Towards Bundling Minimal Trees in Polygonal Maps

Victor Parque
Waseda University
3-4-1 Okubo, Shinjuku-ku, Tokyo, Japan
parque@aoni.waseda.jp
Egypt-Japan University of Science and Technology
Borg Al Arab, Alexandria, Egypt
victor.parque@ejust.edu.eg

Tomoyuki Miyashita
Waseda University
3-4-1 Okubo, Shinjuku-ku, Tokyo, Japan
tomo.miyashita@waseda.jp

ABSTRACT

Minimal trees in polygonal maps aim at minimizing the connectivity in a network while avoiding obstacle collision. Being closely related to the Steiner Tree Problem, yet with a different scope, minimal trees aim at connecting origin-destination pairs, given in a bipartite network, to allow the joint transport of information, goods, resources and people. In this paper, we propose a method to tackle the bundling problem of minimal trees in modular bipartite networks by using a two-layer optimization based on Differential Evolution with a convex representation of coordinates. Our computational experiments in polygonal domains considering both convex and non-convex geometry show the feasibility and the efficiency of the proposed approach.

CCS CONCEPTS

• **Computing methodologies → Continuous space search**; **Motion path planning**; **Discrete space search**;

KEYWORDS

Modular Bipartite Networks, Modular Minimal Tree

ACM Reference Format:
Victor Parque and Tomoyuki Miyashita. 2018. Towards Bundling Minimal Trees in Polygonal Maps. In *GECCO '18 Companion: Genetic and Evolutionary Computation Conference Companion, July 15–19, 2018, Kyoto, Japan,* Jennifer B. Sartor, Theo D'Hondt, and Wolfgang De Meuter (Eds.). ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3205651.3208316

1 INTRODUCTION

Being ubiquitous in scenarios in which the means for transport and communication is scarce, and the environment is either hard o impossible to navigate, minimal trees aim at compounding paths in a network in order to achieve the minimal-length connectivity and thus efficient transport and communication in a cluttered environment. As such, the result is a tree minimizing the inter-connectivity among source and destination nodes.

The research on minimal trees has its origins in the 30's[1], with the *Steiner tree* problem, which was popularized in the 40's[2]. Also, in VLSI systems, the Obstacle-Avoiding Rectilinear Steiner (OARST) given n nodes in a polygonal map has received attention[3–7]. Also, the Obstacle-Avoiding Steiner Tree (OAST) has an approximation scheme running in polynomial-time with $O(nlog^2n)$, for n bring the number of terminals and obstacle vertices[8]. However, the exhaustive study of global optimization and gradient-free approaches on Minimum Steiner Trees in polygonal maps has received little attention.

Related studies on minimal trees are also studied in wireless networks [9–14], and network visualization [15–20]. In line of the above, and in more particular domains, the closest developments to minimal trees is related to edge bundling problem in network visualization field. Here, the conventional works have focused on clustering the geometry of edges[15], the edge bundling based on force attraction[15, 16], the attraction to the skeletonization[17], and the optimization of the centroid points of close edges[18].

Indeed, also path planning is related to the computation of minimal trees. Here, Dijsktra [21] and A* [22], along with their extensions are well-known. It has been argued that path planning in triangulated space is highly accurate and efficient. As such, for a map with polygonal obstacles having$n$ vertices, the Delaunay Triangulation ca be computed to render a connected graph with $O(n)$ nodes, each of which represents the triangles conforming the free-space. Then, by using the Delaunay-based connected graph, path planning can be efficiently performed by graph search methods on the adjacency matrix of the Delaunay-based connected graph. In particular, If one uses the Dijkstra-based search[21, 23], time complexity ca be achieved by $O(n.log(n))$, yet it is possible to use A* search[22, 24] along with a funneling algorithm[25, 26] to obtain a quasi-linear time path planning.

Other related works are the path bundling in non-modular bipartite networks[27–29], and the minimal trees in n-star networks with fixed roots[30]. However, path planning considering minimal Steiner trees and flexible root configuration has received little examination in the literature.

However, due to having a different scope, the above algorithms have rendered compounded networks which are aesthetic and compact, but not necessarily optimal in the sense of minimizing a global connectivity metric among nodes. In this paper, we consider modular bipartite networks and compute minimal trees on polygonal maps. The basic idea of our approach consists in compounding edges in modular bipartite networks by a two-layer optimization, in which we first compute the optimal roots of path bundles for

GECCO '18 Companion, July 15–19, 2018, Kyoto, Japan
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5764-7/18/07…$15.00
https://doi.org/10.1145/3205651.3208316

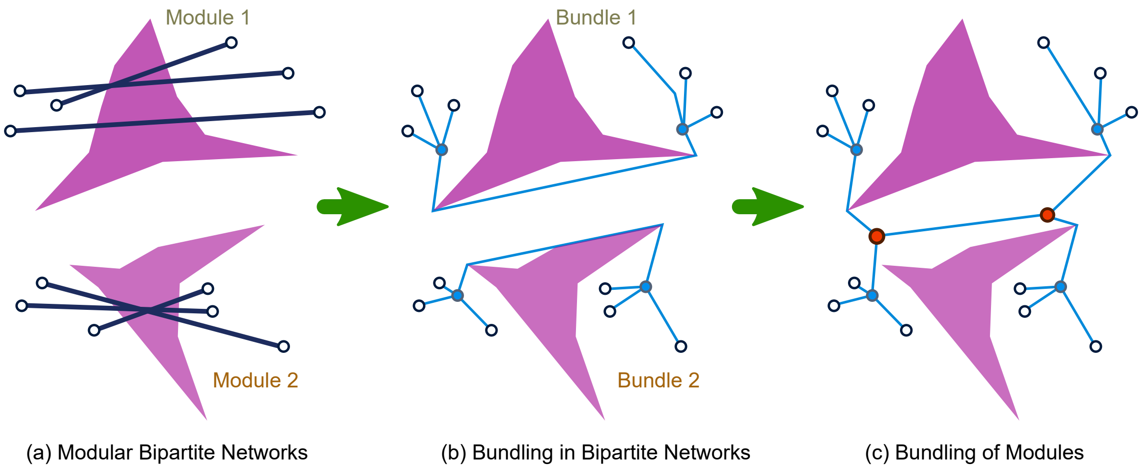

Module 1
Module 2
Bundle 1
Bundle 2
(a) Modular Bipartite Networks
(b) Bundling in Bipartite Networks
(c) Bundling of Modules

Figure 1: Basic Idea of Modular Minimal Trees

each module of the bipartite network, and then compute the optimal roots of path bundles across modules. Our contributions are as follows:

- We propose a method to tackle the bundling problem of minimal trees in modular bipartite networks by using a two-layer optimization based on Differential Evolution with a convex representation.
- Computational experiments consisting of 1250 problem scenarios, and 2.5 million instances of functions evaluations show the (1) the feasibility to compute minimal trees in modular bipartite networks, and (2) convergence within [400, 800] function evaluations.

In the rest of this article, after describing the key components in our proposed approach, we discuss our findings though our computational experiments, and finally summarize our insights and future work.

2 COMPUTING MODULAR PATH BUNDLES

2.1 Basic Idea

The basic concept of minimal trees in modular bipartite networks is depicted by Fig. 1 and Algorithm 1, in which the input is a polygonal map and a bipartite network, and the output is a tree structure with paths avoiding obstacles and minimizing the total distance metric. The following is assumed as inputs:

- A modular bipartite graph $G = (V, E)$ wherein edge $e \in E$ represents the needs in communication/transportation between origin-destination pairs, in which the edges in the set E are grouped in M modules, following a cluster-like configuration such that $E = E_1 \cup E_2 \cup \ldots E_m \ldots \cup E_M$, and $V = V_1 \cup V_2 \cup \ldots V_m \ldots \cup V_M$. An example of a modular bipartite network is depicted by Fig. 1-(a), in which the bipartite network has 6 edges, grouped into two modules, namely Module 1 and Module 2, each of which has distinct edges and vertices.
- A polygonal map configuration $\rho = \{\rho_1, \ldots, \rho_s\}$, in which a-priori knowledge of obstacles $\rho \in M$ define unfeasible areas for navigation/transportation. An example of a polygonal configuration is shown by Fig. 1-(a), in which the map involves the presence of two polygons $\rho = \{\rho_1, \rho_2\}$. In this paper, we assume polygons are in 2D.

2.2 Bundle Representation

In order to represent path bundles of modular bipartite networks, we use the tuple x_m to encode the root for each module ($m = 1, 2, \ldots, M$), and the tuple x_o to encode the root across modules. In the following paragraphs we describe the key encoding mechanisms.

2.2.1 Representation for each Module. We represent minimal trees by using the roots of a tree encoded in a triangular (convex) space; thus the bundle of the m-th module of the bipartite network is represented by the following tuple:

$$x_m = \Big(\underbrace{(i_P^m, \alpha_P^m, \beta_P^m)}_{x_m^P}, \underbrace{(i_Q^m, \alpha_Q^m, \beta_Q^m)}_{x_m^Q} \Big) \tag{1}$$

, where i_P^m is the integer number related to the i-th triangle of the m-th module related to the coordinate P, and $\alpha_P^m, \beta_P^m, \alpha_Q^m, \beta_Q^m \in [0, 1]$ are Real numbers.

The basic idea of the above implies representing coordinates in a triangular encoding, in which in order to encode a coordinate, the following tuple is used:

$$P = (i, \alpha, \beta), \tag{2}$$

$$(P_x, P_y) = (1 - \alpha)A_i + \sqrt{\alpha}(1 - \beta)B_i + \sqrt{\alpha}\beta C_i, \tag{3}$$

where P is the coordinate in triangular space, for $i \in [|T|]$, $[a] = \{1, 2, \ldots, a\}$ and $T = \{t_1, t_2, \ldots, t_i, \ldots, t_n\}$ is the set of Delaunay triangles of the free space of the polygonal map ρ, $\alpha, \beta \in [0, 1]$ are arbitrary constants, P_x and P_y is the horizontal and vertical cartesian coordinates of point P, and A_i, B_i, C_i are the coordinates of the vertices of the i-th triangle $t_i \in T$. The above encoding enables to represent coordinates which guarantee to be inside the free

Algorithm 1 Bundling of Modular Bipartite Networks

```
1: procedure BUNDLE($G$)
2:     Input $G$                                              ▷ Modular Bipartite Graph
3:     Output $(x_o, x_1, x_2, ..., x_m, ..., x_M)$   ▷ Roots of Path Bundle
4:     for $m \leftarrow 1$ to $M$ do                     ▷ $M$ Minimization Problems
5:         $x_m \leftarrow$ Minimize $F_m(x)$ with $x \in \mathbf{T}_m$
6:     end for
7:     $x_o \leftarrow$ Minimize $F_o(x)$ with $x \in \mathbf{T}_o$
8: end procedure
```

space, and which is meritorious to encode obstacle-free coordinates for path planning problems.

Furthermore, the following relation holds: $i_P^m, i_Q^m \in [|T_m|]$, for the set T_m being computed as follows:

$$T_m = DT\Big(\mathbb{C}_{V_m} - \{\rho_1 \cup \rho_2 \ldots \cup \rho_s\}\Big) \tag{4}$$

, where $DT(.)$ represents the Delaunay Triangulation, the geometry $\mathbb{C}_{V_m}$ encodes the Convex Hull of the union of all vertices in the m-th module of the bipartite network G, and $\rho_1 \cup \rho_2 \ldots \cup \rho_s$ represents the union of obstacles. Basically, T_m encodes the partition, in terms of a convex set, of the free space which is most relevant to the m-th module of the bipartite network. Naturally, $x_m \in \mathbf{T}_m$, for $m = 1, 2, ..., M$.

2.2.2 Representation across Modules. We represent bundles by considering the grouping of modules with the following:

$$x_o = \Big(\underbrace{(i_P^o, \alpha_P^o, \beta_P^o)}_{x_o^P}, \underbrace{(i_Q^o, \alpha_Q^o, \beta_Q^o)}_{x_o^Q}\Big) \tag{5}$$

, in which the following holds: $i_P^o, i_Q^o \in [|T_o|]$, and the set T_o is computed as follows:

$$T_o = DT\Big(\mathbb{C}_{x_1 \cup x_2 \cup \ldots \cup x_m} - \rho_1 \cup \rho_2 \ldots \cup \rho_s\Big) \tag{6}$$

, where $DT(.)$ represents the Delaunay Triangulation operator, $\mathbb{C}_{x_1 \cup x_2 \cup \ldots \cup x_m}$ represents the Convex Hull of the roots of m modular bundles in the bipartite network G. In line of the above, the following holds: $x_o \in \mathbf{T}_o$.

The basic idea in the above representation is to encode the potential locations of the roots x_o by triangulating the relevant free search space (which is derived by geometric operations). Furthermore, by computing the triangulation sets $\mathbf{T}_m$ and $\mathbf{T}_o$, we aim at constructing the focalized search spaces of roots of modular bundles, which is then used by optimization algorithms.

2.3 Optimization Problems

The basic idea of computing path bundles in modular bipartite networks is depicted by two-level optimization, each of which computes the root x of bundled paths *for each* and *across* modules.

The basic idea to solve the two-level optimization is portrayed by Algorithm 1, and in the following we explain its dynamics.

2.3.1 Optimization in Modules. Here, the optimization aims at computing the roots x_m of path bundles for each module $m \in [M]$, as follows:

$$\begin{aligned} \underset{x}{\textbf{Minimize}} \quad & F_m(x) \\ \text{subject to} \quad & x \in \mathbf{T}_m \end{aligned} \tag{7}$$

, where $x \in \mathbf{T}_m$ implies the feasible search space for the tuple x. Following the definition of Eq. (1), the lower bound of x is $(1, 0, 0, 1, 0, 0)$, and the upper bound of x is $(|T_m|, 1, 1, |T_m|, 1, 1)$ for $m \in [M]$.

The objective function F_m measures the length of the route bundle for each module, as follows:

$$F_m(x) = \sum_{e \in E_m} d(e_S, x_m^P) + d(x_m^P, x_m^Q) + \sum_{e \in E_m} d(e_F, x_m^Q) \tag{8}$$

where, $d(.,.)$ is the Euclidean obstacle-free shortest distance between two points, e_S and e_F are coordinates of the *origin* and *destination* of the edge $e \in E_m$ in the m-th module, and x_m^P and x_m^Q are roots of bundles for each module.

2.3.2 Optimization across Modules. Here, the optimization aims at computing the location of roots x_o of path bundles *across modules*, as follows:

$$\begin{aligned} \underset{x}{\textbf{Minimize}} \quad & F_o(x) \\ \text{subject to} \quad & x \in \mathbf{T}_o \end{aligned} \tag{9}$$

In line of the above, here, the lower bound of x is $(1, 0, 0, 1, 0, 0)$, and the upper bound of x_o is $(|T_o|, 1, 1, |T_o|, 1, 1)$ for $m \in [M]$.

Also, here, the objective function measures the length of the route bundle across modules, as follows:

$$F_o(x) = \sum_{m \in [M]} d(x_m^P, x_o^P) + d(x_o^P, x_o^Q) + \sum_{m \in [M]} d(x_m^Q, x_o^Q) \tag{10}$$

where, $d(.,.)$ is the Euclidean obstacle-free shortest distance between two points, x_o^P and x_o^Q are roots of bundles across modules.

2.4 Differential Evolution

This subsection describes the optimization algorithm used to compute the optimal route bundles.

We use *Differential Evolution with Neighborhood and Convex Encoding* (DENC)[28] both global-local interpolation vectors, as well as convex representation of the geometric search space. Sampling is based on the following:

$$x_{t+1} = \begin{cases} u_t & F(u_t) \le F(x_t) \\ x_t & \text{otherwise} \end{cases} \tag{11}$$

where,

- x_t represents x_m or x_o at iteration t, in other words x_t is the *individual* (route bundle),
- $F(.)$ is the objective function, which is either F_m or F_o (*denoting minimization*), and
- u_t is the *trial* route bundle solution at iteration t.

In the above, the representation of variable x is convex, in which coordinates are in triangular space, as follows:

$$x_t \equiv \Big(\underbrace{(i_P, \alpha_P, \beta_P)}_{\textit{Coordinate P}}, \underbrace{(i_Q, \alpha_Q, \beta_Q)}_{\textit{Coordinate Q}} \Big) \tag{12}$$

, where *Coordinate P* and *Coordinate Q* imply Euclidean coordinates in the plane. The definitions of i, α, β follow the same principles of Eq. (1) and Eq. (5). Also, the conversion to the cartesian coordinates is straightforward due to the definition of Eq. 3.

The trial vector u_t is computed from global-local interpolation vectors, as follows:

$$u_t = x_t^c + b_t \circ (v_t - x_t^c) \tag{13}$$

$$b_t = [b_{t,1}, b_{t,2}, b_{t,3}, ..., b_{t,D}] \tag{14}$$

- $\circ$ is the *Hadamard* product (element-wise).
- x^c is the *crossover* individual at iteration t.
- v_t is the *mutant* individual at iteration t.
- b_t is a vector of masks containing zeros and ones.

The mask b_t is computed as follows:

$$b_{t,j} = \begin{cases} 1, & \mathbf{r}_{t,j} \leq CR \text{ or } j = jrand \\ 0, & \text{otherwise} \end{cases} \tag{15}$$

$$v_t = w^{x_t}.g_t + (1 - w^{x_t}).l_t \tag{16}$$

where,

- $\circ$ is the *Hadamard* product (element-wise).
- $\mathbf{r}_{t,j}$ and *jrand* are random numbers uniformly distributed in $\mathbb{R}^{[0,1]}$ and $\mathbb{N}^{[D]}$ respectively.
- *CR* is the probability of crossover.
- $D = 6$ is the dimensionality of the route bundling problem.

In the above, the reader may note that high crossover rates *CR* incite in large number of ones in the mask vector b_t, implying a highly explorative sampling.

The *global* and *local* interpolation vectors are computed by using neighborhood concepts, as follows:

$$g_t = x_t + \alpha(x_{gbest} - x_t) + \beta(x^1 - x^2) \tag{17}$$

$$l_t = x_t + \alpha(x_{nbest_x} - x_t) + \beta(x^p - x^q) \tag{18}$$

$$w^{x_t} = w^{x_t} + \alpha(w_{gbest} - w^{x_t}) + \beta(w^1 - w^2) \tag{19}$$

where,

- g_t is the *global* donor individual.
- l_t is the *local* donor individual.
- $\{x_t^1, x_t^2\} \subset \mathbf{P}$, are random individuals sampled from the population $\mathbf{P}$ for $x^1 \neq x^2 \neq x_t$.
- x_{gbest} is the *global best* in the population $\mathbf{P}$ at iteration t.
- x_{nbest_x} is the *local best* in the *neighborhood* $\mathbf{N}(x_t)$ of individual x_t at iteration t.
- the *neighborhood* $\mathbf{N}(x_t)$ of vector x_t is the set of individuals contiguous to x_t by radius $\delta = \frac{|\mathbf{P}|.\eta}{2}$ in a *ring topology*.
- $\{x^p, x^q\} \subset \mathbf{N}(x_t)$, are random individuals for $x^p \neq x^q \neq x_t$.
- w^{x_t} denotes the coefficient of individual x_t,
- w_{gbest} is the coefficient associated to x_{gbest},
- w^1, w^2 are the coefficients associated to the vectors x^1, x^2 respectively,and
- any coefficient $w^{x_t} \in U[0, 1]$ is set randomly at initial iteration.

The main reason of using *Differential Evolution* with *global* and *local* interpolation vectors is due to its advantage to balance both exploration and exploitation over the entire search space $x \in \mathbf{T}$, wherein the trade-off between the global and the local search is self-adapted throughout the iterations.

3 COMPUTATIONAL EXPERIMENTS

To evaluate the feasibility and efficiency of our proposed approach, we used polygonal domains with both convex and non-convex geometry, as well as distinct configurations of modular bipartite networks. In this section, we describe our experimental results and obtained insights.

3.1 Experimental Settings

Our computing environment was an Intel i7-4930K @ 3.4GHz with Windows 8.1, and our algorithms were implemented in Matlab 2018a. In order to enable a meaningful evaluation of our proposed approach, we considered the following environmental settings:

- No. of Modules in the bipartite graph = $M = \{2, 4, 6, 8, 10\}$
- No. of Edges in each module of bipartite graph, $|E_m| = \{5, 10, 15, 20, 25\}$,
- Number of Polygonal Obstacles: $\{1, 2, 3, 4, 5\}$,
- No. Sides in each Polygonal Obstacle: $\{5, 10\}$.
- 5 Independent runs of the above experimental setting, with the maximum number of functions evaluations being set between 2000 and 2500,
- In each independent experiment, the initial solutions of route bundles $x_m \in \mathbf{T}_m$ and $x_o \in \mathbf{T}_o$ are initialized randomly and independently.

As a result of the above, we used 1250 problem scenarios, and 2.5 million instances of functions evaluations.

As for parameters in Differential Evolution, we used:

- probability of crossover $CR = 0.5$,
- scaling factor $\alpha = \beta = |ln(U(0, 1))/2|$,
- population size with 10 individuals, and
- neighborhood ratio $\eta = 0.1$

The main reason of using crossover probability $CR = 0.5$ is to give the same importance to the sampling local-global interpolations and historical search directions. Also, the small values of scaling factors α, β as used in this paper allow to search in small steps when computing the self-adaptive directions. Furthermore, small values of population size and neighborhood factor η allow to sampling efficiently within the local polygonal neighborhood[28].

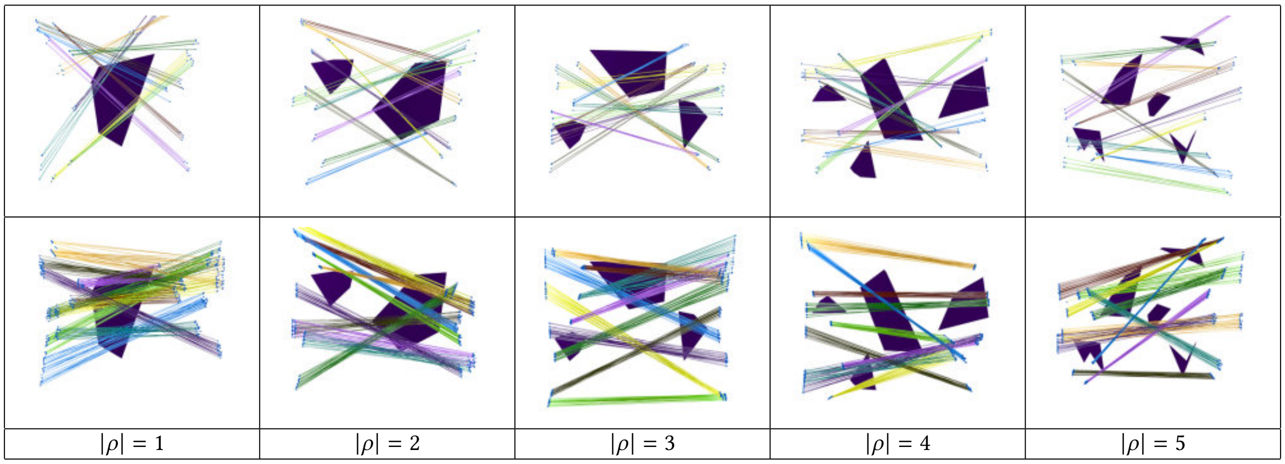


Figure 2: Polygonal Map with 5 sides and Bipartite Networks with $|\rho|$ modules, each with 5 (top) edges, 10 edges, 15 edges, 20 edges and 25 (bottom) edges.

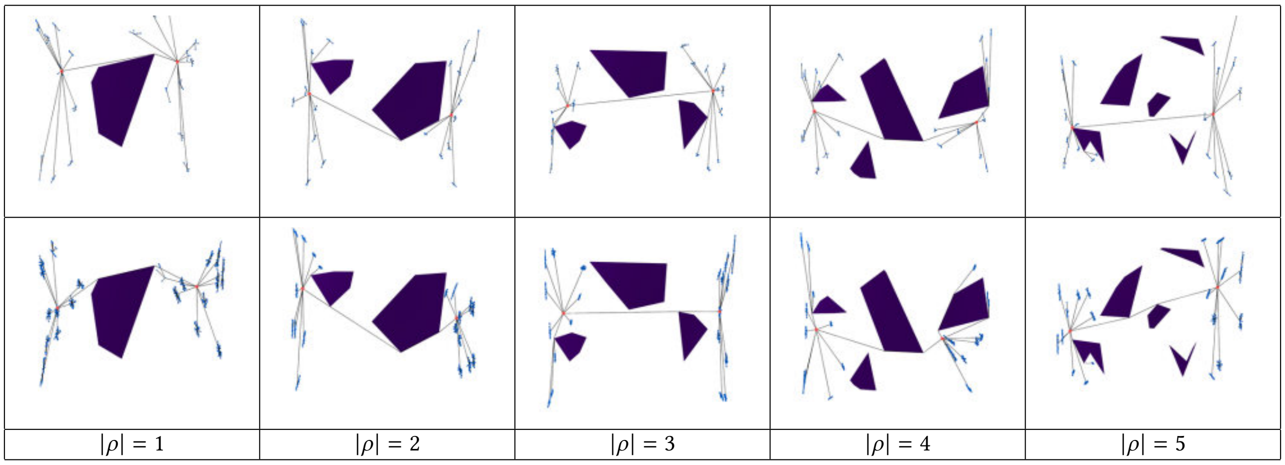


Figure 3: Polygonal Map with 5 sides and Bipartite Networks with $|\rho|$ modules, each with 5 (top) edges, 10 edges, 15 edges, 20 edges and 25 (bottom) edges.

Fine tuning of Differential Evolution Parameters is out of the scope of this paper.

The key motivation behind using the number of edges up to 25, and modules up to 10 is due to our interest in applications relating coordination and communication in indoor environments, in which the complexity of the environment is controlled by (1) the number of obstacles in the polygonal map, and/or (2) the number of sides for each obstacle. These two factors increase the search space, thus, it is relevant in our focus to evaluate the convergence speed through independent runs.

3.2 Results

To exemplify the kind of tree structures obtained by our proposed method, Fig. 2 - Fig. 5 show examples of modular bipartite networks and their optimized minimal trees. In regards to the obtained minimal trees, by observing the results being rendered in Fig. 2 to Fig. 5, we can confirm the following facts:

- Regardless of the configuration of the polygonal domain, it becomes possible to compute tree structures of minimal trees which aim at minimizing the total tree length of modular bipartite networks.
- The location of the anchoring points are not necessarily close to the centroid of origin-destination pairs.
- Regardless of increasing the edges for each module of the bipartite graph, a number of sub-trees overlap in geometry, which implies the need to increase the depth to compute anchoring points in each module.

To show the convergence speed of our algorithms, Fig. 6 - Fig. 10 show the convergence of the fitness function F_o, denoting the performance of the length across modules, for different number of obstacles in the map $|\rho|$. In regards to the convergence behaviour of the proposed algorithm, by observing Fig. 6 - Fig. 10, it is possible to confirm the following facts:

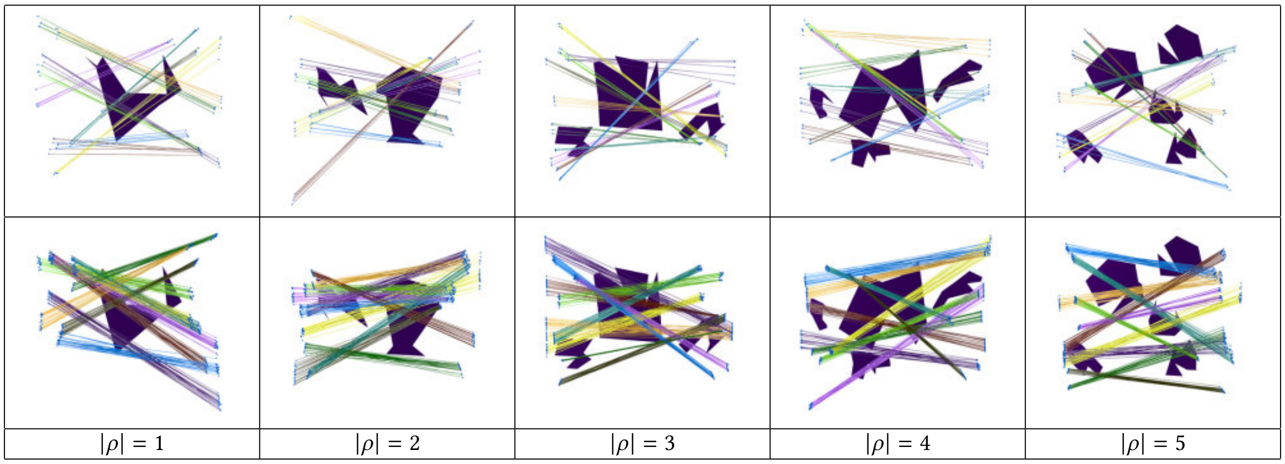


Figure 4: Polygonal Map with 5 sides and Bipartite Networks with $|\rho|$ modules, each with 5 (top) edges, 10 edges, 15 edges, 20 edges and 25 (bottom) edges.

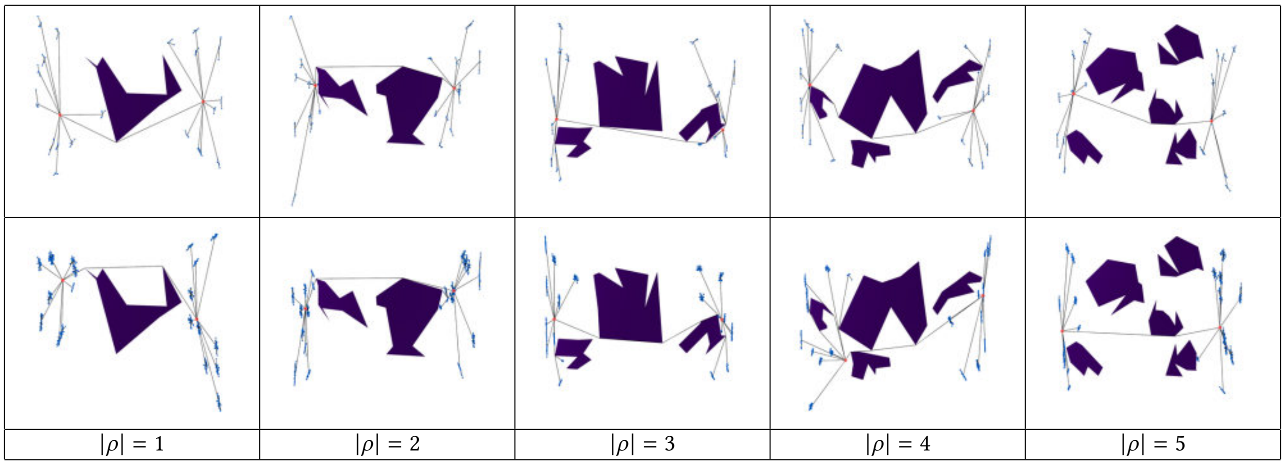


Figure 5: Polygonal Map with 5 sides and Bipartite Networks with $|\rho|$ modules, each with 5 (top) edges, 10 edges, 15 edges, 20 edges and 25 (bottom) edges.

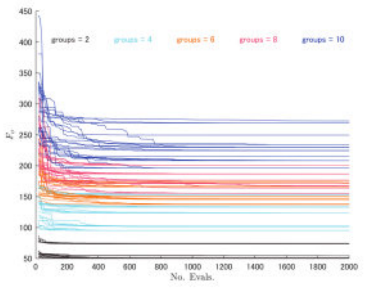
- Regardless of the configuration of the polygonal, it is possible to converge to the bundled paths which minimize the global distance metric within [400, 800] function evaluations.
- Increasing the number of edges has a direct effect on increasing the length of the minimal tree by some small factor smaller than 1.
- Increasing the number of modules of bipartite networks has a direct effect on increasing the length of the minimal tree by a factor smaller than 3 (upper bound).

We believe that the above observations are relevant to design higher-order minimal trees for modular networks. In line of this, we propose the following:

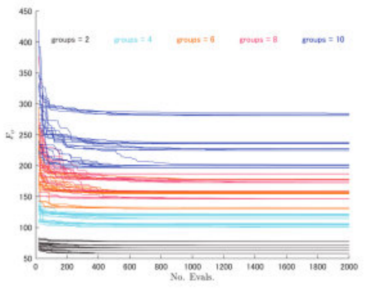
- Whenever modular bipartite networks or polygonal maps are expected to change, it may be possible to use pre-computed minimal trees as initial solutions further adaptation, since the location of roots is expected to be near.
- A hierarchical convex search space, as the one proposed in this paper, may be key for effective and efficient sampling of the search space for higher-order minimal trees.

The above insights implies the possibility to build more efficient algorithms for computing minimal trees in polygonal environments with both convex and non-convex obstacles. In our future agenda, we aim at performing further computational experiments using large number of edges and diverse obstacle configurations reminiscent of outdoor environments.

In our future work, we aim at investigating improved approaches to compute higher-order bundling of minimal trees, as well as improved learning algorithms. For instance, it may be possible to use canonical encodings in directed graphs[31] and undirected graphs[32] to explore diverse topologies of network connectivity;

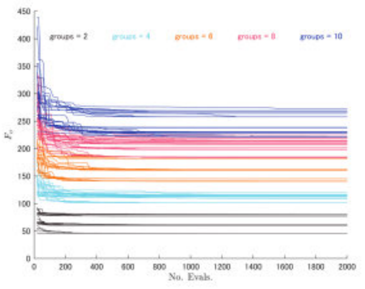


(a) Polygons with 5 sides

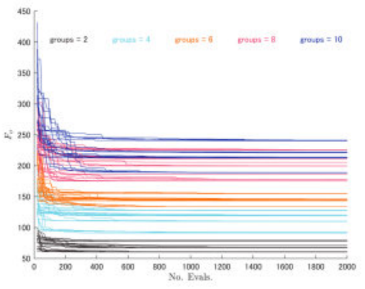


(b) Polygons with 10 sides

Figure 6: Convergence of fitness functions for $|\rho| = 1$
.



(a) Polygons with 5 sides



(b) Polygons with 10 sides

Figure 7: Convergence of fitness functions for $|\rho| = 2$
.

and it may be possible to use of concurrency concepts in exploration-exploitation[33, 34]. Furthermore, studying the formation of modules in hierarchical bundles by succinct subset partitions is in our agenda[35].

4 FINAL REMARKS

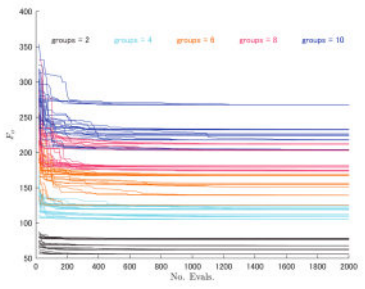
We have presented an approach to compute minimal trees by compounding edges in modular bipartite networks. Our computational experiments show the feasibility to obtain minimal trees in environments with non-convex obstacles. In future work, we aim at evaluating the generalization ability in environments reminiscent of outdoor configurations, and evaluate in problems involving communication, transportation and network design.

ACKNOWLEDGEMENT

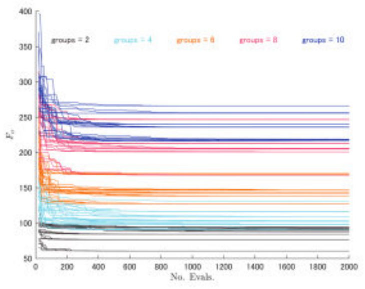
Funding this work by JSPS Kakenhi No. 15K18095 is appreciated.

REFERENCES

[1] J. Vojtěch and M. Kössler. *O minimálních grafech, obsahujících n daných bodů.* Časopis pro pěstování matematiky a fysiky 063.8, pp. 223-235, 1934.
[2] H. Robbins and R. Courant. *What is Mathematics?* Oxford University Press, 1941.
[3] H. Zhang, D. Ye, and W Guo. *A heuristic for constructing a rectilinear Steiner tree by reusing routing resources over obstacles.* INTEGRATION, the VLSI journal, Vol. 55, pp. 162-175, 2016.
[4] P. Winter, M. Zachariasen, and J. Nielsen. *Short trees in Polygons.* Discrete Applied Mathematics, Vol. 118, pp. 55-72, 2002.
[5] T. T. Jing, Y. Hu, Z. Feng, X. Hong, X Hu, and G. Yan. *A full-scale solution to the rectilinear obstacle-avoiding Steiner problem.* INTEGRATION, the VLSI journal, Vol. 41, pp. 413-425, 2008.
[6] P. Winter. *Euclidean Steiner minimal trees with obstacles and Steiner visibility graphs.* Discrete Applied Mathematics, Vol. 47, pp. 187-206, 1993.
[7] W. Chow, L. Li, E. Young, and C. Sham. *Obstacle-avoiding rectilinear Steiner tree construction in sequential and parallel approach.* INTEGRATION, the VLSI journal, Vol. 47, pp. 105-114, 2014.
[8] M. Müller-Hannemann, , and S. Tazari. *A near linear time approximation scheme for Steiner tree among obstacles in the plane.* Computational Geometry: Theory and Applications, Vol. 43, pp. 395-409, 2010.
[9] R. Falud. *Building Wireless Sensor Networks. O'Reilly Media*, Sebastapol, 4th edition, 2014.
[10] P. Wightman and M. Labardor. *A family of simple distributed minimum connected dominating set-based topology construc tion algorithms.* Journal of Network and Computer Applications, Vol. 34, pp. 1997 - 2010, 2011.
[11] J. A. Torkestani. *An energy-efficient topology construction algorithm for wireless sensor networks.* Computer Networks, Vol. 57, pp. 1714 - 1725, 2013.
[12] N. Panigrahi and P. M. Khilar. *An evolutionary based topological optimization strategy for consensus based clock synchronization protocols in wireless sensor network.* Swarm and Evolutionary Computation, Vol. 22, pp. 66 - 85, 2015.
[13] J. Szurley, A. Bertrand, and M. Moonen. *An evolutionary based topological optimization strategy for consensus based clock synchronization protocols in wireless sensor network.* Signal Processing, Vol. 117, pp. 44 - 60, 2015.
[14] S. P. Singh and S.C. Sharma. *A Survey on Cluster Based Routing Protocols in Wireless Sensor Networks.* Procedia Computer Science, Vol. 45, pp. 687 - 695, 2015.
[15] W. Cui, H. Zhou, P. C. Wong H. Qu, and X. Li. *Geometry-based edge clustering for graph visualization.* IEEE Transactions on Visualization and Computer Graphics, Vol. 14, pp. 1277 - 1284, 2008.
[16] D. Selassie, B. Heller, and J. Heer. *Divided Edge Bundling for Directional Network Data.* IEEE Transactions on Visualization and Computer Graphics, Vol. 17, No. 12, pp. 2354 - 2363, 2011.
[17] O. Ersoy, C. Hurther, F. Paulovich, G. Cabtareiro, and A. Telea. *Skeleton-Based Edge Bundlig for Graph Visualization.* IEEE Transactions on Visualization and Computer Graphics, Vol. 17, No. 12, pp. 2364 - 2373, 2011.
[18] E. R. Gansner, S. North Y. Hu, and C. Scheidegger. *Multilevel agglomerative edge bundling for visualizing large graphs.* IEEE Pacific Visualization Symposium, pp. 187 - 194, 2011.
[19] D. Holten and J. J. van Wijk. *Force-Directed Edge Bundling for Graph Visualization.* Eurographics, IEEE-VGTC Symposium on Visualization, 2009.
[20] B. Chazelle R. Osada, T. Funkhouser and D. Dobki. *Shape Distributions.* Eurographics, ACM Transactions on Graphics, Vol. 21, No. 4, pp. 807-832, 2002.



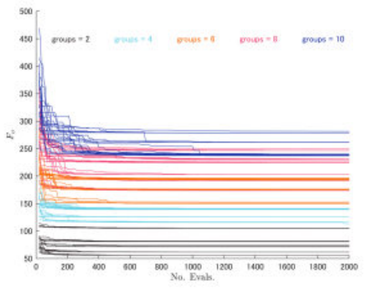
(a) Polygons with 5 sides



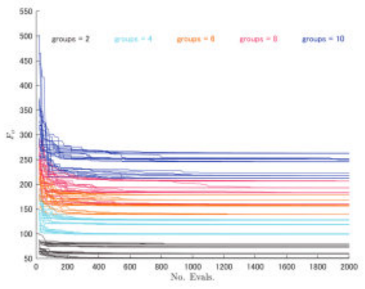
(b) Polygons with 10 sides

Figure 8: Convergence of fitness functions for $|\rho| = 3$

.



(a) Polygons with 5 sides



(b) Polygons obstacles with 10 sides

Figure 9: Convergence of fitness functions for $|\rho| = 4$

.

[21] E. W. Dijkstra. *A note on two problems in connexion with graphs.* Numerische Mathematik,1:269-271, 1959.

[22] B. Raphael P.E. Hart, N.J. Nilsson. *A formal basis for the heuristic determination of minimum cost paths.* IEEE Transactions on Systems Science and Cybernetics, 4(2):100-107, 1968.

[23] T. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms.* MIT Press, Cambridge, MA, 1993.

[24] M. Kallmann. *Path Planning in Triangulations.* In Proceedings of the Workshop on Reasoning, Representation, and Learning in Computer Games, IJCAI, pp. 49 - 54, 2005.

[25] B. Chazelle. *A Theorem on Polygon Cutting with Applications.* In Proceedings of the 23rd IEEE Symposium on Founda tions of Computer Science, pp. 339-349, 1982.

[26] D. T. Lee and F. P. Preparata. *Euclidean Shortest Paths in the Presence of rectilinear barriers.* Networks. 14(3), pp. 393-410, 1984.

[27] Victor Parque, M Kobayashi, and M Higashi. *Optimisation of Bundled Routes.* 16th International Conference on Geometry and Graphics, pp. 893-902, 2014.

[28] Victor Parque, Satoshi Miura, and Tomoyuki Miyashita. Route bundling in polygonal domains using differential evolution. *Robotics and Biomimetics*, 4(1):22, Dec 2017.

[29] Victor Parque, S. Miura, and T. Miyashita. *Optimization of Route Bundling via Differential Evolution with a Convex Representation.* IEEE International Conference on Real-time Computing and Robotics, Okinawa, Japan*forthcoming*, 2017.

[30] Victor Parque and T. Miyashita. *Bundling n-Stars in Polygonal Maps.* 29th IEEE Int. Conf. on Tools with Artificial Intelligence, Nov. 6-8, Boston, U.S., 2017.

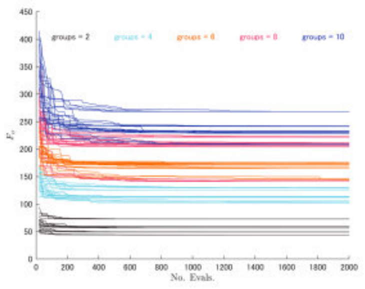
[31] V. Parque and T. Miyashita. *On succinct representation of directed graphs.* IEEE International Conference on Big Data and Smart Computing, pp. 199-205, 2017.

[32] V. Parque, M. Kobayashi, and M. Higashi. *Bijections for the numeric representation of labeled graphs.* IEEE Int. Conf. on Systems, Man and Cybernetics, pp. 447-452, 2014.

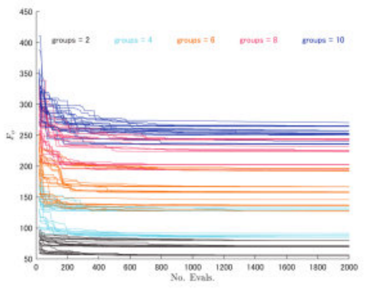
[33] Victor Parque, Masakazu Kobayashi, and Masatake Higashi. Neural computing with concurrent synchrony. In *Neural Information Processing*, pages 304–311, Cham, 2014. Springer International Publishing.

[34] V. Parque, M. Kobayashi, and M. Higashi. *Searching for Machine Modularity using Explorit.* IEEE Int. Conf. on Systems, Man and Cybernetics, pp. 2599-2604, 2014.

[35] V. Parque and T. Miyashita. On k-subset sum using enumerative encoding. In *IEEE Int. Symp. on Signal Processing and Information Technology*, pages 81–86, 2016.



(a) Polygons with 5 sides



(b) Polygons with 10 sides

Figure 10: Convergence of fitness functions for $|\rho| = 5$

.