ISSN: 2029-7955 / eISSN: 2029-7947

JOURNAL of ARCHITECTURE & URBANISM

2025
Volume 49
Issue 1
Pages 87–99
https://doi.org/10.3846/jau.2025.22138

EXPLORING SPATIAL PROGRAMMING THROUGH MODULARITY BASED EVOLUTIONARY COMPUTATION

Resza RISKIYANTO[✉], Ahmad Bayu WIBISONO, Arnis Rochma HARANI

Universitas Diponegoro, Semarang, Indonesia

Article History:
- received 08 October 2024
- accepted 10 June 2025

Abstract. The application of evolutionary computing in architecture has advanced beyond active feedback to designers by integrating natural processes with computation to synchronize input from the final solution. Following knowledge in digital morphogenesis, new approaches can be formed by examining design issues deemed non-pragmatic and abstract, such as function in spatial programming. The study presented in this paper explores an approach based on the principle of modularity, which describes a biological system's ability to organize distinct, independent units to increase the system's adaptability. By employing modularity in evolutionary computation, we can characterize function as an abstract feature of phenotypes. The basic modularity method is simulated by developing a spatial program with dynamic programmatic functions to see how adaptable units are as spatial program components.

Keywords: morphogenesis, evolutionary computation, genetic algorithm, spatial programming, modularity.

[✉]Corresponding author. E-mail: *reszariskiyanto@lecturer.undip.ac.id*

1. Introduction

Morphogenesis can be seen as "the origin of form," including the processes of evolution and development that culminate in the formation of an organism's structure and form (Gökmen, 2020; Rees, 2018). According to Leach (2009), the application of morphogenesis in architecture has led to a paradigm shift known as "Digital Morphogenesis." Architects used simulations of development processes across micro to macro dimensions to understand natural and computer domains as distinct ontologies. The goal of merging digital morphogenesis with natural processes continues to drive computational-nature integration within architectural design. Despite initial challenges, the development of modeling tools that allow for real-time input is consistently promoted. It is evaluated whether evolutionary computing, particularly genetic algorithms, can offer a sophisticated and flexible technique akin to biological systems (Kamaoğlu, 2023; Miikkulainen & Forrest, 2021).

The addition of evolutionary development components enriches this architectural design methodology even more. The principles of Digital Morphogenesis represent a paradigm change as they have evolved from the pre-Darwinian Lamarckian trial-and-error method to a Darwinian strategy emphasizing natural selection (Leach, 2009; Leach et al., 2004). A new paradigm for phenotypic investigation is introduced by Navarro-Mateu and Cocho-Bermejo (2019) through a case study that combines Darwin's ideas alongside Gregor Mendel's genetic inheritance theory, mutations, and population genetics. This framework emphasizes growth through form and transformation by integrating ideas like body plan, allometry, and homeobox genes. However, it does not elucidate the morphological evolution within the context of the functional system of the phenotypes since this is deemed impracticable and therefore not prioritized. Interestingly, building programmatic functions is naturally flexible due to their non-rigidity and abstract nature, which opens up new possibilities for improving genetic algorithm uses for layout optimization.

This study explores the potential of evolutionary algorithms to improve optimization systems by integrating morphogenesis mechanisms. Specifically, this study aims to understand better how spatial programming can be investigated through phenotypes. A deeper understanding of the development of functional aspects of phenotypes is proper because, although abstract, the notions of emergence, change, and self-organization have been consistently used to determine functions in building programs. Focusing on interior applications as a basis for introducing fundamental rules can form a combination of mechanics that can be applied to the full programmatic function of a building, as in the projects carried out by Canestrino et al. (2020) and Suharjito and Muslim (2023). Evolutionary algorithms have the potential to be a system to optimize layouts in interior settings, focusing on compliance with established regulatory rules. This system encompasses self-organization mechanisms that have not been applied to analyze other dynamic mechanisms of programmatic functions, including emergence or transformation that can occur in building programs. The capacity to translate these mechanisms using genetic algorithms can be used to develop sustainable design solutions that can adapt to changing social, environmental, technological, and economic situations.

2. Current development of morphogenesis in architecture

2.1. Mechanism of morphogenesis

Morphogenesis is a basic biological phenomenon in biology that is defined by the dynamics of self-organization in living things. This complex process, which controls the formation and development of a living organism's anatomical form and structural organization, is regulated by an interaction of environmental stimuli and genetic instructions. From the early stages of embryonic development, known as embryogenesis, to the later development of specialized organs during organogenesis, morphogenesis integrates evolutionary, differentiative, transformative, and distributive interactions among the constituent parts of an organism (Gökmen, 2020; Kasyanov, 2020). The primary purpose of morphogenesis is to clarify the underlying mechanisms that control the molecular events that occur during development and adaptation, resulting in morphological changes in organisms (Kasyanov, 2020; Roudavski, 2009). The mechanisms underlying morphogenesis drive an interdisciplinary approach that unifies many fields of study to advance scientific inquiry. Alan Turing's contributions to computer systems have made it possible to simulate the evolutionary processes found in nature. Through the study of organic laws, mathematical and physical frameworks using algorithms clarify the development and structure of organisms (Kamaoğlu, 2023; Thompson, 1992).

Throughout history, people have attempted to understand how nature works and have used their newfound understanding to create objects. Nature is an endless source of inspiration and solutions, fostering creativity and connections across many interdisciplinary fields, including architecture (Kamaoğlu, 2023; Kasyanov, 2020). Nature has many hidden aspects only revealed by careful examination; therefore, several viewpoints are required to produce complex results. This research goes beyond merely drawing inspiration for formative processes; it explores natural systems' mechanical and functional characteristics as possible sources of architectural innovation (Ball, 2009; Harani et al., 2021; Riskiyanto et al., 2021). Morphogenesis is a broad category of digital processes that includes generative and speculative design techniques. Computational geometry, mathematics, properties of materials, fabrication procedures, and algorithms are all integrated into these techniques. As generative algorithms progress, it is possible to simulate self-organizing behavior and achieve optimal design solutions. Architectural applications based on biological sciences, specifically in comprehending emergent behavior in multi-agent systems, represent a process-oriented methodology associated with several ideas such as form-finding, emergence, and self-organization (Gökmen, 2020; Roudavski, 2009). It's possible that architectural design is not directly impacted by the mechanisms driving morphogenesis. However, the fundamental ideas behind these procedures may be used to create control systems that can effectively handle intricate and dynamic architectural configurations (Roudavski, 2009).

2.2. Exploration and exploitation within evolutionary algorithm

Using parametric and generative methods in computational design and high-quality mass production capabilities allows geometry to be thoroughly examined. The notion of morphogenesis in architecture employs several digital methods to conjecture and produce designs, combining computational geometry, mathematics, materials characteristics, fabrication methods, and algorithms (Dixit & Stefańska, 2023; Gökmen, 2020). Formation processes may now be simulated using evolutionary methods, developmental growth models, and reaction-diffusion mechanisms thanks to the development of complexity theory and the accessibility of computing power. Due to the algorithm's ability to "grow" and change, this method allows for developing design concepts through a formative process similar to the production of biological entities (Kamaoğlu, 2023). Within computational intelligence, Evolutionary Computation (EC) uses artificial systems with lifelike qualities to mimic natural evolutionary processes. When it comes to using Evolutionary Algorithms (EAs) to solve scientific and engineering problems, EC includes a variety of approaches, including Genetic Algorithms (GA), Evolutionary Programming (EP), Evolutionary Strategies (ES), and Genetic Programming (GP).

Evolutionary programming was first developed to develop finite automata. Later, its use was extended to include solving numerical optimization problems (Fogel, 1999). Genetic algorithms were first developed by Holland in 1973. They are mainly used to solve combinatorial problems and use binary strings influenced by the genetic code found in real organisms. In the early 1990s, genetic programming became a targeted program optimization technique (Bartz-Beielstein et al., 2014). These techniques work by applying selection, recombination, and mutation to populations of digital entities iteratively to produce unique solutions (Miikkulainen & Forrest, 2021).

Since Darwin's hypothesis was put forth 160 years ago, the fundamentals of biological evolution have become refined and defined. The current agreement acknowledges that the primary mechanisms guiding evolutionary processes are random drift, recombination, mutation, and natural selection (Miikkulainen & Forrest, 2021). Rather than using a single solution, these evolutionary approaches operate by managing individual populations. For future generations to have better traits, these systems integrate and retain the genetic representations of possible individuals. In this case, "genes" represent problem parameters, and their values are equivalent to "alleles". These techniques improve solution performance iteratively through this mechanism. Each individual is given a "fitness" score by evaluation, which guides evolution toward the intended solution domains. This strategy makes parallel search operations possible, which is necessary for effectively handling large combinatorial problem spaces. Given the enormous parameter variances, serial examination of all options would be unfeasible, if not impossible. The time needed for exploration is drastically cut down to a manageable level using parallel search.

2.3. Implementation of modularity through genetic algorithm

A standard evolutionary algorithm for optimizing objective functions with ambiguity, noise, and discontinuity is the Genetic Algorithm (GA). Genetic algorithms perform at the genotypic level, selecting populations according to their phenotypic expressions after altering populations through mutation and crossover procedures (Baden & Taghizadeh, 2023; Bartz-Beielstein et al., 2014; Harding & Brandt-Olsen, 2018; Katoch et al., 2021). As an optimization technique, GA begins with a starting set of chromosomes representing potential solutions. Then, the new set of chromosomes is created to perform better than the original by using genetic operators such as crossing, mutation, and selection. This iterative process continues until the termination requirements are met, usually achieved by obtaining a particular level of convergence or a satisfactory solution. The optimal answer is typically acknowledged to be the chromosome with the highest performance from the preceding generation (Baden & Taghizadeh, 2023; Katoch et al., 2021; El-Shorbagy & El-Refaey, 2020).

Application in the field of architecture is firmly rooted in the development of digital morphogenesis, a system of computation closely interwoven with its role as an architect's collaborator and its integration with the processes of nature. This cooperative model with architects lays out the authority and responsibility for addressing various issues, including construction, environmental, and financial aspects, in addition to the more ethereal fields of dwellings, aesthetics, and programmatic considerations (Davis, 2009; Leach, 2009). There remains a tendency in genetic algorithm implementations to give practical problems a higher priority than less practical ones, especially when it comes to programming difficulties in developing programs. These programming efforts are closely associated with configurations intended to meet specific needs within building constructions, closely related to setups designed to satisfy particular requirements inside building structures. Analyzing geometry as a typology for defining spatial functions in a building program presents a new way of thinking about phenotypic optimization problems, which are typically less practical in digital morphogenesis. Understanding the mechanism of phenotypes' interaction in the system is made possible by classifying and describing them according to different typologies. This system is the cornerstone of a dynamic building program that puts users' needs first and embraces emerging opportunities to help with decision-making. This system forms the foundation for a dynamic building program that prioritizes user-centric adaptations, facilitating decision-making by embracing emergent possibilities.

Modularity is applied to explain the manifestation of functionality in the phenotype. Here, "modularity" refers to a biological system's ability to organize distinct, independent parts, which increases the system's overall flexibility. This feature tends to encourage selective pressures, whereas integration helps to bring these modules together and make them more cohesive. This idea is fundamental to evolutionary biology since it examines the relationships between relatively independent parts of a system. Drawing on Darwin's concept of "correlations of growth, modularity is about adaptation, which gives rise to modular variational structures and allows for the emergence of complexity and diversity found in nature. Additionally, it provides insights into the anticipated patterns of species diversity under genetic drift and directed selection. It can be used to explore the macroevolutionary scale of evolutionary drivers of diversification (Melo et al., 2016).

Modularity extends to adult functional relationships, wherein modules comprise components that collaborate in executing specific physiological functions. Traits that are developmentally and functionally linked tend to exhibit relatively strong intercorrelation. This principle is a framework for categorizing phenotypes based on their functional attributes as modules. These modules are classified into two categories: developmental modularity, which influences phenotype formation, and variational modularity, which contributes to phenotypic diversity (Melo et al., 2016; Minter et al., 2012).

3. Method

Several experiments were carried out in constructing a modularity-based evolutionary computation methodology by selecting issues on programmatically dynamic buildings, which is an office space programming. The methodology constructs phenotypes based on a catalog of furniture modules as units divided according to function, size, and modularity categories, including developmental and variational modularity. The module catalog imposes rules on population and zoning to provide sufficient restrictions for the simulation to work. From the simulations that have been carried out, programmatic functions can be observed from several types of solution selection resulting from the evolutionary algorithm, such as through Pareto front selection, fittest solution selection, and relative difference & average fitness solution selection.

The variety of solutions in selecting the ideal office program can be seen from two types of solution selection: the all-population Pareto front selection and the relative difference selection. The function of office programs, which are starting to diversify due to the decline in unit population and the formation of dominated zones, can be observed through the latest Pareto front selection and fittest solution selection. Finally, function programs that drastically change with the emergence of variational modularity or the formation of total zones can be observed through the latest Pareto front selection, fittest solution selection, and average fitness solution selection. These experiments were divided into phases as follows.

3.1. Experimental setup I – evolutionary and selection strategy

This study aims to determine whether it is feasible to design an associative framework that incorporates evo-devo's qualities to handle specific problems that fall under the scope of the building's programmed functions. The process involves creating interior object prototypes in an exploratory manner to clarify the functional links between their morphologies and the building's spatial arrangement. The programmatic role of office buildings was selected as the experiment's topic. This subject was chosen due to the typological nature of office buildings, which enables different changes in the operation of a single program within or between various departments within a corporation. Office building occupancy can also decline due to new work patterns, such as remote working, potentially replacing the office work paradigm as technology advances. These circumstances may lead to complete or dynamic modifications to the office building's program, an appropriate setting for using the modularity model inside evolutionary computation.

Several plugins for the modeling program Rhinoceros 3D were used in experiments to create a model that might simulate the dynamic nature of an office building's programmatic function. Kangaroo, Human, and Wallacei primarily use the Grasshopper3D platform (visual algorithmic modeling). The model's parametric and algorithmic strategies are built using this plug-in. The circle packing feature is implemented using Kangaroo, which can automatically pack as many circles as possible inside the boundaries of space without causing them to collide. This system mimics the spatial conditions of interior objects within the given area. Then, Human is used to display inside objects based on each circle's size and position parameters, to better visualize the phenotypes within the circle packing bounds. Wallacei is utilized in the algorithmic simulation to examine and model the growth of the multi-objective optimization of office interior objects, which are broken down into five main objectives:

- S, a high number of interior objects in the sociability function;
- P, a high number of interior objects in the productivity function;
- HW, high number of interior objects in health and wellness functions;
- PO, low population of all interior objects;
- ZO, a low number of existing zones.

This simulation is run with the following settings:

- Generation size: 25;
- Generation count: 50 (1 + 49);
- Crossover Probability: 0.9;
- Mutation Probability: 1/n;
- Mutation Rate: 20;
- Crossover Rate: 20.

The objectives for numbers 1, 2, and 3 are set to maximize the number of furniture units present in each zone. These conflicting criteria can give better variation of combinations within the system, depending on how developmental and variational modularity are present in the solution. On the contrary, the objectives for numbers 4 and 5 are set to minimize the total amount of unit furniture and the number of existing zones in the given space. This converging criterion is introduced to provide the optimal scenarios for the emergence of variational modularity as a balance condition considered unfavorable for the simulation to occur more often. Also, with these objectives, the combination of the present zone within the space can be better selected and analyzed as balanced, dominated, and total zones.

As for selection strategy, three methods are used: Pareto front selection, fittest solution selection, and relative difference & average fitness solution selection. This method of selection is being used to observe the phenotype regarding the function of each of the solutions that are being selected and to give helpful and well-versed feedback to the designer to better understand which solution is better suited to the particular design consideration.

3.2. Experimental setup II – modules

To incorporate modularity into the simulation, furniture as units combined to make a module must be identified in a module library (Figure 1). The module library is separated into three criteria, namely function, size, and modularity categories. To define the primary purpose for furnishing the office, function groupings are created according to the ideal function of furniture in an office building. However, this will not limit how the furniture's function is generated or updated; it will clarify the module's zoning. Each furniture unit's size will be determined by measuring its floor area, with a radius starting from the center point and adding a 0.6-meter-wide region for circulation around the outside. To construct the function of a module, the extent of this unit will create a hierarchy according to its occurrence level.

Developmental modularity and variational modularity make up the two divisions of the modularity categories. The furniture elements that comprise developmental modularity are frequently seen in office spaces, making them the primary source for figuring out how an office program operates. Variational modularity is characterized by furniture units that serve purposes not often seen in office settings. The modularity of variations is directed towards

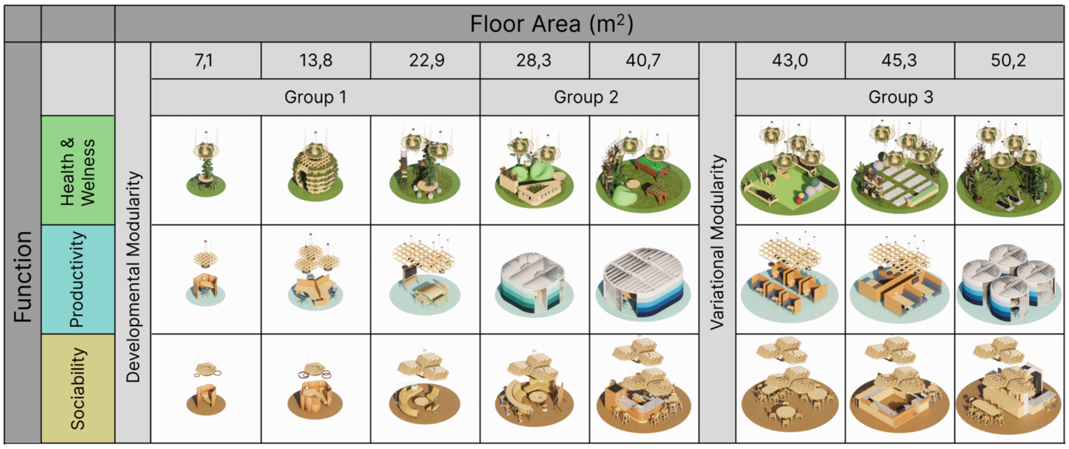


Figure 1. The module library catalog contains furniture units divided based on function, size, and modularity categories

furniture from other programs, namely for health & wellness towards the gym program, for productivity towards the co-working space program, and sociability towards the cafeteria program. The size of units in variational modularity is larger than that of developmental modularity, allowing it to better express its role as a module that clarifies changes in the system's overall function while adding variation. A special grouping is also established for the productivity function, which serves as the primary stimulus in the developmental modularity category. This special grouping is based on its intended use, particularly for staff, managers, or executives.

3.3. Experimental setup III – population

The population establishes the most minor and significant number of furniture units that can be incorporated into the system based on the unit size. To initiate the circle packing function, the floor space of the unit is calculated as a circle area, which determines the population groups used for the initialization procedure. The population model that is being used in this evolutionary algorithm simulation contains two distinct gene types, which are as follows:

1. The population of furniture units is known as the initial population and will be addressed following the established restriction.
2. The unit appears randomly in the designated location, which is known as the population seed.

The maximum and minimum number of genes in the initial population will be restricted due to the drawbacks of the circle packing feature (Figure 2). The ratio of available space area to the ratio of units produces the maximum population, while the minimum population is set at one. These limitations will ensure that the simulation operates smoothly and provide flexibility so that it can be applied to different regions of space. This ratio is used as the upper bound for the two modularity categories that follow one another, which are as follows:

1. The foundation of developmental modularity consists of unique classifications that apply to every unit in these categories, which are separated into ⅓ for manager and executive units and ⅓ for staff units. To ensure that high density in circle packing is given priority, the tolerance limit with this ratio is two times the area of each population.
2. Variational modularity happens when the population of modular units used for development is less than ⅔ of the total space and more than ⅓ of the total space.

Another limitation is placed on both categories: randomly removing a portion of the group's population until the ratio is fit. This prevents the simulation from reaching the null solution. These limitations will enable developmental modularity in all simulation scenarios and reach high population densities per zone without reducing the diversity of unit types. One of the requirements for how developmental and variational modularity can be articulated into building programming functions is the set of units that will be created based on this modularity.

3.4. Experimental setup IV – zoning

Zoning occurs when the unit is produced and dynamic borders are placed inside the available space, with the central point of the unit's representative circle being assigned

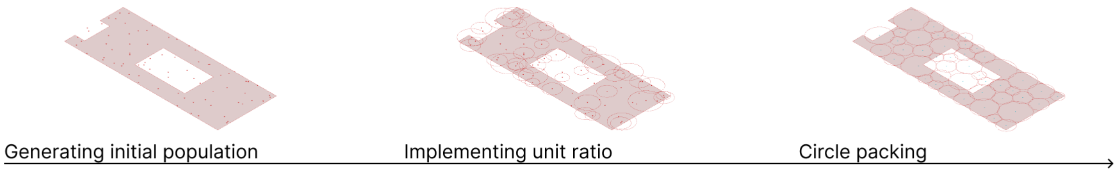


Figure 2. The rules of the initial population define the phenotype construction process

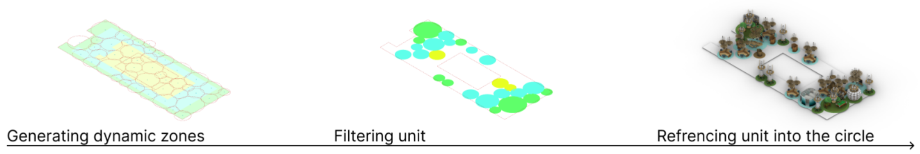


Figure 3. Rules of zoning define the construction process

to those boundaries (Figure 3). This zone is formed by the two inner projections of the available space at different scales, and it is bounded by the grouping of functions in the preset module library. With a minimum restriction of 0.2 times the initial size, this scale will be the third gene utilized for simulations in evolutionary algorithms. It can adjust to more complicated geometries and prevent the simulation's null solution from arising from a failure to specify boundaries by utilizing the scale of each relative boundary formed by available space. Two restrictions are applied to the formation of zoning boundaries:

1. A defined zone order on the dynamical limits, ranging from the inner to the outside region of the available space, should be imposed: sociability, productivity, health, and wellness. The degree of access to the central core area and the surrounding views/outside locations determines this arrangement.
2. The circulation zone within each unit, along the perimeter of the space area and core area, has a tolerance of 0.6 m for collision restriction to available space.

The simulation will count the number of circles that occur within the dynamic bounds and omit any circles that, while changing dynamically, happen to develop outside of the space region and core area. Depending on the size of the zone generated, these constraints will also enable the available space to be realized as balanced, dominated, and total zones. This quality is one of the requirements for expressing variational and developmental modularity in developing programmed functions.

4. Result

4.1. Data flow

Several processes are established, including development, genetic algorithm, and result, based on the constructed experimental setting. Therefore, the following data flow shown in Figure 4 is recommended by the authors for use in replicating modularity.

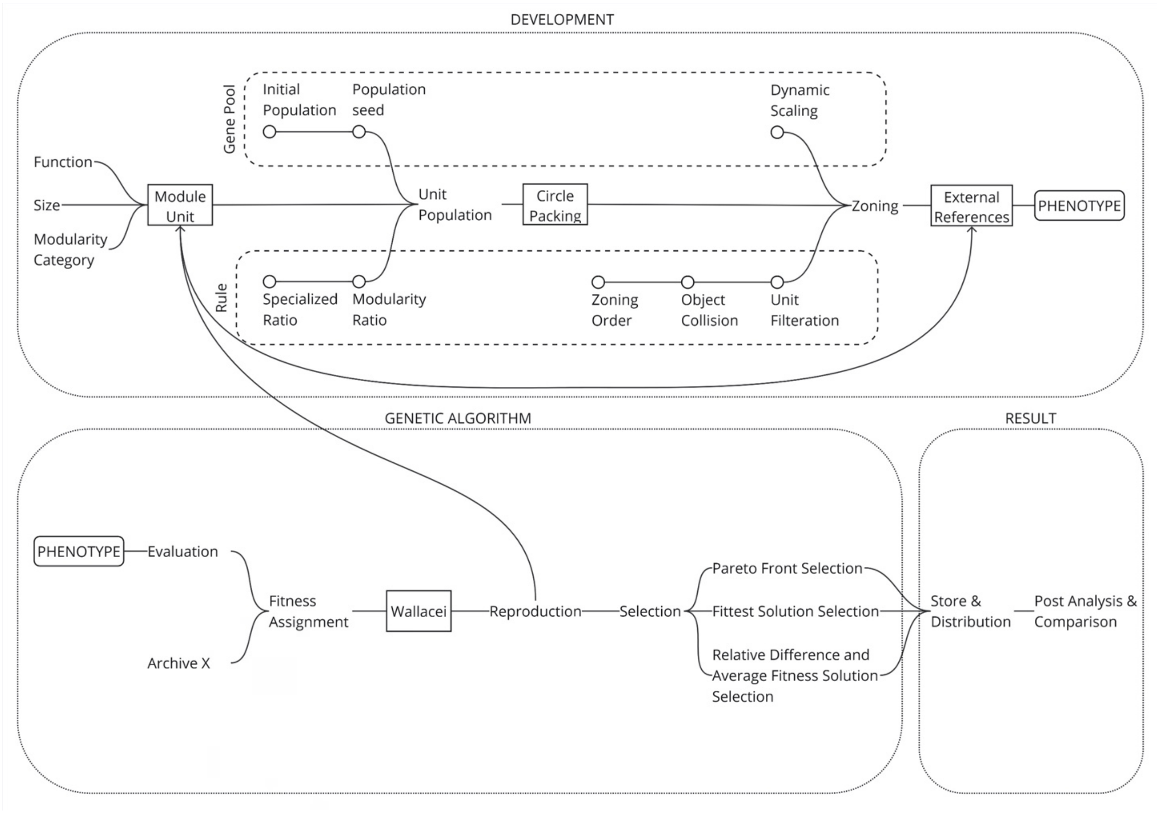


Figure 4. Proposed data flow to parametrize modularity within the digital genetic algorithm (GA) framework in visual programming

4.2. Algorithm result

As one of the essential aspects of constructing the phenotypes, the overview of the simulation needs to be analyzed to see the interaction between each setup that will influence the algorithm's result. Figure 5 shows that each of the five fitness functions exhibits a different degree of convergence towards local optima, with Fitness Functions 2, 3, and 4 (productivity population, sociability population, and total population) exhibiting a more noticeable increase in fitness throughout developed generations. By comparison, the two other Fitness Functions (health and wellness population and existing zone) are relatively less so. Overall, they show better fitness, but their convergence towards an optimum is slower. Figure 6 demonstrates that, despite the simulation aim being designed to prioritize a lower total population, this objective can fall to a no longer desirable level, and the frequency of occurrence of a total population below 300 m^2 is relatively low. These two analyses revealed that the design was incorrectly formulated, making it difficult to select for the emergence of variational modularity appropriately. Even so, the simulation's results can still be applied and deemed successful because they demonstrate how the algorithm repeatedly reformulates the design problem to find a balance between the fitness function, phenotypes, and chromosomes.

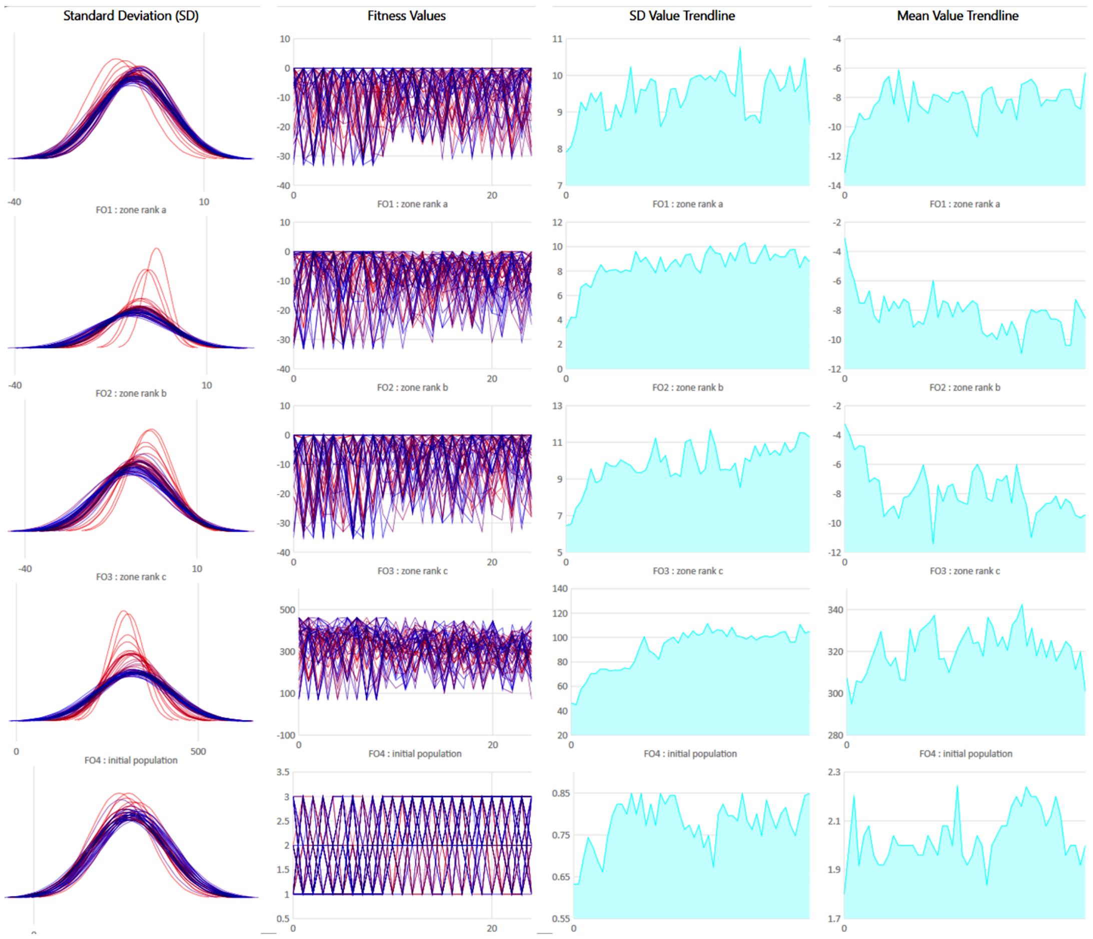


Figure 5. The evolutionary algorithm's output. Four important metrics were examined independently for each fitness function (from left to right): fitness values, standard deviation, mean value trendline, and standard deviation trendline

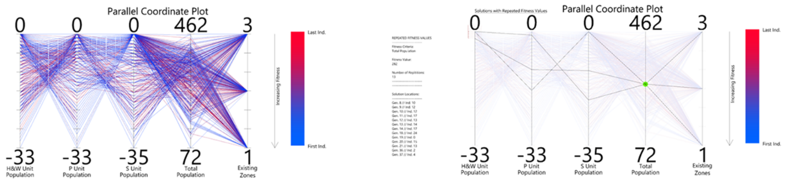


Figure 6. Parallel coordinate plot graph and the projection of the number solution in the total population below 300 m^2 repeats less and occurs less frequently

4.3. Pareto front selection

Using two approaches, the Pareto front's 25 solutions from the latest generation (generation 49) were compared to the 192 solutions that comprised the population's whole Pareto front. Using an average cluster size of nine solutions over the Pareto front and a K-value of 20, hierarchical (average linkage) clustering was performed. This is done to simplify the 192 options by choosing the one closest to the center of each cluster. Figures 7 and 8 depict the occurrence of dominance in solutions that emphasize development modularity, with the frequency of occurrence ordered from most frequent, namely in the balanced, dominated, and total zones. Out of the 20 solutions, three exhibit variational modularity. These three solutions are

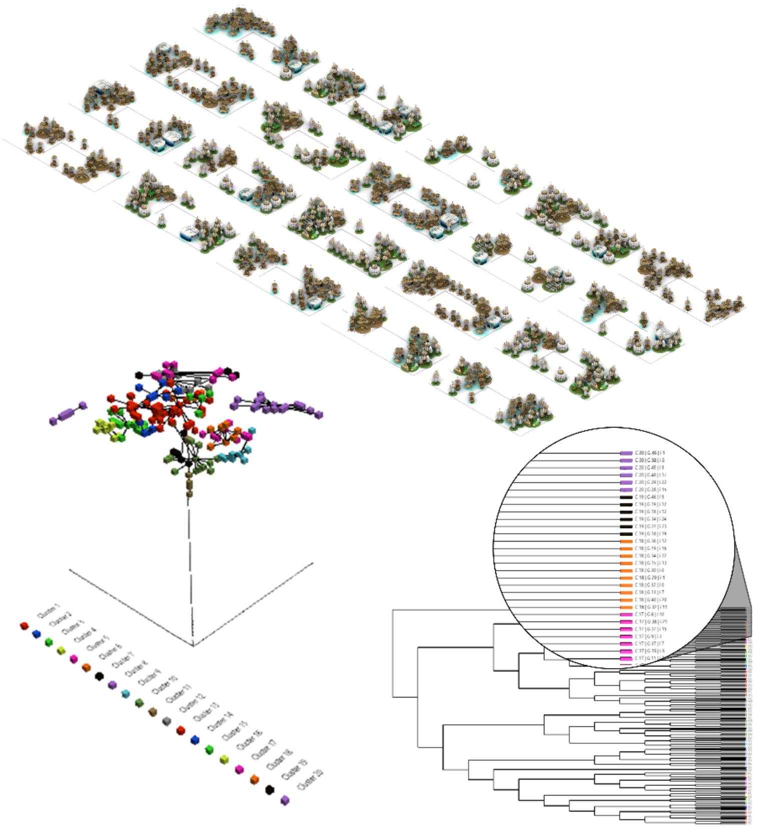


Figure 7. The selected phenotypes from the solution closest to each cluster with Pareto front solutions' hierarchical clustering with a K-value of 20 are displayed using the objective space and dendrogram

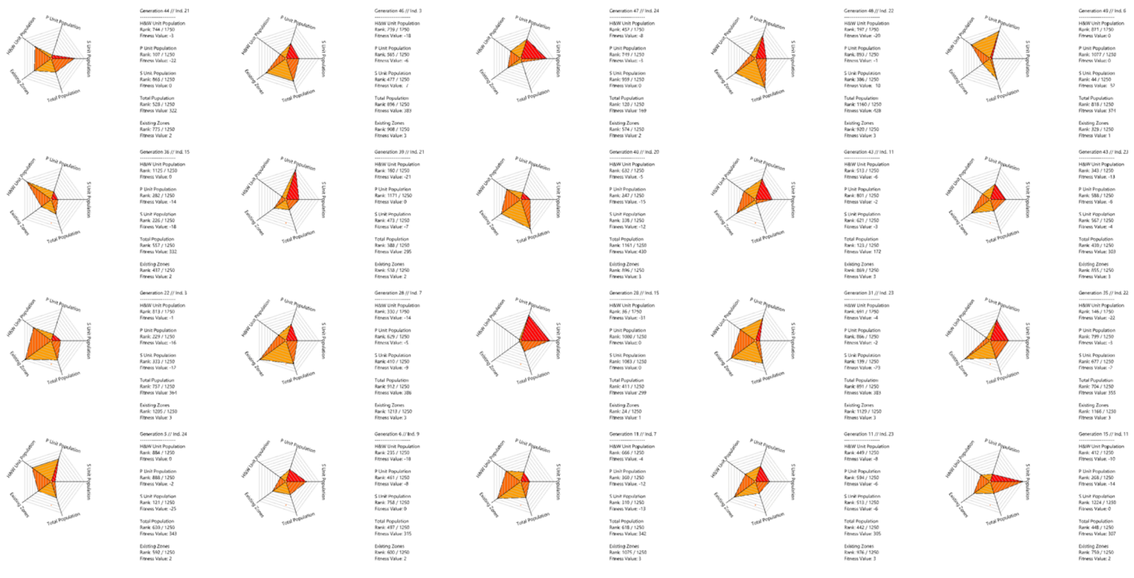


Figure 8. The description of each solution from the selected phenotypes of the Pareto front in all populations is presented as a list and a diamond diagram to make it easier to observe each vitality value

found in the balanced, dominated, and total zones, each of which is a solution from generation 28 individual 25, generation 43 individual 11, and generation 47 individual 25. From the Pareto front for the entire population, more varied results were obtained to determine the level of emergence of ideal office programs and drastic changes in office programs.

The Pareto front solution in generation 49, seen in Figures 9 and 10, shows the same dominance in developmental modularity, with half of it emerging in total zones. The emergence of variational modularity increases to 9 types, with four solutions appearing in total zones, four in dominated zones, and one in balanced zones. Suboptimal solutions can also be identified in populations that

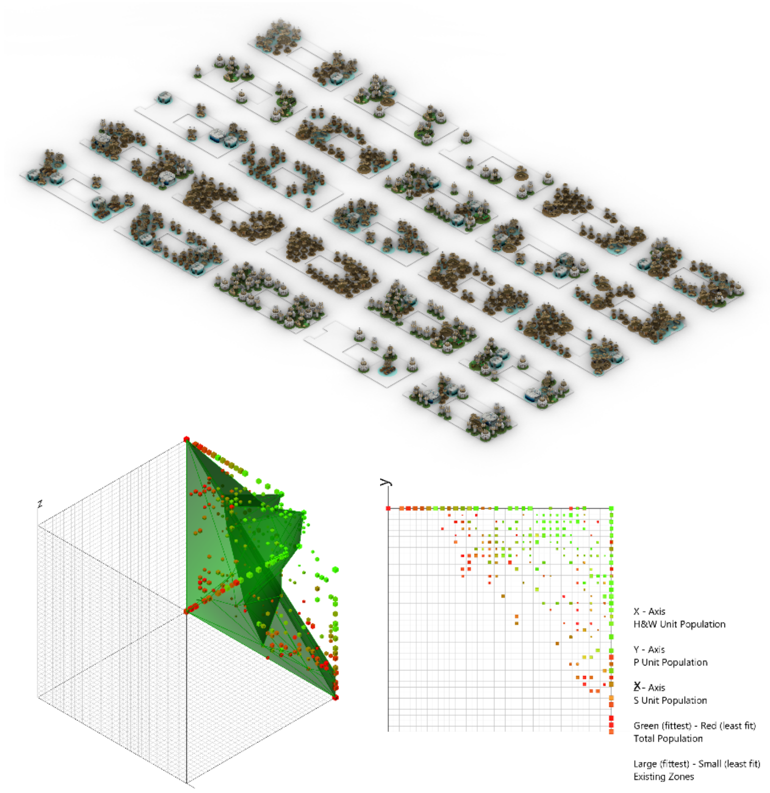


Figure 9. The selected phenotypes from the Pareto front solution in generation 49 are also displayed in the objective space

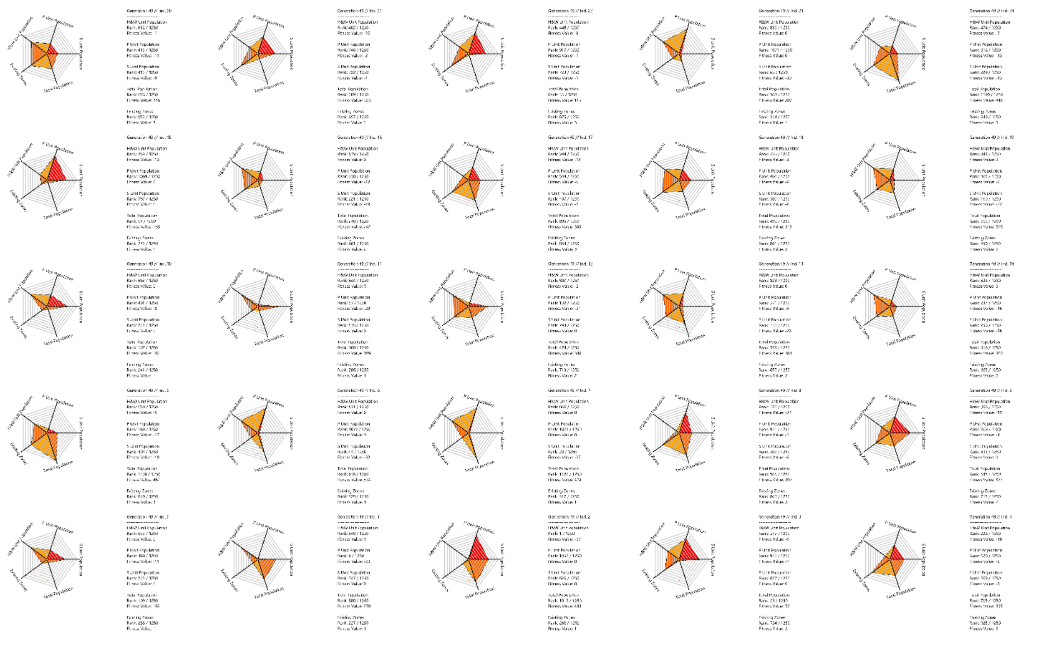


Figure 10. The description of each solution from the selected phenotypes of the Pareto front solution in generation 49 is presented as a list and a diamond diagram to make it easier to observe each vitality value

are too low, namely less than 200 m^2, which is caused by the failure of the emergence of variational modularity in 5 solutions. From the Pareto front in the last generation, more drastic variations appeared in the function of office programs, characterized by variational modularity and dominated and total zone types.

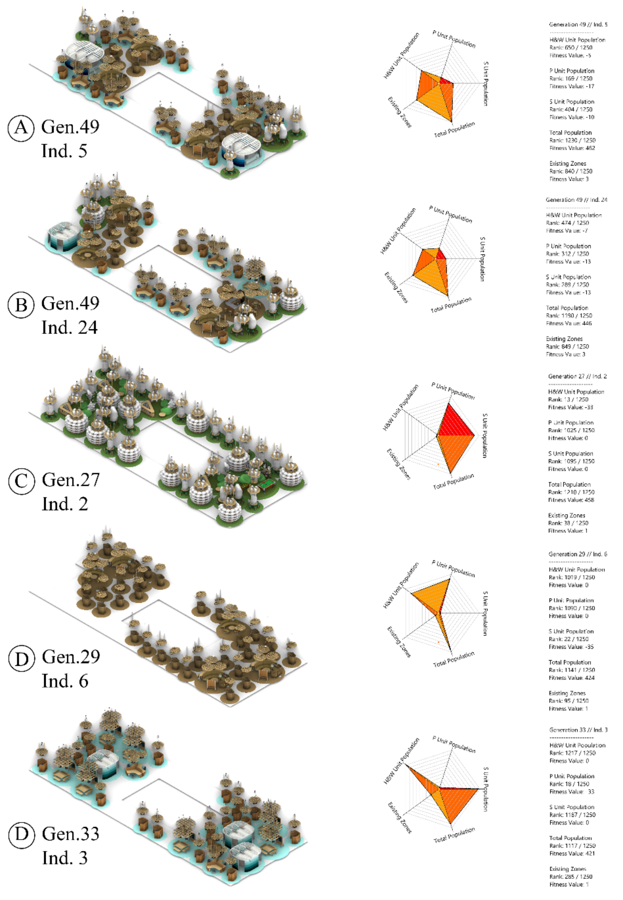
4.4. Fittest solution selection

As shown in Figure 11, the best-fit solution is chosen for every fitness function. The fittest solution for one function typically shows low fitness for other tasks because the fitness functions in the design challenge clash. The Fitness Function 1, 2, 3 (health & wellness, productivity, and sociability unit) that represents the maximum total of units in each function does come from the total zones in each of the respective objectives. This can indicate how drastic changes can happen within office programs with no variational modularity. However, it is still considered a change from one office program to another. With reverse input, Fitness Function 4 and 3 (total population and existing zone) can be identified to maximize the population with balanced zones.

4.5. Relative difference and average fitness solution selection

This selection uses the relative difference between rankings and fitness average ranking to sort the Pareto front individuals using the Parallel Coordinate Plot. The solution was selected based on the highest rank in each category, which was sorted into different solutions. Individuals with relative differences tend to find an equilibrium between all the fitness criteria. The relative difference between individuals usually helps achieve a balance between all fitness requirements. Meanwhile, the fitness average has the potential to introduce extreme individuals who specialize in one criterion. By weakening the other criteria, this specialisation enables the individual to achieve high ranks.

In Figure 12, four solutions can be found that have a balanced number of units in each of the functions, which can also be found in balanced zones. However, the total population is considered small because fitness function 4 (total population) is optimized to have a small unit population. Still, it can be regarded as an ideal office program variation. In Figure 13, three solutions can be found: 3 solutions with a drastic number of units in each function that can also be found in total zones. As with the solution for the relative difference, the total population is considered small. However, in this condition, variational modularity appears and becomes a variation of drastic changes in the office program into another program.



Generation 49 // Ind. 5

H&W Unit Population
Rank: 650 / 1250
Fitness Value: -5

P Unit Population
Rank: 169 / 1250
Fitness Value: -17

S Unit Population
Rank: 404 / 1250
Fitness Value: -10

Total Population
Rank: 1230 / 1250
Fitness Value: 462

Existing Zones
Rank: 840 / 1250
Fitness Value: 3

Generation 49 // Ind. 24

H&W Unit Population
Rank: 474 / 1250
Fitness Value: -7

P Unit Population
Rank: 312 / 1250
Fitness Value: -13

S Unit Population
Rank: 289 / 1250
Fitness Value: -13

Total Population
Rank: 1190 / 1250
Fitness Value: 446

Existing Zones
Rank: 849 / 1250
Fitness Value: 3

Generation 27 // Ind. 2

H&W Unit Population
Rank: 13 / 1250
Fitness Value: -33

P Unit Population
Rank: 1025 / 1250
Fitness Value: 0

S Unit Population
Rank: 1095 / 1250
Fitness Value: 0

Total Population
Rank: 1210 / 1250
Fitness Value: 458

Existing Zones
Rank: 38 / 1250
Fitness Value: 1

Generation 29 // Ind. 6

H&W Unit Population
Rank: 1019 / 1250
Fitness Value: 0

P Unit Population
Rank: 1090 / 1250
Fitness Value: 0

S Unit Population
Rank: 22 / 1250
Fitness Value: -35

Total Population
Rank: 1141 / 1250
Fitness Value: 424

Existing Zones
Rank: 95 / 1250
Fitness Value: 1

Generation 33 // Ind. 3

H&W Unit Population
Rank: 1217 / 1250
Fitness Value: 0

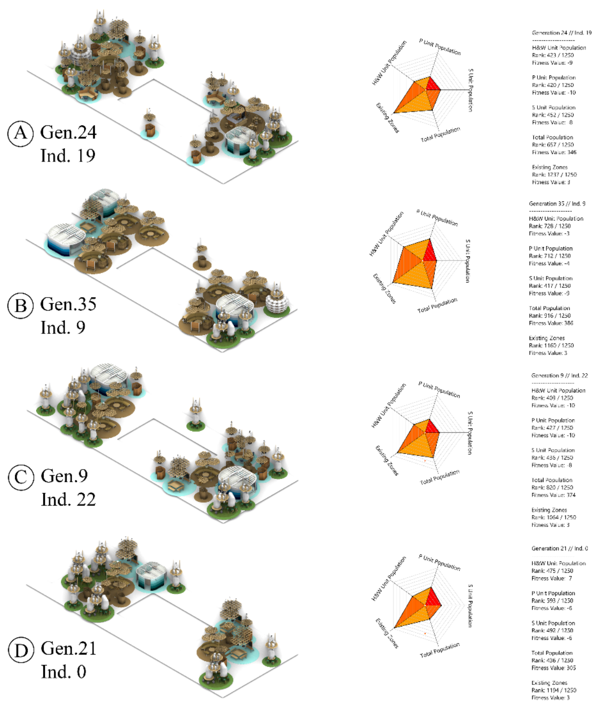
P Unit Population
Rank: 18 / 1250
Fitness Value: -33

S Unit Population
Rank: 1187 / 1250
Fitness Value: 0

Total Population
Rank: 1117 / 1250
Fitness Value: 421

Existing Zones
Rank: 285 / 1250
Fitness Value: 1

Figure 11. The selected phenotypes from the fittest solution on each of 5 objectives that are present in the simulation and the description of the details on each solution that are presented in the list and diamond chart



Generation 24 // Ind. 19

H&W Unit Population
Rank: 423 / 1250
Fitness Value: -9

P Unit Population
Rank: 420 / 1250
Fitness Value: -10

S Unit Population
Rank: 452 / 1250
Fitness Value: -8

Total Population
Rank: 657 / 1250
Fitness Value: 346

Existing Zones
Rank: 1237 / 1250
Fitness Value: 3

Generation 35 // Ind. 9

H&W Unit Population
Rank: 728 / 1250
Fitness Value: -3

P Unit Population
Rank: 712 / 1250
Fitness Value: -4

S Unit Population
Rank: 417 / 1250
Fitness Value: -9

Total Population
Rank: 916 / 1250
Fitness Value: 386

Existing Zones
Rank: 1160 / 1250
Fitness Value: 3

Generation 9 // Ind. 22

H&W Unit Population
Rank: 409 / 1250
Fitness Value: -10

P Unit Population
Rank: 427 / 1250
Fitness Value: -10

S Unit Population
Rank: 436 / 1250
Fitness Value: -8

Total Population
Rank: 820 / 1250
Fitness Value: 374

Existing Zones
Rank: 1064 / 1250
Fitness Value: 3

Generation 21 // Ind. 0

H&W Unit Population
Rank: 475 / 1250
Fitness Value: 7

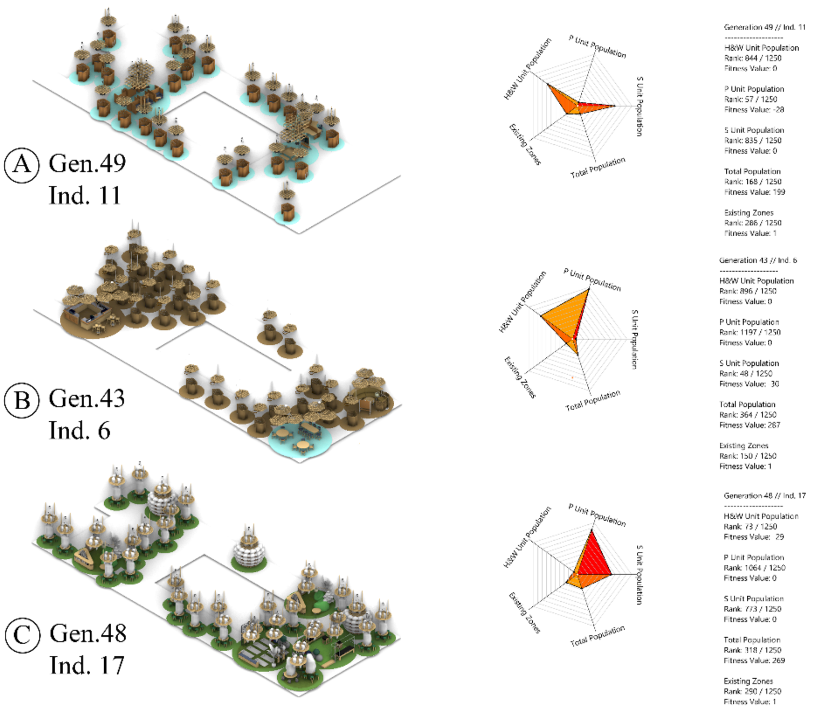
P Unit Population
Rank: 593 / 1250
Fitness Value: -6

S Unit Population
Rank: 492 / 1250
Fitness Value: -6

Total Population
Rank: 436 / 1250
Fitness Value: 305

Existing Zones
Rank: 1194 / 1250
Fitness Value: 3

Figure 12. The selected phenotypes from the relative differences between fitness rank present in the simulation and the description of the details on each solution are presented in the list and diamond chart



Generation 49 // Ind. 11

H&W Unit Population
Rank: 844 / 1250
Fitness Value: 0

P Unit Population
Rank: 57 / 1250
Fitness Value: -28

S Unit Population
Rank: 835 / 1250
Fitness Value: 0

Total Population
Rank: 168 / 1250
Fitness Value: 199

Existing Zones
Rank: 286 / 1250
Fitness Value: 1

Generation 43 // Ind. 6

H&W Unit Population
Rank: 896 / 1250
Fitness Value: 0

P Unit Population
Rank: 1197 / 1250
Fitness Value: 0

S Unit Population
Rank: 48 / 1250
Fitness Value: -30

Total Population
Rank: 364 / 1250
Fitness Value: 267

Existing Zones
Rank: 150 / 1250
Fitness Value: 1

Generation 48 // Ind. 17

H&W Unit Population
Rank: 73 / 1250
Fitness Value: -29

P Unit Population
Rank: 1064 / 1250
Fitness Value: 0

S Unit Population
Rank: 773 / 1250
Fitness Value: 0

Total Population
Rank: 318 / 1250
Fitness Value: 269

Existing Zones
Rank: 290 / 1250
Fitness Value: 1

Figure 13. The selected phenotypes from the average fitness rank that are present in the simulation and the description of the details on each solution that are presented in the list and diamond chart

5. Discussion

The study's results above show a meaningful interrelated relationship between the use of the algorithm concept in the morphogenesis framework to explore optimization variations. The modularity concept and algorithms form a close relationship, as shown by several forms of phenotypes selected through Pareto front, fittest Solution, relative difference, and Average Fitness Solution Selection. This relationship becomes stronger when placed in the morphogenesis framework, where form is not seen as a static end product, but rather as a result of growth, adaptation, and selection in a dynamic parameter environment. Morphogenesis places form as it grows through responses to internal and external pressures. The phenotypes produced in this study are real expressions of modular configurations that have gone through an algorithmic selection process in a multi-objective space. Several phenotypes manifest physical or morphological forms as end products resulting from the interaction between genotype (population parameters and zoning) and basic modules, as performative feedback that occurs during the morphogenesis process.

The modularity concept used in this study is building a system through small units that can be rearranged or combined to produce various optimization variations. The application of modularity increases the possibility of interior configurations in space programming to determine the optimum quality produced. The more modules used, the wider the optimization variations must be evaluated. However, not all modular combinations produced show optimal variations against the desired criteria related to space efficiency and functional flexibility. This is where the role of evolutionary algorithms becomes crucial. Multi-objective optimization algorithms explore, evaluate, and filter these complex optimization variations. The algorithm used in this study allows for systematic and efficient variation searches, considering more than one objective at a time.

Several experiments in building a modularity-based evolutionary computing methodology by selecting problems in programmed dynamic construction show that the genetic algorithm's optimization results (phenotype) are more than the extent to which it succeeds in meeting its design objectives. Evolutionary computing methods generally focus more on the shape of the resulting phenotype to produce the desired optimization. At the same time, the characteristics are only additional data to describe each objective. This method can facilitate more abstract problems, such as space programming, where the objective factors are essential as a benchmark for combinations between several types of units, where form only plays a role in facilitating the observation process.

The genetic algorithm used in this experiment only functioned as a "tool" system. However, the entire framework (data flow) obtained can be reconstructed to develop other evolutionary development (evo-devo) theories and produce forms that are more than just the evolutionary process results. The modularity-based evolutionary development applied in this experiment must be seen from how it works regarding the relationships between components to form a system. So, it depends more on which scale you want to choose, for example, a space program that includes furniture as part of the unit, or even smaller if the furniture is seen as a separate system with various other components that make it up.

The result of the algorithmic process in this study is shown through identifying phenotypes as a set of modular variations that dominate each other. Each phenotype represents an optimal balance between various objectives, where no single solution can be satisfied in one aspect without sacrificing other aspects. Thus, the relationship between modularity and algorithms in the morphogenesis framework is essential in multi-criteria design and decision making. Modularity enriches the possibilities, algorithms process and optimize, and phenotypes become visual and conceptual representations of the optimal variations available.

6. Conclusions

Developing a mechanism combining evolutionary computation with natural processes to obtain direct feedback has become essential in applying digital morphogenesis to enrich architectural design methods. By exploring design solutions that can be optimized with evolutionary computing in architecture, a type of problem that is not considered pragmatic but is very important in the architectural design process, such as the programmatic function of the building, can be developed and optimized. By applying the theory of morphological integration in biology, namely modularity, a method can be constructed to optimize more abstract problems in phenotypes and enrich the use of evolutionary computation step by step in synchronizing feedback with natural processes.

This study demonstrated that an evolutionary algorithm that focuses on the relationship between modularity and algorithms played an essential role in shaping the process of morphogenesis, namely the process of growth and formation of form through dynamic interactions between parts of the system. Modularity becomes the foundation for morphogenesis by providing flexible units that can be combined, reconfigured, or transformed as context, functional needs, and form aspirations change. Through this relationship, design is no longer simply looking for one "best" form, but rather celebrating a spectrum of responsive, adaptive, and diverse phenotypes following the dynamics of complex goals.

The simulation results show a series of configurations made from small unit components with large spatial programming functions that can adapt to changes in other programs, both significant and insignificant. These various options can be used to support the needs of space programming in controlling internal operations and estimating the use of different existing units so that they can be reused for other program initiatives. This research is limited to its application on the scale of the interior space of the building. Still, it does not rule out the possibility that this evolutionary algorithm can be applied to other studies, such as studying interior programs in an urban context. However, the larger the modular categorization used, the more variations will be produced. As knowledge, the results of this study have the potential to be developed as a modular-based generative design method to define adaptive architecture.

Acknowledgements

This research was financially supported by Universitas Diponegoro, Indonesia, through the International Reputable Research Grant (RPI) 2024.

References

Baden, N., & Taghizadeh, M. (2023). *Building performance optimisation tools for the decarbonisation of Swedish buildings*. http://lup.lub.lu.se/student-papers/record/9130848

Ball, P. (2009). *Nature's patterns: A tapestry in three parts*. Oxford University Press.

Bartz-Beielstein, T., Branke, J., Mehnen, J., & Mersmann, O. (2014). Evolutionary algorithms. *WIREs Data Mining and Knowledge Discovery*, *4*(3), 178–195. https://doi.org/10.1002/widm.1124

Canestrino, G., Laura, G., Spada, F., & Lucente, R. (2020). Generating architectural plan with evolutionary multiobjective optimization algorithms: A benchmark case with an existent construction system. In *Blucher Design Proceedings* (pp. 149–156). Blucher. https://doi.org/10.5151/sigradi2020-21

Davis, D. (2009). *Evolving digital morphogenesis by means of biology and computer science* [Thesis, Victoria University]. https://www.danieldavis.com/evolving-digital-morphogenesis/

Dixit, S., & Stefańska, A. (2023). Bio-logic, a review on the biomimetic application in architectural and structural design. *Ain Shams Engineering Journal*, *14*(1), Article 101822. https://doi.org/10.1016/j.asej.2022.101822

El-Shorbagy, M. A., & El-Refaey, A. M. (2020). Hybridization of Grasshopper optimization algorithm with genetic algorithm for solving system of non-linear equations. *IEEE Access*, *8*, 220944–220961. https://doi.org/10.1109/ACCESS.2020.3043029

Fogel, D. B. (1999). An overview of evolutionary programming. In L. D. Davis, K. De Jong, M. D. Vose, & L. D. Whitley (Eds.), *Evolutionary algorithms* (Vol. 111, pp. 89–109). Springer. https://doi.org/10.1007/978-1-4612-1542-4_5

Gökmen, S. (2020). Rediscovering Goethe's concept of polarity: A new direction for architectural morphogenesis. *Metu Journal of the Faculty of Architecture*, *37*(1), 51–72. https://doi.org/10.4305/METU.JFA.2020.1.5

Harani, A. R., Atmodiwirjo, P., Yatmo, Y. A., & Riskiyanto, R. (2021). The existence of a shortcut as an urban space system to support physic and mental health. *IOP Conference Series: Earth and Environmental Science*, *623*(1), Article 012041. https://doi.org/10.1088/1755-1315/623/1/012041

Harding, J., & Brandt-Olsen, C. (2018). Biomorpher: Interactive evolution for parametric design. *International Journal of Architectural Computing*, *16*(2), 144–163. https://doi.org/10.1177/1478077118778579

Kamaoğlu, M. (2023). The idea of evolution in digital architecture: Toward united ontologies? *International Journal of Architectural Computing*, *21*(4), 622–634. https://doi.org/10.1177/14780771231174890

Kasyanov, N. (2020). Research on the similarities of morphogenesis in architecture and nature. In *Proceedings of the 2nd International Conference on Architecture: Heritage, Traditions and*

Innovations (*AHTI 2020*) (Vol. 471, pp. 260–264), Moscow, Russia. https://doi.org/10.2991/assehr.k.200923.045

Katoch, S., Chauhan, S. S., & Kumar, V. (2021). A review on genetic algorithm: Past, present, and future. *Multimedia Tools and Applications*, *80*(5), 8091–8126. https://doi.org/10.1007/s11042-020-10139-6

Leach, N. (2009). Digital morphogenesis. *Architectural Design*, *79*(1), 32–37. https://doi.org/10.1002/ad.806

Leach, N., Turnbull, D., & Williams, C. (Eds.). (2004). *Digital tectonics*. Wiley-Academy.

Melo, D., Porto, A., Cheverud, J. M., & Marroig, G. (2016). Modularity: Genes, development, and evolution. *Annual Review of Ecology*, *Evolution*, *and Systematics*, *47*(1), 463–486. https://doi.org/10.1146/annurev-ecolsys-121415-032409

Miikkulainen, R., & Forrest, S. (2021). A biological perspective on evolutionary computation. *Nature Machine Intelligence*, *3*(1), 9–15. https://doi.org/10.1038/s42256-020-00278-8

Minter, N. J., Franks, N. R., & Robson Brown, K. A. (2012). Morphogenesis of an extended phenotype: Four-dimensional ant nest architecture. *Journal of The Royal Society Interface*, *9*(68), 586–595. https://doi.org/10.1098/rsif.2011.0377

Navarro-Mateu, D., & Cocho-Bermejo, A. (2019). Evo-devo algorithms: Gene-regulation for digital architecture. *Biomimetics*, *4*(3), Article 58. https://doi.org/10.3390/biomimetics4030058

Rees, J. M. (2018). Surform: An architectural vocabulary of morphogenesis. *Footprint*, (22), 113–122. https://doi.org/10.7480/footprint.12.1.1751

Riskiyanto, R., Andri Yatmo, Y., & Atmodiwirjo, P. (2021). Reading (Hidden) dialogue of organic tectonics. *The Plan Journal*, *6*(2). https://doi.org/10.15274/tpj.2021.06.02.5

Roudavski, S. (2009). Towards morphogenesis in architecture. *International Journal of Architectural Computing*, *7*(3), 345–374. https://doi.org/10.1260/147807709789621266

Suharjito, S., & Muslim, M. (2023). Optimization of facility layout problems using genetic algorithm. *Syntax Literate*; *Jurnal Ilmiah Indonesia*, *7*(9), 16058–16077. https://doi.org/10.36418/syntax-literate.v7i9.13787

Thompson, D. W. (1992). *On growth and form* (J. T. Bonner, Ed.). Cambridge University Press. https://doi.org/10.1017/CBO9781107325852