Advances in Intelligent Systems Research, volume 174

Proceedings of the 8th Scientific Conference on Information Technologies for Intelligent Decision Making Support (ITIDS 2020)

Method for Improving the Accuracy of Search Algorithms in Socially Significant Events

Vyacheslav Antonov
Faculty of Computer Science and Robotics
Ufa State Aviation Technical University
Ufa, Russia
antonov.v@bashkortostan.ru

Ansaf Abdulnagimov
Faculty of Computer Science and Robotics
Ufa State Aviation Technical University
Ufa, Russia
ansafufa@mail.ru

Veronika Suvorova
Faculty of Computer Science and Robotics
Ufa State Aviation Technical University
Ufa, Russia
milana_da@mail.ru

Yana Mikhailova*
Faculty of Computer Science and Robotics
Ufa State Aviation Technical University
Ufa, Russia
ysm.act@mail.ru

***Abstract*—This article presents the results of calculations that allow us to visualize the connections of graphs of socially significant events, which are revealed through the tools of categorical logic as homomorphisms in the conditions of algorithmization of search mechanisms using combinatorial analysis and the principle of greedy maximization of information growth. The main result is a new method, which is based on the grouping of graph structures by highlighting their main ties in spirtah and creating their unique homomorphic image due to the greedy principle of maximizing the information gain in conjunction with combinatorial logic combinatorial analysis that allows the interpretation of socially significant events in the form of detail relationships creating its elements and forming them as an object and, subsequently, improve the accuracy of the algorithms.**

Keywords—graph, morphism, category, algorithm

I. INTRODUCTION

Properties of graphs allow us to expand the limits of measurement in a variety of applications. The most common ones are Economics, logistics, chemistry, and many others. This article reveals the possibility of increasing accuracy in the field of search algorithmization by combining graph theory with various mathematical tools, such as set theory, game theory, and combinatorial analysis.

II. SEARCH ALGORITHMS

Initially, referring to the basics of graph construction as a theory, the graph is the preferred element for further calculus produced using it. Based on the conditions for organizing automated search, the connection between graphs is used as the basis for establishing relations between objects. Referring to objects and designating them as a system for further description, we present an array of statistics of ordered quantitative data of socially significant processes that can be disclosed in the criminalization of society and considered as a socially dangerous trend of each individual event in statistics.

For example, such an element can be decomposed into separate types of crimes, and they, in turn, into the properties of each of the types of committed acts. The connection between all types and their resonance on society (which can also, of course, be considered as a connection, only for events), determines the preference for algorithmization of search in graph connection structures. The algorithmization condition shown in the flowchart (Fig. 1) is to follow the formulation rule until all exceptions are taken into account and the information void is reduced. Due to these rules, the accuracy of the calculations given below can be achieved. Let's assume that event statistics are Let's assume that event statistics are formulated as Cartesian sets based on parameters of conditional equivalence of set values

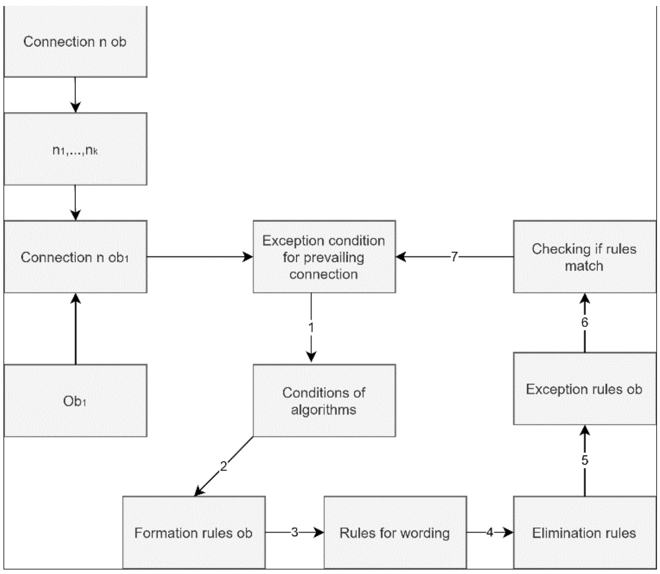

Fig. 1. A block diagram of the algorithmic rules of search in the structures of the connections graphs.

on graphs as separate elements. Then the graph G with vertices H and arcs E is considered as a signal in the form of nodal intersections by cause- and-effect connections between variables (signals) of the connection system.

Establishing a functional connection involves reducing the complexity of the graph structure and converting graphs in topology if they are equivalent, and forms their homomorphic image and homomorphic connections for connected subgraphs of sets of preimages [1]. The graph-intersection of the selected connections contains vertices and arcs that are common for further multiplication of graphs, which can be assumed as a necessary element for the formation of Cartesian products of all sets of graphs

(multiplication of topological spaces).

Let the count $G = (H, E)$ includes a set of vertices and a set of edges that are a subset of the Cartesian square of the set of vertices (i.e., each edge connects exactly two vertices). Then, conditionally, we have some relations $G_1=(H_1,E_1)$ и $G_2=(H_2,E_2)$. Product of graphs $G_1 \times G_2$ there will be such a graph G, for which it is feasible

$$H_G = H_1 \times H_2 \quad (1)$$

what it represents cartesian product of the sets of nodes (links) of the original graphs as objects, and there is also an edge (v,w), where $v=(v_1,\dots,v_k)$ and $w=(w_1,\dots,w_k)$, when there is a family of edges

$$E_1=[v_1; w_1],\dots,E_k=[v_k; w_k],E_i \subset H_i \quad (2)$$

Based on this, it becomes expedient to introduce the concept of the spectrum of graphs in the formed objects. The spectrum of a graph is the set of all eigenvalues of the adjacency matrix taking into account multiple edges, and the adjacency matrix is a square matrix of order n, in which the value is a_{ij} an element located in place (i,j) equal to the number of edges (arcs) starting at the vertex h_i , and ending at the top h_j.

Matrices are convenient for various permutations between iterations over relations. For example, the reachability matrix on graphs derived from it reveals the structure of graph connections. And the vertex connectivity and edge connectivity can be defined in order to create conditions specified sets of the graph as edge-connected. The main task will be to memorize the rules for selecting connections in graph tree structures and form them as a functional object (functor).

Then we can Express the order of iterations with graph trees as the decisive rules under which it becomes possible to form features in graph spectra through the most prevailing connections, where the connections are nothing more than an input functor of a topological space. To form the main tree structure, you need to check the static influence of connections on the Hamiltonian paths of graph trees, which also allows you to double-check the complexity of the graph structure. Including the minimum number in the graph k of new vertices by the set influence, the graph becomes Hamiltonian ($if\ k > 0$). Let us denote the Hamiltonian cycle $v_1, v_k, \dots, v_1$ graph G', if $v_1 \in V$. Any vertex of the graph G' is adjacent, with the exception of the reverse, to the vertex $v_к$, which assumes the total number of $n + k$ vertices and new vertices is at least

$$2\left(\frac{n}{2} + k\right), \quad (3)$$

which makes the graph Hamiltonian and does not contradict ore's theorem. Before calculating the spectrum of graphs, the combinatorial apparatus of combination can be used first. The validity of the algorithm is achievable by induction over n, if we note that the algorithm generates first all subsets

$$A = \{1,2,\dots,n-1\}, \quad (4)$$

then repeats this sequence, adding n to each subset.the Last term in the sequence will be the set A itself, and the algorithm will automatically stop. The descriptive part of the algorithm is as follows

$$a_1 = 1, a_2 = 2,\dots, a_r = r \quad . \quad (5)$$

If an r-combination is generated $a_1, a_2,\dots, a_r$, then the largest one is defined a_i, such that $a_i + 1$ it is not included in this combination. If there a_i are none this only happens if, then

$$a_1 = n - r + 1,\dots, a_r = n \quad (6)$$

urther conversions are stopped [2].

Additionally, a conjunction and disjunction check can be performed, after which the entire original structure is converted to a spectrum $Sp(G)$, reflecting the General properties of all graphs. To do this, you should translate the graph data G to the adjacency matrix A=(a_{vi}), where v - variable (next v_k , k - graph vertex, $k \in V$), but i - value of the neighboring vertex. In this case, we use the condition

$$\lambda v = Av, \quad (7)$$

where v - vector-column with elements v_k, in which it is obvious

$$|\lambda I - A| = P_G(\lambda) = 0, \quad (8)$$

where P_G (λ) the preliminary result of permutations of adjacency matrices and the spectrum is defined as

$$Sp(G) = [\lambda_1, \lambda_2,\dots, \lambda_i] \quad , \quad (9)$$

In a spectrum with the same event sampling conditions, n-links can be allocated as a separate structural unit, which, based on the General problem, must be described as a spectrum with conditionally assigned values by their weights [3]. In this case, the weight functions are fair

$$A = \{(a_{ij}), i = 1,\dots,n;\ j = 1,\dots,n; \left(a_{ij}\right) = f(V_i, V_j)\}, \quad (10)$$

where V_i, V_j – vertices of the graph, $f(V_i, V_j)\}$ – the weight function that determines the covariance of communication. Based on what, the wood frame T - homomorphic image or rule for conglomeration by relations V_i, V_j in $Sp_n(G)$ and its corresponding allocation on the grounds of connections. In this case, it is excluded that the tree T has cycles, except for their image for the algorithm, but also has chains or, in other words, trunks T, which are conditionally, two-way-infinite (Fig. 2).

In this case, it is possible to use the output of the center of mass of the tree. Take the classical representation of tree weights for clarity. For a vertex v from the tree T, denote by

$$E_i=[v,w_i] \quad (11)$$

an arbitrary edge with the end of v. All edges belonging to chains from v with the first edge E_i form a part of

$$B_i = B[v,E_i], \quad (12)$$

which forms a branch, forms a branch defined by E_i and v; in this case, the edge E_i is included in B_i. The number of edges in B_i, is $v_e(B_i)$, so that

$$v_e(T) = \sum_{i=1}^{p(v)} v_e(B_i). \quad (13)$$

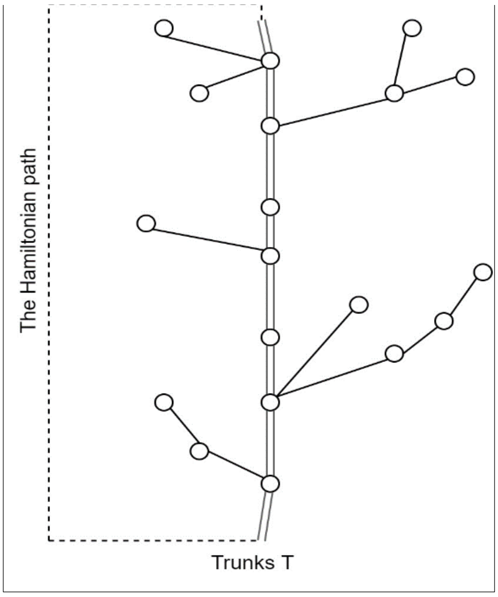

Fig. 2. The Hamiltonian path in algorithmization procedures.

Numeric value

$$v(u) = \max_{i=1,2,..,p(v)} v_e(B_i) \quad (14)$$

is the weight of the tree in v, and any branch B_i for which

$$v_e(B_i) = v(u) \quad (15)$$

is true and is the definition of a branch with weight. Then the initial center of mass, from pre-conglomerated connections, is given m_0 as the vertex with the minimum weight and also the weight of the tree T for which

$$u(m_0) = u(T) \quad (16)$$

is valid [4]. Now we propose a series of events as a phenomenon of K_n in the description of $Sp_n\ (G)$ and the prevailing V_i, V_j, then, when formulating a conditional feature, a set of logical connections formed earlier by disjunction is derived from the operations. The preparatory iteration for describing the further algorithm can be indexing the search steps. Let's assume that the search procedures are performed on a chain conditionally set as S_0 and possible chains $S_1, S_2, \ldots, S_n$.

Let's consider a vertex of one of these chains (shown in Fig. 3), different from the initial and final. Since this vertex is already included in the chain using two arcs, it is obvious that all other arcs entering or exiting such a vertex can be removed from the graph. For any starting vertex of the above chains, you can delete all arcs that originate from it (except for the arc that includes this vertex in the chain), and for any ending vertex, you can delete all arcs that end in it. In addition, except for the case when there is only one chain (for example, S_0) passing through all the vertices of the graph G (i.e., when S_0 is a Hamiltonian chain), any existing arc leading from the end of any chain to the beginning vertex of the same chain can be deleted, since such an arc closes non — Hamiltonian cycles and can only be considered when formulating the algorithm. Removing all these arcs will produce a graph with many vertices — all the vertices of chains-in which only one arc ends at each vertex and only one arc originates from it. All these vertices and arcs that are incident to them are removed from G, and instead a single arc is introduced for each chain, running from the chain's starting vertex to its ending vertex.

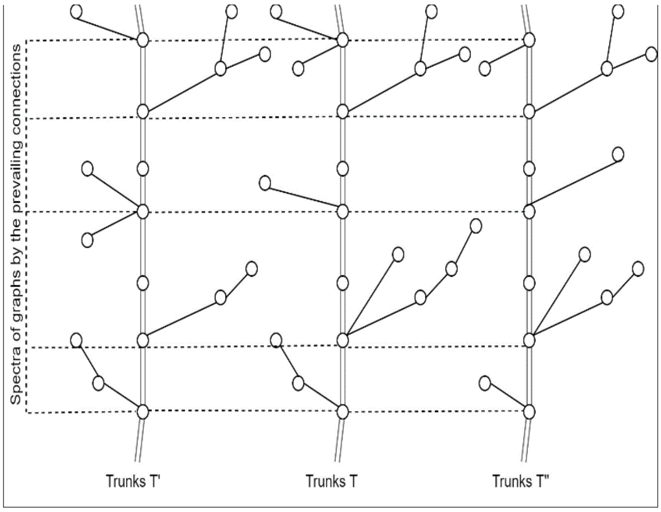

Fig. 3. The resulting spectra of the translation data.

As a result, we get a reduced graph $G_k=(H_k,E_k)$, where κ is an index that shows the step number of searching for informative links and partially solves the problem of reducing complexity, and also prepares the algorithm for greedy maximization of information growth (hereinafter) [5].The most important, at this stage, is the search for the Hamiltonian cycle itself, and this problem can be solved classically in the code of Listing 1.

It follows that you need to rebuild the tree so that it is possible to search for the average and closest value in the spectrum of sets up to the conditions of the approximate value by weight [6]. The preferred solution here is to reveal the spectrum weights by comparing the tree weights, which is a connecting link and teaches search abbreviations. On the weight allocated to the categories of expressions can be enclosed in the test substitution by cycles.

We have a set set of graph values, for example N_m, in the substitution of this set P, where is any element of the array $a_m \in N_m$. The desired condition for this problem is repeatable each cycle of links for each search pattern events which can be characterized as allocated to the category provided a cyclic repetition on a set of similar events, and therefore can be confirmed in the grounds of a particular communication structure [7]. In this case, the tree structure can be formulated for further training and focused on the conglomeration of connections, as well as the subsequent formation of a knowledge base for search terms.

```
/* function to verify that a specified solution is indeed a hamiltonian
* cycle of the graph
* solution is represented by an array of vertices
*
* returns HC_VERIFY if it is a solution
* returns HC_NOT_VERIFY if it is not
*/
int
hc_verify_solution(
 graph_type *graph,
 int solution[])
{
 int loop;
 int vertcount[MAXVERT];

 /* first verify that the solution is indeed a path of length = numvert */
 for (loop = 0; loop < (graph->numvert)-1; loop++)
 {
   if ( check_if_edge(graph, solution[loop], solution[loop+1]) != EDGE_EXIST)
     return(HC_NOT_VERIFY);
 }

 /* check if endpoints are connected (have a cycle) */
 if ( check_if_edge(graph, solution[graph->numvert-1], solution[0])
          != EDGE_EXIST)
   return(HC_NOT_VERIFY);

 /* check to see that there are no repeats of vertices */

 /* initialize array */
 for (loop = 0; loop < graph->numvert; loop++)
 {
   vertcount[loop] = 0;
 }

 /* count # of times each vertex appears */
 for (loop = 0; loop < graph->numvert; loop++)
 {
   vertcount[solution[loop]] ++;
 }

 /* check that each vertex only appeared once */
 for (loop = 0; loop < graph->numvert; loop++)
 {
   if (vertcount[loop] != 1)
     return(HC_NOT_VERIFY);
 }

 return(HC_VERIFY);

}  /* end of hc_verify_solution() */


/***************************************************************
* A set of functions to handle an edgestack
* (a stack which stores edges)
*/
/***************************************************************/
```

Listing 1. Code of the Hamiltonian cycle search program.

This task can be solved by further algorithmization of the selected actions, which can be based on the principle of greedy maximization of information growth – at each step, the attribute is selected, when dividing by which the increase in information is greatest. Then the procedure is repeated recursively until the entropy is equal to zero or some small value, in order to avoid unnecessary retraining procedures. Similarly, each element selected by attribute is assigned a sufficient number of unique values, and not all thresholds for connections can be selected, but only those that give the maximum increase. That is, in fact, a tree of depth "1" is built for each threshold, it is calculated how much entropy has decreased, and only the most significant ones are selected, with which the attribute should be compared. In this case, the selected features are added to the units of functors.

Then, let's assume that using functors, a correspondence was established between objects of different categories, each of which was interpreted as a General model of characteristics of socially significant processes. As indicated earlier, functors are allocated to establish connections between elements that were previously transformed as graphs connections (expressed as prevailing by weights in subsystems) and their corresponding models (systems). Take a functor translation from a graph given by some assumed binary relation

$$r =< x, R >, R \subseteq X * X \quad (17)$$

(elements $x_i \in X$, interpreted in the nodes of the graph, pair $< x_i, x_j > \in R$ - as edges of a graph), in the category of dynamical systems. To ensure that the axiomatics of category theory is fulfilled, the graph is considered as transitive and reflexive (conditionally). Given this conclusion, the graph can be considered as a category R, then the vertices of the graph are objects of this category, the edges are morphisms, and the constructing functor $G : R \to r$ set a match between vertexes $x_i \in R$ and built dynamic systems $D_i \in r_n$ between the edges of the graph

$$< x_i, x_j > \in Mor_R\ (x_i, x_j), \quad (18)$$

and their corresponding morphisms $f: Mor_r\ (D_i\ ,\ D_j\ )$. Accordingly, the conditions that allow us to deduce a homomorphic object in the conditions of the previously selected operation on graph structure reduction and homomorphic image formation can assume the output of sets of vertices of two graphs, which maps adjacent vertices to analogous ones [4,8].

One of the possible options for implementing the applications of the considered functor construction based on system logic is the dynamization of a set of processes expressed in the trend, in which each vertex of the graph can be interpreted as an operation of vector data processing, and the corresponding object in the category, conditionally r, - as a dynamic (acceptable deviation from approval) model for this operation. Under the condition of the described structuring, it is possible to introduce control functions as an object during General iterations over the system.

From a meaningful point of view, a Cartesian closed category will be the smallest calculus whose formulas include the vertices of each graph taken and whose proofs include its arrows. A deductive category that is freely generated from a graph, by imposing restrictions in the form of identities between proofs, becomes valid for Cartesian closed deductive categories [9-16].

III. CONCLUSION

This will reduce in the smallest equivalence relation between proofs that fulfills the substitution laws and corresponding identities of a Cartesian closed deductive category, in which the functors preserve a Cartesian closed structure on the categories. In this case, it may be fair to use the functor in strict relation to the rules of axiomatics of category theory, taking into account the connections of the raised graph, transferred to the category of dynamical systems.

The result of the calculations is the possibility of applying a new method to increase the accuracy and consistently exclude the properties of data insufficiency, expressed in the connections of graphs of socially significant events with the tools of categorical logic. The proposed method combines multi-level modeling based on the General principle of increasing accuracy and eliminating errors in the output data. An innovation in the given material is the allocation of graph connections to the object of categorical logic and the use of such an object as the main result [17-25].

ACKNOWLEDGMENT

The study is conducted with financial support from the Ministry of Education and Science of the Russian Federation as part of the basic part of the state assignment to higher education educational institutions # FEUE-2020-0007.

REFERENCES

[1] Theory of finite graphs, A. A. Zykov, Ed. Nauka, Siberian branch Novosibirsk, pp. 322-343, 1969.

[2] Combinatorics and graph theory, V. A. Nosov, Moscow state Institute of electronics and mathematics, M., pp. 32-50, 1999.

[3] Spectra of graphs. Theory and applications, D. Cvetkovic, M. Doob, J. Sachs, Science. Dumka, Kiev, pp. 22-281, 1984.

[4] Graph theory, O. Ore, Nauka, Main edition of physical and mathematical literature, M., 1980.

[5] Graph theory. An algorithmic approach., N. Christofides, Mir, M., 1978.

[6] http://webdocs.cs.ualberta.ca/~joe/Theses/HCarchive/main.html, Basil Vandegriend

[7] Combinatorics and graph theory, V. A. Noskov, Moscow state Institute of electronics and mathematics (Technical University), M., pp. 32-37, 1999.

[8] Category logic, V. L. Vasyukov, ANO Institute of logic, M., 2005.

[9] Spectral Graph Theory, Fan R. K. Chung, Regional Conference Series in Mathematics, Number 92, California, 1994.

[10] “Graph Theory”, R. Diestel, Springer, 2016.

[11] Introduction to Categories and Categorical Logic, S. Abramsky, N. Tzevelekos, 2011.

[12] Category theory, 2nd edition, S. Awodey, Oxford logic guides, 2010.

[13] 'What is a Thing?': Topos Theory in the Foundations of Physics, A. Doering, C. Isham, 2008.

[14] Probabilistic method, 2nd edition, N. Alon, J. Spencer, Binom. Knowledge lab, M., 2013.

[15] Information Theory, Inference, and Learning Algorithms, D.J.C. MacKay, Cambridge University Press, 2005.

[16] Analytic Combinatorics. Symbolic Combinatorics. P. Flajolet, R. Sedgewick, Philippe Flajolet and Robet Sedgewick, France-USA, 2002.

[17] Graph theory, Ed. 4th, F. Harari, Librokom, M., 2009.

[18] Graph theory, A. V. Omelchenko, MTSNMO, M., 2018.

[19] Graph theory—Handbooks, manuals, JL. Gross, J. Yellen, CRC PRESS LLC, D.S., 2003.

[20] Combinatorial Methods with Computer Applications, JL. Gross, Taylor & Francis Inc, Boca Raton, FL, United States, 2007.

[21] Handbook of Discrete and Combinatorial Mathematics, KH. Rosen, CRC PRESS LLC, D.S., 2000.

[22] Search Games, L.A. Petrosyan, A.Y. Garnaev, SPBU publishing house, SPb., 1992.

[23] Toposes. Logic of categorical analysis., R. Goldblatt, Mir, M., 1983.

[24] Inverses of triangular matrices and bipartite graphs, R.B. Bapat, E. Ghorbani, Indian Statistical Institute, Delhi Centre, K.N. Toosi University of Technology, India, Iran, 2018.

[25] The Algorithm Design Manual, S.S. Skiena, BHV Petersburg, SPb., 2011.