



**QUEEN'S
UNIVERSITY
BELFAST**

Evolving novel gene regulatory networks for structural engineering designs

Dubey, R., Hickenbotham, S., Colligan, A., Friel, I., Buchanan, E., Price, M., & Tyrrell, A. M. (2024). Evolving novel gene regulatory networks for structural engineering designs. *Artificial Life*. Advance online publication. https://doi.org/10.1162/artl_a_00448

Published in:
Artificial Life

Document Version:
Peer reviewed version

Queen's University Belfast - Research Portal:
[Link to publication record in Queen's University Belfast Research Portal](#)

Publisher rights

© 2024 The MIT Press

This work is made available online in accordance with the publisher's policies. Please refer to any applicable terms of use of the publisher.

General rights

Copyright for the publications made accessible via the Queen's University Belfast Research Portal is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy

The Research Portal is Queen's institutional repository that provides access to Queen's research output. Every effort has been made to ensure that content in the Research Portal does not infringe any person's rights, or applicable UK laws. If you discover content in the Research Portal that you believe breaches copyright or violates any law, please contact openaccess@qub.ac.uk.

Open Access

This research has been made openly available by Queen's academics and its Open Research team. We would love to hear how access to this research benefits you. – Share your feedback with us: <http://go.qub.ac.uk/oa-feedback>

Artificial Life Manuscript Submission

Evolving Novel Gene Regulatory Networks for Structural Engineering Designs

Rahul Dubey ¹, Simon Hickenbotham ¹, Andrew Colligan ², Imelda Friel ², Edgar Buchanan ¹, Mark Price ², Andy M Tyrrell ¹

Corresponding: Rahul Dubey (rahul.dubey@york.ac.uk)

1. School of Physics, Engineering and Technology, University of York, UK
2. School of Mechanical and Aerospace Engineering, Queen's University Belfast, NI

Abstract. Engineering design optimization poses a significant challenge, usually requiring human expertise to discover superior solutions. While various search techniques have been employed to generate diverse designs, their effectiveness is often limited by problem-specific parameter tuning, making them less generalizable and scalable. This paper introduces a framework inspired by evolutionary and developmental (Evo-Devo) concepts, aiming to automate the evolution of structural engineering designs. In biological systems, Evo-Devo governs the growth of single-cell organisms into multi-cellular organisms through the use of Gene Regulatory Networks (GRNs). GRNs are inherently complex and highly nonlinear, and this paper explores the use of neural networks and genetic programming as artificial representations of GRNs to emulate such behaviors. In order to evolve a wide range of Pareto fronts for artificial GRNs, this paper introduces a new technique, a real-value encoded neuro-evolutionary method termed “real-encoded NEAT” (RNEAT). The performance of RNEAT is compared with two well-known evolutionary search techniques across different 2D and 3D problems. The experimental results demonstrate two key findings: Firstly, the proposed framework effectively generates a population of GRNs that can produce diverse structures for both 2D and 3D problems. Secondly, the proposed RNEAT algorithm outperforms its competitors on more than 50% of the problems examined.

Keywords: Evolutionary Search, Gene Regulatory Networks, NEAT, CGP, Design Optimization

1 Introduction

Evolutionary developmental (Evo-Devo) biology is the part of biology that tries to explain the evolution of growth patterns in organisms, which determine how they develop from a single cell to adulthood (Hall, 2012). Every cell of an organism contains the same DNA yet each can function in different ways to generate different organs like the heart or eyes (Olson, 2006). These differences in functionality are caused by gene regulation that turns on and off different parts of the genome depending on local environmental factors, endowing cells with *state*. The DNA/genome of a cell consists of a set of genes but only a fraction of these are activated (turned ON) to create a specific organ. The cohort of gene regulations happening in parallel can then generate a complex multi-organ organism. These different molecular gene regulators combined together create a Gene Regulatory Network (GRN) (Cussat-Blanc et al., 2019). The GRN is the central and the most crucial component of the evolutionary developmental cycle.

This paper takes inspiration from the Evo-Devo concept to generate different engineering designs by evolving an artificial gene regulatory network. In the field of engineering, structural design optimization has been a topic of study for several decades, involving the use of domain expert knowledge (Christensen & Klarbring, 2008) and computational intelligence techniques (Chi et al., 2021). This optimization process encompasses various aspects such as size, shape, and topology optimization, either individually or in combination. Conventional approaches to design optimization focus on finding solutions for specific problems by directly encoding the structural representation of the genome. This approach often results in a large search space that needs to be explored through automated evolution. Direct encoding implies that these approaches can only be problem specific, and so face limitations in both scalability and generalizability.

To address these challenges, a framework based on Evo-Devo principles to generate and evolve diverse designs using artificial gene regulatory networks is introduced here. The

27 evolutionary component offers a direct encoding of GRNs, while the developmental aspect
28 utilizes these GRNs to update the structures. The primary objective of this approach is to
29 evolve GRNs that are both generalizable and scalable, enabling them to effectively control
30 the growth of engineering designs in order to optimize the structure for given performance
31 targets (e.g. structural loads). In the existing literature, various computational models of
32 GRNs have been proposed, ranging from low complexity with better explainability to high
33 complexity with lower explainability (Karlebach & Shamir, 2008). However, there is cur-
34 rently no consensus regarding the most suitable method or methods for their applicability.
35 GRNs are inherently non-linear and thus different non-linear models have been studied to
36 mimic the behavior of GRNs.

37 In this paper, an artificial gene regulatory network is represented by two different mod-
38 els: neural networks (NN) and computer programs encoded in graphs where problems
39 are formulated as multi-objective optimization problems. The nonlinearity of both NNs
40 and graphs can be varied by changing their hyper-parameters. Since the problems are
41 multi-objective in nature, two different evolutionary search methods have been employed:
42 multi-objective cartesian genetic programming (CGP) (Miller & Harding, 2008), and multi-
43 objective neuro-evolution of augmenting topologies (NEAT) (Stanley & Miikkulainen, 2002)
44 are used to evolve complex computer programs and neural networks respectively. How-
45 ever, extending NEAT to evolve for multi-objective problems while maintaining speciation
46 poses a significant challenge (van Willigen et al., 2013). To evolve multi-objective neural
47 network topologies, this paper introduces a CGP-style encoding to encode a neural archi-
48 tecture, where the number of hidden layers and nodes in each hidden layer can be defined.
49 CGP-style network encoding allows tuning of the connections between nodes, weights, and
50 biases. The entire architecture can be encoded into a real-value chromosome and thus is
51 referred to as “Real-encoded NEAT” (RNEAT).

52 By employing the proposed Evo-Devo-based approach, a range of experiments were car-

53 ried out on diverse 2D and 3D engineering structural design problems. In these experi-
54 ments, a GRN receives inputs derived from the local geometry of the structure, including
55 the cross-sectional (CS) area of members, node locations, as well as information obtained
56 through finite element analysis (FEA), such as member force or strain energy (SE). The GRN,
57 encoded as either a NN or a computer program, subsequently generates small changes
58 to update (i.e. grow) the physical structure, aiming to minimize both the volume and max-
59 imum deflection. The experimental results obtained under various settings demonstrate
60 the viability of the Evo-Devo-based approach for controlling the growth of structures while
61 achieving the desired objectives. Furthermore, the results reveal that RNEAT-evolved so-
62 lutions outperformed its competitors on more than 50% of the problems, indicating its
63 superiority in terms of effectiveness and performance.

64 The two major contributions of this paper are as follows: 1) the paper introduces a gener-
65 alizable and scalable approach based on the Evo-Devo concept to evolve diverse Pareto
66 front engineering designs, and 2) the paper presents RNEAT by taking inspiration from
67 NEAT and CGP to evolve multi-objective NNs. The rest of this paper is structured as fol-
68 lows. Section 2 discusses the literature on Evo-Devo approaches, GRNs, and evolutionary
69 search in engineering design. Section 3 presents the proposed Evo-Devo framework, and
70 section 4 outlines different GRN representations and the methodology of the proposed
71 RNEAT algorithm. The experimental setup and resulting data are presented and discussed
72 in Section 5, while section 6 provides the conclusions and suggests areas for future re-
73 search.

74 **2 Related Work**

75 Numerous difficulties emerge when employing evolutionary search techniques to evolve
76 solutions for intricate real-world problems, including the challenges of selecting relevant
77 parameters for tuning and formulating the fitness function (Goldberg, 2002; Osaba et al.,

78 2021). The complexity of the search space further exacerbates the situation in engineering
79 design problems, as these are characterized by a nonlinear relationship between parame-
80 ters that generate phenotypes (Zou et al., 2022). This paper introduces an Evo-Devo-based
81 approach to address these challenges in evolving GRNs for controlling the growth of struc-
82 tural designs.

83 Approaches based on Evolutionary development have been studied to improve the perfor-
84 mance of both hardware and software in different domain-specific tasks that include de-
85 signs (Richards et al., 2012), digital architecture (Navarro-Mateu & Cocho-Bermejo, 2019),
86 music (Albarracín-Molina et al., 2016), and color-based pattern generation (Navarro-Mateu
87 & Cocho-Bermejo, 2020). For instance, Vuk (Vujovic et al., 2017) introduced an Evo-Devo
88 strategy to evolve the morphology of physical robots. Their proposed approach enabled
89 robots to grow their leg size to simulate ontogenetic morphological changes in three de-
90 velopmental steps. In each step either the length of robot legs changes or the length and
91 thickness both change.

92 Wu (Wu et al., 2022) proposed an evolutionary developmental framework to facilitate robotic
93 Chinese stroke writing. Here a genome encodes stroke trajectory points, and the fitness of
94 the genome is computed using a developmental learning algorithm. However, the genome
95 encoding is not scalable and evolved solutions are problem specific. Jon (McCormack &
96 Gambardella, 2022) presented “growing and evolving 3D printable designs” where a covari-
97 ance matrix adaptation evolutionary strategies algorithm (CMA-ES) tunes five genetic pa-
98 rameters for optimizing 3D printable structures. Bidlo (Bidlo & Dobeš, 2020) presented an
99 evolutionary developmental method for the design of arbitrarily-growing sorting networks.
100 The proposed method basically evolves a grammar, an alphabetically encoded genome, to
101 generate complex strings, and these strings are later converted onto comparator structures
102 which are the building blocks of sorting networks.

103 These Evo-Devo approaches use evolutionary algorithms to tune numerical or alphabetical

104 parameters to evolve solutions and are limited in terms of scalability and generalizability.
105 Unlike these, the Evo-Devo approach proposed in this paper relies on GRNs to govern the
106 growth of design in developmental steps. In the field of engineering design, Evo-Devo
107 inspired approaches have not been investigated significantly to evolve designs.

108 The GRN plays a crucial role in the development of organisms and different computational
109 techniques that act as GRN representations have been studied in the literature e.g. (Cussat-
110 Blanc et al., 2019; Delgado & Gómez-Vela, 2019; Karlebach & Shamir, 2008; Schlitt &
111 Brazma, 2007). These techniques range from simple logical functions such as Boolean
112 functions to complex non-linear functions represented by neural networks. In the litera-
113 ture, various evolutionary and swarm algorithms have been used to evolve GRNs. In 2005
114 Swain (Swain et al., 2005) used evolutionary algorithms to generate computational mod-
115 els of GRNs using data obtained from observations. Xu (Xu et al., 2007) used differential
116 evolution (DE), partial swarm optimization (PSO), and a hybrid of DE and PSO to optimize
117 the hyperparameters of recurrent neural networks to model the behavior of a GRN on time
118 series data-set, and Sylvain (Cussat-Blanc et al., 2015) used NEAT to evolve neural network
119 topologies. In a manner similar to neural networks, computer graphs (evolving using ge-
120 netic programming) have also been studied as a representation for GRN e.g. (Streichert et
121 al., 2004). In this paper, neural networks and genetic programming are considered proxies
122 for artificial gene regulatory networks that govern the growth of structural designs and are
123 evolved using CGP and NEAT. CGP was chosen to evolve computer programs because it
124 has been shown in the literature that other types of GPs suffer from bloating whereas CGP
125 does not (Turner & Miller, 2014).

126 Evolving multi-objective neural architectures has been recognized as a challenging prob-
127 lem. Willigen (van Willigen et al., 2013) modified standard NEAT using strength Pareto
128 evolutionary algorithms (SPEA2) to evolve multi-objective network topologies. However, in
129 order to preserve speciation in the population, fitness-based domination is used to convert

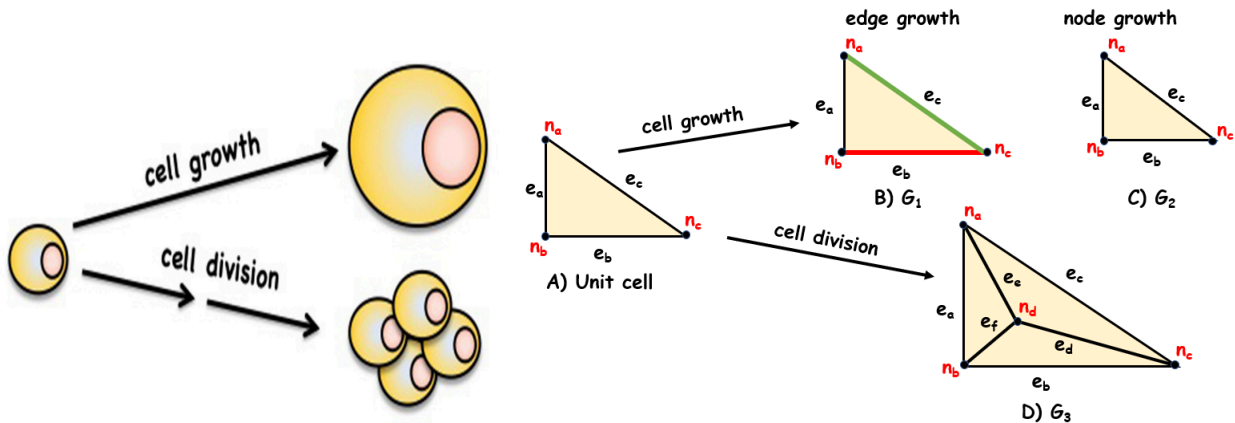
130 multi-objective fitness into single-objective fitness and thus is not a true multi-objective
131 NEAT. Schrum (Schrum & Miikkulainen, 2008) used NSGA2 (Deb et al., 2002) to modify
132 NEAT and evolved different topologies. The concept of speciation is not used here be-
133 cause fitness sharing in multi-objectives is difficult, as there is more than one objective
134 value. This paper presents a hybrid of CGP and NEAT, where encoding is inspired by CGP
135 and information flow between layers is inspired by NEAT.

136 Apart from Evo-Devo-based approaches, in the literature, different bio-inspired algorithms
137 have been used to evolve design (Balamurugan et al., 2008; Perez & Behdinan, 2007;
138 Yildiz, 2013) where problems are categorized into topology, size, and shape optimization.
139 In topology optimization, a bit-array representation scheme has been frequently used to
140 formulate the design problem where the design space is discretized into a grid where
141 rectangular blocks are filled with materials to generate different topologies (Wang et al.,
142 2006). The optimal solution in this representation is dependent on the resolution of the
143 grid where the smaller the resolution, the better chance of searching for optimal solutions.
144 As the grid resolution increases, the search space increases, and thus these approaches
145 are not scalable as well as generalizable. The next section describes in detail the new
146 algorithm and framework that form the foundation work of this paper.

147 **3 Methodology**

148 **3.1 Biological to Structural Cell Analogy**

149 As biological cells are the basic building block of any organ (Kaldis, 2016), this paper
150 considers a triangular shape cell for 2D problems and a tetrahedral shape cell for 3D prob-
151 lems as building blocks to create engineering designs. Figure 1 (a) illustrates biological
152 cell growth where the size of the cell changes and the cell multiplies (by dividing a cell) to
153 generate multiple cells. Cell growth and division mechanisms are controlled by a GRN or



(a) biological cell growth and division (Kaldis, 2016)

(b) 2D triangular cell, its growth, and division

Figure 1: Biological and structural cell analogy. Figure show (a) an example biological unit cell, its growth, and division, (b) show examples of cell growth in three different types of growth mechanisms: \mathcal{G}_1 edge growth, \mathcal{G}_2 node growth, and \mathcal{G}_3 cell division.

154 a set of GRNs based on local environmental factors.

155 Figure 1(b) shows an artificial cell, its growth, and multiplication mechanisms. An artificial
 156 2D triangular shape cell¹ consists of three nodes ($n_a, n_b,$ and n_c) and three edges ($e_a, e_b,$ and
 157 e_c) where the cell grows by changing the properties of nodes and edges. For consistency,
 158 cell division is also referred to as a type of growth mechanism here. A cell grows in one
 159 of the following ways: edge growth (\mathcal{G}_1), node growth (\mathcal{G}_2), and cell division (\mathcal{G}_3) as shown
 160 in Figure 1(b). A GRN takes the state information of a cell (such as the cross-section area
 161 of edges, locations of nodes, and other properties computed using finite element analysis
 162 such as edge force and strain energy) and generates a response to growing the cell.

163 In the first type of growth \mathcal{G}_1 , a GRN governs/updates the thickness of the edge/member's
 164 CS area as shown in Figure 1(b)(B) where the red line shows an increment in CS area, the
 165 green line indicates a reduction in CS area. By changing the CS area of members, the
 166 size of the structure can be optimized. In a similar fashion, the second type of growth

¹This type of cell representation has been chosen because the aim here is to evolve 2D and 3D truss-like structural designs. This cell definition can be easily changed as per the problem requirement.

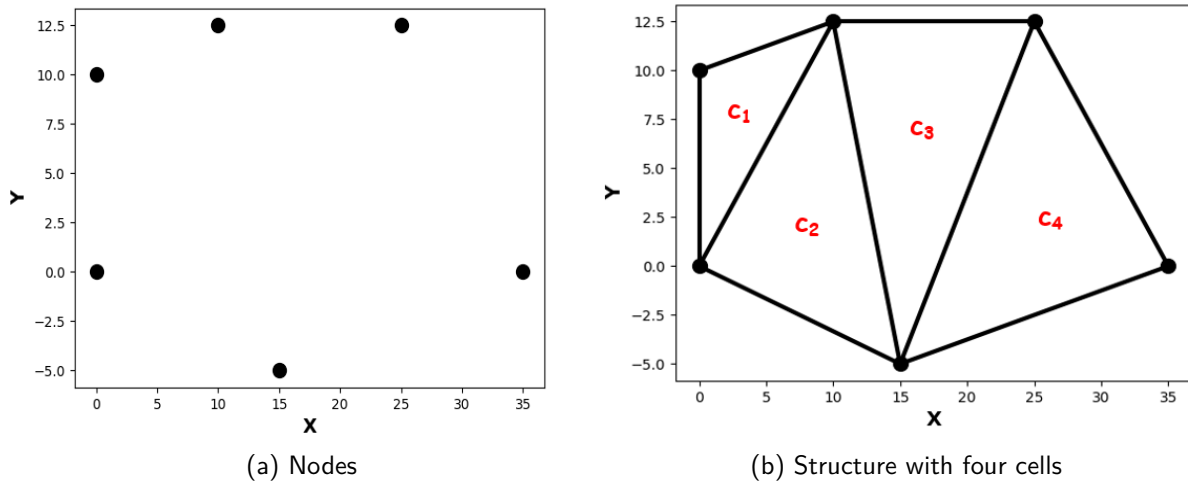


Figure 2: Illustrates an example of the initial seedling/structure generated using a set of points, (a) shows a set of points which are locations of supports and loads, and (b) shows the resulting geometry with four cells (c_1, c_2, c_3, c_4) using Delaunay triangulation.

167 mechanism (\mathcal{G}_2) updates the location of the nodes as shown in Figure 1(b)(C). Again, by
 168 changing the node locations, the size of the structure can be optimized. Finally, the topol-
 169 ogy of the structure can be modified by adding nodes to the structure, the third type of
 170 growth mechanism (\mathcal{G}_3) where a single cell is divided into three cells by placing a node
 171 (n_d) at the centroid of the parent cell Figure 1(b)(D). In the proposed Evo-Devo framework,
 172 a node can be added at the centroid of a cell. The same types of growth mechanisms are
 173 also applicable to 3D structures where a cell is tetrahedral.

174 3.2 Structure Initialization

175 The previous subsection discussed artificial cell representation and different types of
 176 growth mechanisms. These cells are the basic building blocks of design. To produce
 177 designs, an initial structure is generated within an environment that represents the design
 178 context, for example including support points, load points, load magnitude with direction,
 179 and material properties. One such example of a 2D initial structure is shown in Figure 2
 180 where (a) shows the location of points, and (b) shows the resulting geometry. The initial

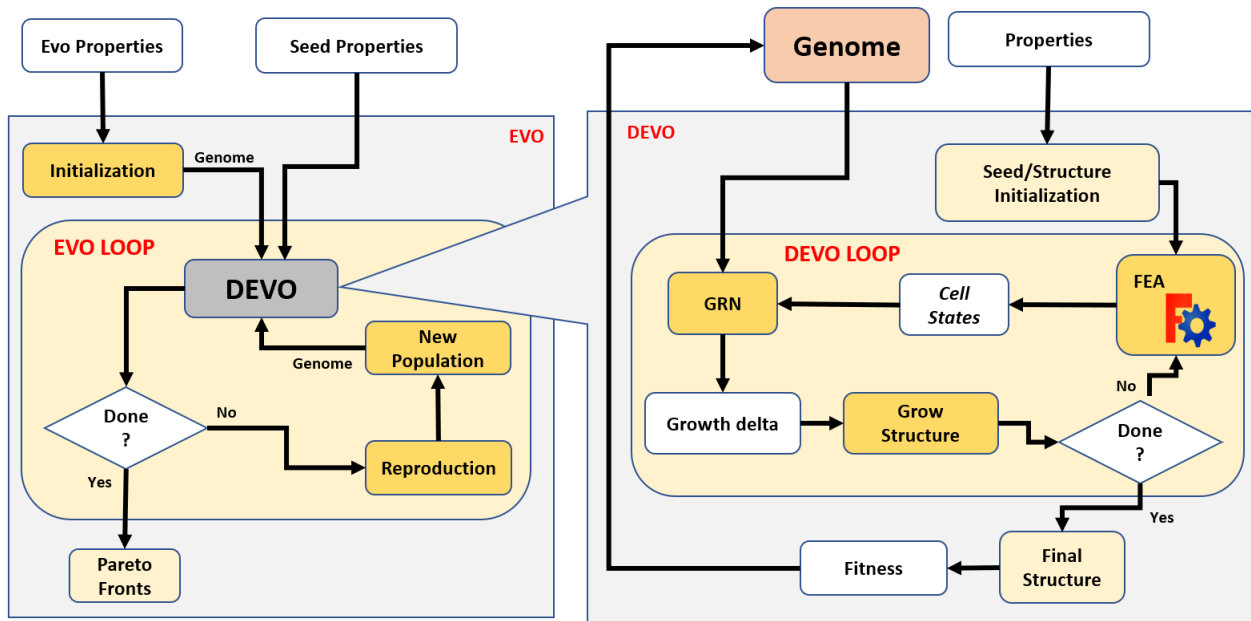


Figure 3: Block diagram representation of the Evo-Devo framework for growing structural design where Evo evolves solutions/GRNs and Devo grows the structure for a predefined number of developmental steps for each GRN. A GRN controls the growth and at the end, the physical properties of the structure are assigned as the fitness to the GRN.

181 topology is called the ‘seedling’, and it is from this basis that all growth steps proceed. In
 182 the case of the simple frame structures being used here, a Delaunay triangulation (DT) can
 183 be used to create the topology of the seedling/structure. The initialized structure is made
 184 up of four cells and each cell has three nodes and three edges/members. Once the ini-
 185 tial structure/seed is generated, this structure can grow depending on local environmental
 186 factors in the evo-devo phase.

187 The block diagram in Figure 3 illustrates the Evo-Devo framework for the evolutionary de-
 188 velopment of engineering designs. The framework consists of two main components: Evo
 189 and Devo. The Evo component focuses on the representation and evolution of GRNs, with
 190 Devo serving as the evaluator of the GRN’s quality, measured by its ability to generate
 191 Pareto-optimal designs.

192 **3.3 Evo: Evolving GRNs**

193 The Evo stage randomly initializes a population of solutions/genomes where each solu-
194 tion encodes an artificial GRN. Here the encoding differs depending on the type of GRN
195 representation. To evaluate the quality/fitness of each genome, the devo component is
196 initialized where the GRN controls the growth of a structure for d developmental steps. At
197 the conclusion of the devo process, the physical properties (e.g. volume and deflection)
198 of the updated structure are assigned as fitness to the genome. Based on the fitness
199 of GRNs, the selection, crossover, and mutation operators evolve GRNs for a predefined
200 number of generations within the evo-loop. In this work NNs and computer programs are
201 used as GRNs representation where different multi-objective evolutionary search tech-
202 niques evolve a Pareto front of GRNs that generates diverse designs. Section 4 discusses
203 in detail GRNs representation and the proposed hybrid evolutionary search algorithm.

204 **3.4 Devo: Growing Structure**

205 In the Devo component, a structure undergoes in a growth phase through various mecha-
206 nisms such as altering cross-sectional areas, relocating nodes, adding new nodes, or com-
207 binations thereof. Within the evo-devo framework, Evo provides an indirect encoding of
208 growth rules, while Devo carries out the actual growth process based on local environmen-
209 tal factors. Figure 3, right, shows the block diagram representation of the Devo component.
210 Here, first, a structure is generated using pre-defined structural properties, and a GRN is
211 generated by decoding the genome. In artificial evolution, the introduction of a develop-
212 ment step allows a structure to be updated through different growth mechanisms. In the
213 literature, the principles of Devo have been demonstrated on simple engineering design
214 problems such as brackets (Price et al., 2022) and trusses (Hickinbotham et al., 2022).

Algorithm 1: Developmental Phase

Input : GRN, d, \mathcal{S}

```
1 for  $i$  in  $d$  do
2    $cells_p = FEA(\mathcal{S})$ 
3    $cells_g = getGeometry(\mathcal{S})$ 
4    $cells_{norm} = normState(cells_p, cells_g)$ 
5    $\delta = []$ 
6   for  $cell$  in  $cells_{norm}$  do
7      $\delta_{cell} = GRN(cell)$ 
8      $\delta.append(\delta_{cell})$ 
9   end
10   $\mathcal{S} = GrowStructure(\mathcal{S}, \delta)$ 
11 end
```

215 **3.4.1 Growth Phase**

216 In Devo, after the initial structure/seedling is generated, for example as shown in Fig-
217 ure 2(b), the structure grows for a predefined number of developmental steps where the
218 growth is regulated by a GRN. Algorithm 1 presents the developmental algorithm that takes
219 a genome from Evo, initialized structure (\mathcal{S}), and the maximum number of developmental
220 cycles (d).

221 During each step, local physical states of the structure are recorded such as member
222 forces, stress, and strain energy ($cells_p$) using an open-source finite element analysis (FEA)
223 tool, CalculiX (Dhondt, 2017) and node locations from structure geometry ($cells_g$). As dis-
224 cussed earlier, artificial cells are the basic building block of a structure, thus physical
225 properties of the structure are divided into a number of cells. The cell properties are nor-
226 malized in each developmental step. Each cell's state information, iteratively, is then fed
227 to the GRN that generates delta (δ). Depending on the δ generated by GRNs, the structure
228 grows by changing the cross-section area of members in each cell, by moving nodes, or by
229 adding new nodes in cells. At the end of d cycles, the fitness of the final modified structure
230 is computed and assigned to the genome.

231 In the field of engineering design, these growth mechanisms are referred to as size, shape,

232 and topology optimization (Dhondt, 2017) respectively. Although there are several other
 233 challenges involved in growing a seedling/structure such as the representation of the struc-
 234 ture, utilization of the global structure state information, and more, the scope of this study
 235 is limited to the search for a controller or regulator using evolutionary search algorithms.

236 **3.4.2 Fitness and Constraints Formulation**

237 At the end of the developmental steps, the fitness of the modified structure is calculated
 238 and assigned to the GRN. The experiments outlined in this paper aim to minimize the to-
 239 tal volume and minimize the maximum deflection recorded at nodes. Volume is selected
 240 alongside deflection as this represents a stiffness to weight optimization problem which
 241 is commonplace in engineering design especially in structural design. When evolving de-
 242 signs, several constraints must be taken into consideration. These constraints are volume
 243 and max deflection to restrict the evolved structure from being too thin to manufacture or
 244 too heavy.

$$\begin{aligned}
 \min \quad & (f_1, f_2) = \frac{\sum_{i=0}^m A_m L_m}{V_{d0}}, \frac{\max[nd_0, \dots, nd_n]}{MD_0} \\
 \text{s.t.} \quad & C1 : 1 - f_1 \leq 0 \\
 & C2 : 1 - f_2 \leq 0
 \end{aligned} \tag{1}$$

245 Equation 1 shows the two objectives (f_1, f_2) subjected to two inequality constraints $(C1, C2)$.
 246 Assuming the structure has m members where A_i and L_i are the cross-sectional area and
 247 length of the i^{th} member respectively. Assuming there are n nodes in the structure, each
 248 node's deflection (nd) is computed using finite element analysis, and the maximum is taken
 249 as the objective. Depending on material type, loading conditions, and size of the structure,
 250 the objective values can have different ranges. Thus, objectives are normalized by dividing
 251 V_{d0} and MD_{d0} which are the volume and max deflection of the initial structure/seedling
 252 respectively.

253 **4 GRN Representations and Evolution**

254 Gene regulatory networks are complex and highly non-linear functions, thus GRNs are rep-
255 resented by non-linear models. In this work, neural network and symbolic programs are
256 used as proxies for GRNs where multi-objective NEAT and CGP evolve Pareto optimal so-
257 lutions.

258 **4.1 NEAT and CGP based GRN representations**

259 In the literature, both NEAT and CGP have been used to evolve complex non-linear sys-
260 tems. NEAT evolves neural network architecture where during the evolutionary phase, it
261 randomly adds or removes nodes to the networks, and due to this, the non-linear response
262 of the networks can change. These random changes often result in a reduction in fitness,
263 and so to keep diverse NN architectures in the population speciation was introduced in
264 NEAT for single-objective problems (Stanley & Miikkulainen, 2002). However, when NEAT
265 is modified with NSGA-II to evolve Pareto fronts of neural network architectures, the con-
266 cept of speciation is not used – instead, Pareto ranking and the crowding distance-based
267 matrix maintain the diversity in the population. This diversity is purely based on the fit-
268 ness values, and not on the network topologies (Schrum & Miikkulainen, 2008). Deeper
269 inspection reveals that when Pareto ranking is combined with standard NEAT, the evolved
270 networks are not complex (i.e. evolved networks have fewer hidden nodes), and thus are
271 not suitable for complex non-linear problems. This is one of the drawbacks of the NEAT
272 algorithm in a multi-objective setting.

273 Similar to NEAT, genetic programming has also been employed for searching non-linear
274 functions. However, when evolving functions or programs for intricate problems using stan-
275 dard GP, bloating becomes a significant concern, leading to uncontrolled growth during
276 evolutionary searches. To mitigate this issue, cartesian genetic programming was intro-
277 duced, representing a type of GP where the tree depth is predefined. When utilizing CGP,

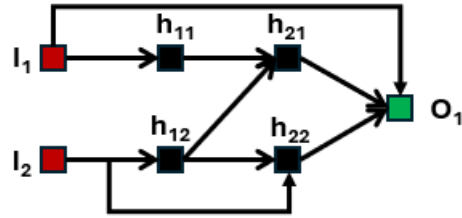


Figure 4: An example of a two-input and one-output neural network encoding in CGP style genotype representation where a grid 2×2 is defined as hidden layers.

278 users are required to define the number of rows and columns based on the number of
 279 inputs, outputs, and desired tree depth (Miller & Harding, 2008). In this setting, nodes at
 280 lower depths can receive inputs from any higher tree nodes, offering flexibility in generat-
 281 ing various functions. However, CGP does not scale well with complex problems. (O’Neill
 282 et al., 2010).

283 For multi objective problems, neural networks can scale well as problem complexity in-
 284 creases, but NEAT algorithm is unable to evolve complex networks, whereas CGP provides
 285 flexible tree structure representation but does not scale with problem complexity. This
 286 paper combines the useful features from NEAT and CGP, and presents a new algorithm as
 287 described in the next subsection. Figure 4 shows a neural network representation using
 288 this hybrid approach which has two inputs (I_1, I_2), one output node (O_1), and two hidden
 289 layers with two nodes in each layer. The output node can receive inputs from the input
 290 nodes or any other hidden layer nodes, and hidden layer nodes can receive inputs from
 291 any node in previous layers similar to CGP.

292 4.2 Real-value encoded NEAT (RNEAT)

293 To evolve multi-objective neural network topologies in a controlled manner, a hybrid repre-
 294 sentation based on NEAT and CGP is presented where connection, weights, and biases can
 295 be tuned using existing evolutionary search algorithms. The primary motivation of RNEAT
 296 is to evolve neural networks of different architectures for multi-objective problems.

Algorithm 2: RNEAT Algorithm

Input : $popSize, maxItr, numH, numN, \mathcal{S}$

```
1  $numInput, numOutput = Size(\mathcal{S})$ 
2  $C_l = Chromosome\ Size(numInput, numOutput, numH, numN)$ 
3  $P_0 = Initialize\ Population(popSize, C_l)$ 
4  $Evaluation(P_0, \mathcal{S}, numInput, numOutput, numH, numN)$ 
5 for  $t$  in  $maxItr$  do
6    $P_c = Reproduction(P_t)$ 
7    $Evaluation(P_c, \mathcal{S}, numInput, numOutput, numH, numN)$ 
8    $Fronts = Ranking(P_t, P_c)$ 
9    $cwd = Crowding\ Distance(Fronts)$ 
10   $P_{t+1} \leftarrow NextGenIndividuals(Fronts, cwd)$ 
11 end
12 return  $Best\ Front$ 
```

Algorithm 3: Evaluation

Input : $genomes, \mathcal{S}, numInput, numOutput, numH, numN$

```
1  $GRN = Network\ Architecture(numInput, numOutput, numH, numN)$ 
2 for  $g$  in  $genomes$  do
3   /** Decode neural network architecture from the genome */
4   for  $node$  in  $GRN.nodes()$  do
5      $node.weights = weights(g, node)$ 
6      $node.connection = Connections(g, node)$ 
7   end
8   /** Growth in Developmental Phase */
9    $\mathcal{S} = Developmental\ Phase(GRN, d, \mathcal{S})$ 
10   $g_f, g_c = computePerformance(\mathcal{S})$  (using eq.1)
11 end
```

297 Algorithm 2 shows the proposed RNEAT algorithm that takes population size (popSize),
298 maximum number of generations (maxItr), number of maximum hidden layers (numH),
299 maximum number of nodes in a hidden layer (numN), and initial seedling/structure (S) as
300 inputs, and returns the best evolved Pareto solutions. To generate the initial population,
301 first, the number of inputs and outputs are computed using the initial seedling/structure,
302 and then the length of the chromosome is calculated. The length depends on the type
303 of experiment being performed. Table 1 shows the number of inputs and outputs for dif-
304 ferent experimental setup. Figure 5 shows the genome encoding of the network shown

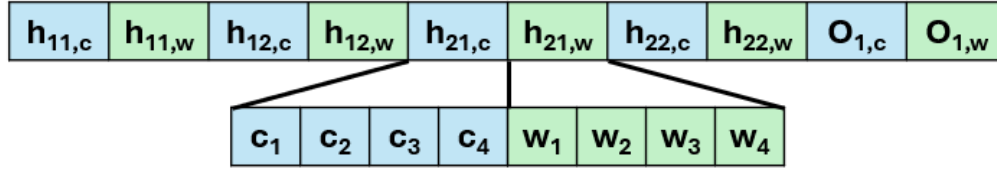


Figure 5: Genome encoding of the example network as shown in Figure 4 where each hidden and output node has two types of genes: connection genes and weight genes. The connection genes (c) are represented by blue color and weight genes (w) by green color. Hidden and output nodes can take inputs from all previous nodes. For example, hidden node h_{21} has four connection (c_1, c_2, c_3, c_4) and four weight (w_1, w_2, w_3, w_4) genes.

305 in Figure 4. Note that here each node (hidden and output) has two different types of
 306 genes: connections genes and weight genes. For example, node h_{21} can take inputs from
 307 four nodes (I_1, I_2, h_{11}, h_{12}) and thus has four connection and weight genes which can be
 308 evolved in multi objective setting. Once the initial population is generated Algorithm 3
 309 evaluates each individual's fitness. The rest of Algorithm 2 implements tournament selec-
 310 tion, simulated binary crossover, polynomial mutation, and Pareto ranking operators, as in
 311 standard NSGA-II.

312 Algorithm 3 takes genomes, initial structure, number of inputs, outputs, hidden layers, num-
 313 ber of nodes in a hidden layer, and computes the fitness of each genome. Here a genome
 314 encodes the network architecture and weights between nodes. Note that similar to CGP,
 315 the dimension of the network is pre-defined whereas a real-value encoded chromosome
 316 defines the connectivity and associated weights. Line 1 of the Algorithm 3 creates a fixed
 317 size network (which is termed as GRN), and then connections and weights are decoded
 318 from each genome. The network's architecture follows feed-forward information propa-
 319 gation, enabling nodes from previous layers to connect to the current layer nodes. This
 320 decoded neural network is the gene regulatory network that controls the growth of the ini-
 321 tial structure in d developmental steps as presented in Algorithm 1. The performance of
 322 the full grown structure is computed using eq. 1.

323 Since RNEAT draws inspiration from NEAT and CGP, it shares some common features, but

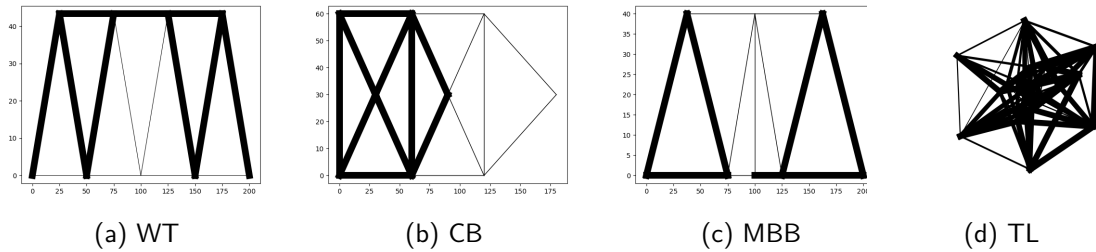


Figure 6: Initial seedling of (1) Warren truss with seven cells, (b) cantilever with nine cells, (c) MBB with six cells, and (d) 3D trabecular lattice. These initial seedlings are designed to be sub-optimal, by having different cross-sectional areas of members, and will be modified by GRNs to improve the quality in terms of volume and maximum deflection.

324 also has some differences. For example, NEAT relies on complexification to add hidden
 325 layer nodes and uses speciation to preserve the diversity in the population, whereas RNEAT
 326 defines a fixed architecture similar to CGP. NEAT focuses more on topological evolution
 327 through crossover and mutation, whereas weights and biases are only mutated. In contrast,
 328 when using RNEAT, both topology, and weights get a fair chance of being searched through
 329 crossover and mutation.

330 5 Experimental Results and Discussion

331 This section presents and analyzes experimental results pertaining to studies formulated
 332 as size, shape, and topology optimization problems. Note that, the growth of a structure
 333 can occur in three ways: by altering the cross-sectional area of members \mathcal{G}_1 , by relocating
 334 nodes \mathcal{G}_2 , or by adding nodes \mathcal{G}_3 . The objective is to evolve a gene regulatory network
 335 that utilizes local structural state information to govern the growth of the structure, with
 336 the aim of minimizing both volume and maximum deflection. To facilitate a comparative
 337 analysis, experiments on four distinct problems were conducted, and employed three dif-
 338 ferent algorithms¹ RNEAT, NEAT, and CGP. The results obtained from these experiments
 339 are examined and discussed.

¹We also compared CPPN but results of NEAT and CPPN were almost the same.

Table 1: Problem specifications: structures support and load points, number of inputs, and outputs under different types of growth/update mechanisms.

	Problems			
	WT	CB	MBB	3L
Support Points	(0, 0, 0) (200, 0, 0)	(0, 0, 0) (0, 60, 0)	(0, 0, 0) (200, 0, 0)	(0, 0, 0) (0, 0, 20) (0, 20, 0) (0, 20, 20)
Load Points	(100, 50, 0)	(180, 30, 0)	(100, 40, 0)	(20, 0, 0) (20, 0, 20) (20, 20, 0) (20, 20, 20)
\mathcal{G}_1 : In \mathcal{G}_1 : Out		3 m_{se} , 3 m_{cs} 3 δ_{cs}		6 m_{se} , 6 m_{cs} 6 δ_{cs}
$\mathcal{G}_1, \mathcal{G}_2$: In $\mathcal{G}_1, \mathcal{G}_2$: Out		3 m_{se} , 3 m_{cs} , 9 n_l 3 δ_{cs} , 9 δ_n		6 m_{se} , 6 m_{cs} , 12 n_l 6 δ_{cs} , 12 δ_n
$\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_3$: In $\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_3$: Out		3 m_{se} , 3 m_{cs} , 9 n_l 3 δ_{cs} , 9 δ_n , 1 δ_{cd}		6 m_{se} , 6 m_{cs} , 12 n_l 6 δ_{cs} , 12 δ_n , 1 δ_{cd}

340 This paper encompasses a total of 11 diverse experiments from three 2D problems and
341 one 3D problem, wherein the performance of the three algorithms is compared. In the
342 first four experimental setups, only one type of growth mechanism, \mathcal{G}_1 was used to update
343 the structure design, in the next four experiments two types of growth rules ($\mathcal{G}_1, \mathcal{G}_2$) were
344 applied together, and in the last three experiments, all three growth rules were applied
345 concurrently. Results on these diverse problems show the generalizability and the scala-
346 bility of the proposed approach and the effectiveness of the proposed RNEAT in evolving
347 Pareto optimal/near-optimal solutions/GRNs.

348 5.1 Experimental Set-up

349 The experiments encompassed the evaluation of GRNs on a Warren Truss (WT), a Cantilever
350 Beam (CB), Messerschmitt-Bolkow-Blohm (MBB) beam structures in a 2D space, as well as
351 a Trabecular Lattice (TL) in a 3D space as shown in Figure 6. In literature, these sample
352 structures are commonly used to test the effectiveness of new algorithms and approaches.
353 Note that structures are made of cells, where the WT has seven cells, CB nine cells, and MBB

354 has six cells. Experiments were conducted considering different types and combinations
355 of growth mechanisms such as \mathcal{G}_1 , \mathcal{G}_1 and \mathcal{G}_2 , and \mathcal{G}_1 , \mathcal{G}_2 , and \mathcal{G}_3 . Table 1 shows support and
356 load locations for these problems, and local structure state information as input to GRN,
357 in addition to output deltas suggested by the GRNs. In each experiment, the loads act in
358 the direction of the negative y-axis.

359 These experiments aimed to evolve the Pareto front of solutions using different parame-
360 ters tailored to the complexity of each problem, allowing for meaningful comparisons. To
361 assess and compare the Pareto fronts obtained through various algorithms, the hypervol-
362 ume (HV) (Guerreiro et al., 2021) performance indicator was employed. The selection of HV
363 was based on its Pareto-agnostic nature, meaning it does not rely on a true Pareto front,
364 thus enabling effective comparisons across different algorithms.

365 **5.2 Evolving GRNs for size optimization**

366 In size optimization, the CS area of members is modified to minimize material usage. Note
367 that, a structure is made up of cells, and a GRN takes input from each cell and generates
368 deltas to update the state of each cell iteratively. In this case, the gene regulatory network
369 takes into account the strain energy (m_{se}) and the CS area (m_{cs}) of each member of a
370 cell as input and provides a delta change in the CS area (δ_{cs}) for a predefined number
371 of developmental steps. For 2D size optimization problems, a GRN takes six inputs and
372 provides three outputs as shown in Table 1 while \mathcal{G}_1 type of growth mechanism happens
373 whereas, for the 3D size optimization problem, a GRN has 12 inputs and six outputs.

374 These initial structures, shown in Figure 6 are suboptimal designs, and the aim here is to
375 evolve a Pareto front of GRNs that can generate a diverse set of optimal or near-optimal
376 structures. Figure 7(a) shows the Pareto fronts obtained using the three algorithms on
377 the WT problem over 10 runs where red dots are RNEAT evolved solutions, blue and aqua
378 represents NEAT and CGP solutions respectively. Here a solution represents a GRN (ei-

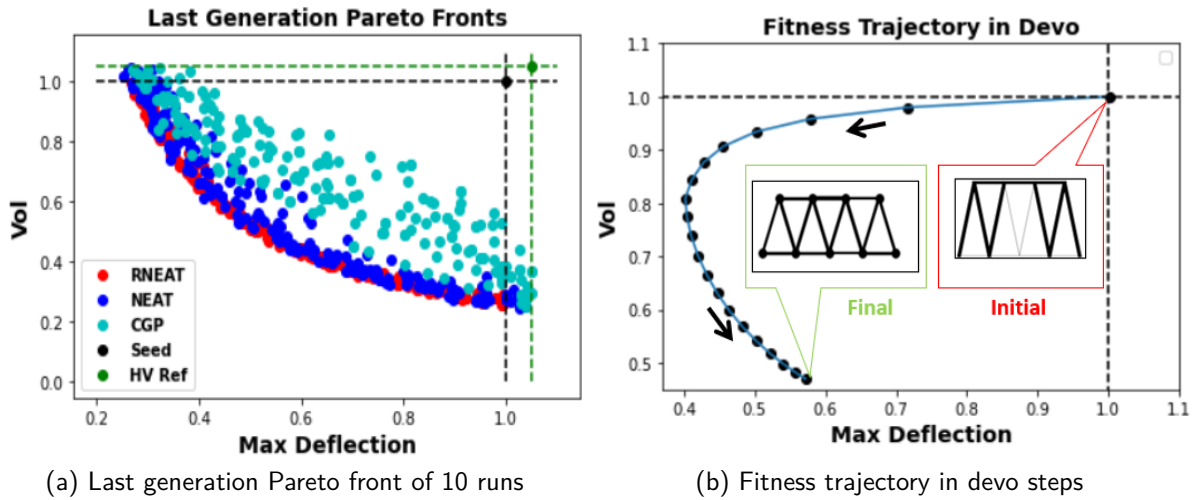


Figure 7: (a) the Pareto fronts obtained using the three algorithms in 10 runs on the WT problem, and (b) an example of fitness movement in different developmental steps where the last developmental step fitness is recorded as the fitness of the genome.

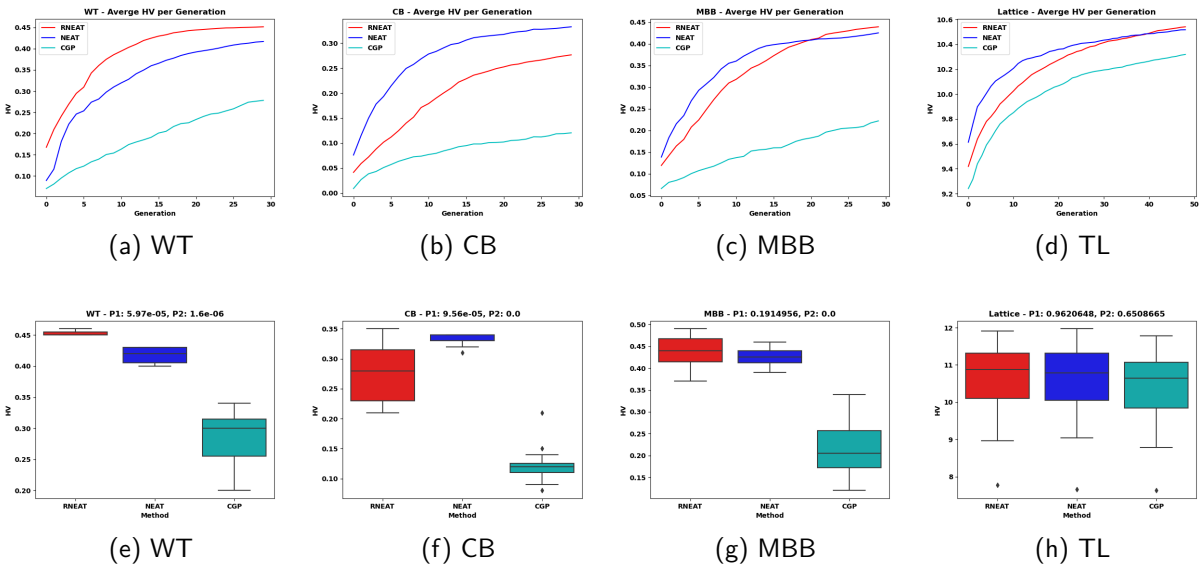


Figure 8: Size optimization: Figures (a), (b), (c), and (d) compare the average value HV per generation obtained using different algorithms over 10 runs, and (e), (f), (g), and (h) compare the distribution of HV from the last generation over 10 runs.

379 ther a NN or a computer program). The black dot is the initial structure fitness and the
380 green dot is the reference point to compute the hyper-volume performance indicator. The
381 initial structure objectives are (1, 1) because each objective is normalized where the initial
382 structure objectives are taken as a reference.

383 Figure 7(a) shows that evolved GRNs were able to grow the structure such that both volume
384 and max deflection reduces. Note that a GRN grows the structure for d developmental steps
385 and then at the end the fitness of the final modified structure is computed and assigned
386 as the fitness to the GRN. Figure 7(b) shows the fitness trajectory in different devo steps
387 indicating that GRN, chosen from a Pareto front, learned to minimize both objectives in the
388 initial few steps and then a compromise has been made between the two objectives. This
389 behavior is expected because reducing the volume often tends to increase the deflection at
390 nodes. Incorporating each devo step fitness into final fitness might be helpful in evolving
391 even better quality Pareto solutions, but this is part of the future endeavors of this work.
392 Similar experiments were conducted on the other three problems: CB, MBB, and TL.

393 Experimental results are presented in Figure 8, where subfigures (a), (b), (c) and (d) com-
394 pare the average hypervolume obtained in each generation over 10 runs using the three
395 algorithms. Subfigures (e), (f), (g), and (h) compare the distribution of HV values of the last
396 generation Pareto fronts. It is important to note that a higher HV value indicates a better
397 quality Pareto front (Guerreiro et al., 2021).

398 From the complexity perspective, the size optimization problem is the least complex com-
399 pared to other experiments conducted in this paper. Figures 8(a) and (c) demonstrate that
400 RNEAT achieves higher HV values compared to the other two algorithms on the WT and
401 MBB problems, indicating superior performance. However, on CB problem, RNEAT shows
402 inferior results compared to NEAT, but better than CGP. The distribution figures, Figure 8
403 (e, f, g, and h), also reveal that RNEAT's performance is better than NEAT on WT and MBB,
404 comparable on TL, and better than CGP on all four problems.

Table 2: Comparing the best, median, and worst values of HV obtained from the last generation Pareto front on four problems over 10 runs, for size optimization problem

	RNEAT	NEAT	CGP	RNEAT	NEAT	CGP	RNEAT	NEAT	CGP	RNEAT	NEAT	CGP
	WT			CB			MBB			TL		
Best	0.46	0.43	0.34	0.35	0.34	0.21	0.49	0.46	0.34	11.91	11.97	11.78
Median	0.45	0.42	0.3	0.28	0.33	0.12	0.44	0.425	0.20	10.87	10.795	10.64
Worst	0.45	0.4	0.2	0.21	0.31	0.08	0.37	0.39	0.12	7.78	7.66	7.63

405 Statistical analysis has been conducted from the data obtained using three methods and
 406 is indicated in Figures 8(e, f, g, and h) where P1 refers to the p-value calculated using the
 407 distribution of HV obtained using RNEAT and NEAT, and P2 when the p-value is computed
 408 using the HV distribution obtained using RNEAT and CGP. Experimental results show that
 409 RNEAT is statistically significantly better than NEAT on WT problem with a p-value of less
 410 than 0.05, and better than CGP on WT, CB, and MBB problems again with p-values less
 411 than 0.05. Table 2 shows the best, worst, and median HVs from the last generation Pareto
 412 front over 10 runs indicating that on three out of four problems, the median performance
 413 of RNEAT is better than others.

414 Note that the three 2D problems have different search space sizes where WT, CB, and
 415 MBB have 15, 17, and 13 members respectively. As the number of members in the struc-
 416 ture increases, the complexity also increases, and it would be difficult for standard GAs
 417 to manage the problem's complexity. However, the proposed Evo-Devo approach grows
 418 cells of the structure iteratively and is thus applicable to these complex problems. This
 419 shows the scalability property of the proposed approach. These results provide encour-
 420 aging evidence that the Evo-Devo-based approach can effectively evolve a diverse set of
 421 Pareto-optimal GRNs that have learned how to control the growth of initial structures, re-
 422 sulting in simultaneous minimization of both volume and deflection.

423 5.3 Evolving GRNs for Topology Optimization

424 In a similar manner to size, the topology of the structure can be optimized with minimal
 425 modifications to GRN's inputs and outputs. In the case of topology optimization, the GRN

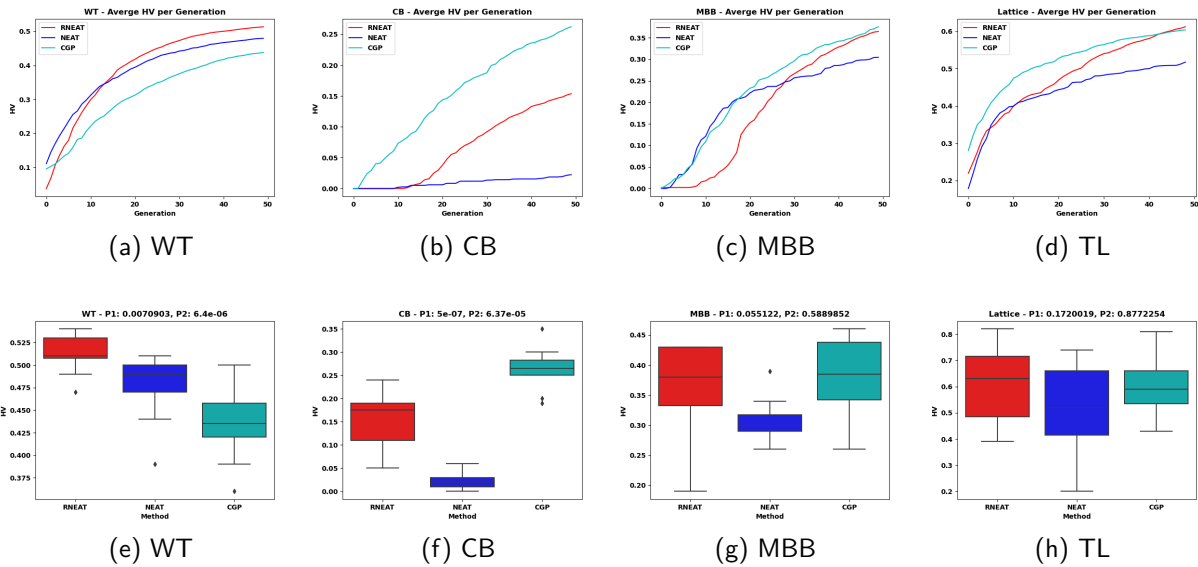


Figure 9: Results for topology optimization. The figure compares the average HV and distribution of HV over 10 runs when a GRN control both node movement and edge/member thickness of structures.

426 governs both member CS area and node movement. In this scenario, the GRN incorporates
 427 node location (n_l) information, along with member thickness (m_{cs}) and strain energy (m_{se}),
 428 to generate deltas for both node location (δ_n) and member thickness (δ_{cs}), thus two types
 429 of growth mechanism happen together, \mathcal{G}_1 , and \mathcal{G}_2 . The number of inputs and outputs for
 430 topology optimization is 15 and 12 respectively for 2D problems, and 24 and 18 for 3D
 431 problems as shown in Table 1.

432 As different types of growth mechanisms are combined to update/grow a given structure,
 433 the complexity of the problem increases, however, the same GRN representations can
 434 be used to evolve diverse designs. This shows the generalizability characteristics of the
 435 proposed Evo-Devo approach.

436 Similar experiments have been conducted and comparative results are presented. The
 437 average HV values obtained in each generation using different techniques, along with the
 438 distribution of HVs from the last generation Pareto fronts over 10 runs, are illustrated in
 439 Figure 9. Figure 9 (a) shows that on WT, RNEAT's HV is higher than the other two, and

Table 3: Comparing the best, median, and worst values of HV obtained from the last generation Pareto front on four problems over 10 runs, for topology optimization problem

	RNEAT	NEAT	CGP	RNEAT	NEAT	CGP	RNEAT	NEAT	CGP	RNEAT	NEAT	CGP
	WT			CB			MBB			TL		
Best	0.54	0.51	0.5	0.24	0.06	0.35	0.43	0.39	0.46	0.82	0.74	0.81
Median	0.51	0.49	0.435	0.175	0.02	0.265	0.38	0.3	0.385	0.63	0.52	0.59
Worst	0.47	0.39	0.36	0.05	0.0	0.19	0.19	0.26	0.26	0.39	0.2	0.43

440 Figure 9 (e) shows the distribution of last generation HV values over 10 runs. The figure
 441 also shows p-values obtained using the distribution of RNEAT and NEAT HV values (P1), and
 442 RNEAT and CGP HV values (P2). Both P1 and P2 are less than 0.05 indicating statistical
 443 significance, on the WT problem. Table 3 compares the best, worst, and median HV values
 444 from the last generation Pareto front in 10 runs. From this table, it is clear that on the WT,
 445 RNEAT's best, and median HVs are better (higher) than the others.

446 A similar performance comparison is made on CB, MBB, and TL. On CB and MBB, CGP is
 447 the best-performing algorithm. On TL, the performance of RNEAT is comparable to CGP
 448 and better than NEAT. However, Table 3 shows that the best and the median HV values
 449 obtained using RNEAT are higher than both NEAT and CGP. Statistical analysis shows that
 450 RNEAT is better than NEAT on WT, CB, and MBB problems.

451 5.4 GRNs for Size, Shape, and Topology Optimization

452 In previous experiments, the initial structure/seedlings had more than one cell, as shown
 453 in Figure 6, and in the growth phase, only the member's thickness and node locations were
 454 changed. In this subsection, experiments are devised to closely mimic the Evo-devo-based
 455 concept.

456 In the proposed Evo-devo-based approach, the integration of size, shape, and topology
 457 allows for a unified problem setting, wherein GRNs govern the cross-sectional area of
 458 members, the relocation of nodes, and even the addition of new nodes to modify the struc-
 459 ture's geometry and topology. Among the various experimental setups, this particular con-

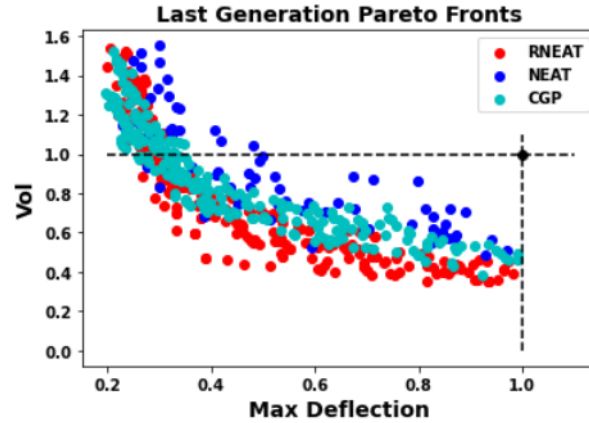


Figure 10: Figure shows the Pareto fronts obtained using the three algorithms in 10 runs on the TL problem. The black dot on the top right show the fitness of the initial seedling/structure.

460 figuration is the most complex due to the significantly larger search space compared to
 461 optimizing for size or topology alone. Here all three types of growth mechanisms (\mathcal{G}_1 , \mathcal{G}_2 ,
 462 and \mathcal{G}_3) can happen simultaneously at every growth step.

463 At the start of the simulation, initial seedlings are generated with minimal cells (minimum
 464 of one). In each developmental step, the GRN takes m_{cs} , m_{se} , and n_l as input and deter-
 465 mines δ_{cs} , and δ_n , and whether to add a node by dividing a cell (δ_{cd}). If the cell division
 466 takes place, then new members are added to the structure, and this results in volume in-
 467 crement. Due to fitness-based selection pressure, evolution prefers solutions/GRNs that
 468 decide not to divide cells (to keep the volume lower) which over the generations decreases
 469 the diversity in the population. Thus to evolve a diverse set of structures, the constraint
 470 limit on this experiment is relaxed. Note that Table 1 shows the design space boundary
 471 of three problems considered here. Throughout the simulation process, it is possible to
 472 adjust the initial design in a manner that leads to a deflection exceeding the prescribed
 473 space constraints. Such outcomes represent solutions that are deemed infeasible. When
 474 computing the HV of the rank zero Pareto front from each generation, obviously infeasible
 475 solutions were excluded, if there were any.

476 Experiments were conducted on two 2D problems: CB, MBB, and a 3D problem, TL, where

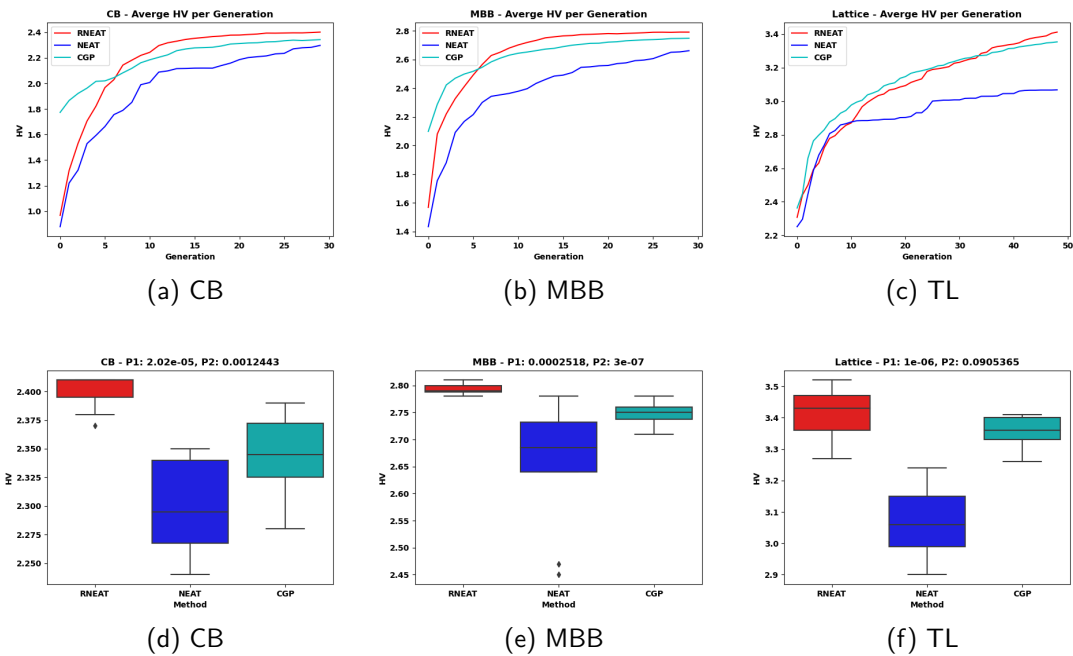


Figure 11: Results for Size, shape, and topology. Figure compares average HV and distribution of HV over 10 runs when a GRN control both node movement, edge thickness, and cell division.

477 the initial seedling for CB and MBB is a single triangular cell structure, and TL's initial
 478 seedling has more than one tetrahedral cell. Figure 10 shows the Pareto fronts obtained
 479 using the three algorithms over 10 runs on the 3D TL problem. The figure shows that RNEAT
 480 evolved Pareto solutions are better than the other two methods. In this set of experiments,
 481 cell division adds new nodes and edges/members to the structure resulting in the volume
 482 increment and whereas modifications in the CS area and node movement can lead to a
 483 reduction in deflection.

484 Comparative results are shown in Figure 11 where subfigures (a), (b), and (c) compares the
 485 average HV per generation over 10 runs. Figure 11 (d), (e), and (f) show the distribution
 486 of HVs from the last generation over 10 runs. Figure 11 shows that RNEAT is better than
 487 the other two on all three problems. Additionally, p-values (P1 and P2) are less than 0.05
 488 indicating that RNEAT performance is statistically significantly better than NEAT and CGP
 489 on CB, MBB, and TL.

Table 4: Comparing the best, median, and worst values of HV obtained from the last generation Pareto front on four problems over 10 runs, for combined optimization problem

		RNEAT	NEAT	CGP	RNEAT	NEAT	CGP	RNEAT	NEAT	CGP
		CB			MBB			TL		
HV	Best	2.41	2.35	2.39	2.81	2.78	2.78	3.52	3.24	3.41
	Median	2.41	2.295	2.34	2.79	2.685	2.75	3.43	3.06	3.36
	Worst	2.37	2.24	2.28	2.78	2.45	2.71	3.27	2.9	3.26

Table 5: Ranking three algorithms based on the quality of evolved Pareto fronts, the distribution of HV values, and statistical analysis

Problems and Types of Growth											
Methods	\mathcal{G}_1				$\mathcal{G}_1, \mathcal{G}_2$				$\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_3$		
	WT	CB	MBB	TL	WT	CB	MBB	TL	CB	MBB	TL
RNEAT	I	II	I	II	I	II	II	I	I	I	I
NEAT	II	I	II	I	II	III	III	II	III	III	II
CGP	III	III	III	III	III	I	I	III	II	II	III

490 Table 4 compares the best, worst, and median HV values from the last generation Pareto
491 front obtained using the three algorithms. For each problem and in each case, RNEAT’s HV
492 is higher than NEAT and CGP. These results (Figure 11 and Table 4), provide evidence that
493 as the complexity of the problems starts increasing, RNEAT performs better than NEAT and
494 CGP.

495 To summarize these findings, Table 5 provides rankings for the different algorithms on all
496 problems. This ranking is derived from results such as average HV per generation, the dis-
497 tribution of last-generation HV values, and statistical analysis under different experimen-
498 tal setups. Out of the total of 11 experiments, RNEAT outperformed the other algorithms
499 seven times, while NEAT achieved the best performance two times, and CGP emerged as
500 the top performer twice. These promising results serve as compelling evidence that first,
501 Evo-devo-based approaches can effectively generate Pareto fronts of GRNs, that gener-
502 ate designs, and second, RNEAT (a hybrid of NEAT and CGP) performs well against NEAT
503 and CGP, particularly for more complex design problems. The experimental outcomes
504 across these four problems demonstrated the generalizability and scalability of the pro-
505 posed method. Nonetheless, because RNEAT evolves non-linear neural networks as black

506 box models, delving into the behaviors of the evolved network will necessitate additional
507 experimental analysis.

508 **6 Conclusions and Future Work**

509 This paper introduces a framework based on evolutionary developmental biology (Evo-
510 Devo), in which the growth of structures is controlled by gene regulatory networks repre-
511 sented by neural networks and genetic programming. To evolve high-quality Pareto solu-
512 tions (GRNs), the paper presents a multi-objective neuro-evolutionary algorithm termed
513 RNEAT, which draws inspiration from NEAT and utilizes a CGP-style encoding of geno-
514 types. The performance of RNEAT is compared to NEAT, and CGP on various 2D and 3D
515 problems, considering different initialization and types of growth. A total of 11 different
516 experiments were conducted to assess the viability of the proposed Evo-Devo approach
517 for evolving structural designs and to compare the effectiveness of RNEAT with the other
518 two algorithms.

519 The results indicate that by considering diverse growth mechanisms and structural initial-
520 ization, evolved GRNs were able to enhance performance in terms of objective functions.
521 This provides compelling evidence that the proposed Evo-Devo approach can effectively
522 facilitate the growth of designs under different environmental conditions. Evaluating the
523 performance using the hypervolume as a performance indicator, RNEAT outperformed the
524 other algorithms on seven problems, while NEAT and CGP only performed better on two
525 problems, each. These results on the different problems under different types of growth
526 rules show that the proposed approach is generalizable and scalable.

527 The experimental outcomes demonstrate the efficacy of Evo-Devo when local information
528 is incorporated into a gene regulatory network. Further work will involve extending the pro-
529 posed approach to evolve even more complex engineering designs, incorporating global
530 structural information to determine local growth. Additionally, the proposed approach will

531 be applied to problems characterized by non-linear loading conditions, where the search
532 space is highly constrained.

533 **Acknowledgement**

534 This work was supported by the Engineering and Physical Sciences Research Council (EP-
535 SRC) Programme under Grant EP/V007335/1.

536 **References**

- 537 Albarracín-Molina, D. D., Moya, J. C., & Vico, F. J. (2016). An evo-devo system for algo-
538 rithmic composition that actually works. *Proceedings of the 2016 on Genetic and*
539 *Evolutionary Computation Conference Companion*, 37–38.
- 540 Balamurugan, R., Ramakrishnan, C., & Singh, N. (2008). Performance evaluation of a two
541 stage adaptive genetic algorithm (tsaga) in structural topology optimization. *Ap-*
542 *plied Soft Computing*, 8(4), 1607–1624.
- 543 Bidlo, M., & Dobeš, M. (2020). Evolutionary development of growing generic sorting net-
544 works by means of rewriting systems. *IEEE Transactions on Evolutionary Computa-*
545 *tion*, 24(2), 232–244. <https://doi.org/10.1109/TEVC.2019.2918212>
- 546 Chi, H., Zhang, Y., Tang, T. L. E., Mirabella, L., Dalloro, L., Song, L., & Paulino, G. H. (2021).
547 Universal machine learning for topology optimization. *Computer Methods in Applied*
548 *Mechanics and Engineering*, 375, 112739.
- 549 Christensen, P. W., & Klarbring, A. (2008). *An introduction to structural optimization* (Vol. 153).
550 Springer Science; Business Media.
- 551 Cussat-Blanc, S., Harrington, K., & Banzhaf, W. (2019). Artificial gene regulatory networks—a
552 review. *Artificial Life*, 24(4), 296–328. https://doi.org/10.1162/artl_a_00267

553 Cussat-Blanc, S., Harrington, K., & Pollack, J. (2015). Gene regulatory network evolution
554 through augmenting topologies. *IEEE Transactions on Evolutionary Computation*,
555 19(6), 823–837. <https://doi.org/10.1109/TEVC.2015.2396199>

556 Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective ge-
557 netic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2), 182–
558 197.

559 Delgado, F. M., & Gómez-Vela, F. (2019). Computational methods for gene regulatory net-
560 works reconstruction and analysis: A review. *Artificial intelligence in medicine*, 95,
561 133–145.

562 Dhondt. (2017). Calculix crunchix user's manual version 2.12. *Munich, Germany*, accessed
563 Sept. http://www.dhondt.de/ccx%5C_2.19.pdf

564 Goldberg, D. E. (2002). *The design of innovation: Lessons from and for competent genetic*
565 *algorithms* (Vol. 1). Springer.

566 Guerreiro, A. P., Fonseca, C. M., & Paquete, L. (2021). The hypervolume indicator: Compu-
567 tational problems and algorithms. *ACM Computing Surveys (CSUR)*, 54(6), 1–42.

568 Hall, B. K. (2012). Evolutionary developmental biology (evo-devo): Past, present, and future.
569 *Evolution: Education and outreach*, 5(2), 184–193.

570 Hickinbotham, S., Dubey, R., Friel, I., Colligan, A., Price, M., & Tyrrell, A. (2022). Evolving
571 design modifiers. *2022 IEEE Symposium Series on Computational Intelligence*
572 *(SSCI)*, 1052–1058. <https://doi.org/10.1109/SSCI51031.2022.10022087>

573 Kaldis, P. (2016). Quo vadis cell growth and division?

574 Karlebach, G., & Shamir, R. (2008). Modelling and analysis of gene regulatory networks.
575 *Nature reviews Molecular cell biology*, 9(10), 770–780.

576 McCormack, J., & Gambardella, C. C. (2022). Growing and evolving 3-d prints. *IEEE Trans-*
577 *actions on Evolutionary Computation*, 26(1), 88–99. [https://doi.org/10.1109/TEVC.](https://doi.org/10.1109/TEVC.2021.3095156)
578 [2021.3095156](https://doi.org/10.1109/TEVC.2021.3095156)

579 Miller, J. F., & Harding, S. L. (2008). Cartesian genetic programming. *Proceedings of the*
580 *10th annual conference companion on Genetic and evolutionary computation*, 2701–
581 2726.

582 Navarro-Mateu, D., & Cocho-Bermejo, A. (2019). Evo-devo algorithms: Gene-regulation for
583 digital architecture. *Biomimetics*, 4(3), 58.

584 Navarro-Mateu, D., & Cocho-Bermejo, A. (2020). Evo-devo strategies for generative archi-
585 tecture: Colour-based patterns in polygon meshes. *Biomimetics*, 5(2), 23.

586 Olson, E. N. (2006). Gene regulatory networks in the evolution and development of the
587 heart. *Science*, 313(5795), 1922–1927.

588 O’Neill, M., Vanneschi, L., Gustafson, S., & Banzhaf, W. (2010). Open issues in genetic pro-
589 gramming. *Genetic Programming and Evolvable Machines*, 11, 339–363.

590 Osaba, E., Villar-Rodriguez, E., Del Ser, J., Nebro, A. J., Molina, D., LaTorre, A., Suganthan,
591 P. N., Coello, C. A. C., & Herrera, F. (2021). A tutorial on the design, experimentation
592 and application of metaheuristic algorithms to real-world optimization problems.
593 *Swarm and Evolutionary Computation*, 64, 100888.

594 Perez, R. E., & Behdinan, K. (2007). Particle swarm approach for structural design optimiza-
595 tion. *Computers & Structures*, 85(19-20), 1579–1588.

596 Price, M., Zhang, W., Friel, I., Robinson, T., McConnell, R., Nolan, D., Kilpatrick, P., Barbhuiya,
597 S., & Kyle, S. (2022). Generative design for additive manufacturing using a biological
598 development analogy. *Journal of Computational Design and Engineering*, 9(2), 463–
599 479.

600 Richards, D., Dunn, N., & Amos, M. (2012). An evo-devo approach to architectural design.
601 *Proceedings of the 14th annual conference on genetic and evolutionary computa-*
602 *tion*, 569–576.

603 Schlitt, T., & Brazma, A. (2007). Current approaches to gene regulatory network modelling.
604 *BMC bioinformatics*, 8(6), 1–22.

- 605 Schrum, J., & Miikkulainen, R. (2008). Constructing complex npc behavior via multi-objective
606 neuroevolution. *Proceedings of the AAAI Conference on Artificial Intelligence and*
607 *Interactive Digital Entertainment*, 4(1), 108–113.
- 608 Stanley, K. O., & Miikkulainen, R. (2002). Evolving neural networks through augmenting
609 topologies. *Evolutionary computation*, 10(2), 99–127.
- 610 Streichert, F., Planatscher, H., Spieth, C., Ulmer, H., & Zell, A. (2004). Comparing genetic
611 programming and evolution strategies on inferring gene regulatory networks. *Ge-*
612 *netic and Evolutionary Computation–GECCO 2004: Genetic and Evolutionary Com-*
613 *putation Conference, Seattle, WA, USA, June 26–30, 2004. Proceedings, Part I*, 471–
614 480.
- 615 Swain, M., Hunniford, T., Mandel, J., Palfreyman, N., & Dubitzky, W. (2005). Modeling gene-
616 regulatory networks using evolutionary algorithms and distributed computing. *CC-*
617 *Grid 2005. IEEE International Symposium on Cluster Computing and the Grid,*
618 *2005., 1*, 512–519.
- 619 Turner, A. J., & Miller, J. F. (2014). Cartesian genetic programming: Why no bloat? *Genetic*
620 *Programming: 17th European Conference, EuroGP 2014, Granada, Spain, April 23-*
621 *25, 2014, Revised Selected Papers 17*, 222–233.
- 622 van Willigen, W. H., Haasdijk, E., & Kester, L. J. (2013). Evolving intelligent vehicle control
623 using multi-objective neat. *2013 IEEE Symposium on Computational Intelligence*
624 *in Vehicles and Transportation Systems (CIVTS)*, 9–15.
- 625 Vujovic, V., Rosendo, A., Brodbeck, L., & Iida, F. (2017). Evolutionary developmental robotics:
626 Improving morphology and control of physical robots. *Artificial Life*, 23(2), 169–185.
627 https://doi.org/10.1162/ARTL_a_00228
- 628 Wang, S., Tai, K., & Wang, M. (2006). An enhanced genetic algorithm for structural topology
629 optimization. *International Journal for Numerical Methods in Engineering*, 65(1),
630 18–44.

- 631 Wu, R., Chao, F., Zhou, C., Huang, Y., Yang, L., Lin, C.-M., Chang, X., Shen, Q., & Shang, C.
632 (2022). A developmental evolutionary learning framework for robotic chinese stroke
633 writing. *IEEE Transactions on Cognitive and Developmental Systems*, 14(3), 1155–
634 1169. <https://doi.org/10.1109/TCDS.2021.3098229>
- 635 Xu, R., Venayagamoorthy, G. K., & Wunsch II, D. C. (2007). Modeling of gene regulatory
636 networks with hybrid differential evolution and particle swarm optimization. *Neural*
637 *Networks*, 20(8), 917–927.
- 638 Yildiz, A. R. (2013). Comparison of evolutionary-based optimization algorithms for struc-
639 tural design optimization. *Engineering applications of artificial intelligence*, 26(1),
640 327–333.
- 641 Zou, F., Chen, D., Liu, H., Cao, S., Ji, X., & Zhang, Y. (2022). A survey of fitness landscape
642 analysis for optimization. *Neurocomputing*, 503, 129–139.