https://orcid.org/0000-0002-4168-7190

UDC 623.764

P.S. SAPATY*

MANAGING DISTRIBUTED SYSTEMS WITH SPATIAL GRASP PATTERNS

*Institute of Mathematical Machines and Systems Problems of the National Academy of Sciences of Ukraine, Kyiv, Ukraine

Анотація. *Патерни – це все навколо нас. Вони можуть відображати закономірності світу, створену людиною розробку, модель, план чи діаграму, стандартний спосіб моделювання, дії та мислення, характерний стиль чи форму, поєднання якостей і тенденцій тощо. Саме тому теорія, дослідження і практична робота з патернами настільки важливі для різних наукових і технологічних галузей. Це також спонукало до підготовки та написання цієї статті. У роботі наводиться огляд існуючих публікацій про патерни, здійснюється їх групування за різними категоріями, описуються розроблені модель і Технологія просторового захоплення (ТПЗ) та її Мова просторового захоплення (МПЗ) із розподіленою мережевою реалізацією, які забезпечують ефективні розподілені рішення в системному управлінні, контролі та моделюванні за допомогою активних моделей, що саморозповсюджуються. У статті показано, як практичні патерни можуть бути виражені за допомогою МПЗ, включаючи звичайні шаблони, шаблони конкретних об'єктів, а також різні рішення для здійснення управління на основі шаблонів, такого як координація транспортних колон, пошук координат розподіленої зони і відстеження рухомих об'єктів у просторі. Також у роботі подаються мережеві приклади розподіленого розпізнавання шаблонів і їх зіставлення з використанням саморозповсюджуваних активних мережевих шаблонів, що відображають зображення, які необхідно знайти. Стаття містить класифікований огляд досліджень використання МПЗ для здійснення операцій із патернами в різних сферах, зокрема, з описовими і творчими патернами, патернами як просторовими процесами, розпізнаванням патернів, патернів, що самостійно співставляються, комбінованих, взаємодіючих, суперечливих, психологічних і рекурсивних патернів. Робота завершується висловленням переконання, що МПЗ можна використовувати як справжню, надзвичайно ефективну та компактну мову для вираження патернів та операцій, а ТПЗ має сприяти розвитку теорії патернів та результуючих технологій.*

Ключові слова: *патерни, звичайні патерни, патерни процесів, розпізнавання патернів, зіставлення патернів, Технологія просторового захоплення, Мова просторового захоплення, мережева реалізація, активні патерни, що самостійно співставляються.*

Abstract. *The pattern is everything around us. It can represent the world's regularity, a human-made design, a model, plan, diagram, a standard way of modeling, acting and thinking, a distinctive style or form, a combination of qualities and tendencies, etc. That is why the theory, research, and practical works on patterns are so important for different scientific and technological fields, having also stimulated the preparation and writing of the current paper. The paper reviews existing works on patterns, grouping them by different categories, and briefs the developed Spatial Grasp Model and Technology (SGT) and its Spatial Grasp Language (SGL) with the distributed networked implementation, which provide effective distributed solutions in systems management, control, and simulation by active self-spreading patterns. The article shows how practical patterns can be expressed in SGL, including regular patterns, patterns of concrete objects, and different pattern-based management solutions like coordinating transport columns, finding distributed zone coordinates, and spatial tracking of mobile objects. It also gives network examples of distributed pattern recognition and matching with the use of self-propagating active network templates reflecting images to be found. The paper provides a classified summary of the investigated use of SGL for pattern operations in different areas, which includes descriptive patterns, creative patterns, patterns as spatial processes, pattern recognition, self-matching patterns, combined patterns, cooperating and conflicting patterns, psychological patterns, and recursive patterns. The work concludes with the belief that SGL can be used as a real, very effective, and compact language for pattern representation and operations, and SGT should contribute to the pattern theory and resultant technologies.*

Keywords: *patterns, regular patterns, process patterns, pattern recognition, pattern matching, Spatial Grasp Technology, Spatial Grasp Language, networked implementation, self-matching active patterns.*

DOI: 10.34121/1028-9763-2023-4-11-25

1. Introduction

The pattern is everything in the digital world. Patterns can be seen physically or observed mathematically by applying algorithms. A pattern, for example, can be as follows (see also [1–28]):

- an arrangement of lines or shapes;
- regularity in the world;
- human-made design or abstract ideas;
- unvarying way of acting or doing;
- a model, plan, or diagram used as a guide;
- a standard way of moving, acting, etc.;
- representation of a class or type;
- an example, sample, instance, or specimen;
- a distinctive style, model, or form;
- a combination of qualities, acts, tendencies, etc.;
- an original or model deserving imitation.

The aim of this paper is to investigate and analyze the applicability of the developed Spatial Grasp Model and Technology and especially its basic Spatial Grasp Language for the representation and implementation of different types of patterns that can be used in the simulation and management of a variety of distributed dynamic systems. These systems may cover such areas as education, economy, science, ecology, psychology, security, defense, international relations, space research, and many others.

The rest of the paper is organized as follows. Section 2 reviews existing works on patterns, grouping them by the following categories: pattern definition and theory, different pattern types, pattern matching and recognition, and pattern languages. Section 3 briefs the developed Spatial Grasp Model and Technology, discussing general model features, its Spatial Grasp Language (SGL), and networked SGL implementation. Section 4 shows how practical patterns can be expressed under SGT, like creating regular patterns, patterns of concrete objects, and different pattern-based management solutions (including managing transport columns, finding distributed zone coordinates, and spatial tracking of mobile objects). It also gives some network examples of distributed pattern recognition and matching solutions. Section 5 provides a summary of the investigated use of SGL, which can be represented as a real and universal pattern language. This summary is grouped for descriptive patterns, creative patterns, patterns as spatial processes, pattern recognition, self-matching patterns, combined patterns, cooperating and conflicting patterns, psychological patterns, and recursive patterns. Section 6 concludes the paper with the belief in the applicability and effectiveness of SGL and SGT for expressing, representing, and processing patterns in different areas. The references contain many discovered and analyzed pattern-based sources, as well as published papers and books on SGT and SGL.

2. Review of existing works on patterns

After being searched for, discovered, analyzed, compared, discussed, and classified, the existing pattern-based ideas, sources, and publications can be grouped as follows.

Pattern definition and theory

The pattern [1] is regularity in the world, human-made design, or abstract ideas. As such, the pattern elements repeat in a predictable manner. A geometric pattern is a kind of pattern formed of geometric shapes and typically repeated like a wallpaper design. Any of the senses may directly observe patterns. Conversely, abstract patterns in science, mathematics, or language may be observable only through analysis. See pattern examples in Fig. 1.

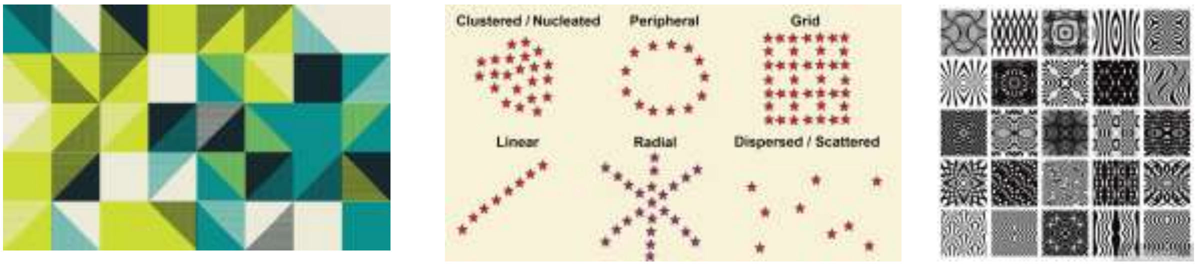


Figure 1 – Geometric pattern examples

Pattern Theory: From Representation to Inference [2] provides a comprehensive and accessible overview of the modern challenges in signal, data, and pattern analysis in speech recognition, computational linguistics, image analysis, and computer vision. It is aimed at graduate students in biomedical engineering, mathematics, computer science, and electrical engineering with a good background in mathematics and probability.

Pattern Theory-based Interpretation of Activities [3] presents a novel framework based on Grenander's pattern theoretic concepts for high-level interpretation of video activities. This framework allows us to elegantly integrate ontological constraints and machine learning classifiers in a single formalism to construct high-level semantic interpretations that describe video activity.

Pattern Theory: A Unifying Perspective [4] introduced the term «pattern theory» as a name for a field of applied mathematics that gave a theoretical framework for a large number of related ideas, techniques, and results from such fields as computer vision, speech recognition, image and acoustic signal processing, pattern recognition, neural nets, and parts of artificial intelligence. Pattern theory contains the foundational elements of a universal theory of thought itself, one of which stands in opposition to the accepted analysis of thought in terms of logic.

Pattern theory [5] is a mathematical formalism used to describe the knowledge of the world as patterns. It differs from other approaches to artificial intelligence in that it does not begin with assigning algorithms and machinery to recognize and classify patterns, it uses a vocabulary to articulate and recast the pattern concepts in precise language. Pattern theory spans algebra and statistics, as well as local topological and global entropic properties.

Different pattern types

Patterns of distributed systems are discussed in [6]. Distributed systems provide a particular challenge to a program. Despite this, many organizations rely on a range of core distributed software for handling data storage, messaging, system management, and computing capability. These systems face common problems which they solve using similar solutions. The paper recognizes and develops these solutions as patterns that help to understand, communicate, and teach distributed system design.

Statistical descriptions of spatial patterns are discussed in [7]. Spatial statistics can be defined as a statistical description of spatial data and a spatial pattern or process. Spatial statistics allow a quantitative description along with indications of statistical significance in observational data on a pattern or a process operating in space. This quantitative and statistical description allows the exploration and modeling of spatial patterns and processes and their relationships with other spatial phenomena.

Recursive pattern: a technique for visualizing very large amounts of data is considered in [8]. An important goal of visualization technology is to support the exploration and analysis of very large amounts of data. This paper proposes a new visualization technique called a «recursive pattern» which has been developed for visualizing large amounts of multidimensional data. The technique is based on a generic recursive scheme that generalizes a wide range of pixel-oriented arrangements for displaying large data sets.

The focus on the hidden pattern: a patternist philosophy of mind is made in [9]. The Hidden Pattern presents a novel philosophy of mind, intended to form a coherent conceptual framework within which it is possible to understand the diverse aspects of mind and intelligence in a unified way. The central concept of the philosophy presented is the concept of «pattern»: minds and the world they live in and co-create are viewed as patterned systems of patterns, evolving over time.

Workflow control-flow patterns are studied in [10]. The Workflow Patterns Initiative aims at delineating fundamental requirements that arise during business process modeling and describing them in an imperative way. The first deliverable of this project was a set of twenty patterns describing the control-flow perspective of workflow systems. These patterns have been widely used by practitioners, vendors, and academics in the selection, design, and development of workflow systems.

Workflow patterns are also considered in [11]. Requirements for workflow languages are indicated through workflow patterns. In this context, patterns address business requirements in an imperative workflow style expression. The paper describes a number of workflow patterns that can identify comprehensive workflow functionality. These patterns provide a basic comparison of a number of commercially available workflow management systems. See related pattern examples in Fig. 2.

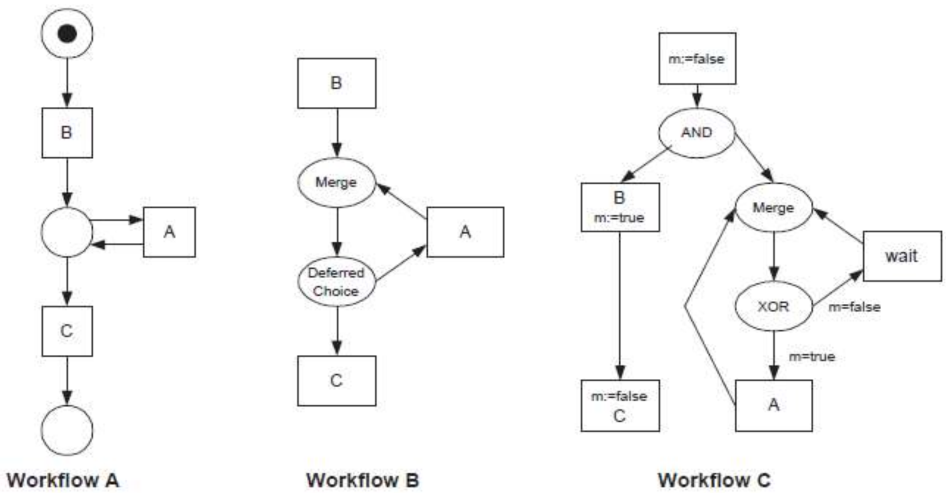


Figure 2 – Examples of workflow patterns

Pattern matching and recognition

Pattern matching is discussed in [12]. In computer science, pattern matching is the act of checking a given sequence of tokens for the presence of the constituents of any pattern. In contrast to pattern recognition, the match usually has to be exact: either it will or will not be a match. The uses of pattern matching include outputting the locations of a pattern within a token sequence to output some component of the matched pattern and to substitute the matching pattern with another token.

Fast graph pattern matching is studied in [13]. Due to the rapid growth of Internet technology and new scientific and technological advances, the number of applications that model data as graphs increases because graphs have high expressive power to model complicated structures. The dominance of graphs in real-world applications requires new graph data management so that users can access graph data effectively and efficiently. This paper studies a graph pattern-matching problem over a large data graph.

Pattern matching in massive metadata graphs at scale is presented in [14]. Pattern matching in graphs, that is finding sub-graphs that match a smaller template graph within the large background graph, is fundamental to graph analysis and serves a wide range of applications. Existing solutions have limited scalability, are difficult to parallelize, support only a limited set of search patterns, and/or focus on only a subset of the real-world problems. This work explores ways of designing a scalable solution for subgraph pattern matching. See the related example in Fig. 3.

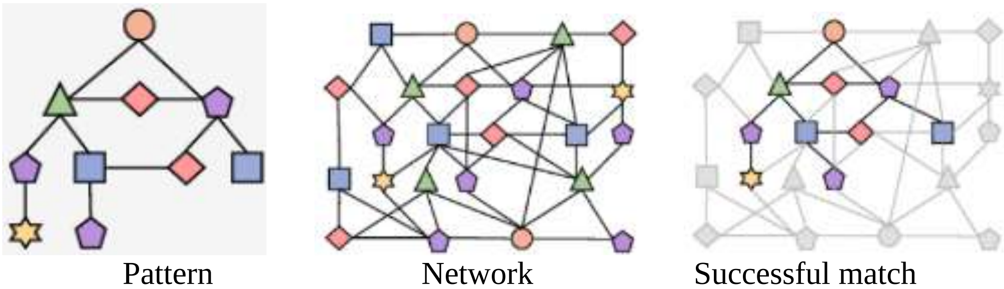


Figure 3 – Network pattern matching example

Pattern recognition [15] is the automated recognition of patterns and regularities in data. It has applications in statistical data analysis, signal processing, image analysis, information retrieval, bioinformatics, data compression, computer graphics, and machine learning. Pattern recognition has its origins in statistics and engineering; some modern approaches to pattern recognition include the use of machine learning.

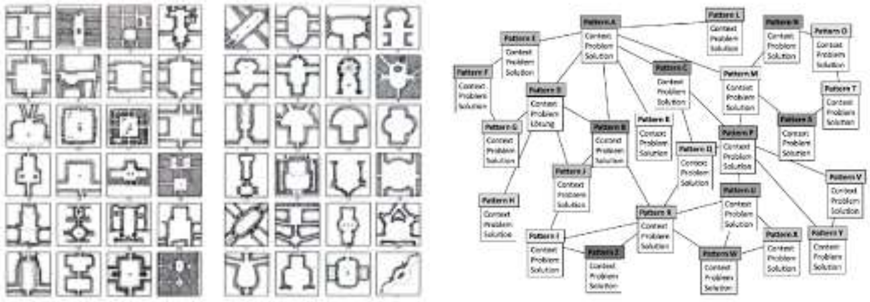
Consciousness as pattern recognition is shown in [16]. This is proof of the strong AI hypothesis that machines can be conscious. It is a phenomenological proof that pattern recognition and subjective consciousness are the same activity expressed using different terms. Therefore, it proves that the essential subjective processes of consciousness are computable and identifies significant features and requirements of a conscious system.

Pattern recognition (psychology) is discussed in [17]. In psychology and cognitive neuroscience, pattern recognition describes a cognitive process that matches information from a stimulus with information retrieved from memory. Pattern recognition occurs when information from the environment is received and entered into short-term memory, causing automatic activation of specific content in long-term memory.

Pattern languages

A pattern language is discussed in [18]. The language description – the vocabulary – is a collection of named, described solutions to problems in a field of interest. These are called design patterns. So, for example, the language for architecture describes items like settlements, buildings, rooms, windows, latches, etc. Each solution includes syntax, a description that shows where the solution fits in a larger, more comprehensive or more abstract design. A pattern language is an organized and coherent set of patterns, each of which describes a problem and the core of a solution that can be used in many ways within a specific field of expertise.

In addition, a pattern language is also considered in [19]. The book creates a new language, what the authors call a pattern language derived from timeless entities called patterns. As they write in the introduction, all 253 patterns together form a language. The pattern language is structured as a network, where each pattern may have a statement referenced to another pattern by placing that pattern's number in brackets. See some related pattern language examples in Fig. 4.



a *b*

Figure 4 – Specific patterns (a) and their network (b) as a pattern language

What is a pattern language is discussed in [20]. In this work, pattern language is an attempt to express the deeper wisdom of what brings aliveness within a particular field of human endeavor, through a set of interconnected expressions arising from that wisdom. Aliveness is one placeholder term for «the quality that has no name»: a sense of wholeness, spirit, or grace, that while of varying form, is precise and empirically verifiable.

«Pattern language: towns, buildings, and construction» is studied in [21]. Patterns, the units of this language, are answers to design problems. More than 250 patterns in this pattern language are given: each consists of a problem statement, a discussion of the problem with an illustration, and a solution. Many of the patterns are archetypal, so deeply rooted in the nature of things that it is likely that they will be a part of human nature and human action (see Fig. 4).

A pattern language for a pattern language structure is analyzed in [22]. This paper aims to help the «writers» of pattern languages create better pattern languages. It focuses not on the aesthetics of pattern languages but on their structure: how patterns work together to build a system. The paper assumes that a pattern language is a designed system and, therefore, the theory of system design and evolution underlies the language.

Growing a pattern language (for security) is discussed in [23]. The paper presents a pattern language containing all security patterns that have been published in various places. It describes the mechanism of growing this pattern language: how to catalog the security patterns from books, papers, and pattern catalogs; how to classify the patterns to help developers find appropriate patterns; and how to identify and describe the relationships between patterns in the pattern language.

The idea of pattern languages as the media for a creative society is described in [24]. This paper proposes new languages for basic skills in the creative society where people create their own goods, tools, concepts, knowledge, and mechanisms with their own hands: the skills of learning, presentation, and collaboration. These languages are written as a pattern language which is a way of describing the tacit practical knowledge.

Change making patterns and a pattern language for fostering social entrepreneurship are presented in [25]. A pattern language named «Change Making Patterns» was created by conducting interviews with social entrepreneurs. The objective of these patterns is to encourage more individuals to take their own actions to make a better world with fewer social problems. The background of the patterns and ideas on how they can be applied for social entrepreneurial education are provided.

Pattern languages in interaction design, structure and organization are discussed in [26]. When individual patterns for interaction design started to appear, the issue of structuring collections of patterns into pattern languages became relevant both from a theoretical and a practical perspective. This paper investigates how pattern languages in interaction design can be structured in a meaningful and practical way. A top-down approach is taken where patterns for interaction design are organized hierarchically.

A pattern language for pattern writing is discussed in [27]. As the pattern community has accumulated experience in writing and reviewing patterns and pattern languages, the authors have begun to develop insight into pattern-writing techniques and approaches that are particularly effective at addressing certain recurring problems. This pattern language attempts to capture some of these «best practices» of pattern writing both by describing them in pattern form and by demonstrating them in action.

3. Spatial Grasp Model and Technology

Only the general features of the developed paradigm and some existing extended publications on its philosophy, features, organization, and numerous applications, some in [28–44] are included.

General information

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 Fig. 5, propagates, covers, and matches the distributed environment in a parallel wave-like mode. Such propagation can result in echoing of the reached states and data, which may be arbitrarily remote, 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.

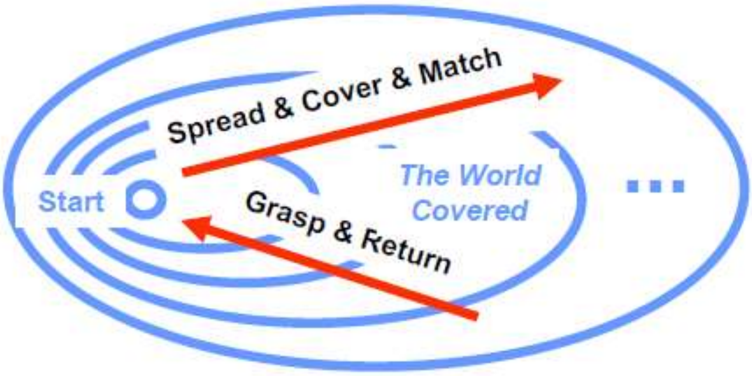


Figure 5 – Distributed matching, collecting, and returning knowledge with Spatial Grasp Model

Spatial Grasp Language (SGL)

SGL (more information about it can be found in [31–43]) 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 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 follows:

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 at some world points, can organize the navigation of the world sequentially, in parallel, or in any combination. They can result in staying at the same application point or cause the movement to other world points with the 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 for this rule. The final world points reached after the rule invocation can themselves become starting ones for other rules. Due to recursive language organization, the rules can form arbitrary operational and control infrastructures expressing any sequential, parallel, hierarchical, centralized, localized, mixed, and up to fully decentralized and distributed algorithms.

Networked SGL implementation

Each SGL interpreter copy can handle and process multiple active SGL scenario codes propagating in space and between the interpreters. Integrated with any distributed systems and networks, the SGL interpretation network can form a spatial computer with unlimited power for processing, simulation, and management of distributed systems and worlds. Having a recursive self-spreading, self-controlled, and super-virus nature, this paradigm can dynamically establish and maintain superior power over any system (including creating them from scratch and providing local and global awareness from any point to the same or other points, whether inside or outside the system).

4. Patterns under SGT

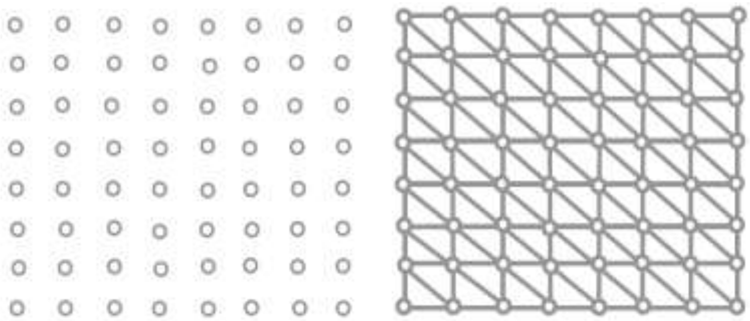
Different types of practical patterns expressed in SGL are discussed below.

4.1. Creating patterns

Regular patterns

The two-step result of the following SGL scenario is shown in Fig. 6.

```
nodal(Xmin = ..., Ymin = ...; Step = ..., Number = ...);
frontal(Xcurr), Link = type);
sequence(
  (Xcoord = array(Xmin, Step, Number);
   Ycoord = array(Ymin, Step, Number);
   parallel(split(Xcoord); Xcurr = VALUE);
   parallel(split(Ycoord); Ycurr = VALUE);
   create_node(Xcurr, Ycurr)),
  (hop_nodes(all);
   parallel(
     linkup(Link, node(shift_X(Step)),
     linkup(Link, node(shift_Y(-Step)),
     linkup(Link, node(shift_X(Step)); shift_Y(-Step)))))
```

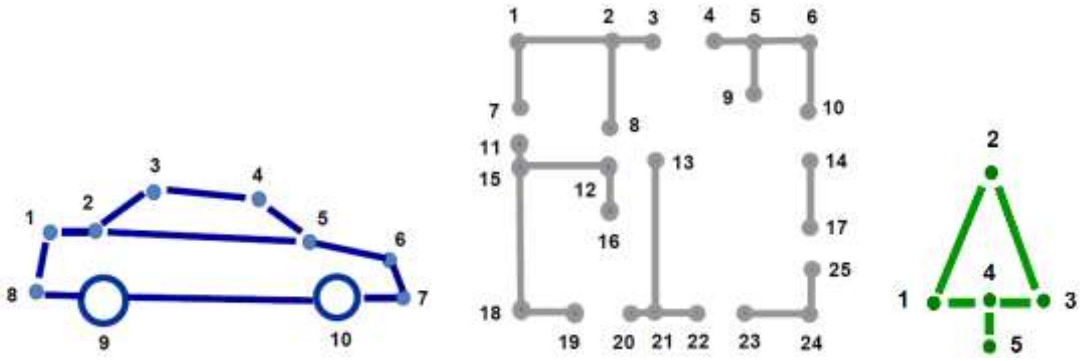


Step 1 Step 2

Figure 6 – Creating a regular pattern

Patterns of concrete objects

The results of the following SGL scenarios are shown in Fig. 7.



Car Flat Tree

Figure 7 – Patterns of different objects

Car pattern

```
frontal(N = ..., L = blue, D = diameter);
create(N:2; L_N:1; L_N:8; L_(N:9, D); L_(N:10, D); L_N:7;
       L_N:6; L_N:5; (L_N:4; L_N:3), stay); linkup(L_N:2)
```

Flat pattern

```
frontal(N = ..., L = grey);
create(
  (N:1; L_N:7, (L_N:2; L_N:8, L_N:3),
  (N:6; L_N:10, (L_N:5; L_N:9, L_N:4)),
  (N:18; L_N:19, (L_N:15; L_N:11, (L_N:12; L_N:16))),
  (N:21: L_N:20, L_N:13, L_N:22),
  (N:24; L_N:23, L_N:25),
  (N:14; L_N:17))
```

Tree pattern

```
frontal(N = ..., L = green);
create(N:5; L_N:4; L_N:1; L_N:2; L_N:3); linkup(L_N:4)
```

4.2. Pattern-based practical solutions

Managing a transport column

Starting from the head vehicle, the SGL scenario propagates along the column and reduces gaps between vehicles, also leaving repetitive gap-analyzing processes in each node, as in Fig. 8.

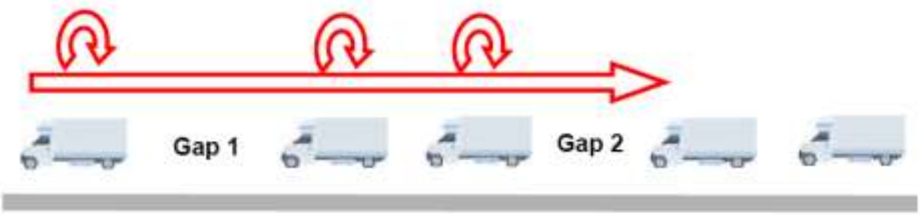


Figure 8 – Correcting the distances in the vehicle column

```
frontal(Standard) = ...;
hop_vehicle(first);
repeat(
  free(nonequal(vehicle, first);
       repeat(if(distance(vehicle_ahead) > Standard,
                 reduce(distance(ahead), Standard),
                 stay)));
  hop_vehicle(behind))
```

Collecting zone coordinates

The following scenario, starting at some point near the distributed area, collects its border coordinates simultaneously in two directions, finally appending the obtained results to each other as the united sequence, as in Fig. 9.

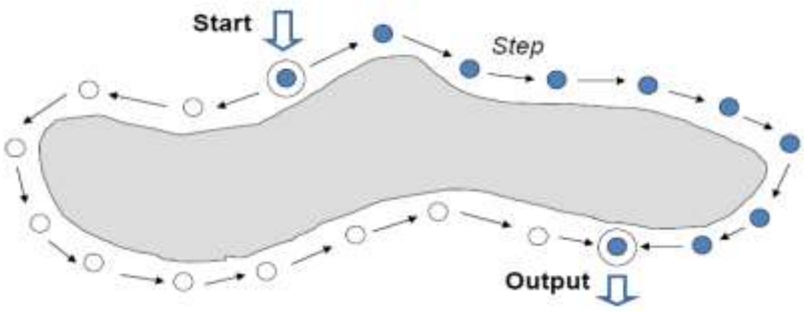


Figure 9 – Collecting zone coordinates in two directions

```
go(Start); frontal(Coord1, Step); nodal(Coord2);
parallel(
  repeat(append(Coord1, WHERE);
         if(seen(close, other),
            done(hop(other; output(append(Coord1, re-
verse(Coord2)));
         go(right, Step, Distance)),
  repeat(go(left, Step, Distance); append(Coord2, WHERE);
         if(seen(close, other), terminate)))
```

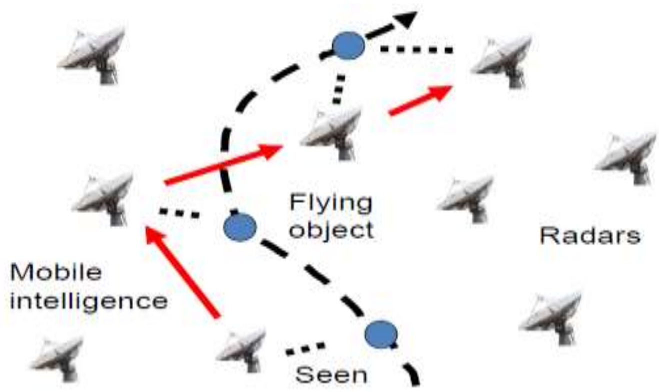


Figure 10 – Tracking of distributed objects with a radar network

Tracking mobile objects

The following self-evolving spatial scenario tracks and controls complex objects propagating via a distributed network of radar stations (see Fig. 10).

```
hop(all_nodes);
frontal(Object);
whirl(
  Object = search(aerial,
new));
  visibility(Object) >
threshold;
  repeat(
```

```
loop(visibility(Object) > Threshold);
max_destination(hop(all_neighbors); visibility(Object));
visibility(Object) > Threshold))
```

4.3. Pattern recognition and matching

Pattern recognition

The following scenarios recognize proper nodes and peculiar structures in a distributed network, as in Fig. 11.

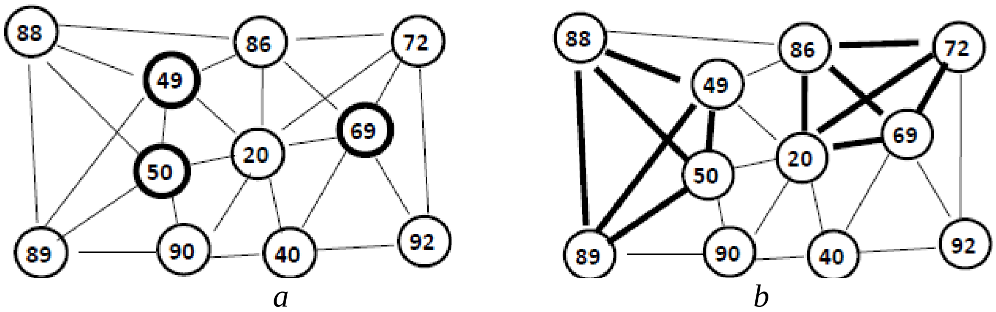


Figure 11 – Finding proper nodes (a); discovering structures (b)

Finding all nodes with 5 links to other nodes

```
output(hop_nodes(all); count(hop_links(all)) == 5; NAME)
```

Answer: 49, 50, 69.

Finding all cliques with 4 nodes.

```
hop_nodes(all);
frontal(Clique = NAME, Number = 4);
repeat(
  hop_links(all); notbelong(NAME, Clique);
  true(and_parallel(hop(link(any), nodes_all(Clique))));
  append(Clique, NAME);
  if(count(Clique) == Number, done(output(Clique))))
```

Pattern matching

The following SGL scenario finds an exact match of the given image in a distributed network, see also Fig. 12.

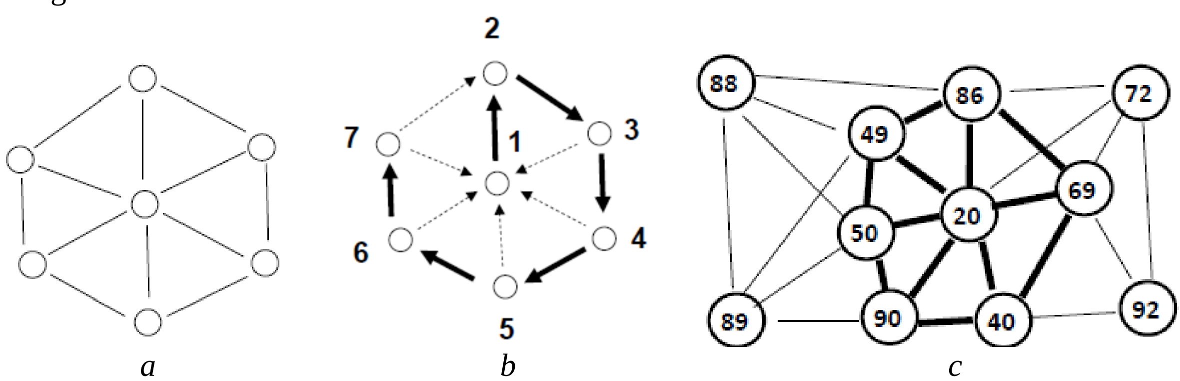


Figure 12 – Graph image (a); its active template for finding an exact match (b); the match found in the network (c)

```
hop_first_nodes(all); COLOR = Match = NAME;
hop_first_links(all); Match = NAME;
repeat(5)(hop_first_links(all);
          true(hop_links(any, node(Match[1])));
hop_links(any, node(match[2]); output(Match)
```

5. SGL as a real pattern language

After the detailed analyses, practical implementation, and use of pattern-based solutions in many areas [28–44], SGL cannot be treated just as a networked set of specific patterns like the existing pattern languages [18–27]. On the contrary, it reflects the universal paradigm capable of expressing and using any individual or general patterns and their systems, whether passive or active, as well as combining descriptive patterns with any centralized and distributed processes. A summary of this may be as follows, which can confirm that SGL can represent a real and very effective pattern language with practical applications in numerous areas.

Descriptive patterns

Any pattern parts, their compositions, and structures, with local or global parameters, can be naturally described in SGL and subsequently used for pattern recognition and matching by special processes.

Creative patterns

Patterns describing the composition and structures of different images, whether regular or arbitrary, can themselves be active parallel processes capable of creating or drawing such images autonomously in distributed spaces.

Patterns as spatial processes

These can provide any investigation, control, management, and simulation of any system, which may have a terrestrial or celestial nature. They can also express and simulate large distributed spatial processes like the spread of global viruses and fighting them.

Pattern recognition

This can be effectively organized in any physical, virtual, or combined system where the pattern recognition process relies on detailed or generalized pattern qualification and representation data expressed in logical, numerical, or structural form.

Self-matching patterns

Any graph-based pattern can be organized as active self-matching templates autonomously evolving and spreading in distributed spaces in a wave-like parallel manner. They can return names, addresses, and/or physical coordinates of the found elements and their structures, as well as influence the matched systems.

Combined patterns

Any combinations of the above-mentioned and following patterns, of any complexity, that can recognize, match, analyze, change, and control any distributed systems can be effectively organized. These patterns may virtually or physically migrate and fly over the covered systems in a controlled virus-like mode.

Cooperating and conflicting patterns

Different self-powered patterns may compete or cooperate in distributed systems and spaces, for example, representing collectives working on common problems or reflecting opposing units on a battlefield competing and destroying each other.

Psychological patterns

As it was already shown and discussed in SGT-related publications ([31, 32, 34] and others), the main gestalt psychology ideas, laws, and examples, based on mentally grasping the whole of concepts and events first, can be effectively represented and modeled with the use of spatial patterns.

Recursive patterns

Patterns of any complexity can be effectively organized in SGL using its universal recursive structure, with any above-mentioned pattern capable of including any other patterns, each of which can do the same, and to an unlimited depth.

6. Conclusions

The paper concludes with a strong belief that SGT reflects the universal paradigm capable of expressing, creating, and using any individual or general patterns and their systems, and SGL can be used as a real, effective, and very compact language for pattern representation and operations, truly named as «pattern language». SGT can believably contribute to pattern theory and resultant technologies as a universal model treating the whole world, whether physical, virtual, psychological, or mental, as spatial patterns. 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 than traditional sequential ones for solving the same problems, just proving the opposite, and especially for the pattern and network-related problems. The plans for the near future include further improvement of pattern-based and SGT-related techniques, also extended publication of this and related material in the new book, currently in preparation.

REFERENCES

1. Pattern. URL: https://en.wikipedia.org/wiki/Pattern.

2. Grenander U., Miller M. Pattern Theory: From Representation to Inference. Oxford University Press, 2007. 596 p.

3. de Souza F.D.M., Sarkar S., Srivastava A., Su J. Pattern Theory-based Interpretation of Activities. *22nd International Conf. on Pattern Recognition*. 2014. URL: https://projet.liris.cnrs.fr/imagine/pub/proceedings/ICPR-2014/data/5209a106.pdf.

4. Mumford D. Pattern Theory: A Unifying Perspective. URL: https://www.dam.brown.edu/people/ mumford/vision/papers/1994c-96--PattThUnifyingPersp-NC.pdf.

5. Pattern theory. URL: https://en.wikipedia.org/wiki/Pattern_theory.

6. Joshi U. Patterns of Distributed Systems. URL: https://martinfowler.com/articles/patterns-of-distributed-systems/.

7. Sankey T. Statistical Descriptions of Spatial Patterns. Encyclopedia of GIS. Springer, Cham, 2017. URL: https://link.springer.com/referenceworkentry/10.1007/978-3-319-17885-1_1351. DOI: 10.1007/978-3-319-17885-1_1351.

8. Keim D.A., Kröger P., Ankerst M. Recursive pattern: A technique for visualizing very large amounts of data. *IEEE Xplore: Conf. Visualization*. 1995. URL: https://www.researchgate.net/publication/3618236_Recursive_pattern_A_technique_for_visualizing_very_large_amounts_of_data. DOI: 10.1109/VISUAL.1995.485140.

9. Goertzel B. The Hidden Pattern: A Patternist Philosophy of Mind. Brown Walker Press, 2006. 424 p.
10. Russell N., ter Hofstede A.H.M., van der Aalst W.M.P., Mulyar N. Workflow control-flow patterns: a revised view. BPMcenter, 2006. 134 p.
11. W.M.P. van der Aalst, A.H.M. der Hofstede, Kiepuszewski B., Barros A.P. Workflow Patterns, 2002. URL: http://www.workflowpatterns.com/documentation/documents/wfs-pat-2002.pdf.
12. Pattern matching. URL: https://en.wikipedia.org/wiki/Pattern_matching.
13. Cheng J., Xu Yu J., Ding B., Yu P.S., Wang H. Fast Graph Pattern Matching. URL: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/icde08gsearch.pdf.
14. Reza T. Pattern matching in massive metadata graphs at scale. University of British Columbia, 2019. URL: https://open.library.ubc.ca/soa/cIRcle/collections/ubctheses/24/items/ 1.0387453. DOI: 10.14288 /1.0387453.
15. Pattern recognition. URL: https://en.wikipedia.org/wiki/Pattern_recognition.
16. V. De Walker R. Consciousness is Pattern Recognition. Copyright, 2016. URL: https://arxiv.org/abs/1605.03009.
17. Pattern recognition (psychology). URL: https://en.wikipedia.org/wiki/Pattern_recognition _(psychology).
18. Pattern language. URL: https://en.wikipedia.org/wiki/Pattern_language.
19. A Pattern Language: Towns, Buildings, Construction. 1977. URL: https://en.wikipedia.org/wiki/A_Pattern_Language.
20. What is a Pattern Language? URL: https://groupworksdeck.org/pattern-language.
21. Alexander Ch., Ishikawa S., Silverstein M., Jacobson M., Fiksdahl-King I., Angel Sh. A Pattern Language: Towns, Buildings, Construction. Oxford University Press, 1977. 1171 p. URL: https://www.amazon.com/Pattern-Language-Buildings-Construction-Environmental/dp/ 0195019199.
22. Winn T., Calder P. A pattern language for pattern language structure. *Third Asian Pacific Conference on Pattern Languages of Programs*. KoalaPLoP, 2002. URL: https://www.researchgate.net/ publication/228944468_A_pattern_language_for _pattern_language_structure.
23. Hafiz M., Adamczyk P., Johnson R. Growing a Pattern Language (for Security). *Proc. of the ACM international symposium on new ideas, new paradigms, and reflections on programming and software*. 2012. October. URL: https://www.researchgate.net/publication/262211609_Growing_a_pattern_ language_for_security. DOI: 10.1145/2384592.2384607.
24. Iba T. Pattern Languages as Media for the Creative Society. *Journal of Information Processing and Management.* 2013. Vol. 55 (12). URL: https://www.researchgate.net/publication/255484746_Pattern_ Languages_as_Media_for_the_Creative_Society. DOI: 10. 1241/johokanri.55.865.
25. Shimomukai E., Nakamura S., Iba T. Change Making Patterns. A Pattern Language for Fostering Social Entrepreneurship. CreativeShift Lab, 2015. 112 p.
26. van Welie M., van der Veer G. Pattern languages in interaction design: structure and organization. October 2011. URL: https://www.researchgate.net/publication/228881522_Pattern_languages_in _interaction_design_Structure_and_organization.
27. Meszaros G., Doble J. A Pattern Language for Pattern Writing. The Hillside Group. URL: https://hillside.net/index.php/a-pattern-language-for-pattern-writing.
28. Sapaty P.S. A distributed processing system. European Patent N 0389655, Publ. 10.11.93, European Patent Office.
29. Sapaty P.S. Mobile Processing in Distributed and Open Environments. New York: John Wiley & Sons, 1999. 410 p.
30. Sapaty P.S., Ruling Distributed Dynamic Worlds. New York: John Wiley & Sons, 2005. 255 p.
31. Sapaty P.S. Managing Distributed Dynamic Systems with Spatial Grasp Technology. Springer, 2017. 284 p.
32. Sapaty P.S. Holistic Analysis and Management of Distributed Social Systems. Springer, 2018. 234 p.
33. Sapaty P.S. Complexity in International Security: A Holistic Spatial Approach. Emerald Publishing, 2019. 160 p.
34. Sapaty P.S. Symbiosis of Real and Simulated Worlds under Spatial Grasp Technology. Springer, 2021. 251 p.
35. Sapaty P.S. Spatial Grasp as a Model for Space-based Control and Management Systems. CRC Press, 2022. 280 p.

36. Sapaty P.S. The Spatial Grasp Model: Applications and Investigations of Distributed Dynamic Worlds. Emerald Points, 2023. 184 p.
37. Sapaty P.S. Relation of Spatial Grasp Paradigm to Higher Psychological and Mental Concepts. *Acta Scientific Computer Science*. 2022. Vol. 4, Issue 12. URL: https://actascientific. com/ASCS/pdf/ASCS-04-0359.pdf.
38. Sapaty P.S. Seeing and Managing Distributed Worlds with Spatial Grasp Paradigm. Acta Scientific Computer Sciences. 2022. Vol. 4, Issue 12. URL: https://actascientific.com/ASCS/ pdf/ASCS-04-0365.pdf.
39. Sapaty P.S. Comprehending Distributed Worlds with the Spatial Grasp Paradigm. *Mathematical machines and systems*. 2022. N 1. URL: http://www.immsp.kiev.ua/publications/articles/2022/2022_1/01_22_Sapaty.pdf.
40. Sapaty P.S. Spatial management of air and missile defence operations. *Mathematical machines and systems.* 2023. N 1. URL: http://www.immsp.kiev.ua/publications/articles/2023/2023_1/01_23_Sapaty.pdf.
41. Sapaty P.S. Providing distributed system integrity under Spatial Grasp Technology. *Mathematical machines and systems*. 2023. N 2. URL: http://www.immsp.kiev.ua/publications/articles/2023/2023_2/02_23_Sapaty.pdf.
42. Sapaty P.S. Providing Global Awareness in Distributed Dynamic Systems. *International Relations and Diplomacy*. 2023. Vol. 11, N 2. URL: https://www.davidpublisher.com/Public/uploads/ Contribute/6486c3d05a6cc.pdf. DOI: 10.17265/ 2328-2134/2023.02.002.
43. Sapaty P.S. Simulating distributed consciousness with Spatial Grasp Model. *Mathematical machines and systems*. 2023. N 3. URL: http://www.immsp.kiev.ua/publications/ articles/2023/2023_3/03_23_Sapaty.pdf.
44. Sapaty P.S. Providing integrity, awareness, and consciousness in distributed dynamic systems (new book summary). *Mathematical machines and systems*. 2023. N 1. URL: http://www.immsp.kiev.ua/ publications/articles/2023/2023_1/01_23_Sapaty_anons.pdf.

Стаття надійшла до редакції 12.07.2023