальной системе. Данная статья содержит информацию об архитектуре этой системы, других ее фундаментальных концепциях, а также некоторых проведенных экспериментах. Для обработки полученных результатов использовался аппарат математической статистики. Рекомендована система критериев для оценки эффективности визуализации. Проведено сравнение популярных языков описания графов и разработаны рекомендации по их улучшению. Данные рекомендации можно успешно реализовать вместе с разработанным гибридным алгоритмом. В статье сделаны выводы о достигнутых результатах, а также представлена информация о дальнейших планируемых исследованиях.