Graph-Based Problem-Solving and Representation: Levels of Deployment in Computational Design Process

Dragana Ćirić
Research Assistant Professor / Scientific Associate, Independent, unit [d], Belgrade, Serbia
unit.d.dciric@gmail.com ORCID: https://orcid.org/0000-0001-5778-5599

Abstract: The paper addresses the question of the deployment of 'dependency graph'-based problem-solving and representation methods. The question is observed in terms of decomposition levels of the computational design process and the subject of computation, or graph representation and modelling. Based on the stated criteria that firstly distinguish process graphs from object-based or formally congruent graphs, and secondly identify sublevels within the formed classes, the paper explains 1. two levels of the first group (1a. the level of a complete design problem-solving algorithm (metalevel process propagation graphs) and 1b. the level of smaller problem-solving clusters of methods (specific operations graphs, or sequential process propagation graphs)) and 2. specificities of the second group at the level of content that is to be modelled as a graph (subject-based graphs, including formal network-graph congruency). The importance of determining deployment levels has been recognised in the need to clarify, in a systematised way, the graphs' use, pursuing elaboration of each application level's scope or problem-solving coverage, and complexity considering the subject of analysis and representation. The study supports the aim of developing new and optimised process workflows and better-informed conduct when approaching computational design problems and tasks, leading to a higher standard of performance at each scale and construction/design stage of the computational problem-solving path and representation.
In that respect, the stated arguments are supported by the examples drawn from the experimental case of dependency (network and navigation) graph application in the field of architectural and urban computing, defined and tested in Grasshopper. The structure and results of the thereby created graphs, based on the investigated subjects of transportation infrastructure design and analytics, dynamic localisation, and path-finding, are parsed in a way that best corresponds to the intended instructional explanation. The case study provides supporting evidence in the form of graph-based workflows and formal geometric outputs resulting from their propagation – i.e., it supplements arguments diagrammatically and provides illustrations of the stated points.

Keywords: Network Graphs, Diagrams, Design Process, Computational Design Methodology, Computational Problem-Solving, Algorithmic Thinking, Urban and Architectural Computation, Smart Cities, Intelligent Cities, Intelligent Architecture

1 Introduction

The introduction lines will first make a reference to the thus far accomplished phases of the project on network design and path generation with the support of graph theory and representation (Ćirić 2023a, 2023b, 2024, 2023-2024). It is through the course of this project that the main reasons that revealed the importance of the topic of the paper have arisen. The preliminary phase of the project had the aim of modelling a network graph of the urban transportation system including all its types, nodes, and connection lines. The network has been modelled in Gephi, which proved itself convenient for large amounts of data, offering an operable interface and ways in which points, edges, and their properties, grouped in defined categories, could have been constructed, visually processed, and represented. A number of graph layouts enabled the easy transition of data to various representation frames and modalities so as to single out targeted network features, while certain actions, such as the shortest path generation, could have also been performed automatically.

Data gathered, organised, and ordered in hierarchical structures for the purposes of the stated network graph design were further processed to present the network content in more detail. Such arrangement enabled easy conversion of data into the chosen representation modalities—e.g., linear and circular dendrograms—clearly displaying the underlying branching logic that can be used to support network description and that can be made operable in further algorithmic definitions and software transitions.

The built network fulfilled basic representation requirements, as well as those of diagrammatic explanation. However, in order for data to be operationalised and made active with regards to the design intentions, the software environment with a wider spectrum of definitions and functions had to be deployed. The representation of the computational design process and methodology based on workflow graphs as design instruments was the key criterion for the selection of the new software environment. The software needed to fulfil the condition of clear design process explanation and visualisation, containing a required range of operations and functions to complete the set design tasks. The first to be opted for was Grasshopper, and the existing network graph data model was adapted, transferred and utilised in its visual computing environment

where network geometry and data could have also been made dynamic, parametrised, and subjected to various instructional, analytical, and transformative operations. This paper draws its theoretical and practical design conclusions based on the results achieved by using and testing Grasshopper definitions and operators against the defined design problems. It relies on the examples of workflow graphs and their dynamic geometric outputs of urban network systems, constructed through Grasshopper design components. Parametric and generative design algorithms and logic that enable the path between the design problem definition and its solution stand behind the workflow and output registers, while the dependency that exists between different design phases and local and final design results/outputs shapes and conditions the development of the chains of connections between the used computational methods (CM), as well as the complete CM-based structure (process workflow). The modelled urban networks and their relational principles, on the other hand, guide the second class of graphs that have been subjected to the study.

The fact that Grasshopper workflows are graph-based was central for opting for its definition with respect to the subject of this study, and the overlapping regarding terminology and use of the graphs initiated the titled research topic and analysis that should have clarified necessary graph distinctions.

2 Graph-Based Problem-Solving Methods within the Computational Design Methodology and Strategy

The central section of the paper considers graph-based problem-solving methodology and graph-based representation/modelling methodology. The first one is pivotal in producing specific abstract diagrammatic forms of process explanation, guidance, control, design, representation, and propagation which enables automatic execution of all functions. The second one refers to the framework for the construction of specific, operable network graph structures, including graph forms that are congruent with their referents from the physical environment (e.g., networks). Named distinction forms different graph classes (process-based and object-based) and implies that graphs communicate different content, at different levels or within the different subject registers, even though they refer to the same mathematical concept and model, as well as a specific type of relational diagrams.

Besides division between the graphs that entail any kind of chain and logic of actions (process/workflow graphs) and graphs corresponding to specific formal relations (object-defined graphs), additional subdivisions can be made following several criteria. They include the relation between the graph topology and its output or referent, which outlines the difference between graphs that express formal congruence with referents and those whose topology greatly differs from their formal propagation results, not only in terms of shape, but in terms of algorithmic description as well, or graph complexity that implies the amount of data and number of functions that graph contains, as well as its position within the complex chain of elements that constitute the whole graph definition. Thus, the section aims to resolve the problem of graph differences based on 1. the subject that undergoes graph transposition (which divides graphs on process- and object-based graphs), 2. the graph relation to referents or outputs (degree of morphological congruence between the graph and referent), and 3. graph complexity and function (the position within the workflow).

With respect to that, the observed levels of graph-based thinking and graphs design methods deployment result from the stated classification criteria. It has been argued that their clarification is important for disciplines that involve and utilise graph-based thinking and design, and the following subsections will provide more details and examples to analyse and support the stated notions.

1.1 Process Graphs and Algorithms

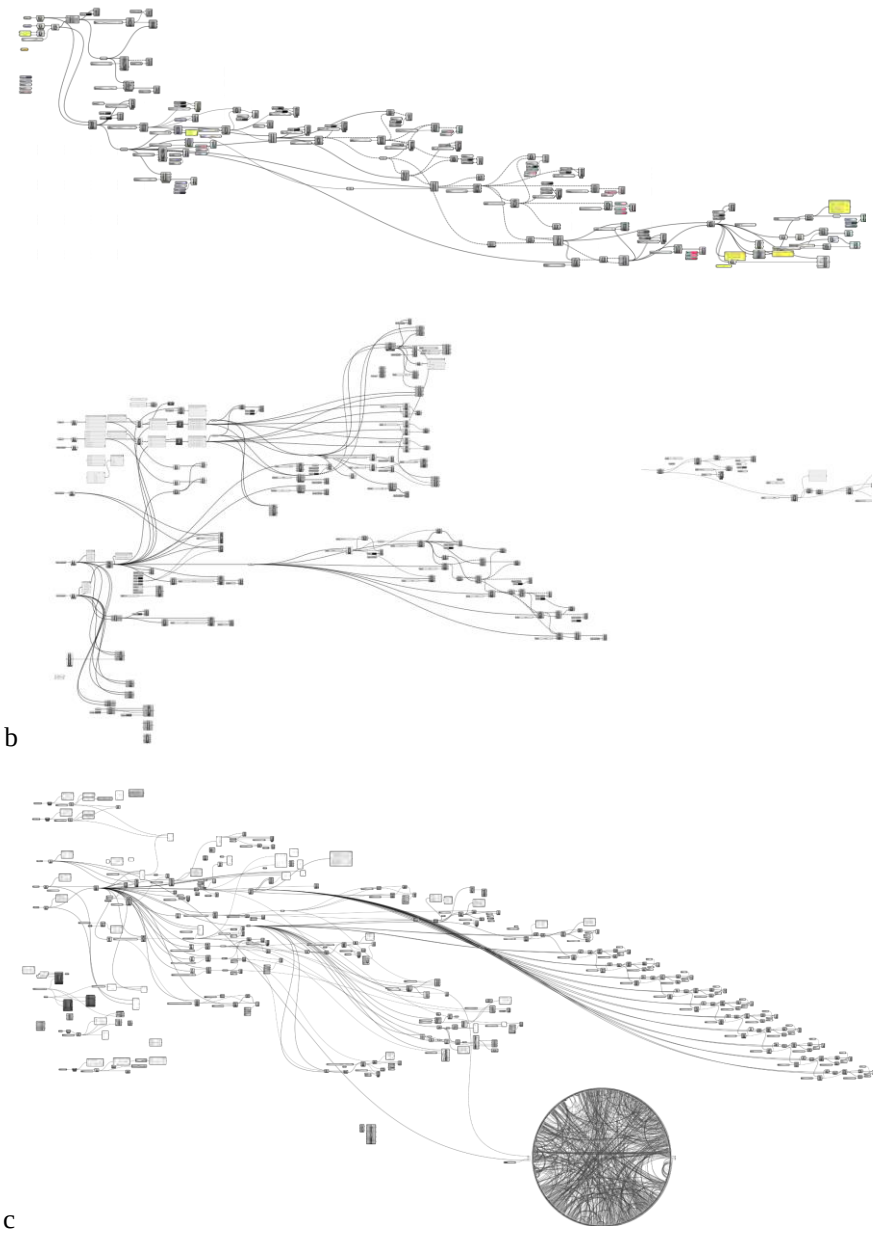
The first subject-based criterion distinguishes process-based and object-based graphs. Starting from the first division and class of process-based graphs, by looking at the graph complexity or function from the widest perspective and macro level towards the problem's micro interior, one can identify the following types: a) the complete process propagation graph and algorithm entailing the whole design problem-solving methodology, or set of methods, and b) single methods or their smaller sequences organised to resolve different parts of the overall problem-solving process (individual smaller chunks and clusters of the problem-solving methodology). The first subclass contains and represents the process at the macro level, while the second one is more specified and narrower in terms of the operations it performs. The topology of both of them usually differs from their outputs in formal terms, but the lower on the complexity scale the graph gets, the greater are chances that it might correspond to its referent with the same recognisable elements.

2.1.1 Metalevel Process Propagation Graphs and Algorithms

The first group of graphs are *process graphs* or *process diagrams* considered as propagation-based systems (Woodbury 2010, p.12) and/or algorithms. Regarding the field of application (in this particular case a design framework), they can be designated as (design) problem-solving process/procedure graphs. Having been

represented in the form of a tree or a rhizome, they stand for the problem-solving strategy and methodology between the inputs and final outputs of the process they describe, within the specific software environment (e.g., Grasshopper for Rhinoceros). They are usually termed *process propagation graphs*. In terms of problem-solving methodology, these graphs represent a problem-solving set of methods or set of decision-making sequences (logical sequences of problem-solving)—a composition of chosen single methods for arriving at the desired design solution, or a composition of computational steps to be performed for such purposes, the latter indicating the logic of state machine. Process propagation graphs are complex structures and are determined to resolve complex problems through the relational structuring of clustered or single functions and their parametrisation.

Figure 1. provides several examples of such complex compositions. Presented propagation process graphs contain several smaller problem-solving clusters (each of them deploying specific operations), which are connected according to the predefined course of actions that need to be performed to arrive at the solution. The smaller cluster of operations in a chain of actions has to be resolved prior to the propagation of another functional cluster, which makes them mutually dependent and renders this dependency, or relations, with full clarity. Thus, one who plans the process needs to think about the assembly and relational flow between the used functions. These higher-level, strategic complex process graphs demand a meta-perspective on the defined problem and control over the functions, singular methods chosen for subproblem-resolution and design, as well as their proper and optimum relational networking, flexible enough to receive possible changes, improvements, and adaptations of each singular part of its complex structure (graph expansions and condensing). Their functioning can be comprehended only through generative process animation and representation as proof and evidence, indicating that their functionality can only be confirmed in another communicative environment (in geometric viewport)—by translating both code and visual coding diagrams (graphs) into the digital environment as a real-world digital twin and/or graphic form of communication of the investigated problem.



b

c

Fig. 1. Process propagation graphs for network and path-generation and design, including operations for dynamic localisation, search for possible moves within the network according to given parameters (such as distance, or a specific class of destination points), tree-like representation of decision-making during the path construction and transition from point to point.

a. formal outputs:

1.https://64.media.tumblr.com/333f2be93ea91c32c04a7fe4ffd502fa/e8adf95518df9889-e8/s1280x1920/5506ddbd0ed2dee16b5a0ce4e2bf74777b1aa86e.gifv

2. https://va.media.tumblr.com/tumblr_sbaeuiOF4C1a96rc8_720.mp4

b. formal outputs:

1.https://64.media.tumblr.com/f875cc057e19fea2fbafd1bb7b18433a/870c1911c3c3ba4f-0e/s1280x1920/b7df0481efcd15f590addb5b397cc959adf3feaf.jpg

2.https://64.media.tumblr.com/320a0d251bd1dd0214d7a9a3d66f0d90/870c1911c3c3ba4f-46/s1280x1920/d95bebe53faa2c630a9d42f635779204c28307f4.jpg

c. formal outputs:

1. https://va.media.tumblr.com/tumblr_s1fik2uIPa1a96rc8_720.mp4

2. https://va.media.tumblr.com/tumblr_sikowzz6Ur1a96rc8_720.mp4

3. https://va.media.tumblr.com/tumblr_s9zori53F11a96rc8_r1_720.mp4

Source: Dragana Ciric, 2023-2024, all rights reserved. Software: Grasshopper, Rhinoceros

2.1.2 Sequential Propagation Graphs

The second group represents a subgroup of the strategic meta-level, and it is focused on specific entities that compose the whole design process and assembly. They are the propagation graphs or parametric propagation diagrams of each part/sequence of the problem-solving composition and set of methods that resolve smaller problems through single operations within the complete problem-solving algorithm (Table 1). This level is traceable due to the decomposable form and nature of the complex problem-solving structures and features of computational thinking and design. The computational thinking method of pattern recognition helps in dividing problems into parts that can be *cleanly and clearly resolved and then combined into a whole* (Woodbury 2010, p. 8). Referring to Woodbury's notions on *divide-and-conquer strategy* (Woodbury 2010, p. 27), this method tends to divide graphs into easily solvable subproblems—implying the use of less complex sequences to solve the whole complex problem (Woodbury 2010, p. 12, 22)—and then compose them back into the initial problem-solving structure. Besides being considered as smaller problem-solving clusters when looking from the meta-perspective, in reverse, this level also provides a set of basic tools or units/blocks for building complex problem-solving and decision-making structures. The smallest units are basic operations, while sub-problems or tasks of a smaller scope are constructed through their networking and parametrisation. Looking at the whole problem-solving set of methods from this perspective, a higher-level graph application requires the designer to focus on the logic that binds the design together (Woodbury 2010, p. 24) and think about the relationships (chains) between the available and composable units of a lower level, as well as assemblies as organizational principles. In such context, at the subproblem level, variations of single methods are initiated in line with a demand that conditions the algorithmic structure to enable data flow from part to part in a clear and explainable manner (Woodbury 2010, p. 28). Besides being recognised as a part of the common algorithmic step-by-step solving and propagation procedure, the subunits can also appear as a result of the expansion or condensing of the graph nodes (Woodbury 2010, p. 30), influencing the meta-level graph scope through their execution.

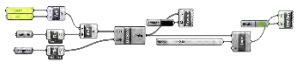
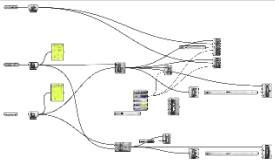
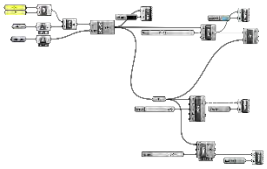
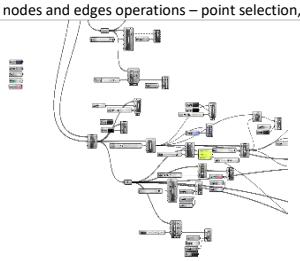
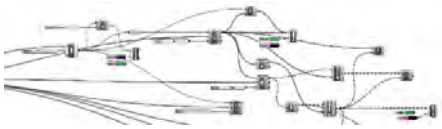
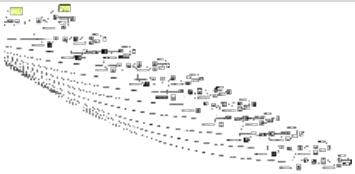
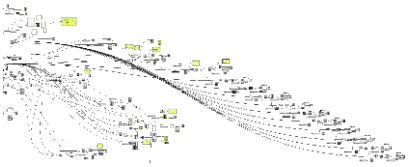
network construction (WGS84 to XY adjustments) – points/nodes representation	network points construction (lat, lng), tagged with IDs; point selection
	
network construction (WGS84 to XY adjustments) – nodes and edges representation	nodes and edges operations – point selection, next state search, connections
	
basic sequence: movement from state S_1 to state S_2, decision parametrization	iterations: cluster of sequential movements from one state to another
	
iterations: cluster of parametrized sequential movements from one state to another	problem division and flow of actions within the complex problem structure
	

Table 1. Sequences and smaller operation clusters of the propagation graphs from Fig. 1. Source: Dragana Ciric, 2024, all rights reserved. Software: Grasshopper, Rhinoceros

Certain software plug-ins are particularly designed to respond to the requirements of specific problem-solving subjects (e.g., Urbano for various urban design and analysis tasks, or networks- and graph-based plug-ins

containing specified graph operations). As such, they can be used as standalone packages to resolve specific problems they have full coverage of, while in other cases their units (operations and operators/components) and written codes are combined in a more flexible and open way, leading to original contributions regarding design methodology, *know-hows* (new problem-solving paths), as well as new plug-ins.

2.2 Object-Based Graphs

Returning to the graph content and object-based graphs, one is drawn to inquire about specific subject that undergoes definition through graphs, and specific relations that enable the analysed object to be transposed into the targeted mathematical model. Within this class, certain formal congruence between the graph morphology or logic and the morphology of the investigated problem can occur. It is expressed either through direct formal correspondence between the network and its mathematical model, or through the capacity of the subject's formative relationships or aspects to be represented in graph-like logic and form.

At this level, graphs are deployed to represent the relational logic within the design task's content and formal spatial elements. While analysing the internal elements of the investigated subject, the relations between them are identified and represented in a diagrammatic manner. Thus an object-based graph emerges; it depicts or translates relations within the subject that need to be analysed to a graph model and mathematical formalisation, communicating these relations in terms of a graph theory. As already stated, there might exist a formal or morphological correspondence between the graph network representation and the subject of the investigation—the subject may be a subject whose different formal aspects can be explained through graphs that retain key invariant values for making the object recognisable, or it can be the network itself (network as a term is usually deployed for real systems (or real-world situations, emphasis added), while graphs indicate mathematical representations and graph theoretical formalisms of these network cases (Barabási with Pósfai, 2018)).

In the exemplary case, a network graph has been deployed to represent the rail infrastructure of the Greater Paris area. The real-world transportation network is translated into the graph as its mathematical formalisation and modality so as to be deployed and operationalised for different analytical, planning, and design actions (Fig.3). The main graph constitutive elements (features/nodes and dependencies/edges) formally directly correspond to the building units of the transportation network system (stations and connection lines/tracks). In this way, the system that can be operationalised for further research and design is generated, as well as the environment for planned and defined operations (e.g., network's incremental growth and restructuring (Fig.4), network analysis and path design and parametrisation (Fig. 5 and 6), path-finding and generation (Fig.1)).

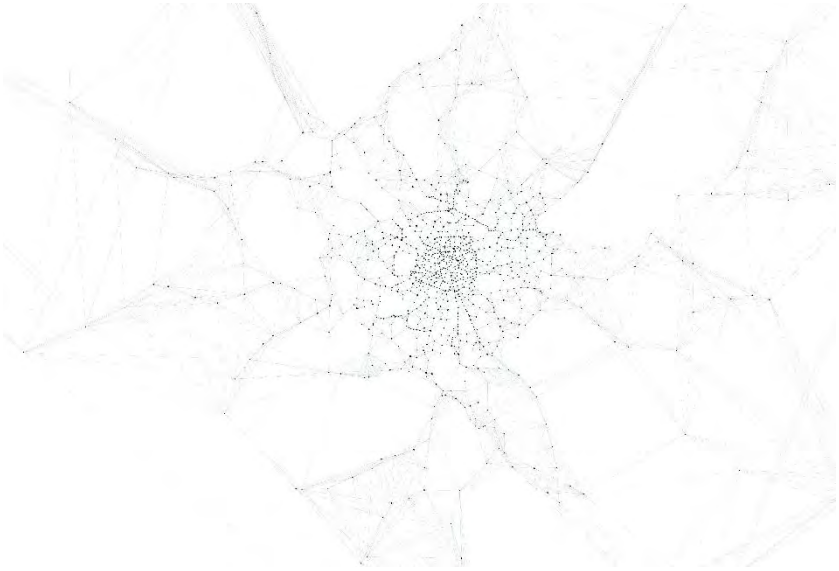


Fig. 3. Île-de-France region rail transportation network generated through Grasshopper workflow graph—second test: workflow graph is constructed and compared to the existing results of the first tests; points have been constructed by assigning them latitude and longitude, including adjustments regarding the transition from WGS84 to XY environment. Source: Dragana Ćirić, 2024, all rights reserved. Software: Grasshopper, Rhinoceros.

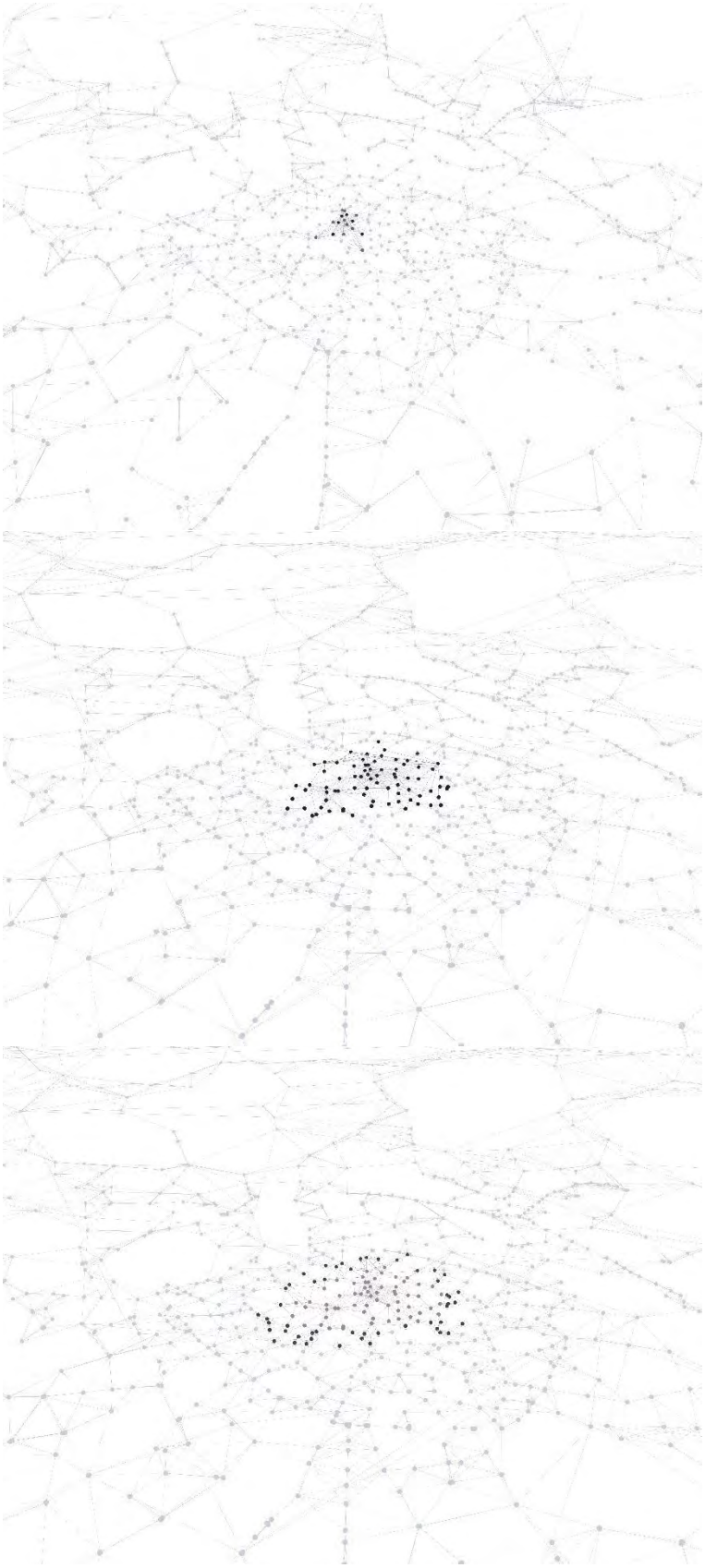
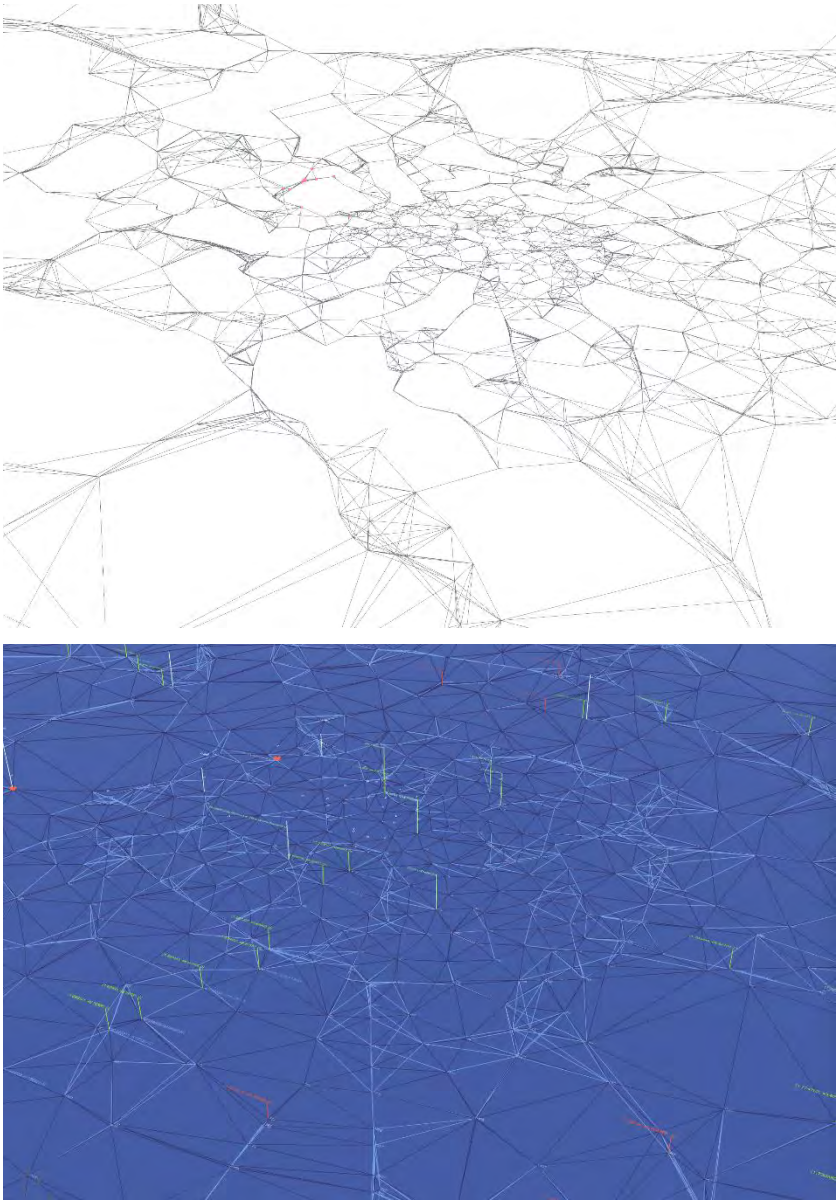


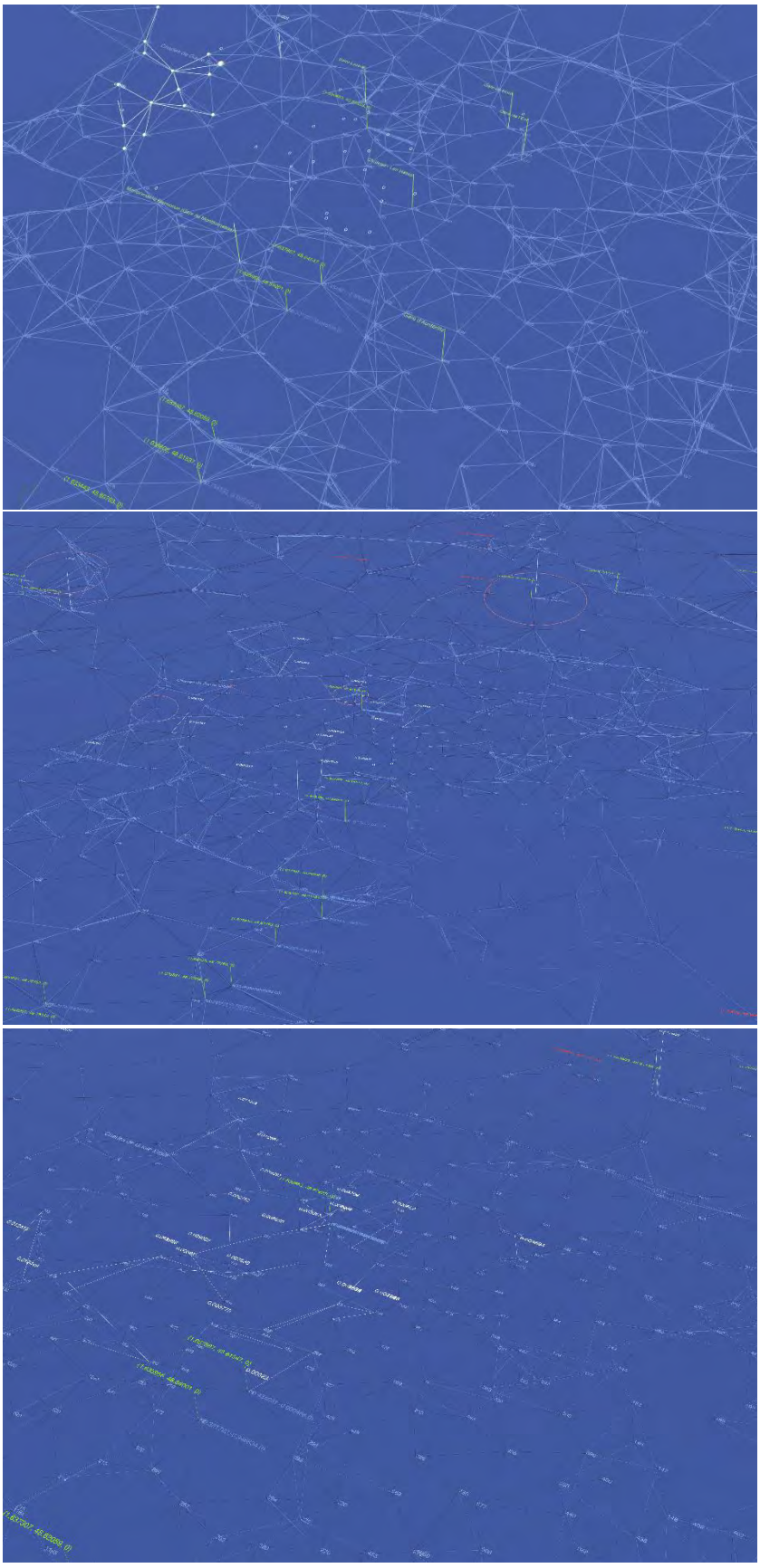
Fig. 4. Network expansion based on the 'proximity search' rule, which guides the analysis of nodes that will be occupied in each incremental growth iteration. 1. https://64.media.tumblr.com/333f2be93ea91c32c04a7fe4ffd502fa/e8adf95518df9889-e8/s1280x1920/5506ddbd0ed2dee16b5a0ce4e2bf74777b1aa86e.gifv 2. https://va.media.tumblr.com/tumblr_sbaeuiOF4C1a96rc8_720.mp4 Source: Dragana Ćirić, 2024. Software: Grasshopper

3 Discussion

Further development of the topic described in the paper can be directed towards its grounding with respect to diagrammatic reasoning and representation theory, cognitive theory, and computer theory and logic. Besides the literature used thus far for resolving practical problems and instances related to algorithmic and parametric design along with some theoretical questions (Woodbury 2010, Jabi 2013, Tedeschi 2014, Madl 2022), the bibliography can be extended so as to encompass the named fields and aspects in more detail.

Diagrams are considered instruments of strategic reasoning, inferring, problem-solving, decision-making, and representation, while graphs, one of their subforms (wiring diagrams), are claimed to have the same capacities and are applied in line with this claim. Regarding such context, comments on interrelation between diagrams and graphs, as well as visual cognition and representation of various processes through graph- or diagram-based forms can be developed in more detail and given proper consideration. Since the logic of thinking that has been investigated is shaped by the computational models of reasoning, while the results appear and are evaluated in another communicative modality and framework (visual and geometric), the relation between these registers can also be referred to. Reasoning *with/through/by* graphs as diagrammatic modalities can finally be supplemented and supported by visual and aesthetic results provided and shaped through graph use, design, and propagation. This research aspect specifically targets the parts and operations within the propagation graphs that address and improve representation and visualisation of the described processes' logical substructure, as well as the very content and spatial actions that have been diagrammatically processed.





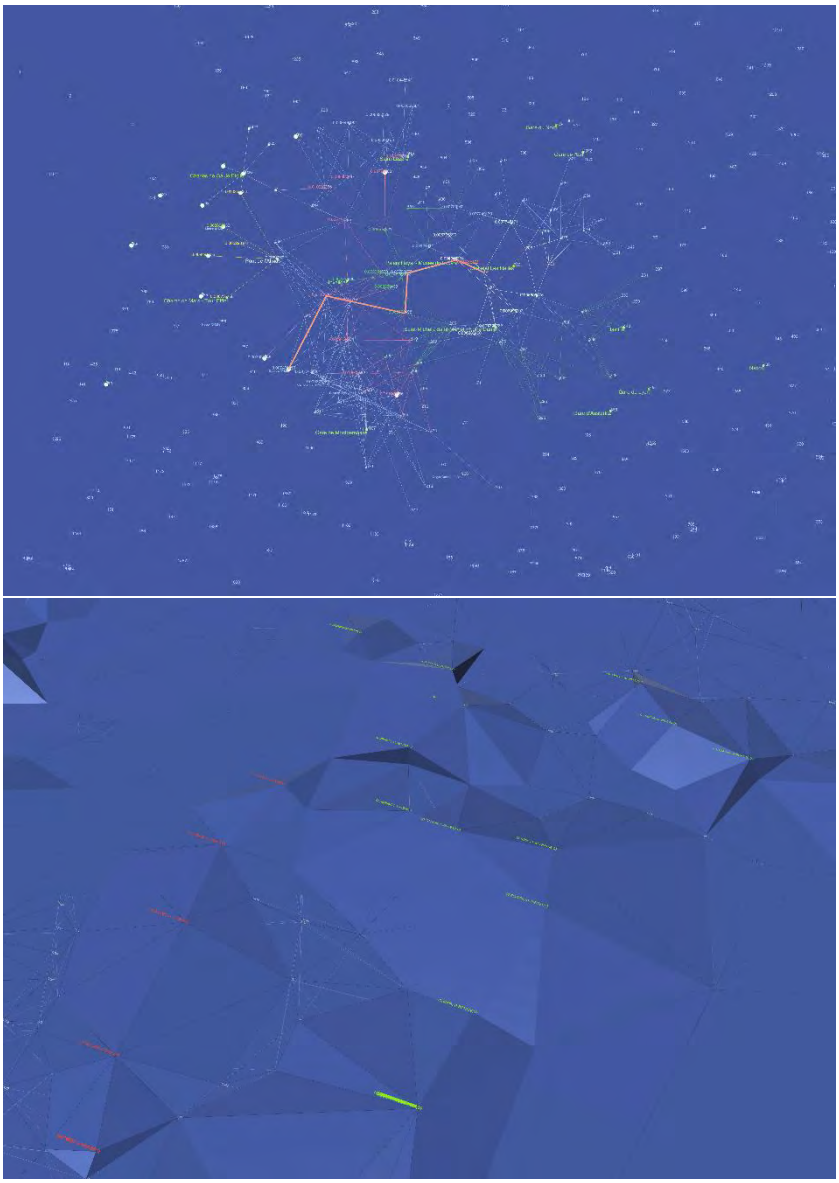


Fig. 5. Operations within the constructed network and tests regarding visualisation of static (mapped) and dynamic (diagrammed) content and data: localisation, labeling, path/point search and evaluation, path generation, network analysis including different context/network representations and data representations. Source: Dragana Ćirić, 2024, all rights reserved. Software: Grasshopper.

4 Conclusion

The paper placed the focus on two specific levels of (computational) design and problem-solving methodology at which graph-based methods can be deployed. The project developed for urban systems design, analytics, and smart operation, having the Greater Paris rail transportation network as a subject, has been used as an example, demonstrating specific reasoning through graphs, the resulting representation, and systematisation of the potential usage of graph-based methods. All steps of the process, including designed exercises that are performed at all mentioned levels, can be found and followed online (Ćirić 2023-2024). The results are supplemented with papers explaining different aspects of graph-based thinking and graphs application in architectural and urban, or more generally spatial design (Ćirić 2023a, 2023b, 2024).

Fig. 6. Network operations, https://dciricnetworks.tumblr.com/. Source: Dragana Ćirić, 2023-2024, all rights reserved.

References

Barabási, A-L. (with M. Pósfai). (2018). *Network science*. Cambridge University Press, Cambridge, UK.

Ćirić, D. (2023a). The Science of Networks: Urban Movement Design, Analytics, and Navigation. In: Bogdanović, R. (ed.). On Architecture – Challenges in Design, Proceedings, pp. 110-129. STRAND – Sustainable Urban Society Association, Belgrade.

Ćirić, D. (2023b). The Science of Networks: Network Graphs in Urban Navigation Design Problems - Mobility, Transportation Systems, and Movement Paths Generation [Revised e-Book Version]. In: Soddu, C. and Colabella, E. (Eds). XXVI Generative Art 2023—Proceedings of the XXVI GA Conference, pp. 105-120. Domus Argenia Publishing, Rome.

Ćirić, D. (2023-2024). Network operations, https://dciricnetworks.tumblr.com/, last accessed 2024/09/16.

Ćirić, D. (2024). Generative and AI Methods for Transportation Systems and Urban Mobility Design, Planning, Operation, and Analysis: Contribution to Theory and Methodology of Urban Computing. In: Đukić, A., Krstić-Furundžić, A., Vaništa Lazarević, E., and Vukmirović, M. (Eds.). Keeping up with Technologies to Imagine and Build Together Sustainable, Inclusive, and Beautiful Cities, 8th International Academic Conference on Places and Technologies Proceedings, Vol.8, pp. 490–503. Belgrade University - Faculty of Architecture Belgrade.

Jabi, W. (2013). *Parametric Design for Architecture*. Laurence King Publishing Ltd., London.

Madl. A. (2022). *Parametric Design for Landscape Architects: Computational Techniques and Workflows*. Routledge, London and New York.

Tedeschi, A. (2014). *AAD_Algorithms-Aided Design: Parametric Strategies Using Grasshopper*. Brienza: Le Penseur Publisher.

Woodbury, R. (with contributions by O. Y. Gün, B. Peters and M. R. Sheikholeslami). (2010). *Elements of Parametric Design*. Routledge, London and New York.