On the Optimization of 2D Path Network Layouts in Engineering Designs via Evolutionary Computation Techniques

Alexandru-Ciprian Zăvoianu*, Susanne Saminger-Platz
Department of Knowledge-Based Mathematical Systems
Johannes Kepler University Linz, Altenbergerstraße 69, 4040 Linz, Austria
Email: {Ciprian.Zavoianu, Susanne.Saminger-Platz}@jku.at

Doris Entner, Thorsten Prante
Design Automation
V-Research GmbH - Industrial Research and Development, Stadtstraße 33, 6850 Dornbirn, Austria
Email: {Doris.Entner, Thorsten.Prante}@v-research.at

Michael Hellwig
Research Centre for Process and Product Engineering
Vorarlberg University of Applied Sciences, Hochschulstrße 1, 6850 Dornbirn, Austria
Email: Michael.Hellwig@fhv.at

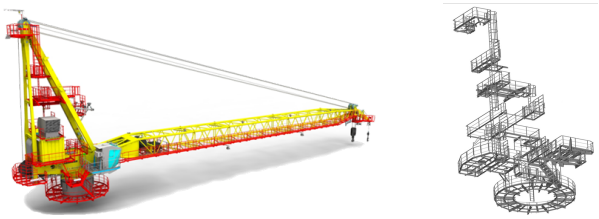
Martin Schwarz, Klara Fink
Technology Management
Liebherr-Werk Nenzing GmbH, Dr.-Hans-Liebherr-Str. 1, 6710 Nenzing, Austria
Email: {Martin.Schwarz, Klara.Fink}@liebherr.com

Summary
We describe an effective optimization strategy that is capable of discovering innovative cost-optimal designs of complete ascent assembly structures. Our approach relies on a continuous 2D model abstraction, an application-inspired multi-objective formulation of the optimal design task and an efficient coevolutionary solver. Results on both artificial problems and an industrial test case empirically support the value of our contribution to the field of design automation.

Keywords: *continuous multi-objective optimization, coevolution, engineering design automation, ascent assemblies*

1 Introduction

The present work is primarily motivated from theoretical and practical considerations from the field of engineering design automation. Concretely, we describe initial results concerning the automatic generation of cost-optimal *complete ascent assembly structures* (CAA-Structures) – external access structures for cranes, building facades, over-sized industrial machines, etc. Figure 1b shows an example of a CAA-Structure that is itself composed from several types of ascent assembly modules (i.e., sub-assemblies) like rectangular or round platforms, stairs and ladders.



(a) (b)

Figure 1: An example of an offshore crane where different ascent assembly modules highlighted in red (a) are combined to form a fairly complex CAA-Structure (b).

The task of designing individual ascent assembly modules, although important, is rather repetitive and time consuming. In recent years, in light of strong financial and operational incentives, there has been a consistent and successful effort to standardize individual ascent assembly modules and to automate their design process.[1] As a result, the task of automating the design of cost-optimal CAA-Structures has itself become a feasible undertaking since it can be regarded as a search for a 3D "skeleton" that indicates which ascent assembly modules are required and where they should be placed in ordered to ensure that the

CAA-Structure provides access with minimal costs.

In spite of the apparent simplicity suggested by a 3D "skeleton", there is still a large set of particularities and uncertainties associated with real-life CAA-Structure design tasks in modern engineer-to-order environments. In order to have a relevant but accessible formulation for analyzing and comparing various proofs of concept, in Section 2.1 we introduce a 2D model abstraction of the optimal design task. The results we obtained for a real-life industrial design scenario (described in Section 4.3) were largely validated by domain experts (design engineers), thus indicating that the cost-optimal and innovative solutions obtained via the introduced 2D model abstraction and the proposed optimization strategy (described in Section 3) are a large step forward towards the final goal of fully automating the design of cost-optimal CAA-Structures.

2 Modelling and formal problem statement

2.1 *Description of model abstraction*

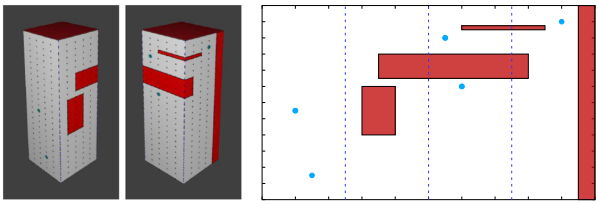
A user that wishes to generate a cost-optimal CAA-Structure is expected to provide at least three inputs: a 3D model of the solid base / support object to which the CAA-Structure is to be attached, a set of desired points of access on this structure and information regarding potential obstacles that are defined on the 3D solid base object. The latter requirement is extremely relevant as obstacles indicate severe restrictions regarding the placement of ascent assembly modules in certain areas.

For example, in Figure 2a we illustrate a simplified design case that involves a cuboid structure, five access points and four obstacle areas that are spread across three faces of the cuboid. A far clearer representation of this academic automated design scenario can be obtained by unfolding the 3D model. The result of the unfolding procedure, shown in Figure 2b, is a 2D design surface that is characterized by a left edge - right edge continuity – i.e., line segments exiting the left edge at a certain height and orientation, should enter the right edge at the same height and orientation in order to model the circular structure of the facade. More importantly, as a result of the unfolding, the task of finding a cost-optimal 3D "skeleton" of the ascent assembly is transformed into that of discovering a simpler 2D design "skeleton": a *cost-optimal 2D path network layout* on the 2D design surface that links all the points of interest while avoiding the obstacle areas.

Although an obvious simplification of the 3D case, we shall see in the next section that the resulting *2D path network layout problem* is not trivial.

2.2 *Formal definition of the path network layout problem*

When considering a set of n user defined *access points* / definition vertices $\{p_1,\ldots,p_n\}$, the goal of the 2D optimal path network layout problem is to discover a (graph) structure T of minimal cost that links all these points. T must obviously span all the access points but it may also contain up to k well-placed extra points (2D vertices)



(a) (b)

Figure 2: A 3D model and the corresponding 2D design plane obtained after unfolding. Access points are marked with blue circles and obstacles are marked with red.

$\{s_1,\ldots,s_k\}$ that help minimize the total cost of T. Thus, E, the set of possible edges that contains all the segments that can be used to construct T, is defined over the union $\{p_1,\ldots,p_n\}\cup\{s_1,\ldots,s_k\}$. When considering a positive cost for connecting any two points, it is obvious that T is in fact a tree. Formally, the resulting minimal path optimization task can be defined as: *"Determine* $k\in\mathbb{N}$ *and* $s_1,\ldots,s_k\in\mathbb{R}\times\mathbb{R}$ *in order to minimize*

$$f_1(p_1,\ldots,p_n,s_1,\ldots,s_k)=\sum_{(ij)\in E}c(i,j)x_{(ij)}, \tag{1}$$

subject to:

$$x_{(ij)}\in\{0,1\},\ \forall(ij)\in E\ \ and$$
$$\sum_{(ij)\in E}x_{ij}=(n+k)-1\ \ and$$
$$\sum_{(ij)\in E,i\in F,j\in F}x_{ij}\leq|F|-1,\ \forall F\subseteq\{p_1,\ldots,p_n,s_1,\ldots,s_k\},$$

where $G=(\{p_1,\ldots,p_n,s_1,\ldots,s_k\},E)$ *is a complete graph."*

The function $c(i,j)$ from Equation (1) denotes the cost of linking vertices i and j. In the case of ascent assemblies, this cost can be defined as the combined price of individual modules (i.e., platform, stair, and ladder segments) and of connecting them (e.g., welding) required to construct a walkway between points i and j. While it is expected that, in the general case, $c(i,j)$ is proportional to the Euclidean distance between the two vertices, in more realistic scenarios, obstacles and other penalties do influence the cost function.

For example, when considering a slightly more realistic description of optimal layouts for CAA-Structures, one would likely consider inside $c(i,j)$ a large penalty $\Gamma_{(ij)}$ for assembly modules that extend into obstacle areas when connecting vertices i and j and another smaller penalty for assembly modules that are not placed at a preset angle requirement – e.g., platforms should be placed at an angle of exactly 0° to the horizontal axis, stairs at 45°, and ladders at 90°. All these "allowed/preferred" design angles should be provided as a user defined set, e.g., $U=\{0,45,90\}$. The resulting angle-aware cost function could be defined as:

$$c(i,j)=dist(i,j)\left(1+\frac{\min B_{(ij)}}{100}z\right)+\Gamma_{(ij)} \tag{2}$$

where z is a parameter ($0 \leq z \leq 4$) that controls the magnitude of the angle penalty and $B_{(ij)}$ is a set that contains the absolute differences between $\alpha_{(ij)}$ – the horizontal angle of the segment (ij) – and the allowed placement angles stored in U. For instance, given the example in the previous paragraph , $B_{(ij)} = \{|\alpha_{(ij)} - 0|, |\alpha_{(ij)} - 45|, |\alpha_{(ij)} - 90|\}$. For the set of tests we report over in the present study, $dist(i, j)$ marks the 2D Euclidean distance between vertices i and j.

It is noteworthy to remark that when $z = 0$ in Equation (2) and one does not consider obstacle areas and a left-right continuity of the design plane, $c(i, j)$ is reduced to the Euclidean distance and Equation (1) gives the definition of the well-known *Euclidean Steiner Tree Problem* (ESTP).[2] Although they represent the simplest cases of the 2D path network layout problems we aim to solve, ESTPs are proven to be NP-hard.[3] Nevertheless, ESTPs have also been intensively studied by mathematicians and computer scientists and this opens up the possibility to compare (in part) our proposed solving strategy with other results from literature on standard benchmarks. In the context of ESTPs, the k points that help minimize the 2D path layout between the access points are called *Steiner points* and throughout this work we shall also maintain this naming in the context of optimal path network layouts for CAA-Structures. Furthermore, the NP-hard nature of ESTPs also motivates our strong preference for a metaheuristic-based solver. As such, we would like to inform the reader that the lexicon throughout the remainder of this work is tailored for the field of evolutionary computation[4] – one of the most (historically) successful global optimization paradigms for tackling complicated optimization problems.

Finally, in the context of path network optimization tasks for ascent assemblies, opting for a value of $z = 0$ in Equation (2) would result in a problem definition that enables the optimizer to freely explore the design space and quite possibly discover innovative designs (i.e., innovative ways of connecting the desired access points). However, the best results of such an "open" definition would (likely) only be interpreted as optimal design "suggestions" as building them to specification would be unfeasible. When opting for a larger value of the penalty parameter z and a realistic list of standard allowed angles, good results of the more "restricted" path optimization problem are far more likely to resemble "blueprints" of the ascent assembly.

3 Optimization procedure

3.1 Solution codification

A very important aspect of trying to solve the problem described in Section 2.2 is represented by the encoding of individuals / candidate solutions. First and foremost, a good encoding should be simple (*general*) in order to be compatible with many fitness assessment strategies and in order to allow for an immediate extension to 3D scenarios. Secondly, the encoding should also be *flexible* as the number of Steiner points required by each problem is unknown. Although the latter characteristic seems to hint towards a variable-length encoding, we argue in favour of a fixed-length variant in which the maximal number of the Steiner points expected to be discovered (i.e., k^*) is preset at a sufficiently large level. For example:

- in the case ESTPs, one can use the mathematically proven[2] upper bound $k^* = n - 2$
- in the case of all the ascent assembly optimization problems presented in Section 4 we experimented with several settings in the range $n \leq k^* \leq 3n$.

The task of "deciding" the exact number of Steiner points required for solving the problem at hand is "passed" to the fitness assessment method described in the next section. Apart from the extra simplicity that enables the usage of various standard genetic operators, our choice for a fixed-length encoding is also motivated by the desire to counteract potential solution bloating - a well-known and harmful phenomenon in terms of both solution quality and convergence speed that is associated in the field of evolutionary computation (genetic programming in particular) with combinations of strong (evolutionary) selection pressure and variable-length encodings.[5]

After opting for fixed-length encodings, we adopted a basic real-valued representation $\vec{x} = (x_1, x_2, \ldots, x_{2k^*-1}, x_{2k^*})$ of potential Steiner points, with the understanding that, given the encoded vertex $v_{(i,\vec{x})}$, $1 \leq i \leq k^*$, x_{2i-1} denotes the horizontal coordinate of $v_{(i,\vec{x})}$ and x_{2i} the vertical one.

The minimum and maximum ranges for $x_i \in \vec{x}$ are set according to the design scenario definition limits in the case of CAA-Structures and to the extreme coordinate values of the definition points in the case of ESTPs.

3.2 Fitness assessment

Let $o_1(\vec{x})$ denote the ability of the vertices encoded in a given candidate solution $\vec{x}$ to minimize Equation (1). In order to estimate $o_1(\vec{x})$, we employ a two-step process:

- Firstly, we build the union between all the k^* vertices encoded by $\vec{x}$ and the n definition points of the optimization scenario: $S_{\vec{x}} = \{v_{(1,\vec{x})}, \ldots, v_{(k^*,\vec{x})}\} \cup \{p_1, \ldots, p_n\}$.
- Secondly, starting with p_1, we apply Prim's algorithm[6] in order to construct $MT_{n,\vec{x}}$ – the *partial minimum spanning tree* (MST) over the set $S_{\vec{x}}$ that contains all n definition points. $MT_{n,\vec{x}}$ is a partial MST because the construction process is interrupted once all the definition points have been added to the tree.

Any vertex encoded in $\vec{x}$ that remains unlinked by $MT_{n,\vec{x}}$ has the property that its placement is highly likely not to improve in any way the formation of an optimal-cost path between all the definition points $\{p_1, \ldots, p_n\}$ – i.e., this vertex is deemed as having a low chance of being a potential Steiner point. We mark with $s_{i,\vec{x}}$, $i \in \{1, \ldots, m\}$, $m \leq k^*$ the vertices encoded in $\vec{x}$ that are part of $MT_{n,\vec{x}}$. Compared

with the unlinked vertices, any $s_{i,\vec{x}}$ has a better chance of being useful in constructing an optimal path between the definition points and is thus deemed a *potential Steiner point*. When considering previous notations, and denoting with $\Phi(MT_{n,\vec{x}})$ the total cost associated with $MT_{n,\vec{x}}$, we have that $\Phi(MT_{n,\vec{x}}) = f_1(p_1,\ldots,p_n,s_{1,\vec{x}},...s_{m,\vec{x}})$ where function f_1 is defined in Equation (1).

We argue that $o_1(\vec{x})$ can be well approximated by $\Phi(MT_{n,\vec{x}})$ because the closer the set of potential Steiner points in $\vec{x}$ is to $\{s_1,\ldots,s_k\}$, i.e., to the actual Steiner point set that represents the solution to Equation (1), the closer $f_1(p_1,\ldots,p_n,s_{i,\vec{x}},...s_{m,\vec{x}})$ is to $f_1(p_1,\ldots,p_n,s_1,\ldots s_k)$.

When considering the main motivation behind the present work (i.e., optimizing real-life industrial designs), it is highly likely that a more advanced future model abstraction might yield *secondary requirements regarding optimality*. For instance, these requirements might relate to: 1. the complexity of the overall CAA-Structure design (i.e., number of different module types that is required), 2. ensuring different levels of ease-of-access for different definition points, 3. CAA-Structure building time given present stocks of individual ascent assembly modules. Such possible secondary requirements appear to be rather conflicting with the currently identified primary one (i.e., cost minimization) and modeling them via penalties and rewards is expected to be extremely cumbersome. Alternatively, formalizing them as optimization objectives in their own right would be more natural and should yield better results from the perspective of a decision maker.

Motivated largely by the previous considerations but also by initial attempts to optimize $o_1(\vec{x})$ on benchmark ESTPs using evolutionary algorithms that were less successful than anticipated (showing signs of premature convergence), we defined an artificial secondary objective $o_2(\vec{x})$. This second objective is designed to be (slightly) conflicting with o_1 and is defined as:

$$o_2(\vec{x}) = \frac{1}{n-1}\left(\sum_{r=2}^{n} impr_{MST(r)} * 2^{size_{MST(r)}}\right) - 1.1^m, \quad (3)$$

where:

$impr_{MST(r)} = \Phi(MT_{r,\vec{x}}) - \Phi(MST_r)$ *and*

$size_{MST(r)} = 3 - \dfrac{3\Phi(MST_r)}{\Phi(MST_n)}$.

Inside Equation (3), when considering that p_r is the r^{th} user defined access point, in an analogous way to $\Phi(MT_{n,\vec{x}})$, we have that:

- $\Phi(MST_r)$ is the cost of the minimum spanning tree constructed over the set $\{p_1,\ldots,p_r\}$, i.e., $\Phi(MST_r) = f_1(p_1,\ldots,p_r)$;
- $\Phi(MT_{r,\vec{x}})$ is the total cost of the *partial minimal spanning tree* constructed over the union $S_{\vec{x}} = \{v_{(1,\vec{x})},\ldots,v_{(k^*,\vec{x})}\} \cup \{p_1,\ldots,p_r\}$

This means that $o_2(\vec{x})$ computes the average level to which $\vec{x}$ is able to solve Equation (1) for every incremental subset of definition points that is obtained when constructing the minimal spanning tree over $\{p_1,\ldots,p_n\}$. The smaller the subset the more important it is weighted inside the average and there is a small bonus for candidate solutions that achieve good results with a reduced number m of potential Steiner points.

Finally we have chosen to solve a multi-objective optimization problem that aims to simultaneously minimize both $o_1(\vec{x})$ and $o_2(\vec{x})$. Although empirically validated by all the results presented in Section 5, the inclusion of $o_2(\vec{x})$ alongside the main path minimization objective is highly counter-intuitive. The reason for this decision is two-fold:

- $o_2(\vec{x})$ is engineered to induce both some level of niching during the evolutionary search as well as a biasing of the multi-objective search towards robust candidate solutions that encode potential Steiner points which are generically well placed – i.e., able to improve the total minimal path in key locations that are common to many sub-paths.
- Having a multi-objective formulation enables us to check whether our assumption that $\Phi(MT_{n,\vec{x}})$ is a good enough approximation for $o_1(\vec{x})$ also holds when faced with a conflicting optimization objective that, to a certain extent, steers the search towards path layouts that are not necessarily cost-optimal.

3.3 The Multi-Objective Solver

In order to solve the previously introduced multi-objective optimization (MOO) problem, we performed a limited set of initial tests with NSGA-II[7] – a classical multi-objective evolutionary algorithm (MOEA) – and with DECMO2.[8] The latter is a newer hybrid and adaptive evolutionary approach specially designed for rapid convergence on a wide class of problems. DECMO2 was designed to capitalize on previous insights[9] that a cooperative coevolutionary strategy can deliver very competitive results on a wide range of MOO problems. As DECMO2 exhibited a better balance between convergence speed and final solution quality, we adopted it as our default solver. The DECMO2 evolutionary model is presented in Algorithm 1 and its main feature is that it effectively integrates three different MOO search space exploration paradigms.

Firstly, P, one of the two equally-sized coevolved subpopulations in DECMO2 implements a SPEA2[10] evolutionary model that is based on *environmental selection* (notation E_{sel}) — a selection for survival mechanism based on Pareto dominance as a primary metric and a crowding distance in objective space as a secondary metric. Apart from E_{sel}, this evolutionary paradigm also relies on the simulated binary crossover (SBX)[11] and polynomial mutation (PM)[12] genetic operators. It is noteworthy that inside E_{sel}, DECMO2 uses a slightly

Algorithm 1 The DECMO2 evolutionary model

function DECMO2($problem$, a_{size}, $maxGen$)
  $P, Q \leftarrow \emptyset$
  $\langle p_{size}, q_{size}, e_{size} \rangle \leftarrow$ **COMPUTESIZES**(a_{size})
  $A \leftarrow$ **INITIALIZEARCHIVE**($problem$, a_{size})
  $i \leftarrow 1$
  while $i \leq a_{size}$ **do**
    $\vec{x} \leftarrow$ **CREATEINDIVIDUAL**($problem$)
    $A \leftarrow$ **INSERTINTOARCHIVE**($A, \vec{x}$)
    if $i \leq p_{size}$ **then**
      $P \leftarrow P \cup \{\vec{x}\}$
    else
      if $i \leq (p_{size} + q_{size})$ and $i > p_{size}$ **then**
        $Q \leftarrow Q \cup \{\vec{x}\}$
      end if
    end if
    $i \leftarrow i + 1$
  end while
  $\phi^P, \phi^Q, \phi^A, t \leftarrow 1$
  while $t \leq maxGen$ **do**
    $p_{bonus}, q_{bonus}, \leftarrow 0$
    $a_{bonus} \leftarrow e_{size}$
    if $t \in \{2k+1 : k \in \mathbb{Z}\}$ **then**
      $p_{bonus}, q_{bonus}, a_{bonus} \leftarrow 0$
      if $\phi^P > \phi^Q$ **and** $\phi^P > \phi^A$ **then**
        $p_{bonus} \leftarrow e_{size} \wedge a_{bonus} \leftarrow 0$
      end if
      if $\phi^Q > \phi^P$ **and** $\phi^Q > \phi^A$ **then**
        $q_{bonus} \leftarrow e_{size} \wedge a_{bonus} \leftarrow 0$
      end if
    end if
    $\langle P, \phi^P \rangle \leftarrow$ **EVOGENSPEA2**(P, $p_{size} + p_{bonus}$)
    $\langle Q, \phi^Q \rangle \leftarrow$ **EVOGENDE**(Q, $q_{size} + q_{bonus}$)
    $\phi^A \leftarrow$ **EVODIRARCHIVEIND**(A, a_{bonus})
    $E \leftarrow E_{sel}(P \cup Q \cup A, e_{size})$
    $P \leftarrow E_{sel}(P \cup E, p_{size})$
    $Q \leftarrow E_{sel}(Q \cup E, q_{size})$
    $t \leftarrow t + 1$
  end while
  return $E_{sel}(P \cup Q \cup A, a_{size})$
end function

modified version of environmental selection that filters objective-wise duplicates from the returned solution set.

Secondly, Q – the other coevolved subpopulation – implements a GDE3-like[13] search behaviour that focuses on exploiting the very good performance of the differential evolution paradigm[14] on continuous optimization problems.

Thirdly, the last MOO paradigm incorporated in DECMO2 comes in the form of an archive of well-spaced elite solutions, A, that is maintained according to a decomposition-based principle similar to the one popularized by MOEA/D-DE.[15] Even though at certain times a limited number of new individuals is evolved directly from the archive, the main purpose of A is to preserve an accurate approximation of the Pareto front.

DECMO2 is also designed to dynamically pivot towards the evolutionary paradigm that was more successful during the latest stage of the run by allowing the part of the algorithm that implements this paradigm to generate a total of $e_{size} = \frac{2}{9}|P|$ individuals more than usual. This means that during the run, a performance bonus is awarded based on perceived current space-exploration performance.

4 Experimental setup

4.1 DECMO2 parameterization

For all the numerical experiments that we report on, we used a total population size of a_{size} =400 for DECMO2 and the standard (literature recommended) parameter settings for the coevolved subpopulations of the solver. Thus, for subpopulation P of size 180, we used a value of 0.9 for the crossover probability and 20 for the crossover distribution index of the SBX operator and a value of $1/|\vec{x}|$ for the mutation probability and 20 for the mutation distribution index of the PM operator. Subpopulation Q was evolved according to a DE/rand/1/bin strategy[14] in which the control parameter F was set at 0.5 and the crossover factor CR was set at 0.3. Spacing inside the archive A was maintained via a weighted Tschebyscheff distance measure.

We evaluated 100.000 solution candidates during each optimization run (i.e., $maxGen = 250$) and we report on the best result out of 3 repeats for each numerical experiment. Since the secondary objective of our MOOP is an artificial placeholder that has no practical importance when assessing the overall result of the optimization, we always only report the best discovered solution with regard to $o_1(\vec{x})$.

4.2 ESTP benchmark and academic test cases

In order to demonstrate the ability of our approach, we compared the results obtained by DECMO2 on 15 problems from a benchmark ESTP set[16] with those of two reference solvers: one based on a geometrically motivated heuristic[17] and another one that uses artificial neural networks.[18]

For initial insight on how our method performs on optimization scenarios that are more representative for the ascent assembly domain, we applied DECMO2 on 4 academic test cases: the one illustrated at the beginning of this paper (A1) in Section 2.1 and three more derivations (A2, A3, and A4) based on the more challenging access point placement from problem no. 12 of the ESTP benchmark set. On all these tests we used the setting $z = 0$ to parameterize the cost function from Equation (2) and thus optimize for the minimum Euclidean distance.

4.3 Industrial test case

The most realistic cost-optimal CAA-Structure design scenario we investigated was proposed by Liebherr-Werk Nenzing GmbH (LWN)[19] – a manufacturer of a wide range of products including various types of cranes. More specifically, we investigated cost-optimal CAA-Structures that allow access to user-specified regions of interest

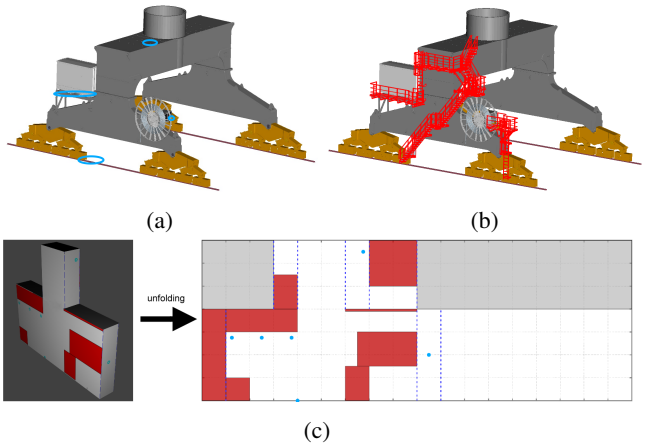


Figure 3: A CAD model of the gantry of a Liebherr mobile harbour crane with highlighted user-specified access points (a) expert-designed ascent assembly solution (b) and complementary 3D and unfolded 2D model abstraction (c).

on the gantry of a mobile harbour crane (please see Figure 3a). Figure 3b presents an expert-designed CAA-Structure attached to the gantry and we aim to use the unfolding-based 2D model abstraction from Figure 3c (obtained by vertically stacking two cuboids) to explore complementary optimal designs that might provide interesting insights to LWN.

In order to provide a balance between innovative and realistic cost-optimal solutions for CAA-Structures, we considered two test case variations (TC1 and TC2) in which the ground access point is placed at different positions along the horizontal axis and four different cost settings:

- C1 — The first cost setting uses a value of $z = 0$ to parameterize the cost function from Equation (2). Given the infinite degrees of freedom, optimal designs discovered for this setting are expected to have the smallest total path network (Euclidean) distance and can be used as a generic structural reference when assessing more constrained cost-optimal designs.

- C2 — The second cost setting uses a value of $z = 4$ and a list of preferred design angles $U = \{0, 45, 90\}$ and aims to deliver ascent assembly designs that only use the three standard assembly components: horizontal platforms, stairs and vertical ladders.

- C3 — The third cost setting uses a softer angle-wise constraint factor of $z = 1$ and a minimal list of preferred design angles $U = \{0, 90\}$ and aims to deliver ascent assembly designs that have a real-life minimal cost as they only use horizontal platforms and vertical ladders.

- C4 — The fourth cost setting uses a value of $z = 4$ and a minimal list of preferred design angles $U = \{0, 30, 45\}$ and aims to deliver CAA-Structures that offer a higher degree of access (no mandatory use of hands) by only using platform modules and two types of staircase modules (mild and regular inclination).

5 Results

5.1 *Performance on artificial problems*

The comparative performance on benchmark ESTPs is presented in Table 1 and indicates that our solving strategy based on DECMO2 and a MOOP formulation is very competitive for ESTP instances that have a low-to-medium number of definition (access) points.

Table 1: Comparative performance of DECMO2 on ESTPs. Best results are highlighted and * marks problems with an unknown optimum.

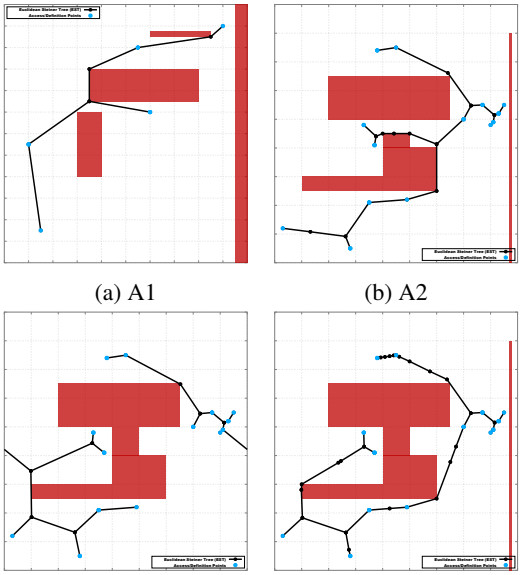
Prob. Id.[16]	n	Minimum Euclidean Steiner Tree			
		Baseline[16]	Ref.[17]	Ref.[18]	DECMO2
1	5	**1.6644**	**1.6644**	**1.6644**	**1.6644**
2B	8	**2.1387**	**2.1387**	2.1393	**2.1387**
2D	12	2.2223	**2.1842**	2.2979	**2.1842**
2G*	7	1.5878	1.6018	1.7019	**1.5594**
3	6	1.6472	**1.5988**	1.6553	**1.5988**
6	9	**1.2733**	1.2862	1.3024	**1.2733**
11*	64	3.8513	3.8380	3.9707	**3.8274**
12*	14	1.7222	1.7222	1.7989	**1.7067**
15A	5	**0.5130**	**0.5130**	0.5236	**0.5130**
18*	12	1.0332	1.0421	1.0782	**1.0241**
19B*	19	2.8567	2.8408	2.9689	**2.8286**
26*	20	**1.9767**	2.2770	1.9785	1.9785
28*	16	2.3671	2.3446	2.4048	**2.3309**
29*	17	2.1974	2.1974	2.2076	**2.1869**
31*	16	1.4220	1.3999	1.4343	**1.3660**

The results for the four academic CAA-Structure design scenarios are presented in Figure 4 and they indicate that the DECMO2-based solving strategy is quite general and able to both efficiently avoid obstacles as well as profit from the left edge - right edge continuity.

5.2 *Results for the LWN industrial test case*

The best solutions discovered for the two test case variants of the Liebherr mobile harbour crane scenario when considering the four different cost settings are plotted in Figures 5 and 6.

A visual inspection of the results for each test case reveals that imposing angle-wise restrictions on the overall design of the ascent assembly can be successfully accommodated by the DECMO2-based optimization strategy. Furthermore, while angle-wise restrictions do influence the optimization outcome, the generic (star-shaped) structure that characterizes the expert CAA-Structure design is confirmed by all the cost-optimal results obtained for this somewhat simplistic test case.



(a) A1 (b) A2

(c) A3 (d) A4

Figure 4: Results for the four academic test cases.

As specific observations related to the discovered optimal CAA-Structure designs, it is noteworthy that:

- The less restrictive setting $z = 1$ in C3 can results in designs that apart from platforms and ladder segments also feature ramps as shown in Figure 5c.
- The cost setting C4 seems to result in designs (e.g., Figure 6d) that resemble more closely the expert-designed CAA-Structure illustrated in Figure 3b. This indicates that accounting for ease-of-access concerns should be enforced in future extensions of the model abstraction.
- Shifting the bottom access point along the horizontal axis induces a small and local effect when unlimited degrees of freedom are allowed (Figures 5a and 6a) but the effect on the overall design of assembly is larger and global when considering angle-wise restrictions. This indicates that using a continuous problem formulation and a limited set of domain-dependent restrictions can yield innovative optimal design suggestions that an expert might overlook.

6 General conclusions and outlook

In the present work we have introduced an initial, practical model abstraction for the task of automating the *cost-optimal design of complete ascent assembly structures*. In order to tackle in a domain-realistic manner the *2D Path network layout problem* that emerges from the aforementioned model abstraction, we propose an optimization procedure based on a multi-objective problem formulation and an advanced coevolution-based solver – DECMO2. As results obtained on benchmark and academic test cases were very encouraging, we also applied our approach on a real-life CAA-Structure design scenario provided by an industrial partner.

The results for the real-life CAA-Structure optimization scenario also empirically support the validity of our approach. Thus, by employing appropriate cost functions, formalizing the CAA-Structure optimization problem on a continuous design space facilitates the discovery of a wide range of innovative designs that provide design engineers with valuable insight regarding the trade-offs between the best theoretical CAA-Structure design (that requires infinite degrees of freedom and new types of assembly modules) and the best practical CAA-Structure design that only requires traditionally used ascent assembly modules.

In the future we plan to investigate the hybridization potential between our current approach and a complementary design strategy[20] that is based on a discretization of the design surface and that delivers competitive results on CAA-Structure optimization scenarios with strong angle-wise restrictions.

Since the search logic of our DECMO2-based solving strategy is very loosely bound to the 2D model abstraction, future work will also revolve around solving the cost-optimal design problems directly in 3D space. The reason is that the simple 2D representation is rather restrictive for several real world applications. For example, it is not possible to model the crane from Figure 1a with it.

Acknowledgment

This work was supported by the K-Project "Advanced Engineering Design Automation" (AEDA) that is financed under the COMET funding scheme of the Austrian Research Promotion Agency.

References

[1] Frank, G., Entner, D., Prante, T., Khachatouri, V., and Schwarz, M. Towards a generic framework of engineering design automation for creating complex CAD models. *International Journal on Advances in Systems and Measurements* **7**(1), 179–192 (2014).

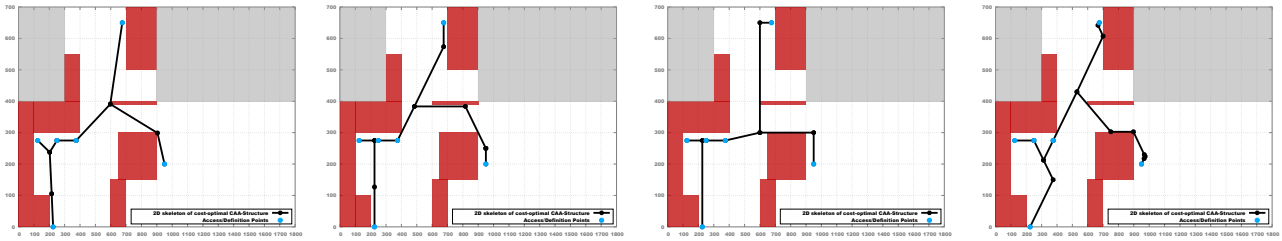
[2] Gilbert, E. and Pollak, H. Steiner minimal trees. *SIAM Journal on Applied Mathematics* **16**(1), 1–29 (1968).

[3] Garey, M. R., Graham, R. L., and Johnson, D. S. The complexity of computing Steiner minimal trees. *SIAM journal on applied mathematics* **32**(4), 835–859 (1977).

[4] De Jong, K. A. *Evolutionary computation: a unified approach*. MIT press, (2006).

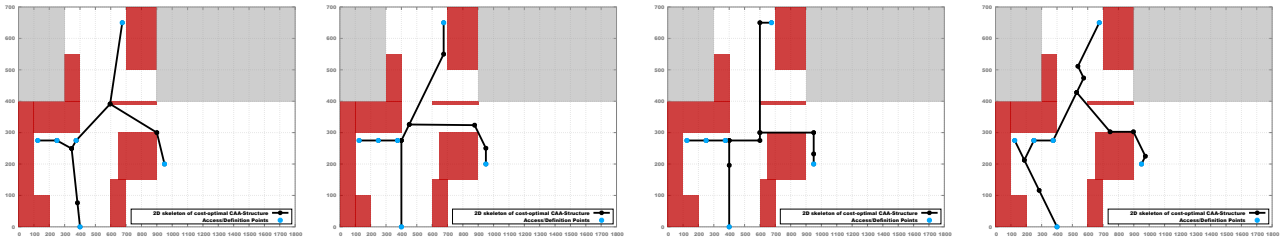
[5] Langdon, W. B. and Poli, R. Fitness causes bloat. In Soft Computing in Engineering Design and Manufacturing, Chawdhry, P. et al., editors, 13–22. Springer (1998).

[6] Prim, R. C. Shortest connection networks and some generalizations. *Bell Labs Technical Journal* **36**(6), 1389–1401 (1957).



(a) TC1 – C1 cost setting (b) TC1 – C2 cost setting (c) TC1 – C3 cost setting (d) TC1 – C4 cost setting

Figure 5: CAA-Structure optimization results for the LWN TC1 optimization scenario using different cost functions.



(a) TC2 – C1 cost setting (b) TC2 – C2 cost setting (c) TC2 – C3 cost setting (d) TC2 – C4 cost setting

Figure 6: CAA-Structure optimization results for the LWN TC2 optimization scenario using different cost functions.

[7] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* **6**(2), 182–197 (2002).

[8] Zăvoianu, A.-C., Lughofer, E., Bramerdorfer, G., Amrhein, W., and Klement, E. P. DECMO2: a robust hybrid and adaptive multi-objective evolutionary algorithm. *Soft Computing* **19**(12), 3551–3569 (2014).

[9] Zăvoianu, A.-C., Lughofer, E., Amrhein, W., and Klement, E. P. Efficient multi-objective optimization using 2-population cooperative coevolution. In *Computer Aided Systems Theory - EUROCAST 2013*, Lecture Notes in Computer Science, 251–258. Springer Berlin Heidelberg, (2013).

[10] Zitzler, E., Laumanns, M., and Thiele, L. SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization. In *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001)*, 95–100. International Center for Numerical Methods in Engineering (CIMNE), (2002).

[11] Deb, K. and Agrawal, R. B. Simulated binary crossover for continuous search space. *Complex Systems* **9**, 115–148 (1995).

[12] Deb, K. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, (2001).

[13] Kukkonen, S. and Lampinen, J. GDE3: The third evolution step of generalized differential evolution. In *IEEE Congress on Evolutionary Computation (CEC 2005)*, 443–450. IEEE Press, (2005).

[14] Storn, R. and Price, K. V. Differential evolution - a simple and effcient heuristic for global optimization over continuous spaces. *Journal of Global Optimization* **11**(4), 341–359 December (1997).

[15] Zhang, Q., Liu, W., and Li, H. The performance of a new version of MOEA/D on CEC09 unconstrained MOP test instances. Technical report, School of CS & EE, University of Essex, February (2009).

[16] Soukup, J. and Chow, W. Set of test problems for the minimum length connection networks. *ACM SIGMAP Bulletin* **15**, 48–51 (1973).

[17] Beasley, J. E. A heuristic for euclidean and rectilinear Steiner problems. *European Journal of Operational Research* **58**(2), 284–292 (1992).

[18] Bhaumik, B. A neural network for the Steiner minimal tree problem. *Biological Cybernetics* **70**(5), 485–494 (1994).

[19] LWN. Liebherr-Werk Nenzing GmbH. `http://www.liebherr.com/en-GB/35267.wfw`, (2017). Accessed: 2017-03-06.

[20] Hellwig, M., Entner, D., Prante, T., Zăvoianu, A.-C., Schwarz, M., and Fink, K. On the optimization of ascent assembly design based on a combinatorial problem representation. In *EUROGEN 2017, Sep. 13-15, Madrid, Spain*, (2017).