Scientia Iranica (2024) 31(14), 1143–1148

Sharif University of Technology
Scientia Iranica
Transactions D: Computer Science & Engineering and Electrical Engineering
https://scientiairanica.sharif.edu

On the maximum triangle problem

A. Jabal Ameli[a] and H. Zarrabi-Zadeh[b,*]

a. *Department of Mathematics and Computer Science, Eindhoven University of Technology, Eindhoven, Netherlands.*
b. *Department of Computer Engineering, Sharif University of Technology, Tehran, Iran.*

Received 17 March 2024; accepted 21 April 2024

KEYWORDS
Maximum triangle;
Min-plus multiplication;
Approximation algorithm;
Computational geometry;
Convex hull.

Abstract. Given a set P of n points in the plane, the maximum triangle problem asks for finding a triangle with three vertices in P that encloses the maximum number of points from P. While the problem is easily solvable in $O(n^3)$ time, it has been open whether a subcubic solution is possible. In this paper, we show that the problem can be solved in $o(n^3)$ time, using a reduction to min-plus matrix multiplication. We also provide some improved approximation algorithms for the problem, including a 4-approximation algorithm running in $O(n \log n \log h)$ time, and a 3-approximation algorithm with $O(nh \log n + nh^2)$ runtime, where h is the size of the convex hull of P.

1. Introduction

Let P be a set of n points in the plane. In the maximum triangle problem, the objective is to find a triangle with three vertices from P, so that the number of points of P enclosed by the triangle is maximum (for an illustration, see Figure 1). Eppstein et al. [1] showed that the problem can be solved in $O(n^3)$ time. They indeed solved a more general problem of finding a convex k-gon enclosing a maximum number of points in $O(kn^3)$ time. They left this question open whether the problem can be solved faster.

Douïeb et al. [2] presented several approximation algorithms for the maximum triangle problem. In particular, they provided a 3-approximation algorithm running in $O(nh^2 \log n)$ time, and a 4-approximation algorithm with $O(n \log^2 n)$ running time. They again posed finding an $o(n^3)$-time exact algorithm as an open problem.

1.1. Our results

In this paper, we revisit the maximum triangle problem, and provide several improved results, as described below:

- We provide the first $o(n^3)$-time exact algorithm for the maximum triangle problem, thereby answering an open problem posed by Eppstein et al. [1]. Our algorithm is based on a reduction to the min-plus matrix multiplication, for which slightly subcubic algorithms are already known in the literature;
- We provide a 3-approximation algorithm for the maximum triangle problem that runs in $O(nh \log n + nh^2)$ time, where h denotes the size of the convex hull of the input point set. Our algorithm improves

*. *Corresponding author.*
E-mail addresses: a.jabal.ameli@tue.nl (A. Jabal Ameli); zarrabi@sharif.edu (H. Zarrabi-Zadeh)

To cite this article:
A. Jabal Ameli and H. Zarrabi-Zadeh "On the maximum triangle problem", *Scientia Iranica* (2024) **31**(14), pp. 1143-1148
https://doi.org/ 10.24200/sci.2024.64287.8852

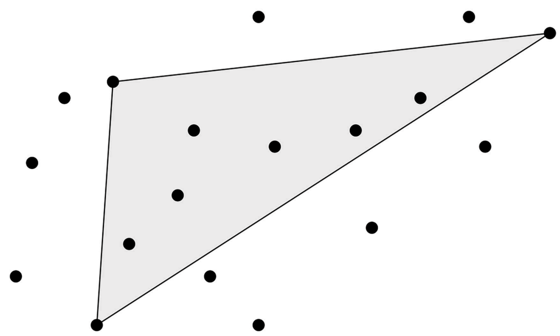

Figure 1. An example of the maximum triangle problem.

Table 1. Summary of the results for the maximum triangle problem.

	Runtime	
Algorithm	**Previous [1,2]**	**This work**
Exact	$O(n^3)$	$n^3/2^{\Omega(\sqrt{\log n})}$
3-approximation	$O(nh^2 \log n)$	$O(nh \log n + nh^2)$
4-approximation	$O(nh^2 \log h)$	$O(nh \log h + h^3)$
4-approximation	$O(n \log^2 n)$	$O(n \log n \log h)$

by a factor of $\min\{\log n, h\}$ the running time of a previous 3-approximation algorithm due to Douïeb et al. [2] that runs in $O(nh^2 \log n)$ time;

- We show how a 4-approximation to the maximum triangle can be computed in $O(nh \log h + nh^3)$ time, improving upon a previous 4-approximation algorithm of Douïeb et al. [2] that runs in $O(nh^2 \log h)$ time. Our algorithm is faster by a factor of $\min\{h, (n/h) \log h\}$ in this case compared to the previous existing algorithm;
- We present another 4-approximation algorithm with a running time of $O(n \log n \log h)$, improving upon a previous 4-approximation algorithm of Douïeb et al. [2] that runs in $O(n \log^2 n)$ time.

A summary of the results provided in this paper and the previous work is presented in Table 1.

1.2. Related work

The problem of finding a convex k-gon with vertices from the input point set maximizing or minimizing a particular function has been widely studied in the literature. For the problem of finding a maximum area and a maximum perimeter convex k-gon, Boyce et al. [3] provided an $O(kn \log n + n \log^2 n)$ time algorithm, which was later improved to $O(kn + n \log n)$ by Aggarwal et al. [4] using a fast matrix search method. Eppstein et al. [1] showed that a minimum area and a minimum perimeter convex k-gon, as well as a convex k-gon enclosing a minimum (or maximum) number of points can be computed in $O(kn^3)$ time.

A related problem of counting the number of triangles in a graph has received considerable attention due to its applications in social network analysis, community detection, and link prediction [5,6]. The best known algorithm for this problem is based on fast matrix multiplication with $O(n^\omega)$ time complexity, where $\omega < 2.372$ [7,8]. The problem is also studied in other models of computation, including parallel and data streaming [9–12]. See also Refs. [13–15] for some recent work on the related triangle detection problem.

The min-plus matrix multiplication, also known as distance product and tropical product, is extensively studied in the literature, due to its connection to several fundamental problems such as all-pairs shortest paths, minimum cycles, and replacement paths [16]. For this problem, slightly subcubic algorithms are known [17–19], the current best of which is due to Chan and Williams [20] with $O(n^3/2^{\sqrt{\log n}})$ time complexity. In fact, it is widely believed that truly subcubic algorithms with $O(n^{3-\varepsilon})$ running time do not exist for min-plus matrix multiplication, based on recent findings in fine-grained complexity [21].

2. Preliminaries

Let P be a set of n points in the plane. Throughout this paper, we assume that the points are in general position, i.e., no three points are co-linear, and no two points have the same x-coordinate.

Given three points $p, q, r \in P$, we call Δpqr a triangle of P, and denote by $|\Delta pqr|$ the number of points *enclosed* by Δpqr, i.e., the number of points contained in the interior or on the boundary of Δpqr. A triangle Δpqr with maximum $|\Delta pqr|$ is called a *maximum triangle* of P, or in short, an *optimal triangle*.

3. A subcubic exact algorithm

In this section, we show how the maximum triangle problem can be solved in $o(n^3)$ time, using matrix multiplication over the $(\min, +)$-semiring, for which slightly subcubic algorithms are available. Recall that the *min-plus* product of two $n \times n$ matrices A and B is defined as:

$$(A \oplus B)_{i,j} = \min_{1 \le k \le n} \{A_{i,k} + B_{k,j}\}. \quad (1)$$

Theorem 1. Let P be a set of n points in the plane. A maximum triangle of P can be found in $O(n^2 + T(n))$ time, where $T(n)$ is the time needed for computing the min-sum product of two $n \times n$ matrices, the best current algorithm for which has $n^3/2^{\Omega(\sqrt{\log n})}$ runtime.

Proof. For each pair of points $p, q \in P$, we denote by $n_{\overline{pq}}$ the number of points from P in the vertical slab (strictly) below the line segment $\overline{pq}$ (see Figure 2). The value of $n_{\overline{pq}}$ for all pairs $p, q \in P$ can be computed in $O(n^2)$ time [1]. For any two points $p, q \in P$, we set

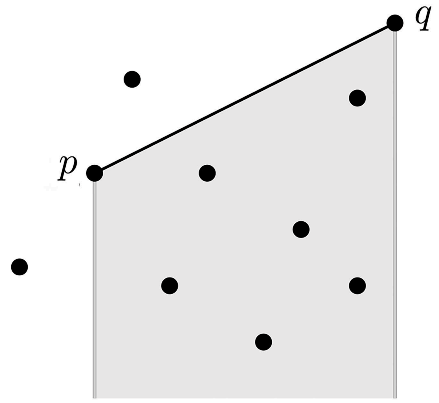

Figure 2. Points below the line segment $\overline{pq}$.

$n_{\vec{pq}} = n_{\overline{pq}} + 1$ if the vector $\vec{pq}$ is directed from left to right, and set $n_{\vec{pq}} = -n_{\overline{pq}}$ otherwise.

Now, for any triangle Δpqr with three vertices p, q, and r in clockwise order, the number of points enclosed by Δpqr can be written as:

$$|\Delta pqr| = n_{\vec{pq}} + n_{\vec{qr}} + n_{\vec{rp}} + 1. \tag{2}$$

Note that Eq. (2) correctly captures the number of points enclosed by the triangle Δpqr, no matter if the triangle is upward or downward (see Figure 3). Moreover, for computing the maximum triangle, we only need to consider the points in clockwise order, as the value of $n_{\vec{pq}} + n_{\vec{qr}} + n_{\vec{rp}}$ for any three points p, q, and r in counter-clockwise order is smaller than the corresponding value in clockwise order.

Let A be a $n \times n$ matrix with $A_{p,q} = -n_{\vec{pq}}$, and let $B = A \oplus (A \oplus A)$. By the definition of the min-plus product, we have:

$$B_{p,p} = \min_{q,r \in P} \{A_{p,q} + A_{q,r} + A_{r,p}\}, \tag{3}$$

for all $p \in P$. Therefore, to obtain a maximum triangle, we just need to check the n values on the main diagonal of the matrix B for the smallest (negative) number, whose absolute value corresponds to the number of points enclosed by a maximum triangle. The optimal triangle itself can be easily found in $O(n^2)$ time by enumerating all $O(n^2)$ triangles with one vertex on the point realizing the smallest value in the diagonal. The whole runtime of the algorithm is therefore bounded by that of computing the min-plus product. □

Our algorithm can be generalized to work for point sets not in general position as well. Let $t_{\overline{pq}}$ denote the number of points lying on the line segment $\overline{pq}$, including p and q, themselves. The value of $t_{\overline{pq}}$ for all pairs $p, q \in P$ can be computed in $O(n^2)$ time [1]. Now, in the proof of Theorem 1, it just suffices to set $n_{\vec{pq}} = n_{\overline{pq}} + t_{\overline{pq}} - 1$ if the vector $\vec{pq}$ is directed from left to right. The rest of the proof remains unchanged.

4. Improved approximation algorithms

Douïeb et al. [2] proposed several subcubic approximation algorithms for the maximum triangle problem. The main idea behind their algorithms is to reduce the number of enumerated triangles by fixing 1, 2, or 3 vertices of the optimal triangle on the convex hull of the input points. They also used this simple observation that if the surface of an optimal triangle is covered by c triangles (for an integer $c \geq 1$), then one of these triangles is a c-approximation of the optimal triangle.

In this section, we improve the runtime of the approximation algorithms proposed by Douïeb et al. [2], using faster methods for counting the number of points in the enumerated triangles.

In the remaining of this section, we assume that P is a set of n points in general position in the plane, H is the convex hull of P, and $h = |H|$. We will use the following two auxiliary lemmas from Douïeb et al. [2].

Lemma 1 [2]. *Among all triangles in P with k vertices on the convex hull ($1 \leq k \leq 3$), there exists a triangle that $(k+1)$-approximates an optimal triangle.*

Lemma 2 [2]. *Given two points $p, q \in H$, the value of $|\Delta pqr|$ for all $r \in P$ can be computed in $O(n \log n)$ time. Furthermore, $|\Delta pqr|$ for all $r \in H$ can be computed in $O(n \log h)$ time.*

Now, we prove three lemmas which are the main

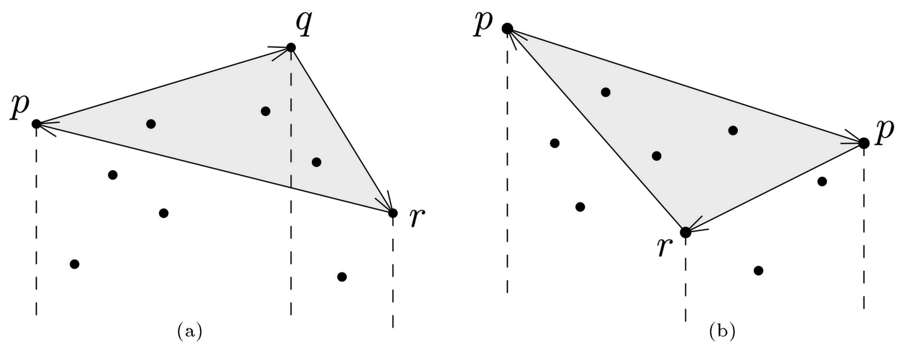

Figure 3. The two possible configurations for the triangle Δpqr.

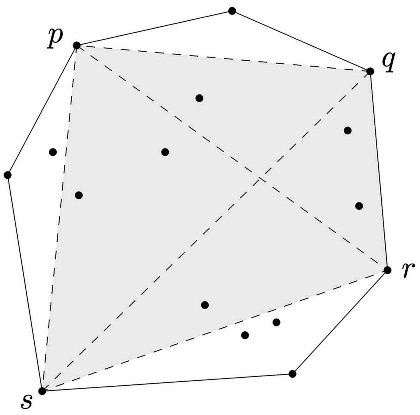

Figure 4. Triangles formed by four points on the convex hull.

ingredients of our improved approximation algorithms.

Lemma 3. *Given a point $p \in H$, the value of $|\Delta pqr|$ for all $q, r \in H$ can be computed in $O(nh \log h)$ time. Furthermore, $|\Delta pqr|$ for all $q \in P$ and $r \in H$ can be computed in $O(nh \log n)$ time.*

Proof. Fix a point q on H. By Lemma 2, $|\Delta pqr|$ for all $r \in H$ can be computed in $O(n \log h)$ time. Since there are $h-1$ option for choosing q, computing $|\Delta pqr|$ for all $q, r \in H$ takes $O(nh \log h)$ time in total. Similarly, if we fix $q \in P$, the algorithm takes $O(nh \log n)$ time by Lemma 2. □

Lemma 4. *The value of $|\Delta pqr|$ for all three points $p, q, r \in H$ can be computed in $O(nh \log h + h^3)$ time.*

Proof. Let p, q, r, and s be four points on H in clockwise order. The value of $|\Delta pqr|$ can be written as $|\Delta spq| + |\Delta sqr| - |\Delta spr|$ (see Figure 4). By Lemma 3 we can compute the number of points enclosed by all triangles on H whose one vertex is fixed on s in $O(nh \log n)$ time. Therefore, after this preprocessing step, we can compute the value of $|\Delta pqr|$ for each $p, q, r \in H$ in $O(1)$ time. Since there are $O(h^3)$ such triangles, the whole process takes $O(nh \log h + h^3)$ time in total. □

Lemma 5. *For all $p, q \in H$ and $r \in P$, the value of $|\Delta pqr|$ can be computed in $O(nh \log n + nh^2)$ total time.*

Proof. For a fixed point s on H, we compute the number of points enclosed by all triangles with one vertex on s, and the other two vertices freely chosen one from P and the other from H in $O(nh \log n)$ time using Lemma 3. Now, for any triangle Δpqr with $p, q \in H$ and $r \in P$, we compute $|\Delta pqr|$ as follows:

(i) If $\overline{rp}$ crosses $\overline{sq}$, then $|\Delta pqr| = |\Delta pqs| + |\Delta qrs| - |\Delta prs|$;

(ii) If $\overline{rq}$ crosses $\overline{sp}$, then $|\Delta pqr| = |\Delta pqs| + |\Delta prs| - |\Delta qrs|$;

(iii) If $\overline{rs}$ crosses $\overline{pq}$, then $|\Delta pqr| = |\Delta prs| + |\Delta qrs| - |\Delta pqs|$;

(iv) If r lies inside Δpqs, then $|\Delta pqr| = |\Delta pqs| - |\Delta prs| - |\Delta qrs| + 5$.

In any of the above cases, $|\Delta pqr|$ can be computed in $O(1)$ time. Since there are $O(nh^2)$ different triangles Δpqr with $p, q \in H$ and $r \in P$, we can compute $|\Delta pqr|$ for all such triangles in $O(nh \log n + nh^2)$ total time. □

Now, Lemmas 4 and 5 together with Lemma 1 yield the following theorem.

Theorem 2. *A 3-approximation of an optimal triangle can be computed in $O(nh \log n + nh^2)$ time. Furthermore, a 4-approximation of an optimal triangle can be found in $O(nh \log h + h^3)$ time.*

Remark 1. Eppstein et al. [1] proved that P can be preprocessed in $O(n^2)$ time, so that for any query triangle Δpqr in P, $|\Delta pqr|$ can be reported in $O(1)$ time. Using this as an alternative way for counting the number of points in the enumerated triangles, we can rewrite the time bounds in Theorem 2 as $O(\min(n^2, nh \log n) + nh^2)$ for the 3-approximation, and $O(\min(n^2, nh \log h) + h^3)$ for the 4-approximation algorithm.

In the following theorem, we present an alternative 4-approximation algorithm for the problem.

Theorem 3. *An optimal triangle can be approximated within a factor of 4 in $O(n \log n \log h)$ time.*

Proof. Let $t_1, t_2, \ldots, t_h$ be the vertices of H in clockwise order, and let $m = \lfloor h/2 \rfloor + 1$. We partition H into two convex polygons $H_1 = t_1, t_2, \ldots, t_m$ and $H_2 = t_m, \ldots, t_h, t_1$. Let P_1 and P_2 be the points of P enclosed by H_1 and H_2, respectively. We use Lemma 2 to compute $|\Delta t_1 t_m p|$ for all $p \in P$ in $O(n \log n)$ time. We then recurse on P_1 and P_2, and return a triangle found containing a maximum number of points.

To prove correctness, we first recall that there exists a triangle $\Delta t_1 pq$ with $p, q \in P$ that 2-approximates an optimal triangle [2]. If $t_1 t_m$ crosses pq, then the two triangles $\Delta t_1 t_m p$ and $\Delta t_1 t_m q$ cover $\Delta t_1 pq$, and hence, one of them is a 2-approximation of $\Delta t_1 pq$, which is in turn, a 4-approximation of an optimal triangle. On the other hand, if pq lies in one side of $t_1 t_m$, the recursive call on that side returns a 2-approximation.

Let $T(n, h)$ be the time required by the algorithm on a point set of size n whose convex hull has size h. Then, $T(n, h) = T(n_1, h_1) + T(n_2, h_2) + O(n \log n)$, where $n_1 + n_2 = n + 2$, $h_1 = \lfloor h/2 \rfloor + 1$, and $h_2 =$

$\lceil h/2 \rceil + 1$. The recurrence tree for this relation has height $O(\log h)$, and yields $T(n,h) = O(n \log n \log h)$.□

5. Conclusions

In this paper, we presented a slightly subcubic algorithm for the maximum triangle problem, and improved the running time of several approximation algorithms available for the problem. A main question that remains open is whether a truly subcubic algorithm with $O(n^{3-\varepsilon})$ time is possible for the problem. It is also interesting to study the generalized maximum k-gon problem, for $k \geq 4$.

Acknowledgments

The authors would like to thank Mohammad-Reza Maleki, Hamed Valizadeh, and Hamed Saleh for their helpful discussions during the early stages of this work.

References

1. Eppstein, D., Overmars, M., Rote, G., et al. "Finding minimum area k-gons", *Discrete Comput. Geom.*, **7**(1), pp. 45–58 (1992).
DOI: 10.1007/BF02187823
2. Douïeb, K., Eastman, M., Maheshwari, A., et al. "Approximation algorithms for a triangle enclosure problem", In *Proc. 23rd Canad. Conf. Computat. Geom.*, pp. 105–110 (2011).
3. Boyce, J.E., Dobkin, D.P., Drysdale III, R.L., et al. "Finding extremal poly-gons", *SIAM J. Comput.*, **14**(1), pp. 134–147 (1985).
DOI: 10.1137/0214011
4. Aggarwal, A., Klawe, M.M., Moran, S., et al. "Geometric applications of a matrix-searching algorithm", *Algorithmica*, **2**, pp. 195–208 (1987).
DOI: 10.1007/BF01840359
5. Al Hasan, M. and Dave, V.S. "Triangle counting in large networks: a review", *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, **8**(2), e1226 (2018).
DOI: 10.1002/widm.1226
6. Ribeiro, P., Paredes, P., Silva, M.E., et al. "A survey on subgraph count-ing: concepts, algorithms, and applications to network motifs and graphlets", *ACM Computing Surveys*, **54**(2), pp. 1–36 (2021).
DOI: 10.1145/3433652
7. Duan, R., Wu, H., and Zhou, R. "Faster matrix multiplication via asymmetric hashing", In *Proc. 64th Annu. IEEE Sympos. Found. Comput. Sci.*, pp. 2129–2138 (2023).
DOI: 10.1109/FOCS57990.2023.00130
8. Williams, V.V., Xu, Y., Xu, Z., et al. "New bounds for matrix multiplica-tion: from alpha to omega", In *Proc. 35th ACM-SIAM Sympos. Discrete Algorithms*, pp. 3792–3835 (2024).
DOI: 10.1137/1.9781611977912.134
9. Jayaram, R. and Kallaugher, J. "An optimal algorithm for triangle counting in the stream", In *Proc. Internat. Workshop Approx. Algorithms*, pp. 11:1–11:11 (2021).
DOI: 10.4230/LIPIcs.APPROX/RANDOM.2021.11
10. Sharafeldeen, A., Alrahmawy, M., and Elmougy, S. "Graph partitioning MapReduce-based algorithms for counting triangles in large-scale graphs", *Scientific Reports*, **13**(1), p. 166 (2023).
DOI: 10.1038/s41598-022-25243-w
11. Dhulipala, L., Liu, Q.C., Shun, J., et al. "Parallel batch-dynamic k-clique counting", In *Proc. Sympos. Algorithmic Principles of Computer Systems*, pp. 129–143 (2021).
DOI: 10.1137/1.9781611976489
12. Ding, X., Sheng, S., Zhou, H., et al. "Differentially private triangle counting in large graphs", *IEEE Trans. Knowledge and Data Engineering*, **34**(11), pp. 5278–5292 (2022).
DOI: 10.1109/TKDE.2021.3052827
13. Chan, T.M. "Finding triangles and other small subgraphs in geometric inter-section graphs", In *Proc. 34th ACM-SIAM Sympos. Discrete Algorithms*, pp. 1777–1805 (2023).
DOI: 10.1137/1.9781611977554.ch68
14. Dumitrescu, A. and Lingas, A. "Finding small complete subgraphs efficiently", In *Proc. 34th Internat. Workshop Combinatorial Algorithms*, pp. 185–196 (2023).
DOI: 10.1007/978-3-031-34347-6 16
15. Dalirrooyfard, M., Vuong, T.D., and Williams, V.V. "Graph pattern detection: Hardness for all induced patterns and faster noninduced cycles", *SIAM J. Comput.*, **50**(5), pp. 1627–1662 (2021).
DOI: 10.1137/20M1335054
16. Williams, V.V. and Williams, R. "Subcubic equivalences between path, matrix, and triangle problems", *J. ACM*, **65**(5), p. 27 (2018).
DOI: 10.1145/3186893
17. Chan, T.M. "More algorithms for all-pairs shortest paths in weighted graphs", *SIAM J. Comput.*, **39**(5), pp. 2075–2089 (2010).
DOI: 10.1137/08071990X
18. Fredman, M.L. "New bounds on the complexity of the shortest path problem", *SIAM J. Comput.*, **5**(1), pp. 83–89 (1976).
DOI: 10.1137/0205006
19. Williams, R.R. "Faster all-pairs shortest paths via circuit complexity", *SIAM Journal on Computing*, **47**(5), pp. 1965–1985 (2018).
DOI: 10.1137/15M1024524
20. Chan, T.M. and Williams, R. "Deterministic APSP, orthogonal vectors, and more: Quickly derandomizing Razborov-Smolensky", In *Proc. 27th ACM-SIAM Sympos. Discrete Algorithms*, pp. 1246–1255 (2016).
DOI: 10.1137/1.9781611974331.ch87

21. Williams, V.V. "On some fine-grained questions in algorithms and complexity", In *Proc. Internat. Congress Math.*, pp. 3447–3487 (2018).
DOI: 10.1142/9789813272880 0188

Biographies

Afrouz Jabal Ameli received his PhD in Computer Science in the Algorithms and Complexity group at the Dalle Molle Institute for Artificial Intelligence Research, Lugano, Switzerland. He is currently a Postdoctoral Researcher at the Department of Mathematics and Computer Science at Eindhoven University of Technology, Netherlands. His research focuses on the design of approximation algorithms for discrete optimization problems, more specifically for survivable network design problems.

Hamid Zarrabi-Zadeh received his PhD in Computer Science from the University of Waterloo, Waterloo, Canada. He is currently an Assistant Professor in the Department of Computer Engineering at Sharif University of Technology. He is also a member of the International Technical Committee for the International Olympiad in Informatics (IOI). His research focuses on the design and analysis of algorithms, computational geometry, approximation and randomized algorithms, and algorithmic graph theory.