Optimization of Multi-branch Truss-Z based on Evolution Strategy

Machi Zawidzki
Department of Architecture,
Massachusetts Institute of Technology
T: +1 617 253 0781, F: +1 627 253 9407, E: zawidzki@mit.edu

Abstract
This paper concerns multi-branch Truss-Z networks (MTZ). A possible scenario for creating a "multi-branch bridge" linking 6 terminals of pedestrian and cycling communication is presented. This process is formulated as a constrained minimization problem. A new, biology-inspired nomenclature for MTZ and encoding for MTZ are introduced. Several operations for MTZs are introduced and illustrated. The functionality of these operations is illustrated with transformation from a random MTZ to a "proper" 6-branch MTZ network. A population-based heuristic experiment is presented to demonstrate that the introduced operators allow us to create any desirable MTZ. A cost function for the considered scenario is introduced. The genetic operations are interpreted and visualized. A number of feasible MTZ layouts produced by an evolution strategy-based algorithm are presented. One of these layouts is used for creation of the spatial 6-terminal MTZ, which is also visualized.

Keywords: modular ramp system, multi-branch network, modular structure encoding, evolution strategy, discrete layout optimization.

1. Introduction

A stairway is the most common means of pedestrian vertical transportation used in the built environment. Elevators and escalators are relatively expensive to install and maintain, and their traffic flow capacity is much lower than that of stairs. Moreover, it is not always possible to install an elevator or escalator due to limited space. However, most people occasionally or temporarily cannot use stairs, as when riding a bicycle, pushing a baby stroller or carrying heavy luggage. For elders and people in wheelchairs, stairs form a permanent and impassable barrier. This is an important social issue, especially since the proportion of elderly people in society is higher than in the past, and some predict that this tendency will continue [1]. For a comprehensive review of the literature on elderly pedestrians, see [2].

2. The concept of Truss-Z

Truss-Z (TZ) is a modular skeletal system for creating free-form ramps and ramp networks among any number of terminals in space. The concept has been introduced in [3]. The motivation for TZ in the context of human mobility, and in particular – the mobility problems of elders is discussed in [4].
The underlying idea of this system is to create structurally sound provisional or permanent structures using the minimal number of types of modular elements. For further discussion on "Modularity vs. free-form" see [5].

2.1 The Modularity of Truss-Z

The TZ structures are composed of four variations of a single basic unit (R), as shown in Figure 1.

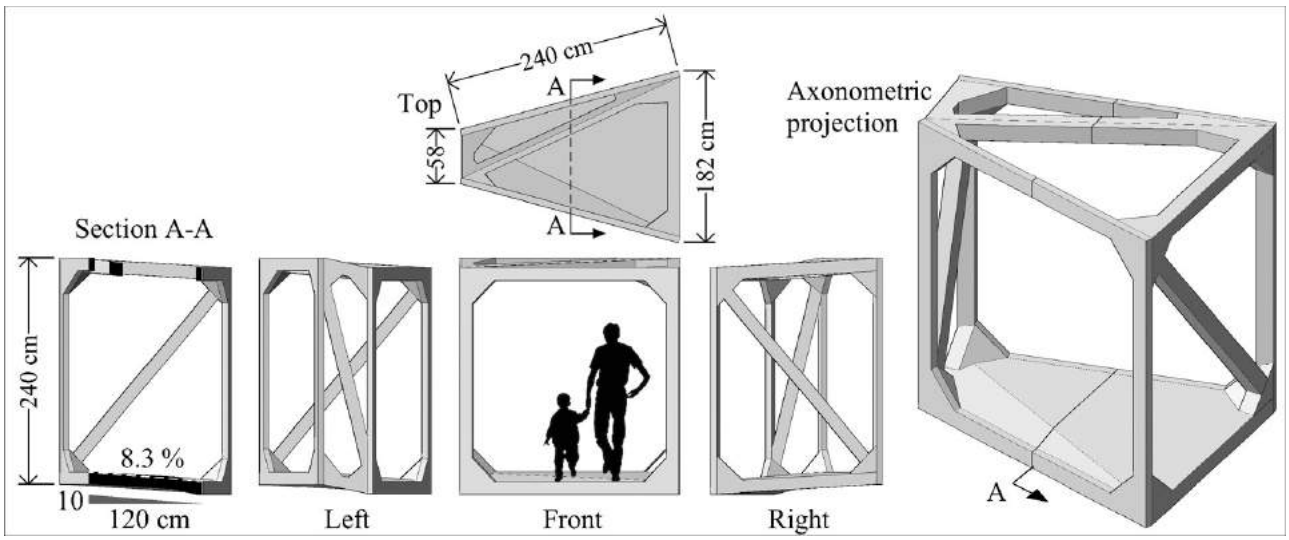

Figure 1: The section showing the slope followed by the planar and axonometric projections of the TZ basic unit (R).

Unit L is a mirror reflection of unit R. By rotation, they can be assembled in two additional ways (R_2 – rotated R and L_2 – rotated L), effectively giving four types of units. Some examples are shown in Figure 2.

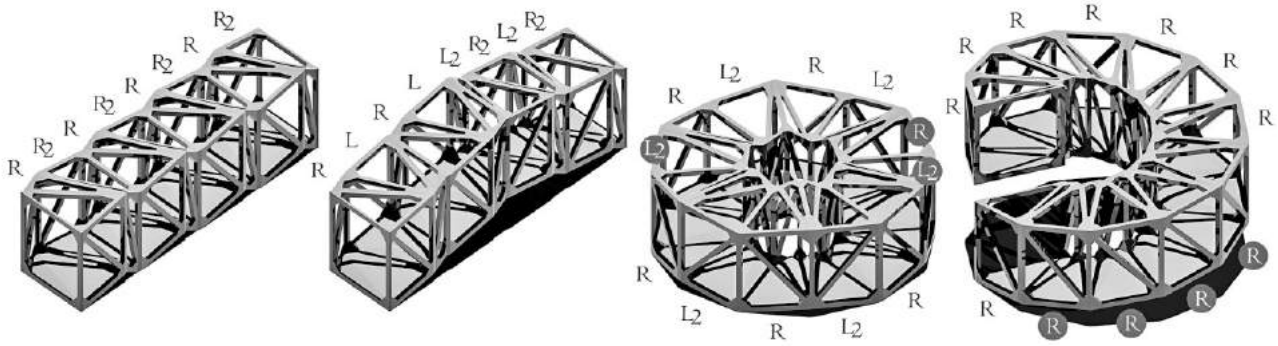

Figure 3 shows an example from the case study of retrofitting an existing concrete overpass system comprised of two bridges connected with three stairways, presented in [6].

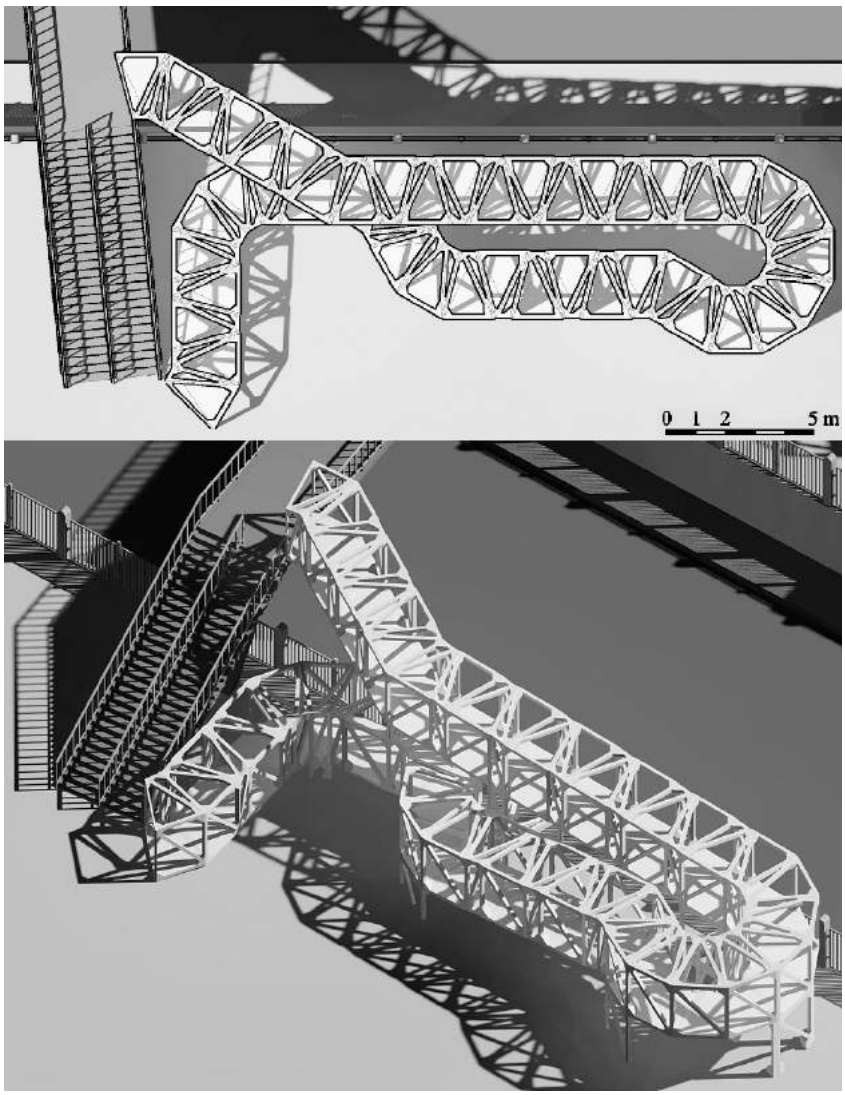

Figure 3: A visualization of a TZ to allow the use of wheelchairs in an existing overpass. The slope of TZ is 1:12 (8.3%). Top and bottom: plan and axonometric views, respectively. The existing stairway (on the left) is very steep and particularly long: 29 rises without an intermediate landing.

Structural rigidity of the TZ module has been demonstrated in [5], along with other topological properties such as nullity, degree of static indeterminacy (DSI = 0), etc. Due to the modularity of this system, it is natural to apply discrete optimization methods for creating TZ linkages and networks. Such structures can be optimized for various criteria: the minimal number of modules, the minimal number of changes in direction, and, in a case of multiple branches, the minimal network distance, etc. Various deterministic and meta-heuristic methods have been successfully implemented for single TZ paths. E.g: backtracking [7], evolution strategy [8], and evolutionary algorithm [9]. These methods produced usually good, but not ideal, solutions. A graph-theoretical exhaustive search method, which produces the best allowable, that is ideal solutions has been described in [6].

2.2 Physical models of TZ

Several case studies of retrofitting with TZ and reduced scale models have been built, as shown in Figures 4 and 5.

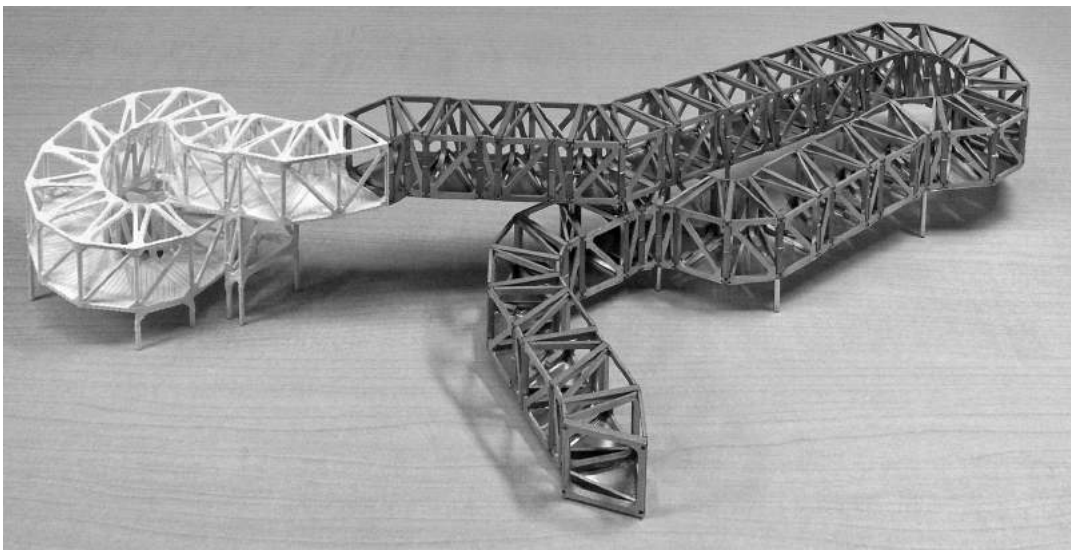

Figure 4: A scale model of a complex TZ overpass. The left part (white) is a 3D-print, the right part – a metal model made of flat pre-cut elements. The latter is the same TZ as visualized in Figure 3.

Although 3D-printing is a common "effortless" method for rapid prototyping, the assembly models are much closer to reality, as they use elements which to a certain extent correspond to the actual truss members, as shown in Figure 5.

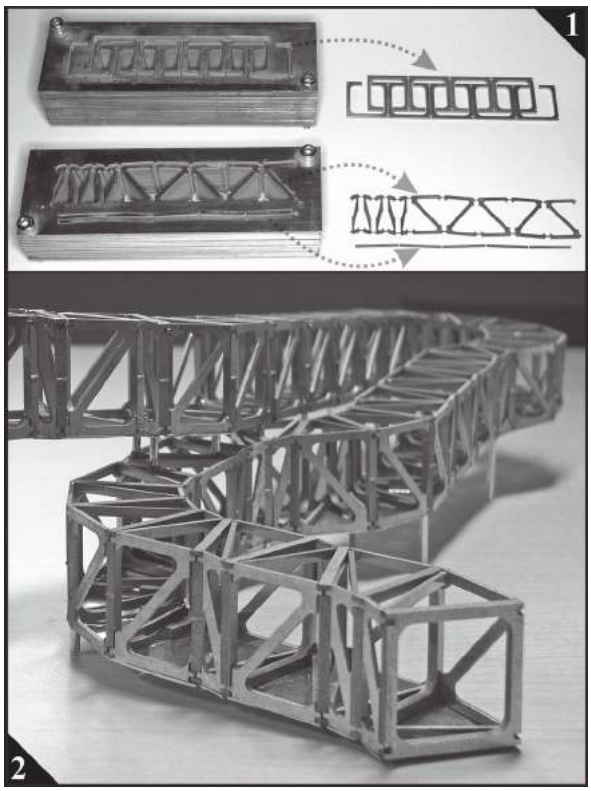

Figure 5: The physical, aluminum scale model. 1) "Mass-produced" aluminum members. 2) Detail showing a gentle ramp slope (approx. 8%) suitable for wheelchairs. TZ allows us to create "meandering" ramps, which is beneficial for safety as the length of straight sections can be reduced. The same TZ as presented in Figures 3 & 4.

Furthermore, the concept of foldable Truss-Z module and the proof-of-concept prototype scale model has been presented in [10]. However, a multi-branch TZ network (MTZ) is a qualitatively more difficult problem. This paper introduces new encoding and operators for efficient optimization of MTZ in realistic situations. Although TZ is a spatial system, for simplicity, in this paper the problem is reduced to the projection on the 2D plane, in other words, to the layout of MTZ. Thus modules R&L_2 and L&R_2 correspond to trapezoidal units 0 and 1, respectively.

3. Possible scenario for a "multi-branch bridge"

The following scenario illustrates a possible application of the concept of a multi-branch Truss-Z network (MTZ):
- There is a canal with walking and cycling paths on both sides.
- The canal is depressed and another, approximately perpendicular walking/cycling path is elevated.
- Create a layout of a network linking 6 elements of pedestrian and cycling communication (terminals).

The situation is shown in Figure 6.

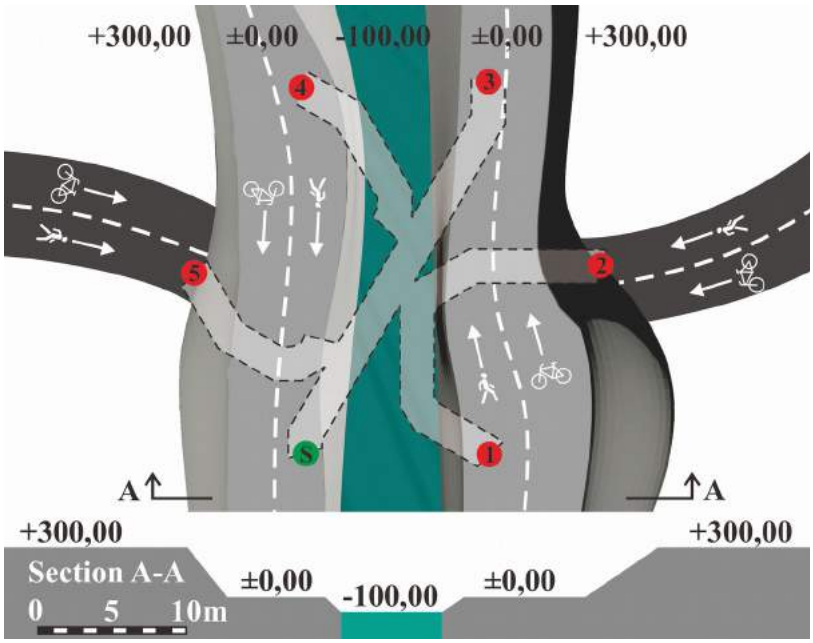

Figure 6: The situation for installing a MTZ. The terminals have been placed to match MTZ_6 (explained further in the text) for comparison. The outline of MTZ_6 connecting all terminals is shown with a dashed line.

The concept of MTZ has been considered in the past [3,5]. An example of a *manually* created layout of a 6-branch Truss-Z (MTZ_6) has been presented in [11]. However, the original encoding has not been suited for genetic operations, the defined genetic operations have not covered all possible transformations, and as a result, the population-based heuristics could not produce satisfactory results. This paper presents: the new encoding, a systematical study of the genetic operations (in particular - mutations), a new and well-converging algorithm based on evolution strategy, and a realistic case-study of retrofitting a traffic junction with a 6-branch MTZ network.

4. The new nomenclature and encoding

According to [12], in computer science, specifically in evolutionary algorithms, "*object forming possible solutions within the original problem context are referred to as phenotypes, their encoding,*

are called genotypes". Usually genotypes are in the form of a bit string, array, tree, list or matrix. The genotype encoding for MTZ introduced in [11] had the form of a nested list. Such an encoding although straightforward, is not very concise or suitable for genetic operations. Figure 7 illustrates the new, nature-inspired naming for the MTZ phenotype components and the new genotype encoding in the form of a relatively simple list of lists.

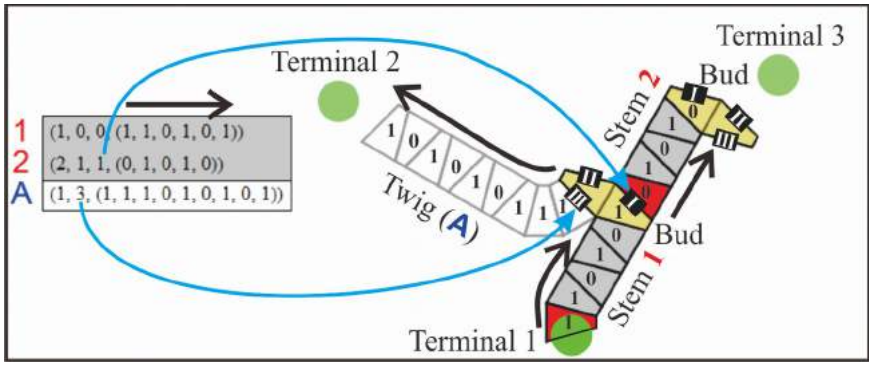

Figure 7. The nomenclature for MTZ. On the left: the genotype tabulated for clarity. On the right: the phenotype. Black arrows illustrate the direction of the sequence of trapezoidal units. The initial units of each *stem*, the *buds* and the *twig* are shown in red, yellow and white, respectively. The first unit of the first stem is *rooted* to the Terminal 1. The blue arrows indicate corresponding *bud* faces.

As figure 7 indicates, MTZ has hierarchical structure, defined as follows:

i. The main structure is composed of *stems*.
ii. Each *stem* ends with a *bud* with three branching faces (*BF*s).
iii. *Stems* must not form closed loops.
iv. A *stem* must have at least one unit (in such a case – that unit is the *bud*).
v. *Twigs* do not have *buds*.
vi. The first *stem* is *rooted* to the first terminal. It is the only stem connected directly to a terminal.
vii. All the terminals except the first one are connected by *twigs*.

A genotype of an i^{th} MTZ is defined as a list of *branches* in a form of two sub-lists of stems and twigs:

$$G_i : (S_i, T_i) \quad (1)$$
$$S_i : (s_1, s_2, \ldots, s_n) \quad (2)$$
$$T_i : (t_1, t_2, \ldots, t_m) \quad (3)$$

where S_i and T_i are the lists of n stems and m twigs of the genotype G_i, respectively.

A j^{th} stem from the list of stems S_i is defined as follows:

$$s_j : (j, p_j, f_j, u_j) \quad (4)$$

where j, p_j, f_j, and u_j are: the index of a j^{th} stem, the index of the *parent*, that is the stem to which the j^{th} stem is attached, the face of the parent stem's bud and the list of units u forming the j^{th} stem, respectively. $u \in (1,0)$, where 1 and 0 stand for a "right turning" and "left turning" trapezoid along the path of a branch. A bud is composed of the last trapezoidal unit in a stem and its mirror reflection along the longer base of the trapezoid. In the case of the first stem, p_1, and f_1 are virtual (since it is not connected to a parent stem).

A k^{th} twig from the list of twigs T_i is defined as follows:

$$t_k : (p_k, f_k, u_k) \quad (5)$$

where p_k, f_k, and u_k are: the parent that is the stem to which the k^{th} twig is attached, the face of the parent stem's bud and the list of units forming the k^{th} twig, respectively. The list of units in a twig

and stem are analogous.

Two MTZs presented in [11] are shown among others in Figure 8 with the new genotype encoding. Sub-figures 8.1 and 8.7 show: the example of a complex network (MTZ_C) and an efficient six-terminal network (MTZ_6) with relatively few units and simple paths, respectively.

5. The creation/transformation algorithm

The creation of any MTZ or transformation of any MTZ_A to any other MTZ_B can be executed in a few steps presented in Table 1.

Table 1: The sequence of steps for creating any MTZ or transforming one to another

Start	Transformation	Operator
General structure	↴	
↓	Add a *stem:* Add *twigs:* Remove *branches:*	$aS\,[G_i, (i_i, BF_i, u_i)]$ $aT\,[G_i, ((p_1, BF_1, u_1),(p_2, BF_2, u_2),...,(p_k, BF_k, u_k))]$ $rB\,[G_i, ((p_1, BF_1), (p_2, BF_2), \ldots,(p_i, BF_k))]$
Substructure @ buds	↲ ↴	
↓	Displace *branches*:	$dB[G_i, (((p_i, BF_i), (p_j, BF_j)),...)]$
Units @ branches	↲ ↴	
↓	Add units @ *branches*: Remove units @ *branches*: Invert units @ *branches*:	$aU[G_i,((p_1,BF_1,(v_{1^1},l_{1^1}),\ldots,(v_{k^1},l_{k^1})),\ldots,(p_j,BF_j,(v_{1^j},l_{1^j}),\ldots,(v_{k^j},l_{k^j})))]$ $rU[G_i,((p_1,BF_1,(l_{1^1},\ldots,l_{k^1})),\ldots,(p_j,BF_j,(l_{1^j},\ldots,l_{k^j})))]$ $iU[G_i,((p_1,BF_1,(l_{1^1},\ldots,l_{k^1})),\ldots,(p_j,BF_j,(l_{1^j},\ldots,l_{k^j})))]$
End	↲	

The operators listed in the right column of Table 1 are defined and explained in the subsection below.

5.1 Transformation operators

The following 7 operators suffice to "manually" construct a genotype of any MTZ or transform one MTZ to another:

Adding a stem

A single stem can be added to a MTZ_i :

$$aS\,[G_i, (i_i, BF_i, u_i)] \quad (6)$$

where G_i, i_i, and BF_i are: the genotype of MTZ_i, the index and bud face of the i^{th} stem to which a new stem is to be added, respectively. The new stem is assigned an index: $\mathrm{Max}[\forall(i_i)] + 1$.

If there was a branch b_j (stem or twig) at BF_i of the i^{th} stem, its parent's index (p_j) is replaced by i_s.

Adding twigs
Adding twigs is similar to adding stems. However, twigs are not indexed and it is possible to add multiple twigs at the same time:

$$aT\,[G_i, ((p_1, BF_1, u_1),(p_2, BF_2, u_2),...,(p_k, BF_k, u_k))] \quad (7)$$

where p_k and BF_k are: the index of the parent (the stem to which the k^{th} twig to be connected) and that parent's bud face, respectively; u_k is the sequence of trapezoidal units of the k^{th} twig.

Removing branches
Removing branches (stems or twigs) from MTZ_t is defined as follows:

$$rB\,[G_i, ((p_1, BF_1), (p_2, BF_2), \ldots,(p_k, BF_k))] \quad (8)$$

where p_k and BF_k are: the index of the parent and the parent's BF from which the k^{th} branch is to be removed. For an illustrative example of this operation see sub-figures 1 and 2 of Figure 8.

Displacing branches
Multiple branches can be displaced among bud faces:

$$dB[G_i, (((p_i, BF_i), (p_j, BF_j)),...)] \quad (9)$$

where p_i and BF_i, describe the original location of the i^{th} branch, namely: the index of the branch's parent and BF to which the i^{th} branch is attached, respectively; p_j and BF_j describe the new location for the i^{th} branch, namely: the index of the j^{th} stem and its BF, respectively.

Adding units at branches
Several units at multiple locations (loci) can be added to multiple branches:

$$aU[G_i,((p_1,BF_1,(v_{1^1},l_{1^1}),\ldots,(v_{k^1},l_{k^1})),\ldots,(p_j,BF_j,(v_{1^j},l_{1^j}),\ldots,(v_{k^j},l_{k^j})))] \quad (10)$$

where p_j and BF_j are: the parent stem and the bud face to which the j^{th} branch is attached, respectively. v_{k^j} and l_{k^j} are the value and position (locus) of the k^{th} unit to be added at the j^{th} branch, respectively. For an example of adding units to stems and twigs see sub-figures 4 and 5 of Figure 8, respectively.

Removing units at branches
Several units at multiple loci can be removed from multiple branches:

$$rU[G_i,((p_1,BF_1,(l_{1^1},\ldots,l_{k^1})),\ldots,(p_j,BF_j,(l_{1^j},\ldots,l_{k^j})))] \quad (11)$$

where p_j and BF_j are: the parent stem and the bud face to which the j^{th} branch is attached, respectively; l_{k^j} is the locus of the k^{th} unit to be removed from the j^{th} branch.

Inverting units at branches
Several units at multiple loci can be inverted at multiple branches:

$$iU[G_i,((p_1,BF_1,(l_{1^1},\ldots,l_{k^1})),\ldots,(p_j,BF_j,(l_{1^j},\ldots,l_{k^j})))] \quad (12)$$

where p_j and BF_j are: the parent stem and bud face to which the j^{th} branch is attached, respectively; l_{k^j} is the locus of the k^{th} unit at the j^{th} branch whose value is to be inverted. For an example of inverting units at stems and twigs see sub-figures 6 and 7 of Figure 8, respectively.

The operators defined above suffice to create any MTZ or, equivalently, to transform any MTZ to any other. The section below illustrates the latter.

5.2 Transformation of MTZ_C to the MTZ_6

Figure 8 shows a transformation from MTZ_C to MTZ_6. Although most of the operators transform branches in general, here, for clarity, the operations on stems and twigs are shown separately.

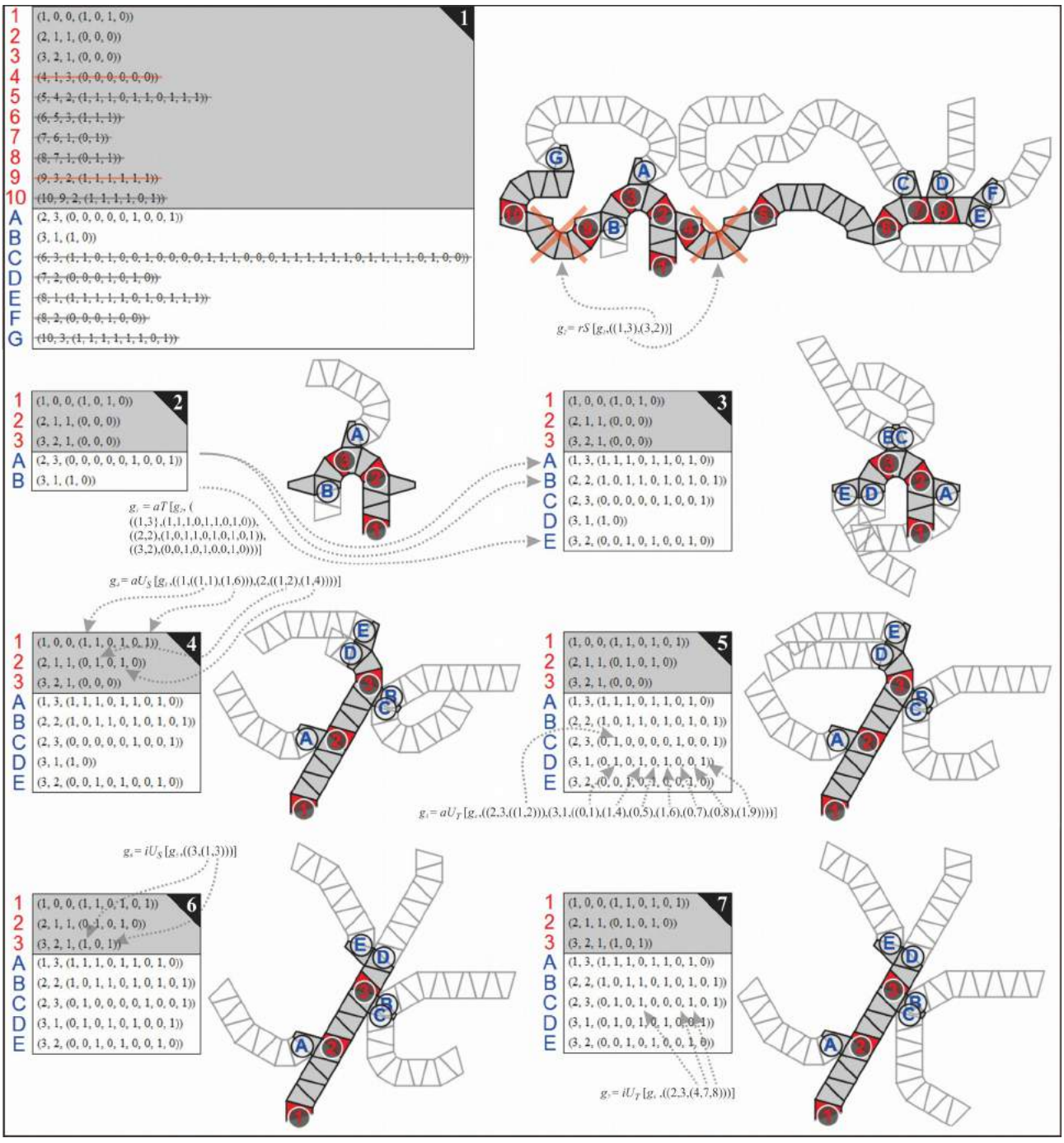

Figure 8. Transition from MTZ_C (sub-figure1) to MTZ_6 (sub-figure 7).

Although it is possible to manually create any MTZ, even for small cases it is quite challenging. This is due to the fact that creating even a single TZ path, and thus obviously a multi-branch TZ

network are NP-hard problems. The number of possible configurations for a layout of a single planar TZ path grows exponentially: 2^n, where n is the number of units. For MTZ the total number of possible configurations can be expressed as:

$$\prod_{i=1}^{s} 2^{u_i} \times u_i \tag{13}$$

where s is the number of *segments* of a MTZ, u_i is the number of units in the i^{th} segment. The notion of segment is explained in Figure 9.

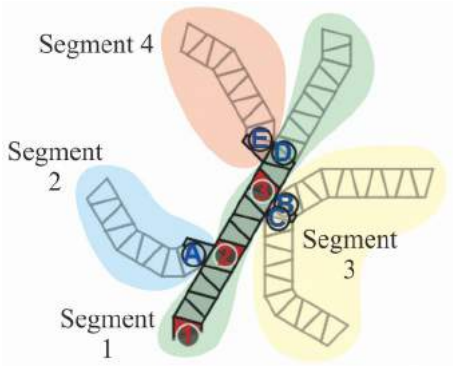

Figure 9: Illustration of the notion of *segments* with MTZ_6. For example *stems* 1–3 and *twig* D form a single segment.

Therefore the number of all possible "siblings" of MTZ_6 , that is networks of similar general structure (4 segments of 23, 10, 21, and 10 units) equals: $23 \times 2^{23} \times 10 \times 2^{10} \times 21 \times 2^{21} \times 10 \times 2^{10} =$ 890,977,738,760,171,343,052,800, which is almost a *septillion* (10^{24}). Meaningful exploration of such an enormous discrete solution space requires a search algorithm. The next subsection describes such a method formulated as a quasi-optimization problem.

6. A quasi-optimization of MTZ

In order to examine whether the transformations described above are effective for practical MTZ applications and also to select the most efficient parameters for the algorithm, a simple population-based experiment has been carried out. In principle, in mathematical optimization the ideal solution is not known. In this experiment, however, the ideal solution is explicitly given – it is MTZ_6. Therefore it is called a quasi-optimization. Nonetheless it also requires a few additional functions and operators inherent to population-based algorithms.

Random MTZ generator

A random MTZ genotype is generated by:

$$rG\,[(s_{\min}, s_{\max}), (t_{\min}, t_{\max}), (u_{\min}, u_{\max})] \tag{14}$$

where s, t and u are the numbers of: stems, twigs and units in all branches, respectively.

Fix genotype (fix MTZ general structure)

Any genotype G_k can be modified to have the desired number of stems and twigs:

$$F\,[G_k, s, t, l] \tag{15}$$

where G_k, s, t, and l are: a genotype to be modified, the desired number of stems, the desired number of twigs, and the number of randomly generated units in any additional branches (if applicable), respectively.

Tabulate genotype

A genotype G_i of MTZ_i relatively concisely, encodes the hierarchy of the elements into a nested list (list of lists). On the other hand, the tabulation transcribes G_i into a less concise, however, more "structured" rectangular matrix $T[G_i]$, in respect of the *buds*, as illustrated in Figure 10.

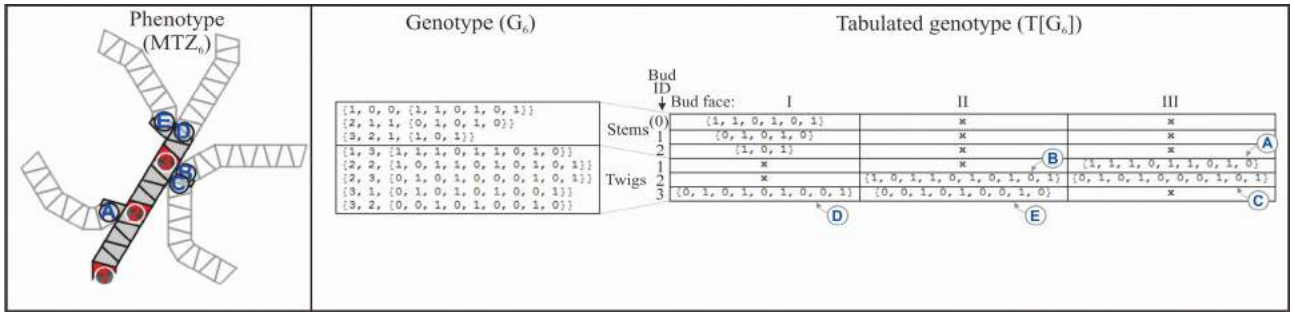

Figure 10: From the left: phenotype, genotype and tabulated genotype of MTZ_6. "x" stands for "nil". For illustration, the positions of twigs (A–E) have been indicated in $T[G_6]$.

Compare genotypes

The difference between two individuals MTZ_j and MTZ_k is measured by comparing their tabulated genotypes:

$$C[G_j, G_k] = \Delta(T^*[G_j], T^*[G_k])$$
$$C[G_j, G_k] \geq 0 \qquad (16)$$

where $T^*[G_j]$ and $T^*[G_k]$ are "standardized" tabulated genotypes G_j and G_k, respectively. Standardization here means that the structures of $T[G_j]$ and $T[G_k]$ have been made equivalent, so if they had different dimensions, blank elements have been added. The difference Δ is the total of differences between the bit strings of the respective branches of MTZ_j and MTZ_k. The differences among respective stems and twigs are calculated independently and summed up, as illustrated in Figure 11.

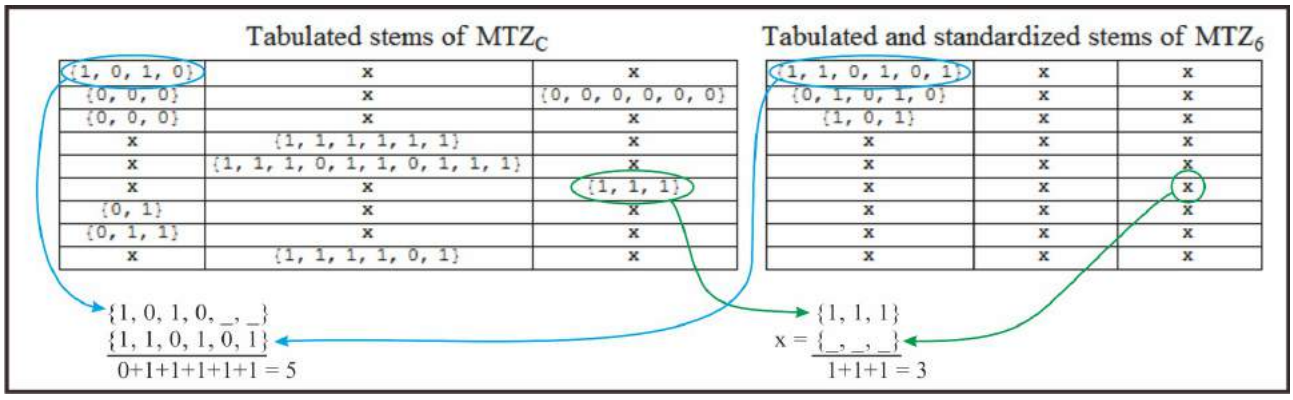

Figure 11: The method for calculating the difference between stems of MTZ_C and MTZ_6. The difference between empty lists is 0. The differences for all corresponding branches is summed up. For clarity only stems have been compared

6.2 Evolution strategy-based experiment (ES*)

This subsection describes a simple experiment based on the principles of the first mutation-selection scheme of the classic meta-heuristic method – evolution strategy (ES) [13]. ES like other evolutionary algorithms, operates on populations of candidate solutions stochastically with bias towards the individuals considered as relatively good. Unlike other EA methods, however, ES does not employ crossover, and is limited to intensive mutation. The general procedure is based on repetition of three simple steps:

1. evaluation of each individual in a population,
2. random selection with a preference for good individuals for the next generation,
3. mutation of the selected individuals,
4. go to 1 until the stop criterion is met.

In this experiment the selection of a genotype G_x is based on the comparison with the known ideal (G_6), that is, $C[G_6, G_x]$. It is a minimization, since the lower this value $C[G_6, G_x]$, the better. Since this is a quasi-optimization based on evolution strategy, it is denoted as ES*. In this experiment, the "multi-mutation" involves four functions directly derived from the transformation operators introduced in Section 5.1 "Transformation operators".

Displace-branch mutation

$$MdB[g_k, m_i] \tag{17}$$

where g_k and m_i are: the genotype of the k^{th} MTZ, and the mutation intensity, respectively; m_i is normalized, so 0 and 1 yield: "displacement of none", and "displacement of all branches", respectively.

Add-unit mutation

$$MaU[g_k, m_i] \tag{18}$$

Here m_i is normalized, so that 0 and 1, "does not add any units", and "doubles the units", respectively. The values of the added units are random integers, 0 or 1. The loci for added units are randomly distributed among all branches.

Remove-unit mutation

$$MrU[g_k, m_i] \tag{19}$$

As mentioned in Section 3, each stem must have at least one unit. There is no such constraint for the twigs. Here m_i is normalized, so that 0 and 1, "does not remove any units", and "removes all the units except one randomly selected unit per stem", respectively. The loci of removed units are randomly distributed among all branches.

Invert-unit mutation

$$MiU[g_k, m_i] \tag{20}$$

Here m_i is normalized, so that 0 and 1: "does not change value of any unit", and "inverts the values of all units" (equivalent to mirror reflection of entire MTZ), respectively. The loci of units for the inversions are randomly distributed among all branches.

Finally, the actual "multi-mutation" is defined as follows:

$$M[G_x, m_i, (w_{dB}, w_{aU}, w_{rU}, w_{iU})] \tag{21}$$

where, G_x and m_i are the x^{th} genotype, and mutation intensity, respectively. *M* randomly selects the mutation type according to w_{dB}, w_{aU}, w_{rU}, w_{iU} which are the weights for respective mutations: *MdB*, *MaU*, *MrU*, and *MiU*.

The experiment has been set up as follows:

- The initial population of 200 MTZs has been generated using the *rTZ* operator with the following parameters:
  - Number of stems: random integer (*i*) from the range [2, 5];
  - Number of twigs: $i \in [4,7]$;
  - Number of units in branches: $i \in [2,10]$;
  - All experiments start from the same initial population P_i.
- Each genotype G_x in the population is evaluated according to $C[G_6, G_x]$, that is compared to the genotype of MTZ_6 and assigned a numerical value (ε). Since it is a minimization problem, the goal is $\varepsilon = 0$.
- 10% of the best genotypes are selected.
- 10 copies of each selected genotype are subject to mutation.
- The stop criterion: the process is repeated for 100 iterations (generations).
- 10 trials for each experiment.

The experiment has been performed for several mutation intensities (m_i). Figure 12 shows selected examples.

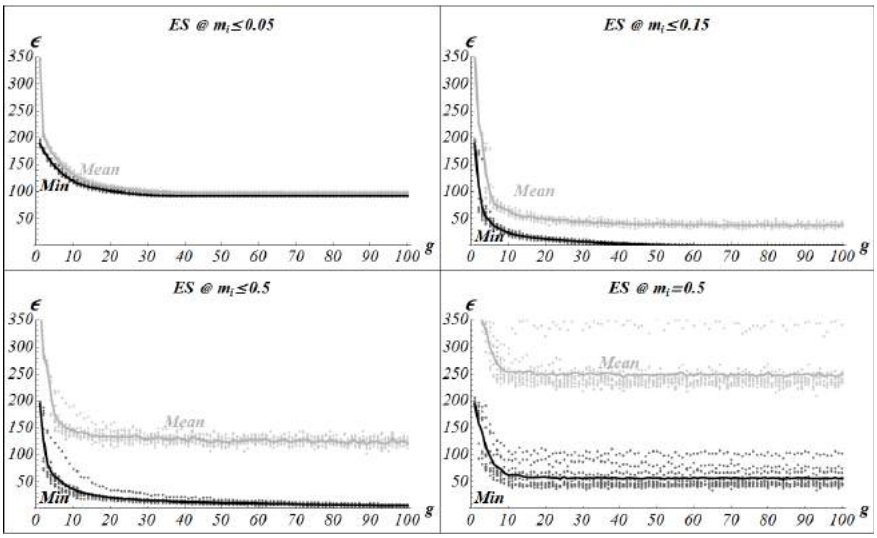

Figure 12: ES* with three variable and one constant mutation intensity (m_i). Dots, and lines indicate the individual, and averaged values for each generation, respectively. Gray and black indicate the mean and minimal values in each generation, respectively.

As Figure 12 indicates, the convergence of this ES* is sensitive to the mutation intensity. Moreover, it should be kept relatively low, that is between 0 and 0.2. In fact, it is a more efficient strategy to randomly draw the m_i value from the given range $[0, m_i^{MAX}]$. Figure 13 compares selected convergences in detail.

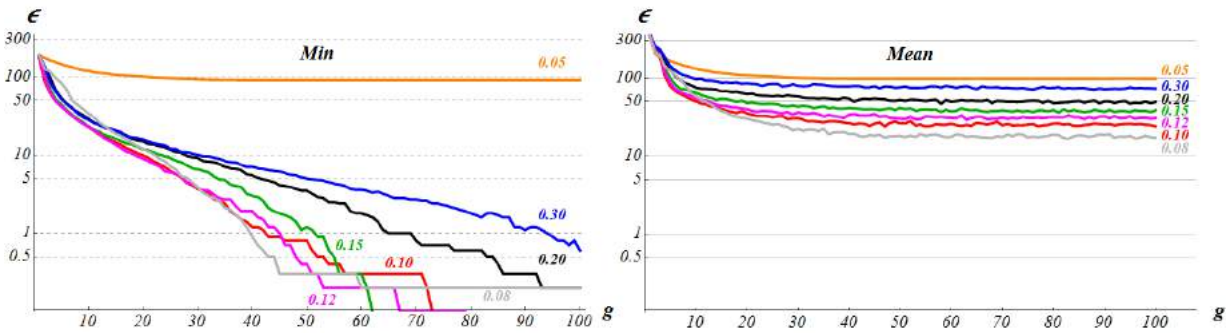

Figure 13: The log-plot emphasizing the convergence of ES* @ different mutation intensities. On the left, and right the minimal and mean values averaged for 10 trials are shown, respectively. The values of m_i^{MAX} are shown for each plot.

As figure 13 indicates, ES* @ m_i^{MAX} from the range [0.1, 0.15] produces the best results. Therefore value 0.12 has been assumed for the subsequent simulations.
In the next experiment one of four mutations has been excluded and the results have been compared in Figure 14.

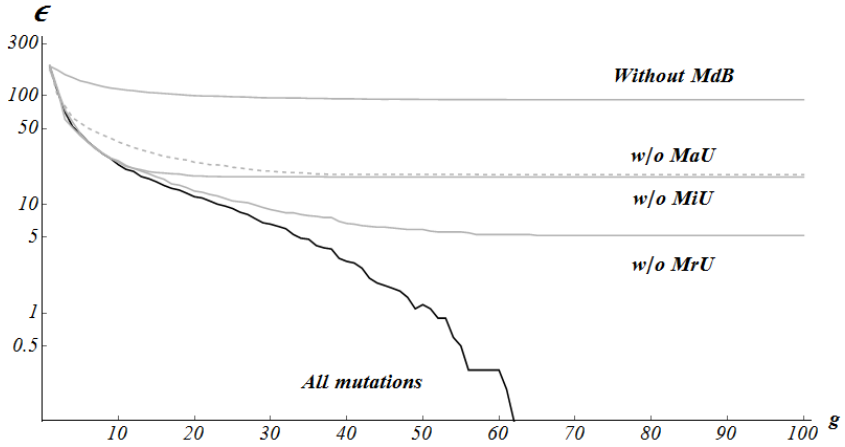

Figure 14: The log-plot showing the convergence of ES* with all four mutations and four setups with one of the mutations excluded. The former reach a certain quality but further improvement is not possible.

As Figure 14 indicates, the mutations are not only sufficient, but all four are necessary for ES* to reach the goal.

6.3 Inside the black box

An additional trial has been performed with the same settings as described above, starting from the same initial population with $m_i = 0.12$, the only difference was the stop criterion, so the procedure stopped after finding the ideal. In this case it was the 50th generation. In this experiment, however, all information about the intermediate steps (generations) has been stored. At each generation, full population, that is 200 individuals have been recorded, including the mutation type and intensity applied to each individual. Moreover, each offspring received a sequential number, which allowed to identify the relationships among the individuals. Figure 15 shows the tree-graph of the algorithm.

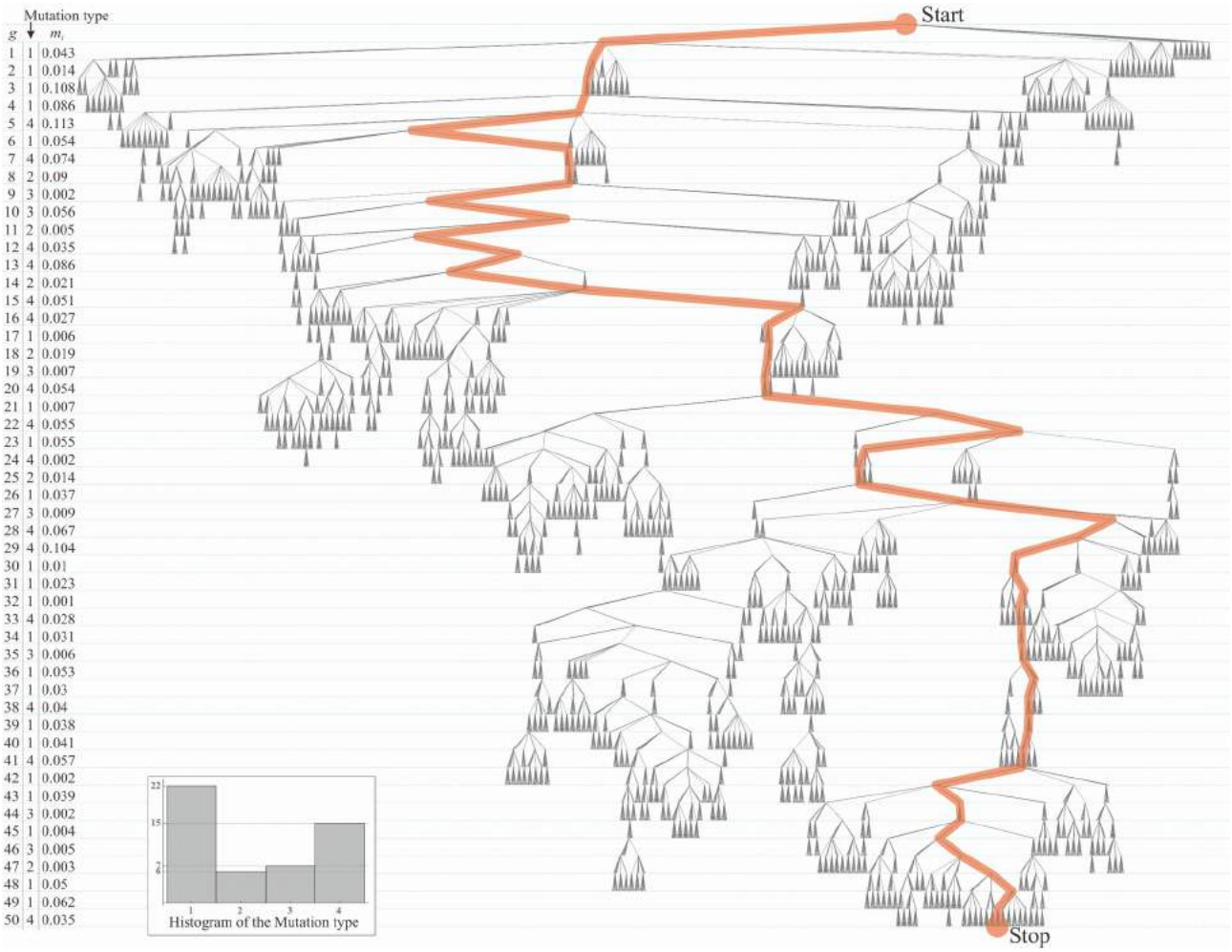

Figure 15: The history of selected trial which produced the genotype of the ideal, MTZ_6 in 50 generations (g). Only the offspring which succeeded to the next generation are shown. The path of the first ideal solution is shown in red. On the left: table showing the generation number, mutation type and intensity applied to the "red" genotype. Bottom left: the histogram of mutation type.

Figure 15 shows the "successful" individuals, that is the best 10% of each population, which have been selected for the subsequent generation. In this trial, 10,000 (200 individuals × 50 generations) genetic operations have been evoked. This number is incomparably smaller than the total number of all possibilities mentioned in Section 4.2 "Transformation of a MTZ to the MTZ_6", that is ~ 8.9×10^{23}. In other words, the ideal solution has been found by exploring merely ~$1.1*10^{-20}$ of the solution space.

Table 2 collects all the intermediate mutations of the successful genotype (referred to as "red" in Figure 15).

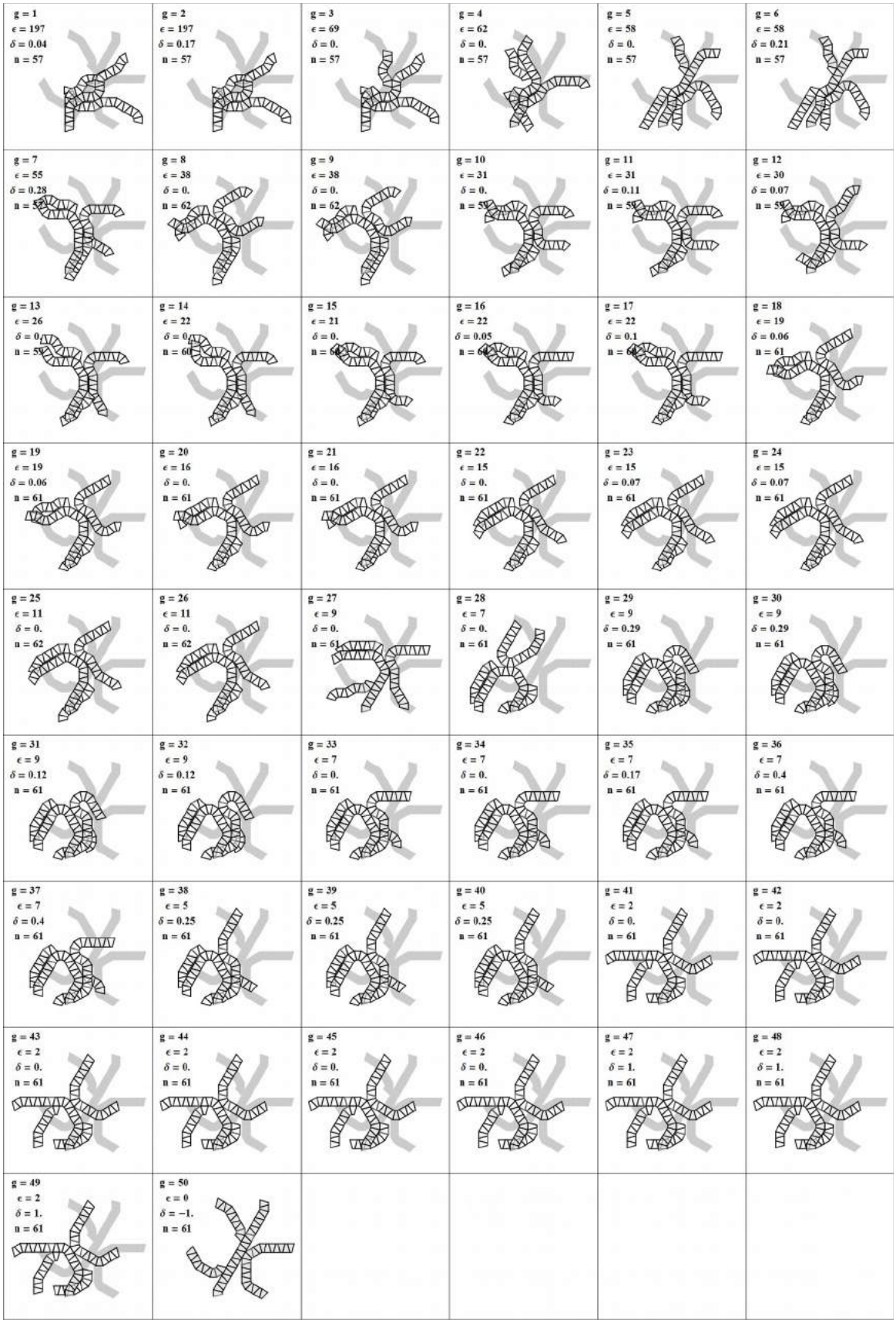

Table 2: All successful mutations of the "red" genotype. For each mutation the number of generation (g), error (ε), relative error to the best individual in the population (δ), and the number of units (n) are shown in the top left corner.

Approximately half of the times the “red” genotype was the best in a population, as illustrated in the Figure 16.

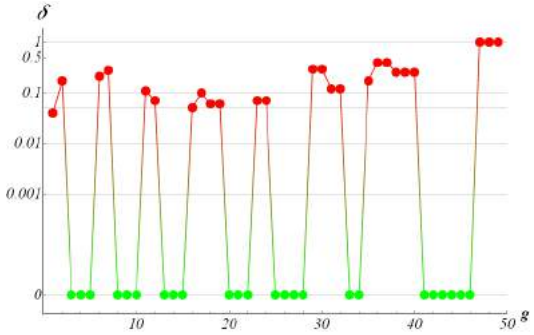

Figure 16: The relative error of the “red” genotype to the best individual in the population at each, but last generation. The best and worse ranks are shown in green and red, respectively.

7. Optimization with Evolution Strategy (ES)

The previous experiment was somewhat unrealistic since the ideal solution was explicitly known. Nonetheless it helped benchmarking and served to understand the general dynamics of the algorithm and to select the most efficient parameters. It was also particularly practical, as it did not employ an objective function which is usually computationally expensive. Thus the simulations were very quick, namely one trial took less than a minute. The following experiment is an optimization in the full sense, and therefore will be denoted as ES (without the asterisk) The known solution (MTZ_6) is not considered as the ideal anymore, but is still assumed to be very good and will be used for comparison. Thus an objective function must be formulated. The problem is defined as follows:

1. There are 6 terminals (T) to be communicated by MTZ
2. The first terminal (T_S) and the initial direction of MTZ are given.
3. MTZ is to have as few modules as possible.
4. The modules must not collide with each other.

The problem is formulated as minimization, namely: the summation of the distances between each terminal and respective twig tip (*tT*), called the *reaching error* (r_E) is to be minimal and the number of units in MTZ (*n*) is also to be minimal, as illustrated in Figure 17. The objective function in minimization is usually called the cost function.

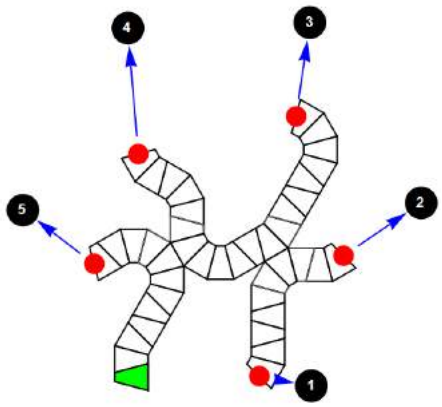

Figure 17: The MTZ optimization problem: each twig tip (*tT*) must come as close as possible to the respective terminal, the total number of units (*n*) is to be minimal, and the MTZ must not intersect with itself. The initial terminal (*Ts*), the remaining terminals, and the twig tips (*tTs*) with corresponding *reaching errors* (r_E) are shown in: green, by black dots, red dots and blue arrows, respectively.

The cost function

The cost function (CF) should consistently give lower values for better candidates. For practical reasons, so that the optimization can be performed in realistic time, CF should also be as computationally inexpensive as possible. Constructing such a function for a constrained problem is usually difficult [14], especially when unacceptable solutions are common in the solution space as shown in Table 2. Moreover, since the solution space has such infeasible areas (corresponding to unacceptable solutions), in order to reach very good solutions it is often necessary to traverse through the unacceptable ones [15]. In other words, the infeasible solutions should not be excluded, but "penalized" by adding to CF a penalty term proportional to the constraint violation [16,17,18,19]. In this experiment, however, the self-intersection prohibition can be ignored altogether. It is intuitive to assume that the best MTZs, that is comprised of the smallest number of units will not self-intersect.

$$CF_{MTZ} = n + w \times r_E \tag{22}$$

$$r_E = \sum_{i=1}^{5} \varepsilon_i \tag{23}$$

where n is the total number of units in MTZ, ε_i is the distance between the i^{th} terminal and corresponding tT; w is a parametric weight.

Since MTZ_6 is comprised of 64 units and its respective tTs reach exactly the 5 terminals ($r_E = 0$), the corresponding cost function CF_{MTZ6} equals 64. Therefore the best MTZs should have similar CFs. In order to promote the MTZ configurations that "reach out for the terminals", the weight w must be greater than 1, otherwise such solutions "do not have motivation to grow towards the terminals" and quickly get stuck in local minima.

7.1 The results

Figure 18 shows the results of 6 and 4 trials for w equals to 3 and 4, respectively.

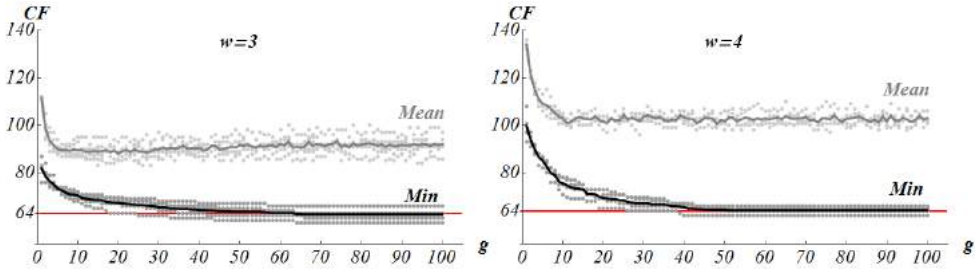

Figure 18: On the left: 6 trials with $w = 3$, on the right: 4 trials with $w = 4$. Dots, and lines indicate the individual, and averaged values for each generation, respectively. Gray, black, and red indicate the mean and minimal values in each generation, and the reference value for MTZ_6, respectively.

Interestingly, all experiments produced MTZs with fewer units than MTZ_6 (although with higher r_E). Figure 19 shows the best results in selected trials.

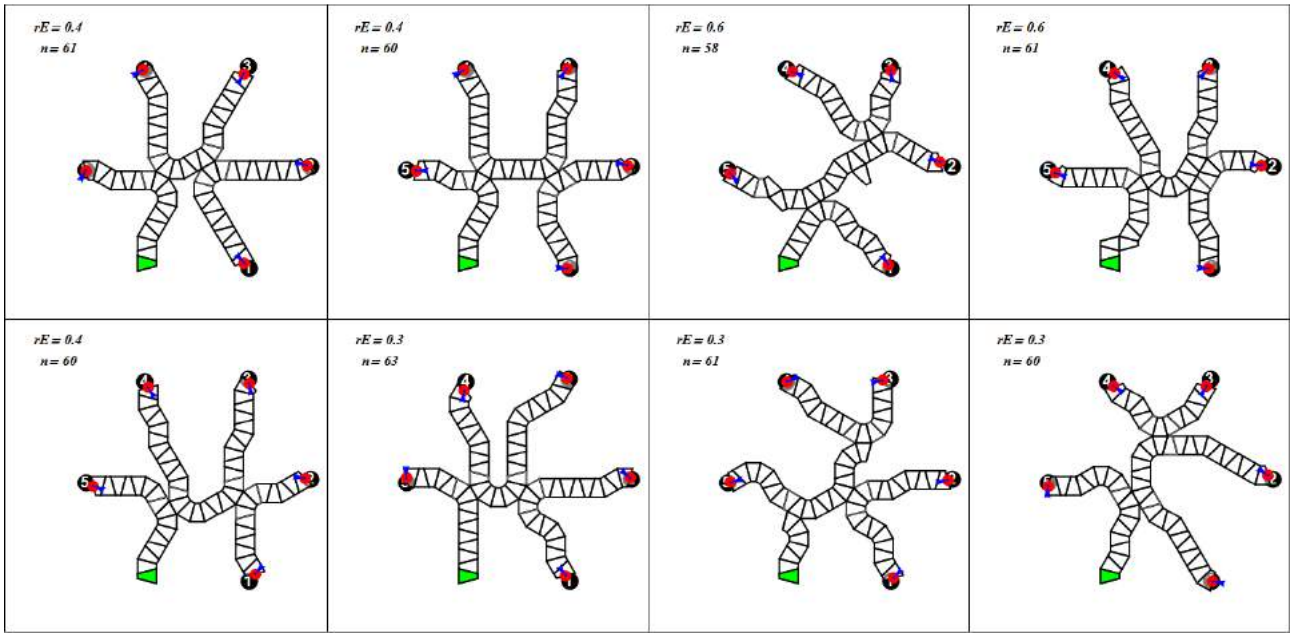

Figure 19: A variety of solutions produced by ES, all are competitive to MTZ_6. The top row: selected 4 trials with $w = 3$. The bottom row: 4 trials with $w = 4$. Merging two stems in the result from the top row and 3rd column further improves this solution (n → 57). For each MTZ the reaching error (r_E), and the number of units (n) are shown in the top left corner.

As Figures 18 and 19 indicate, whether w is set to 3 or 4 the algorithm produces equally competitive results. Moreover, Figure 20 shows selected ("milestone") generations of the 2nd trial of the experiment with $w = 4$, and confirms that the assumption that such simple CF will suffice was correct.

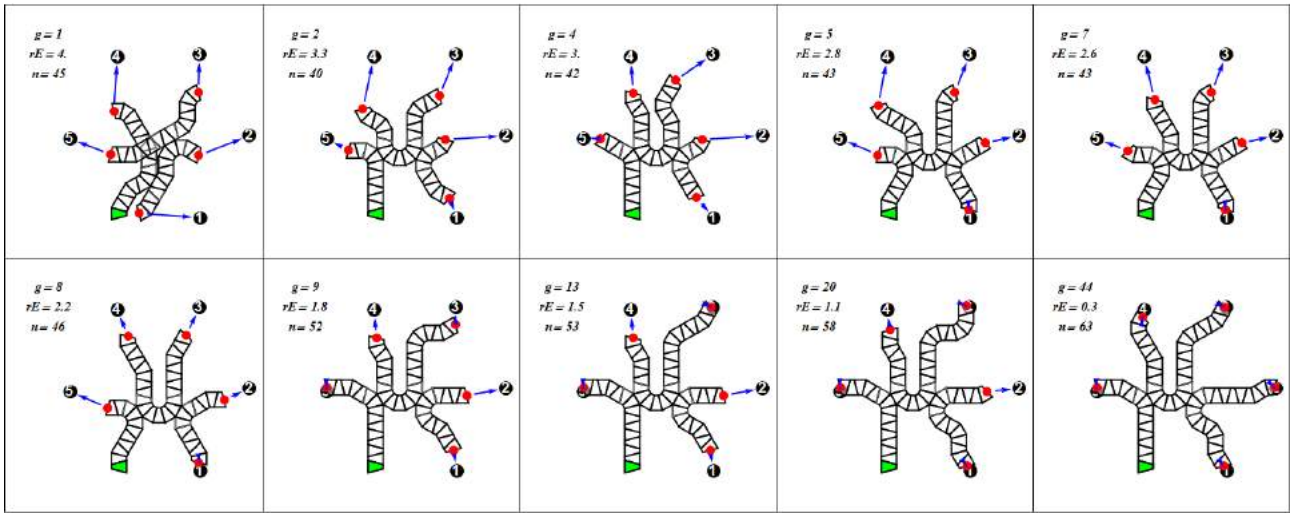

Figure 20: Selected generations of the 2nd ES trial with $w = 4$. For each mutation the generation number (g), reaching error (r_E), and the number of units (n) are shown in the top left corner.

As Figure 20 indicates, the best individual in the first generation was infeasible due to constraint violation, that is self-intersections. However, it soon, that is to say already in the next generation, evolved into feasible offspring, finally producing a satisfactory solution.

7.2 The "multi-branch bridge" based on the result of ES

The solution produced by ES with $w = 3$ in the 2nd trial (ES_{w3t2} for short) has been used as a layout of a TZ network (MTZ_{w3t2} for short) for the problem formulated in Section 3: "Possible scenario: a "multi-branch bridge" (see Figure 6). Figures 21 and 22 show the plan & front views, and perspective view of MTZ_{w3t2}, respectively.

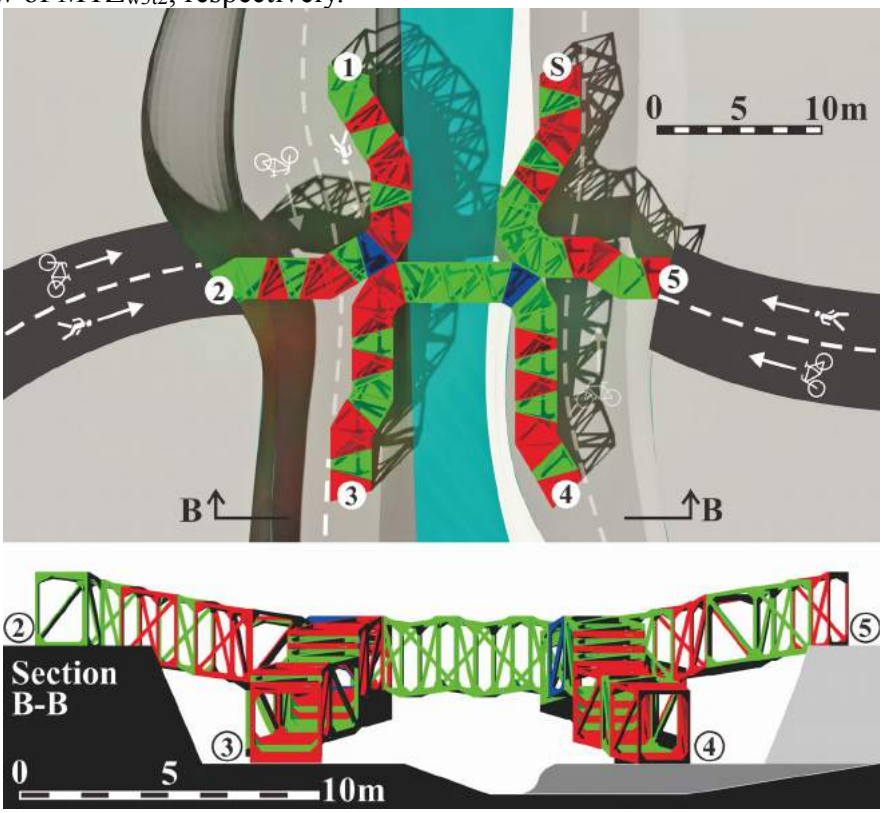

Figure 21: On the top: the plan of MTZ_{w3t2}. On the bottom: the front view transverse to section line B-B. To articulate the modularity, units R & R_2 (rotated R) and L & L_2 (rotated L) are shown in green and red, respectively. The branching units are shown in blue.

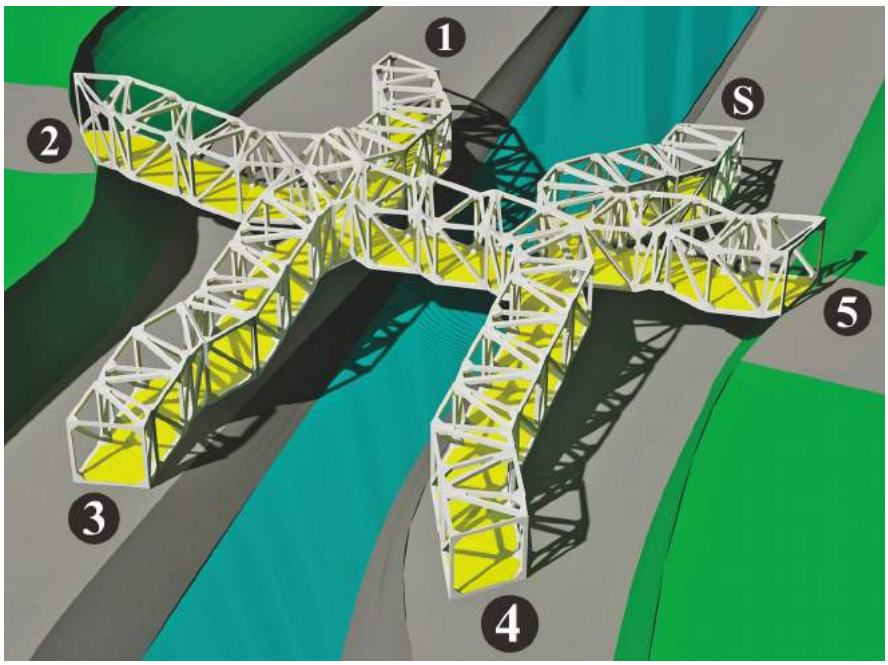

Figure 22: A perspective visualization of the MTZ_{w3t2}.

8. Conclusions and Future Work

The new encoding for multi-branch Truss-Z (MTZ) introduced here is substantially more efficient than the previous encoding presented in [11]. It is not only more intuitive, but also more suitable for "genetic operations" and therefore – better suited for meta-heuristic optimization methods. The genetic operators for MTZ introduced in this paper are efficient as demonstrated in numerous examples and the minimization experiments show good convergence. A 6-branch MTZ for linking 3 pedestrian/cycling paths has been presented. Although the structural considerations are not part of this paper, this modular 6-branch bridge can be considered relatively realistic.

Primary experiments with crossover showed no benefit to the optimization algorithm, therefore have not been presented here. This was most likely due to the "harshness" of the procedure, which produces feasible individuals, but hardly resembling the parents. This phenomenon was also noticeable while calibrating the mutation intensity which also was relatively low (≤ 0.12). Nevertheless, elaboration of crossover is a natural direction for the future research. This paper presents a constrained optimization problem, however, the formal definition of the actual constraints was not necessary. Moreover, the problem of MTZ optimization has been simplified to the MTZ *layout* optimization. Therefore a realistic three-dimensional case study where constraints are explicitly formulated is under consideration. Finally, the problem of intermediate supports has not been addressed sufficiently yet. However, this will be part of the design of the TZ prototype in the future.

References

1. Pollack M.E., Intelligent Technology for an Aging Population: The Use of AI to Assist Elders with Cognitive Impairment, Artificial Intelligence Magazine 26(2), pp. 9-24, 2005.
2. Dunbar G., Holland C., Road Safety Research Report No. 37: Older Pedestrians - A Critical Review of the Literature. Department of Transport: London, England, 2004.
3. Zawidzki M., Creating organic 3-dimensional structures for pedestrian traffic with reconfigurable modular "Truss-Z" system, International Journal of Design & Nature and Ecodynamics 8(1), pp. 61–87, 2013.
4. Zawidzki M., Retrofitting of pedestrian overpass by Truss-Z modular systems using graph-theory approach, Advances in Engineering Software, 81, pp. 41–49, 2015.
5. Zawidzki M., Nishinari K., Modular Truss-Z system for self-supporting skeletal free-form pedestrian networks, Advances in Engineering Software, 47(1), pp. 147–159, 2012.
6. Zawidzki M., Retrofitting of pedestrian overpass by Truss-Z modular systems using graph-theory approach, Advances in Engineering Software, 81, pp. 41–49, 2015.
7. Zawidzki M., Tiling of a Path with Trapezoids in a Constrained Environment with Backtracking Algorithm: http://demonstrations.wolfram.com/TilingOfAPathWithTrapezoidsInAConstrainedEnvironmentWithBack/ Wolfram Demonstrations Project, Published: September 20, 2011.
8. Zawidzki M., Tateyama K., Application of Evolution Strategy for Minimization of the Number of Modules in a Truss Branch Created with the Truss-Z System, in Y. Tsompanakis, B.H.V. Topping, (Editors), Proceedings of the Second International Conference on Soft Computing Technology in Civil, Structural and Environmental Engineering, Civil-Comp Press, Stirlingshire, UK, Paper 9, doi:10.4203/ccp.97.9, 2011.
9. Zawidzki M., Nishinari K., Application of evolutionary algorithms for optimum layout of

Truss-Z linkage in an environment with obstacles, Advances in Engineering Software, 65, pp. 43–59, 2013

10. Zawidzki M., Nagakura T., Foldable Truss-Z module, Proceedings for ICGG 2014: 16th International Conference on Geometry and Graphics, Innsbruck, Austria, August 4 – 8, 2014.
11. Zawidzki M., Nishinari K., Application of Evolutionary Algorithm for Optimization of the Layout of a Multi-Branched Network Constructed with Truss-Z System, Proceedings for WCCM 2012: the 10th World Congress On Computational Mechanics, São Paulo, Brazil, July 8 – 13, 2012.
12. Eiben A.E., Smith J.E., Evolutionary Algorithms, in F. Neri, C. Cotta, and P. Moscato (Eds.): Handbook of Memetic Algorithms, Studies in Computational Intelligence, Vol. 379, pp. 9–27 , Springer, 2012
13. Bäck T., Hoffmeister F., Schwefel H-P., A Survey of Evolution Strategies, in R. K. Belew and L. B. Booker (Eds.): Proceedings of the Fourth International Conference on Genetic Algorithms: University of California, San Diego, July 13-16, 1991, Morgan Kaufmann, 1991.
14. Michalewicz Z, Fogel DB. How to solve it: modern heuristics. Berlin: Springer; 2000.
15. Michalewicz Z. Genetic algorithms + data structures = evolution programs. 3rd ed. New York: Springer-Verlag; 1998.
16. Hasançebi, O., S. Çarbaş, E. Doğan, F. Erdal, and M. P. Saka. Performance evaluation of metaheuristic search techniques in the optimum design of real size pin jointed structures. Computers & Structures 87, no. 5 (2009): 284-302.
17. Hasançebi, O., S. Çarbaş, E. Doğan, F. Erdal, and M. P. Saka. Comparison of non-deterministic search techniques in the optimum design of real size steel frames. Computers & structures 88, no. 17 (2010): 1033-1048.
18. Deb, Kalyanmoy. An efficient constraint handling method for genetic algorithms. Computer methods in applied mechanics and engineering 186, no. 2 (2000): 311-338.
19. Coello, Carlos A. Coello. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. Computer methods in applied mechanics and engineering 191, no. 11 (2002): 1245-1287.