Research Article

Journal of Advances in Artificial Intelligence and Machine Learning

Network Centrality Operations under Spatial Grasp Technology

Peter Simon Sapaty*

Institute of Mathematical Machines and Systems National Academy of Sciences of Ukraine.

***Correspondence:**

Peter Simon Sapaty, Institute of Mathematical Machines and Systems National Academy of Sciences of Ukraine.

Received: 02 Feb 2025; **Accepted:** 18 Mar 2025; **Published:** 09 Apr 2025

Citation: Sapaty PS. Network Centrality Operations under Spatial Grasp Technology. J Adv Artif Intell Mach Learn. 2025; 1(1): 1-11.

ABSTRACT

The paper analyzes rapidly growing popularity and importance of the use of graph and network models in many areas, like sociology, economy, ecology, defense, security, international relations, psychology, and many others. It investigates, evaluates, and models one of the most important features of graphs and networks called Centrality, with its variants Degree Centrality, Closeness Centrality, Betweenness Centrality, and Eigen Vector Centrality. These features are very important for identifying nodes in networks and measuring their importance. It then briefs the developed and patented Spatial Grasp Model and Technology (SGT) which allows for fully distributed and parallel operations on large networked structures, including general technology issues, its basic Spatial Grasp Language (SGL), and the networked language implementation. It then provides in SGL detailed, clear, and extremely compact solutions for different centrality problems, which can operate on arbitrary large and complex networks, and in a highly parallel and fully distributed mode. The paper also reviews and classifies existing publications on networks in different areas, on network centrality problems, and on new prospective applications which can be effectively covered by SGT. The provided solutions were obtained in a direct spatial thinking, spatial pattern recognition and pattern matching mode, rather than under traditional logic-based and algorithmic philosophy and culture. Being highly parallel and fully distributed, they may challenge existing opinions that parallel and distributed algorithms are usually more complex that traditional sequential ones for solving the same problems, just proving the opposite, and especially for the network-related applications.

Keywords

Network Centrality, Degree Centrality, Closeness Centrality, Betweenness Centrality, Eigen Vector Centrality, Spatial Grasp Technology, Spatial Grasp Language, Network implementation, Pattern recognition, Pattern matching.

Introduction

We are witnessing the rapidly growing use of networks and their different types, both physical and virtual, which have become an important part of our lives. The easiest way to expand your network is to build on the relationships with people you know; like family, friends, classmates, colleagues and acquaintance's, what we are doing daily. Networking can be beneficial in both social and professional aspects. Social networking can be defined as using different social media platforms to connect with others who have similar interests. Networks are also created to satisfy objectives or needs. Consider, for example, the transportation network (roads, highways, rails, and so on). If any of these become suddenly unavailable, our ability to distribute food, clothes and products would be seriously compromised. In today's competitive market, responsiveness to customer or supplier demand is often a decisive factor in the success of an organization, and the network is considered one of the most critical resources, both in private and public sectors. Similarly, a computer network is created to provide a means of transmitting data, sometimes essential data, from one computer to another. The accuracy and speed of daily business transactions for large organizations are vital to their success.

The aim of this paper is to investigate, evaluate, and model one of the most important features of graphs and networks, called ***Centrality***, with the use of networking-based Spatial Grasp Model and Technology already tested on different applications, theoretical and practical graphs and networks including. In the graph analytics, Centrality is a very important concept in identifying nodes in a

graph or network, and is used to measure the importance of various nodes, or how "central" they are in the graph. Centrality comes in different flavors and each flavor or a metric defines importance of a node from a different perspective and further provides relevant analytical information about the graph and its nodes.

The rest of the paper is organized as follows. Section 2 reviews publications on different types of networks and their usage, like traffic networks, battle networks, economic networks, virtual networks, and psychological networks. Section 3 provides a brief summary on existing network centrality publications, reflecting: centrality, network centrality, centrality in social networks, centrality and underground network, and centrality in psychological networks. Section 4 briefs the developed and patented Spatial Grasp Model and Technology, which allows for fully distributed and parallel operations on large networked structures. It includes general technology issues, its basic Spatial Grasp Language (SGL), the language interpreter organization, and networked SGL implementation controlled by spatial tracking system. Section 5 shows how arbitrary network can be described in SGL and implemented in parallel. Sections 6, 7, 8, and 9 describe how such network features as Degree Centrality, Closeness Centrality, Betweenness Centrality, and Eigen Vector Centrality can be expressed, implemented, and evaluated in SGL, with detailed examples of solutions provided, which can be organized in parallel and fully distributed manner. Section 10 offers a brief summary on the rapidly growing use of networking models, especially in psychology and psychopathology, for which the results of the current paper may be particularly useful too. Section 11 concludes the paper with availability of practical implementation of the described technology and its use for networking problems in extended areas. References reflect the provided reviews on networking publications and existing sources of the technology used. Appendix provides a summary on SGL syntax and its main constructs.

Different Types of Networks and Their Usage

At the beginning we are providing a brief summary on networks-related publications, explaining their main ideas, orientations, contents, and usage.

Vehicular Ad-hoc Networks [1], Figure 1, offers improved road segment-based geographical routing protocol. The real- time traffic information dissemination among on-road vehicles has been envisioned via realizing vehicular ad hoc networks. The network enables various types of real-time traffic applications related to safety and infotainment for drivers and passengers. This paper presents an improved road segment-based geographical routing protocol focusing on better head node selection for information dissemination. It divides the forwarding area into a number of road segments and selects a head node on each segment.

Battle Networks [2], Figure 2, are used by militaries use to detect what is happening on the battlefield, process that data into actionable information, decide on a course of action, communicate decisions among forces, act on those decisions, and assess the effectiveness of the actions taken. Battle networks are sometimes referred to as the "sensor- to-shooter kill chain" and they are widely acknowledged as an increasingly important element of modern warfare. As battle networks became faster, longer range, and more advantageous to militaries, the networks themselves also became an attractive target.

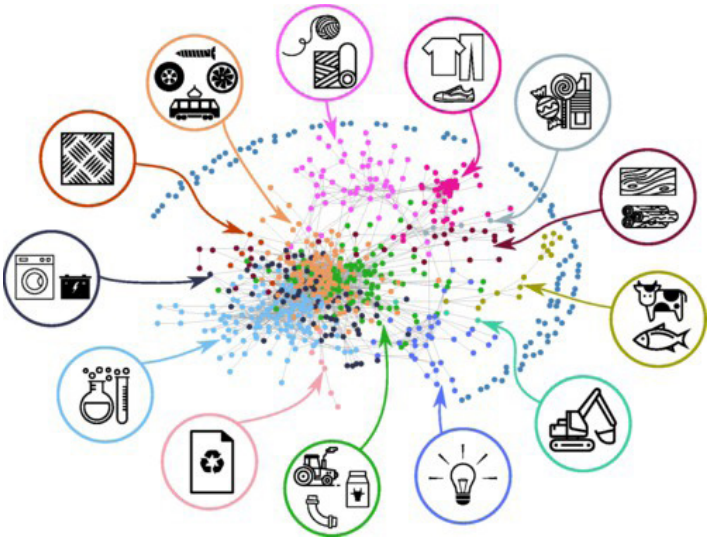

Figure 1: Traffic networks.

Figure 2: Battle networks.

Economic Network [3], Figure 3, is a combination of individuals, groups, or countries that interact to benefit an entire community. The group's main goal in an economic network is to strengthen its position in a market. The composition of these networks may vary. In some economic networks, the members of the network are immutable, in others the members are free to join or leave. Economic networks can include groups of individuals, companies or countries sharing a common goal. Activity in economic networks may include recruiting, surveying, learning, and resource sharing.

Virtual Network [4], Figure 4, allows companies and individuals to create a network that exists between computers and servers. This allows for many benefits from remote access capabilities to making

it easier to troubleshoot and fix issues. The article discusses virtual networking and the role it plays in business. A virtual network is a network where all devices, servers, virtual machines, and data centers are connected through software and wireless technology. This allows the reach of the network to be expanded as far as it needs to for peak efficiency, in addition to numerous other benefits.

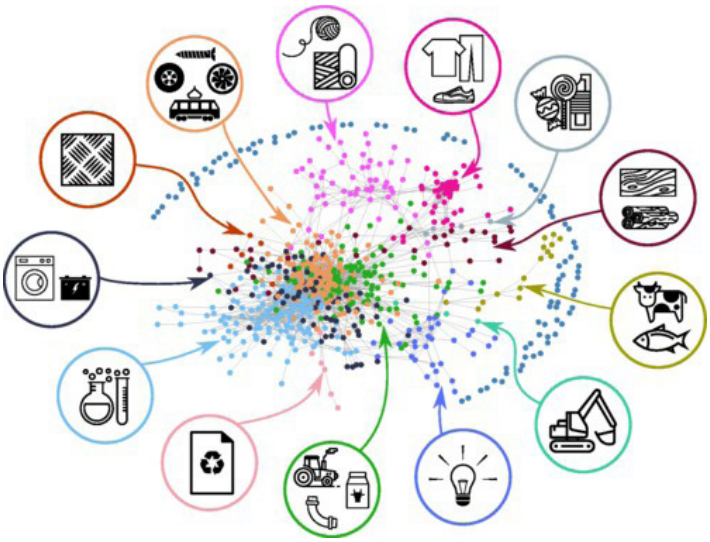

Figure 3: Economic networks.

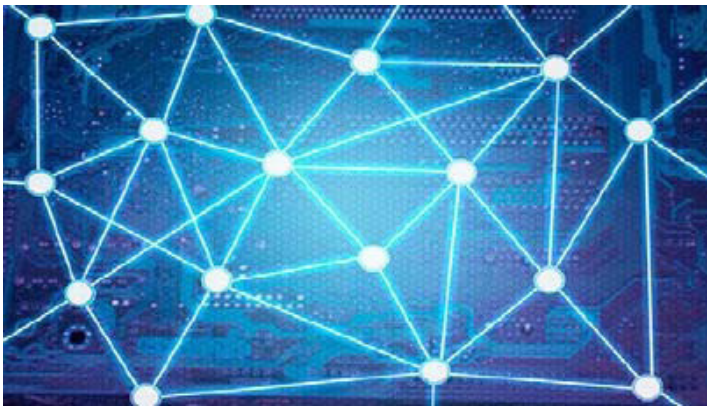

Figure 4: Virtual networks.

Network analysis in Psychological Science [5], Figure 5, has been applied to identify and analyze patterns of statistical association in multivariate psychological data, where network nodes represent variables in a data set, and edges are pairwise associations between variables in the data, while conditioning on the remaining variables. The paper provides an anatomy of these techniques, describes the current state of the art and discusses open problems, also identifies relevant data structures in which network analysis may be applied: cross-sectional data, repeated measures and intensive longitudinal data.

The Concept of Network Centrality and Its Importance

We are offering a brief summary on the existing network centrality publications with their main ideas and applications.

Figure 5: Psychological networks: Stress symptoms (1: Relax, 2: Irritable, 3: Worry, 4: Nervous, 5: Future, 6: Anhedonia); Social symptoms (7: Alone, 8: Social offline, 9: Social online).

Concepts of Centrality [6]. The importance of graph data structures depicting relationships among entities is growing. In graph analytics, Centrality is a very important concept in identifying nodes in a graph. It is used to measure the importance (or "centrality" as in how "central" a node is in the graph) of various nodes in a graph. Each node can be assessed from an angle depending on how "importance" is defined. Centrality comes in different flavors and each flavor or a metric defines importance of a node from a different perspective and further provides relevant information about the graph and its nodes.

Network Centrality [7]. Centrality is a key property of complex networks that influences the behavior of dynamical processes; it can bring important information about the organization of complex systems. There are many metrics to quantify the node centrality in networks. The paper reviews the main centrality measures and discusses their main features and limitations. The influence of network centrality on epidemic spreading and synchronization is also pointed out. Application of centrality measures to understand the function of complex systems is presented, including biological and cortical networks.

Centrality Measures in Social Networks [8]. Centrality measures are vital for understanding networks, also known as graphs. These algorithms use graph theory to calculate the importance of any given node in a network. They cut through noisy data, revealing parts of the network that need. The following centrality measures are popular. Degree centrality assigns an importance score based simply on the number of links held by each node. Betweenness centrality measures the number of times a node lies on the shortest path between other nodes. Closeness centrality scores each node based on their 'closeness' to all other nodes in the network.

Centrality and Underground Networks [9]. Novel centrality indices are defined. PathRank is a generalization of the PageRank algorithm, suitable to rank nodes of undirected graphs according to number and weight of paths in the graph. Icentr ranks nodes

of the graph by means of a combination of the weights of nodes and edges, scaled according to the distance from each node, one at a time. The paper applies the two novel indices to underground transportation networks, since these networks represent an infrastructural backbone for the transportation system of most big cities over the world. A detailed study of Boston network is proposed.

Centrality in Psychological Networks [10]. Centrality indices are a popular tool to analyze structural aspects of psychological networks. As centrality indices were originally developed in the context of social networks, it is unclear to what extent these indices are suitable in a psychological network context. This article critically examines several issues with the use of the most popular centrality indices in psychological networks: degree, betweenness, and closeness centrality. It shows how problems with centrality indices discussed in the social network literature also apply to the psychological networks.

The Spatial Grasp Model and Technology

Only most general features of the developed paradigm are included, with availability of existing extended publications on its philosophy, features, organization, and numerous applications, some in [11-24].

General Issues

Within Spatial Grasp Model and Technology a high-level operational scenario expressed in recursive Spatial Grasp Language (SGL), starting in any world point, as in Figure 6, propagates, covers, and matches the distributed environment in parallel wave-like mode. Such propagation can result in echoing of the reached states and data, which may be arbitrarily remote, to be represented as the final result, or used for higher level decisions and launching other waves. These capabilities altogether provide holistic spatial solutions unachievable by any other models and systems.

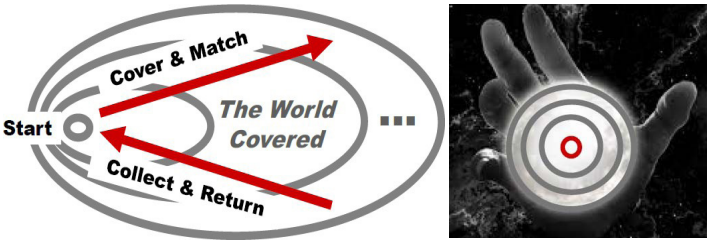

Figure 6: Parallel covering, collecting, and returning knowledge with Spatial Grasp Model.

This concept is based on quite different philosophy of dealing with large distributed systems. Instead of representing systems and solutions in them in the form of communicating parts or agents, the developed Spatial Grasp paradigm is organizing everything by the integral, holistic and parallel substance covering and conquering distributed worlds which may be as follows:

Physical World (PW), considered as continuous and infinite where each point can be identified and accessed by physical coordinates, *Virtual World (VW)*, which is discrete and consists of nodes and semantic links between them,

Executive world (EW), consisting of active "doers", which may be humans or robots, with communication possibilities between them. Different kinds of combinations of these worlds can also be possible within the same formalism.

Spatial Grasp Language (SGL)

The SGL (with many details in [12-24]) allows for organizing direct space presence and operations with unlimited powers and parallelism. Its universal recursive organization with operational scenarios called *grasp* can be expressed just by a single formula:

***grasp → constant | variable | rule* ({ *grasp*, })**

Where SGL *rule* expresses certain action, control, description or context accompanied with operands, which can themselves be any *grasps* too. Top SGL details can be expressed as:

constant	→	*information* | *matter* | *custom* | *special*
variable	→	*global* | *heritable* | *frontal* | *nodal* | *environmental*
rule	→	*type* | *usage* | *movement* | *creation* | *echoing* |
		verification | *assignment* | *advancement* | *branching* |
		transference | *exchange* | *timing* | *qualifying*

The rules, starting in some world points, can organize navigation of the world sequentially, in parallel, or any combinations thereof. They can result in staying in the same application point or can cause movement to other world points with obtained results to be left there, as in the rule's final points. Such results can also be collected, processed, and returned to the rule's starting point, the latter serving as the final one on this rule. The final world points reached after the rule invocation can themselves become starting ones for other rules. The rules, due to recursive language organization, can form arbitrary operational and control infrastructures expressing any sequential, parallel, hierarchical, centralized, localized, mixed and up to fully decentralized and distributed algorithms.

SGL Interpreter Organization

The interpreter consists of a number of specialized functional modules, as in Figure 7 and below, working with specific data structures, both serving SGL scenarios or their parts which happen to be inside this interpreter at this moment of time, also organizing exchanges with other interpreters in case of distributed SGL scenarios. Each SGL interpreter copy can handle and process multiple active SGL scenario code propagating in space and between the interpreters.

Networked SGL implementation

Communicating interpreters of SGL can be in arbitrary number of copies, up to millions and billions, which can be effectively integrated with any existing systems and communications, and their dynamic networks can represent *powerful spatial engines capable of solving any problems in terrestrial and celestial environments*. Such collective engines can simultaneously execute different cooperative or competitive scenarios without any central

resources or control. Hardware or software SGL interpreters, shown in Figure 8 as universal computational, control and management units U, may be stationary or mobile. They can be dynamically installed, and if needed created, in proper physical or virtual world points on the request of self-evolving SGL scenarios.

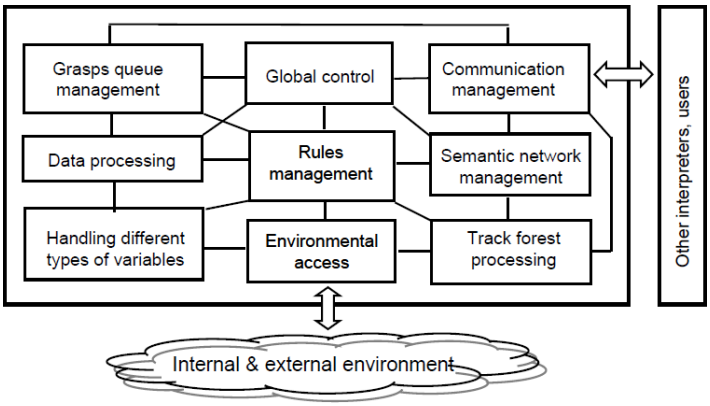

Figure 7: SGL interpreter main components and their interactions.

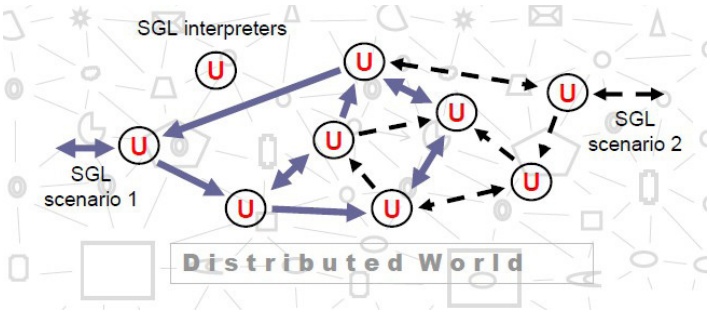

Figure 8: SGL distributed interpretation.

Spatial Tracking System

As *both backbone and nerve system of the distributed interpreter*, its self-optimizing *Spatial Track System* provides hierarchical command and control, as well as remote data and code access, It also supports spatial variables and merges distributed control states for decisions at higher organizational levels. The track infrastructure is automatically distributed between active components (humans, robots, computers, smart-phones, satellites, etc.) during scenario self-spreading in distributed environments. Some stages of track system operation symbolically shown in Figure 9.

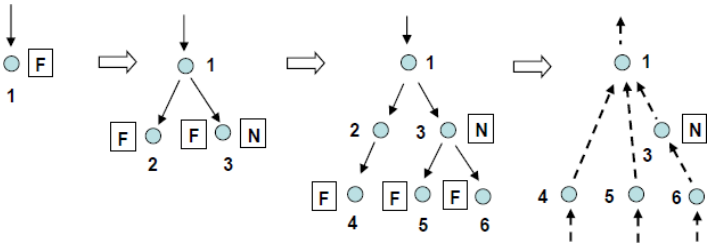

Figure 9: Different spatial track system operations. (a) Start with a single track node and frontal variable; (b) Track grows with replication of frontal variable and fixed support of nodal variable; (c) Further track growing with movement and replication of frontal variables; (d) Echoing via tracks with optimization of tracks structure and continuing support of the nodal variable.

Network Representation and Creation in SGL

The advent of social networks, big data and e-commerce has re-emphasized the importance of analyzing a unique type of data structure- one which depicts relationships among its entities, also known as a Graph. Graphs and networks can be defined as a representation of relationships between "entities" or "things" where as these "entities" are the "nodes" (also known as "vertices") of the graph and the relationships between them are represented by "links" (also known as "edges") of the graph, as shown in Figure 10.

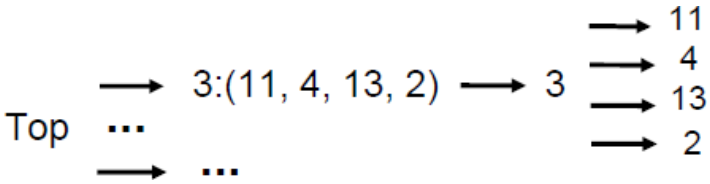

Figure 10: Graph or network example.

Network description

Any graph or network topology can be represented by a list of named nodes, each node accompanied with a set of names of direct neighbors, as follows for the network of Figure 10.

Top = (1:10, 2:(3,5,6,9,7,14,13), 3:(11,2,13,4), 4:(10,11,3,8,12), 5:2, 6:(2,9), 7:(2,9), 8:(12,4,13), 9:(7,2,6), 10:(1,4), 11:(4,3), 12:(4,8), 13:(8,3,2,14), 14:(13,2))

Network creation

The following SGL scenario creates this network from its description in the Top variable in a top-down parallel manner, by first splitting it on all nodes, and then on all neighbors of each node (as symbolically shown in Figure 11), while solving competition to create the same link from neighboring nodes by comparing their names.

```
Top = …;
align(split(Top); frontal(Next) = VAL[2];
create_node(VAL[1]));
split(Next); NAME > VAL; linkup(VAL)
```

Degree Centrality in SGL

Degree of a node is defined as the number of direct connections a node has with other nodes. Mathematically,

Degree Centrality for a node i can be defined as follows:

$D(i) = \sum_j m(i,j)$

where m (i, j) = 1 if there is a link from node i to node j.

The following SGL scenario, applied in parallel to all network nodes of Figure 10, will result with the node degrees shown at nodes in Figure 12.

output(hop(all_nodes);
append(NAME, count(hop(all_neighbors))))

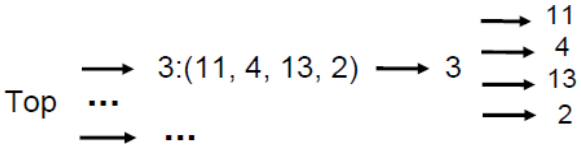

Figure 11: Top-down description splitting in parallel network creation.

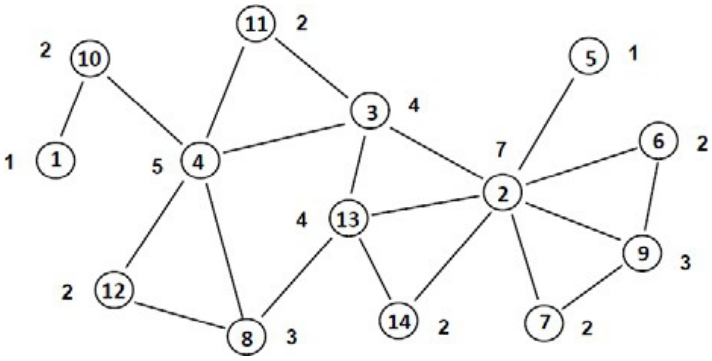

Figure 12: Degree centrality of network nodes.

The printed output will be as follows:
(1:1, 2:7, 3:4, 4:5, 5:1, 6:2, 7:2, 8:3, 9:3, 10:2, 11:2, 12:2, 13:4, 14:2)

Or if shown as a table for convenience:

Nodes:	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Degree Centrality :	**1**	**7**	**4**	**5**	**1**	**2**	**2**	**3**	**3**	**2**	**2**	**2**	**4**	**2**

As can seen, node 2 is degree centrality strongest one (with seven direct neighbors), then go node 4 (with five), nodes 3 and 13 (with four), with the weakest being nodes1 and 5 (with only one).

Closeness Centrality in SGL

Geodesic distance from node *a* to node *b*, or *d(a, b)*, can be defined as the number of edges in the shortest path between *a* and *b* (i.e. with minimum number of edges), if a path exists between them. For the example shown in Fig. 13, the shortest paths between node 13 and all other nodes will be lying on the shortest path tree covering the whole network when starting from node 13.

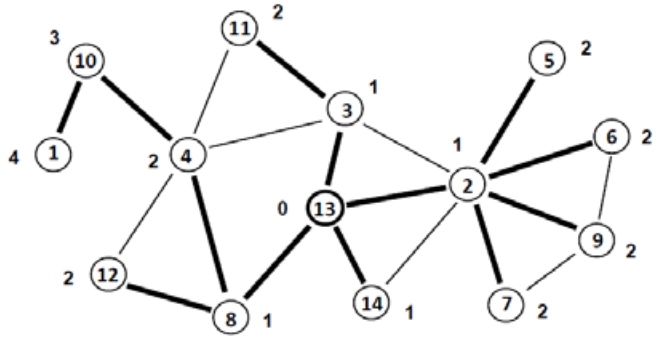

Figure 13: Example of shortest paths from node 13 to other nodes with distances to them.

To obtain the shortest path tree from node 13 embedded into the network body and output shortest distances from node 13 to all other nodes we may write the following SGL scenario, obtaining shortest distances from node 13 in other nodes in variables Distance, and finally printing all these distances together with the node names.

hop_node(13);
frontal(Far) = 0; nodal(Distance);
output(stay(repeat(hop(all_neighbors); Far += 1;
or(Distance == nil, Distance > Far);
Distance = Far));
hop(all_other_nodes); append(NAME, Distance))
SGL output: (1:4, 2:1, 3:1, 4:2, 5:2, 6:2, 7:2, 8:1, 9:2, 10:3, 11:2, 12:2, 14:1)

In the form of a table this will be as follows:

Closeness Centrality C(i) of a node *i* in a graph can be defined by the sum of shortest paths from *i* to all other nodes *j*, as follows.

$$C(i) = \sum_j d(i,j)$$

Distances to all other nodes for each node and their summary can be found by the following SGL scenario (see full results of its operation in Table 1).

output(
hop(all_nodes); IDENTITY = NAME;
frontal(Far) = 0; nodal(Distance);
stay(repeat(hop(all_neighbors); Far += 1;
or(Distance == nil, Distance > Far);
Distance = Far));
append(NAME, sum(hop(all_other_nodes); Distance)))

The above scenario starts in parallel from all nodes of the network and then creates from each node an individual shortest path tree covering the whole network (using for each tree a unique identity corresponding to the name of the node it started, not to interfere between the trees created in parallel). After obtaining the trees, embedded into the network body, the scenario then hops from each node to all other nodes and summarizes distances to them registered in its individual tree (as in Table 1). After this it collects in parallel and outputs all node names with summaries of distances to other nodes (as in the rightmost column of Table 1).

The printed node closeness centrality result reflecting the sum of distances to all other nodes will be as follows: (1:48, 2:23, 3:23, 4:26, 5:35, 6:34, 7:34, 8:28, 9:33, 10:36, 11:30, 12:35, 13:25, 14:32)

As a table this may be as follows (see also Fig. 14, with closeness centrality values associated with nodes).

Nodes:	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Closeness Centrality :	**48**	**23**	**23**	**26**	**35**	**34**	**34**	**28**	**33**	**36**	**30**	**35**	**25**	**32**

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	SUM
1	0	4	3	2	5	5	5	3	5	1	3	3	4	5	48
2	4	0	1	2	1	1	1	2	1	3	2	3	1	1	23
3	3	1	0	1	2	2	2	2	2	2	1	2	1	2	23
4	2	2	1	0	3	3	3	1	3	1	1	1	2	3	26
5	5	1	2	3	0	2	2	3	2	4	3	4	2	2	35
6	5	1	2	3	2	0	2	3	1	4	3	4	2	2	34
7	5	1	2	3	2	2	0	3	1	4	3	4	2	2	34
8	3	2	2	1	3	3	3	0	3	2	2	1	1	2	28
9	5	1	2	3	2	1	1	3	0	4	3	4	2	2	33
10	1	3	2	1	4	4	4	2	4	0	2	2	3	4	36
11	3	2	1	1	3	3	3	2	3	2	0	2	2	3	30
12	3	3	2	1	4	4	4	1	4	2	2	0	2	3	35
13	4	1	1	2	2	2	2	1	2	3	2	2	0	1	25
14	5	1	2	3	2	2	2	2	2	4	3	3	1	0	32

Table 1: Closeness from each node to all other nodes and their summary.

As can be seen, the strongest closeness centrality nodes are 2 and 3 (with same sum of distances 23), then go node 13 (with 25), node 4 (with 26), and the weakest being node 1 (with 48).

Betweenness Centrality in SGL

This metric defines and measures the importance of a node in a network based upon how many times it occurs in the shortest path between all pairs of nodes in a graph. Mathematically, **Betweenness Centrality** B(i) of a node *i* in a graph is defined as follows:

$$B(i) = \sum_{a,b} \frac{g_{aib}}{g_{ab}}$$

Where *a*, *b* is any pair of nodes in the graph, g_{aib} is the number of shortest paths from node *a* to node *b* passing through *i*, and g_{ab} is the number of shortest paths from *a* to *b*.

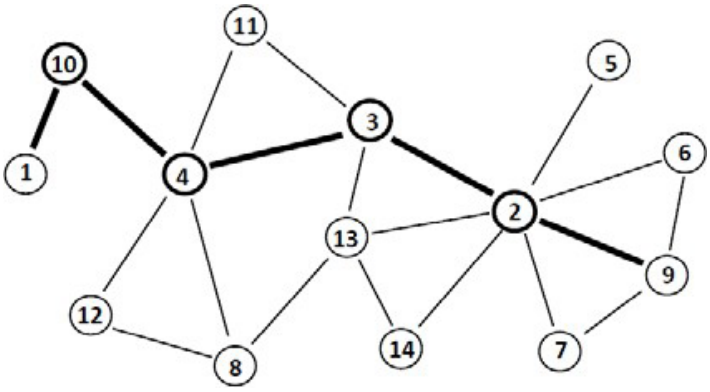

Figure 14: Closeness centrality of network nodes.

An example is shown in Fig. 15 where nodes 10, 4, 3, and 2 are on the shortest path between nodes 1 and 9, which can be found by the SGL scenario below.

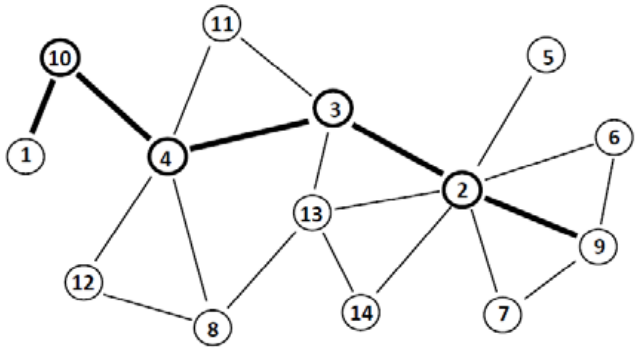

Figure 15: Example of nodes staying on the shortest path between nodes 1 and 9.

```
hop_node(1);
frontal(Far) = 0; nodal(Distance, Prev); stay(repeat(hop(all_
neighbors); Far += 1;
or(Distance == nil, Distance > Far); Distance = Far; Prev =
PREVIOUS));
hop_node(9); repeat(hop(Prev); nonequal(NAME, 1);
output(NAME))
```

SGL output will be: (2, 3, 4, 10).

The following SGL scenario will be finding and summarizing betweenness for all nodes lying on shortest paths between pairs of all other nodes

```
sequence(
(hop(all_nodes); IDENTITY = NAME;
frontal(Far) = 0; nodal(Distance, Prev);
stay(repeat(hop(all_neighbors); Far += 1;
or(Distance == nil, Distance > Far);
Distance = Far; Prev = PREVIOUS));
hop(all_other_nodes);
repeat(hop(Prev); nonequal(NAME, IDENTITY); CONTENT +=
1))),
output(hop(all_nodes); append(NAME, CONTENT)))
```

This scenario starts in parallel from all nodes of the network and then creates from each node an individual shortest path tree covering the whole network (using for each tree a unique identity corresponding to the name of the node it started, not to interfere between the trees created in parallel). After obtaining the trees, embedded into the network body, the scenario hops from each node to all other nodes and repetitively propagates up the recorded individual shortest path tree with incrementing the neutral counter (as content) of each passed node (except the starting ones). After completion of this multiple trees parallel ascending process for all nodes, the scenario then outputs the node names with their resultant betweenness summaries, accumulated as contents. Examples of its operation when repeatedly starting from the node on the left and leading to all the nodes above are shown in Table 2 (from nodes 1 and 14 only, with local summaries below, similarly from others), and the resultant global betweennes summaries on all nodes are provided in Table 3.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
F1		4	7	11						12			1	
F2			5	3				1		1			2	
F3		5		4						1			2	
F4		4	7					2		1			1	
F5		12	5	3				1		1			2	
F6		12	5	3				1	1	1			2	
F7		6	4	3				1	1	1			1	
F8		4		3						1			6	
F9		9	7	3				1		1			1	
F10		4	7	11				2					1	
F11		4	7	4						1			1	
F12		3	4	8				7		1			6	
F13			8	2				2		1				
F14		5	4	2				4		1			7	
FIN		72	70	60				22	2	24			33	

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
F1		4	7	11						12			1	
F2			5	3				1		1			2	
F3		5		4						1			2	
F4		4	7					2		1			1	
F5		12	5	3				1		1			2	
F6		12	5	3				1	1	1			2	
F7		6	4	3				1	1	1			1	
F8		4		3						1			6	
F9		9	7	3				1		1			1	
F10		4	7	11				2					1	
F11		4	7	4						1			1	
F12		3	4	8				7		1			6	
F13			8	2				2		1				
F14		5	4	2				4		1			7	
FIN		72	70	60				22	2	24			33	

Table 2: Nodes staying on shortest paths from a certain node to all other nodes and their summary: (a) From node 1, (b) From node 14.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
F1		4	7	11						12			1	
F2			5	3				1		1			2	
F3		5		4						1			2	
F4		4	7					2		1			1	
F5		12	5	3				1		1			2	
F6		12	5	3				1	1	1			2	
F7		6	4	3				1	1	1			1	
F8		4		3						1			6	
F9		9	7	3				1		1			1	
F10		4	7	11				2					1	
F11		4	7	4						1			1	
F12		3	4	8				7		1			6	
F13			8	2				2		1				
F14		5	4	2				4		1			7	
FIN		72	70	60				22	2	24			33	

Table 3: Resultant betweenness summary from the local summaries on all nodes.

The printed SGL result on the betweenness values for all nodes will be as follows: (1:0, 2:72, 3:70, 4:60, 5:0, 6:0, 7:0, 8:22, 9:2, 10:24, 11:0, 12:0, 13:33, 14:0)

In a more convenient form this may be as follows (see also Figure 16 where betweenness values are shown associated with the related network nodes).

Nodes:	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Betweenness Centrality :	0	72	70	60	0	0	0	22	2	24	0	0	33	0

As can be seen, the strongest betweenness centrality nodes are 2 (72) and 3 (70), then goes node 4 (60), whereas the weakest being nodes 1, 5, 6, 7, 11, 12, and 14 (just 0).

Eigen Vector Centrality in SGL

The last centrality feature considered in this paper is known as the **Eigen Vector Centrality**. This metric measures the importance of a node in a graph as a function of the importance of its neighbors. If a node is connected to highly important nodes, it will have a higher Eigen Vector Centrality score as compared to a node which is connected to lesser important nodes.

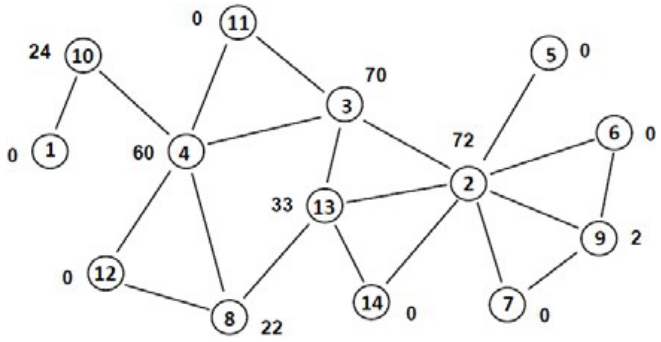

Figure 16: Betweenness Centrality of network nodes.

The following SGL scenario, starting from the degree centrality of each node (i.e. the number of its direct neighbors), calculates importance of each node as the sum of centrality values of all its direct neighbors.

```
nodal(Old, New);
sequence(
(hop(all_nodes); Old = count(hop(all_neighbors)),
output(hop(all_nodes; New = count(hop(all_neighbors); Old);
append(NAME, New)))
```

This scenario first accesses all network nodes in parallel with assigning them their degree centrality value (as an “old” one, by the number of their direct neighbors), and then reassigns to each of them the new value as a sum of previous, or old, values of their direct neighbors, considering new values as their Eigen ones, outputting the latter. The printed SGL Eigen result will be as follows:
(1:2, 2:18, 3:18, 4:13, 5:7, 6:10, 7:10, 8:11, 9:11, 10:6, 11:9, 12:8, 13:16, 14:11)

In a more convenient form (Eigen values shown in red) this will be as follows (see also Fig. 17).

Nodes:	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Eigen Vector Centrality:	2	18	18	13	7	10	10	11	11	6	9	8	16	11

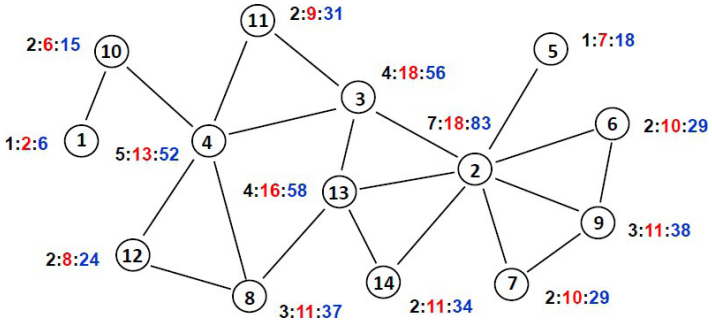

Figure 17: Eigen centrality of all nodes, step 1.

As can be seen, the strongest Eigen Centrality nodes are 2 and 3 (both with value 18), then go nodes 13 (with 16), and 4 (with 13).

This finding of new node values as a summary of previous values of all neighboring nodes may be repeated, each time receiving more global Eigen values of nodes, as follows.

```
nodal(Old, New);
sequence(
(hop(all_nodes); Old = count(hop(all_neighbors))),
loop(2)(
output(align(hop(all_nodes; New = count(hop(all_neighbors);
Old));
Old = New; append(NAME, New))))
```

For the two repetitions of the above scenario we will receive the following summary (see the table below, also Fig. 18, with the second step Eigen values shown in blue):

Nodes:	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Eigen Vector Centrality:	2	18	18	13	7	10	10	11	11	6	9	8	16	11
Eigen Vector Centrality:	6	83	56	52	18	29	29	37	38	15	31	24	58	34

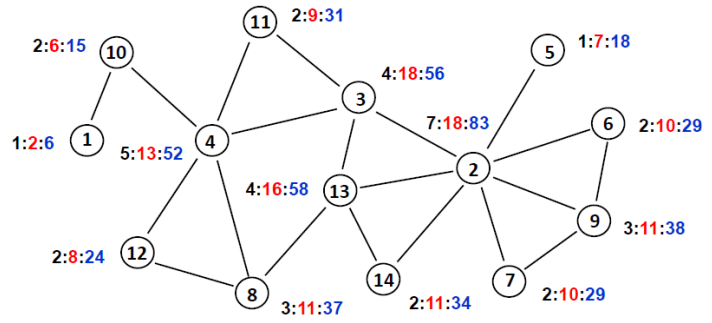

Figure 18: Eigen centrality of all nodes, two steps.

As can be seen, after the second iteration the strongest Eigen Centrality node is 2 (with value 83), then go node 13 (with 58), node 3 (with 56), and node 4 (with 52).

Growing Importance and Extended Use of Networks

A brief summary on the rapidly growing network-related publications, especially in such areas as psychology and psychopathology, with their main ideas and contents is provided below.

Visualizing Psychological Networks [25]. Networks have emerged as a popular method for studying mental disorders. Psychopathology networks consist of symptoms of mental disorders (nodes) and the connections between those aspects (edges). Unfortunately, the visual presentation of networks can occasionally be misleading. Researchers may be tempted to conclude that nodes that appear close together are highly related, and that nodes that are far apart are less related. The paper provides a brief tutorial on several methods including multidimensional scaling, principal components plotting, and eigenmodel networks.

Network Analysis in Psychology [26]. The main goal of this work is to introduce a new approach called network analysis for its application in the field of psychology. This paper presents the network model in a brief, entertaining and simple way and, as far as possible, away from technicalities and the statistical point of view. The aim of this outline is, on the one hand, to take the first steps in network analysis, and on the other, to show the theoretical and clinical implications underlying this model. The concepts of network, node and edge, the types of networks and the procedures for their estimation are all addressed.

Network analysis in psychological science [27]. In recent years, network analysis has been applied to identify and analyze patterns of statistical association in multivariate psychological data. In these approaches, network nodes represent variables in a data set, and edges represent pairwise conditional associations between variables in the data, while conditioning on the remaining variables. The Primer provides anatomy of these techniques, describes current state of the art and discusses open problems. It identifies relevant data structures in which network analysis may be applied: cross-sectional data, repeated measures and intensive longitudinal data.

Networks in psychopathology [28]. In the psychological network approach, mental disorders such as major depressive disorder are conceptualized as networks. The network approach focuses on the symptom structure or the connections between symptoms instead of the severity of a symptom. To infer a person-specific network for a patient, time-series data are needed. By far the most common model to statistically model the person-specific interactions between symptoms or momentary states has been the vector autoregressive (VAR) model. Different challenges and possible solutions are discussed in this review.

Network applications in psychology [29]. Network analysis models have been widely used in psychology research in recent years. Unlike latent variable models which conceive observable variables as outcomes of unobservable latent factors, network analysis models apply the graph theory to construct a network to depict the associations among observable variables. They reveal the

relationships among observable variables and the dynamic system resulted from the interactions between these variables. Network analysis models provide a new perspective for visualization and studying various psychological phenomena.

Conclusions

The results of this work confirm availability and efficiency of using SGT and SGL for investigating, expressing, and evaluating such important network feature as nodes Centrality, with such variants as Degree Centrality, Closeness Centrality, Betweenness Centrality, and Eigen Vector Centrality. SGL allows us to provide detailed, very clear and extremely compact solutions for the centrality problems which can operate on arbitrary large and complex networks, and in highly parallel and fully distributed mode. The solutions provided and discussed in this paper were obtained in a spatial thinking, spatial pattern recognition, and spatial pattern matching mode, rather than in traditional logic- based and algorithmic philosophy and culture. Moreover, being highly parallel and fully distributed, they challenge existing opinions that parallel and distributed algorithms are usually much more complex that traditional sequential ones for solving the same problems, just proving the opposite, and especially for the network-related problems. The latest version of SGL can be quickly implemented by a small group of system programmers, even in standard university environments, as was done for the previous versions in different countries with the author's supervision. Future plans of this work include extended application of SGT and SGL for network-based systems, especially in psychology and psychiatry, also new book on spatial management of advanced systems effectively integrating physical, virtual, and mental aspects and worlds.

References

1. https://www.mdpi.com/2079-9292/9/8/1248
2. https://www.csis.org/analysis/battle-networks-and-future-force
3. https://vietnambiz.vn/mang-luoi-kinh-te-economic-network-la-gi-uu-nhuoc-diem-cua-mang-luoi-kinh-te20191011113428212.html
4. https://www.bmc.com/blogs/virtual-network/
5. https://www.nature.com/articles/s43586-021-00055-w
6. https://towardsdatascience.com/graph-analytics-introduction-and-concepts-of-centrality-8f5543b55de3
7. https://arxiv.org/pdf/1901.07901.pdf
8. https://cambridge-intelligence.com/keylines-faqs-social-network-analysis/
9. https://www.sciencedirect.com/science/article/abs/pii/S0378437121008633
10. https://www.apa.org/pubs/journals/features/abn-abn0000446.pdf
11. Sapaty PS. A distributed processing system, European Patent N 0389655, Publ. 10.11.93, European Patent Office. 35.
12. Sapaty PS. Mobile Processing in Distributed and Open Environments. New York: John Wiley & Sons. 1999; 410.
13. Sapaty PS. Ruling Distributed Dynamic Worlds. New York: John Wiley & Sons. 2005; 255.
14. Sapaty PS. Managing Distributed Dynamic Systems with Spatial Grasp Technology. Springer. 2017; 284.
15. Sapaty PS. Holistic Analysis and Management of Distributed Social Systems. Springer. 2018; 234.
16. Sapaty PS. Complexity in International Security: A Holistic Spatial Approach. Emerald Publishing. 2019; 160.
17. Sapaty PS. Symbiosis of Real and Simulated Worlds under Spatial Grasp Technology. Springer. 2021; 251.
18. Sapaty PS. Spatial Grasp as a Model for Space-based Control and Management Systems. CRC Press. 2022; 280.
19. Sapaty PS. The Spatial Grasp Model: Applications and Investigations of Distributed Dynamic Worlds. Emerald Publishing. 2023.
20. http://www.immsp.kiev.ua/publications/articles/2023/2023_1/01_23_Sapaty.pdf
21. http://www.immsp.kiev.ua/publications/articles/2023/2023_2/02_23_Sapaty.pdf
22. https://www.davidpublisher.com/Public/uploads/Contribute/6486c3d05a6cc.pdf
23. Sapaty PS. Simulating Distributed Consciousness with Spatial Grasp Model, Mathematical machines and systems. 2023.
24. Sapaty PS. Managing Distributed Systems with Spatial Grasp Patterns, Mathematical machines and systems. 2023.
25. https://www.frontiersin.org/articles/10.3389/fpsyg.2018.01742/full
26. https://www.papelesdelpsicologo.es/pdf/2852.pdf
27. https://www.nature.com/articles/s43586-021-00055-w
28. https://www.sciencedirect.com/science/article/pii/S2352250X21000300?via%3Dihub
29. https://journal.psych.ac.cn/adps/EN/10.3724/SP.J.1042.2020.00178

Appendix: Summary of SGL Syntax and Main Constructs

Syntactic categories are shown below in italics, vertical bar separates alternatives, parts in braces indicate zero or more repetitions with a delimiter at the right, and constructs in brackets are optional. The remaining characters and words are the language symbols (including boldfaced braces).

grasp → *constant* | *variable* | [*rule*] [({ *grasp*,})]
constant → *information* | *matter* | *special* | *custom* | *grasp*
information → *string* | *scenario* | *number*
string → '{*character*}'
scenario → **{**{*character*}**}**
number → [*sign*]{*digit*}[.{*digit*}[e[*sign*]{*digit*}]]
matter → "{*character*}"
special → thru | done | fail | fatal | infinite | nil | any | all | other | allother | current | passed | existing | neighbors | direct | forward | backward | synchronous | asynchronous | virtual | physical | executive | engaged | vacant | firstcome | unique
variable → *global* | *heritable* | *frontal* | *nodal* | *environmental*
global → G{*alphameric*}
heritable → H{*alphameric*}
frontal → F{*alphameric*}
nodal → N{*alphameric*}
environmental → TYPE | NAME | CONTENT | ADDRESS | QUALITIES | WHERE | BACK | PREVIOUS | PREDECESSOR | DOER | RESOURCES | LINK | DIRECTION | WHEN | TIME | STATE | VALUE | IDENTITY | IN | OUT | STATUS
rule → *type* | *usage* | *movement* | *creation* | *echoing* | *verification* | *assignment* | *advancement* | *branching* | *ransference* | *exchange* | *timing* | *qualifying* | *grasp*
type → global | heritable | frontal | nodal | environmental | matter | number | string | scenario | constant | custom
usage → address | coordinate | content | index | time | speed | name | place | center | range | doer | node | link | unit
movement → hop | hopfirst | hopforth | move | shift | follow
creation → create | linkup | delete | unlink
echoing → state | rake | order | unit | unique | sum | count | first | last | min | max | random | average | sortup | sortdown | reverse | element | position | fromto | add | subtract | multiply | divide | degree | separate | unite | attach | append | common | withdraw | increment | decrement | access | invert | apply | location | distance
verification → equal | nonequal | less | lessorequal | more | moreorequal | bigger | smaller | heavier | lighter | longer | shorter | empty | nonempty | belong | notbelong | intersect | notintersect | yes | no
assignment → assign | assignpeers
advancement → advance | slide | repeat | align | fringe
branching → branch | sequence | parallel | if | or | and | orsequence | orparallel | andsequence | andparallel | choose | quickest | cycle | loop | sling | whirl | split | replicate
transference → run | call
exchange → input | output | send | receive | emit | get
timing → sleep | allowed
qualification → contain | release | trackless | free | blind | quit | abort | stay | lift | seize | exit