Aggregating Robots Compute: An Adaptive Heuristic for the Euclidean Steiner Tree Problem

Heiko Hamann and Heinz Wörn

Institute for Process Control and Robotics,
Universität Karlsruhe (TH), 76131 Karlsruhe, Germany

Abstract. It is becoming state-of-the-art to form large-scale multi-agent systems or artificial swarms showing adaptive behavior by constructing high numbers of cooperating, embodied, mobile agents (robots). For the sake of space- and cost-efficiency such robots are typically miniaturized and equipped with only few sensors and actuators resulting in rather simple devices. In order to overcome these constraints, bio-inspired concepts of self-organization and emergent properties are applied. Thus, accuracy is usually not a trait of such systems, but robustness and fault tolerance are. It turns out that they are applicable to even hard problems and reliably deliver approximated solutions. Based on these principles we present a heuristic for the Euclidean Steiner tree problem which is NP-hard. Basically, it is the problem of connecting objects in a plane efficiently. The proposed system is investigated from two different viewpoints: computationally and behaviorally. While the performance is, as expected, clearly suboptimal but still reasonably well, the system is adaptive and robust.

1 Introduction

With the increase of interdisciplinary research new concepts were developed in the last decades. For example, swarm intelligence [2] applied to computational problems leads to powerful meta-heuristics [4] and applied to robotics it results in large-scale distributed robotic systems [16]. In this paper we try to pursue these approaches and to combine swarm intelligence with robotics and heuristics.
The scientific approach to computation was significantly governed by the computational devices used in the past. Thus, it began with sequential devices, later parallel machines with shared memory were studied, and even later the focus was on distributed but fully connected systems. All these approaches have determinism and explicit communication in common.
A new philosophy is introduced by applying concepts of swarm intelligence, e.g. simple local rules, indirect communication (stigmergy), and cooperation [2]. We can consider swarm intelligence as the final step of the process of getting away from centralized and deterministic systems towards fully distributed and probabilistic systems. Swarm intelligence was applied to computational problems by using software agents. We provide our computational devices with actuators

making them mobile. Hence, they become real, embodied agents in the form of robots. The idea of using a group of autonomous agents as processing elements, that are embedded in the environment, that sense and compute based only on local information was published by Payton et al. [14] and propagated as 'world-embedded computation'. This is related to an old question, whether an ant colony's struggle of survival might be viewed as computation or not [9]. Thus, we note that problem solving by adaptive and cooperative behaviors might be considered computation.

Combining the local and randomized approach of swarm intelligence with an emphasis on positional information, results in an interesting computing paradigm or also in a method of generating emergent behavior. The position of a robot in the physical world becomes the building block of collective information processing [8]. In a recent work Litus et al. [13] give a good summary of this idea:

> The key insight that underlies our methods is that the physical locations of the robots themselves could be considered as an approximate solution to the entire problem. An individual robot can move itself, thus refining the current solution approximation. No representation of the problem, or the current solution, needs to be held by any robot: they manifest the solution by their physical configuration.

Although these approaches suffer in principle from the same problems of intractability as classical approaches concerning hard problems, they might lead, nevertheless, to more efficient implementations. Such systems might be cheaper than classical devices and easier to maintain due to less complexity.

We focus on the Euclidean Steiner tree (EST) problem which is basically the problem of connecting objects in a plane efficiently. The agent-based heuristic, that we investigated here, was shortly introduced in [8]. The objects in the plane are connected by placing mobile relay stations, that we call robots in the following, instead of using wires. Starting with a uniformly distributed population of robots they aggregate in a way similar to diffusion-limited aggregation (DLA) [17]. Unlike DLA all robots are always moving and turn to avoid collisions between two moving robots. The objects to be connected serve as seeds from which trees of aggregated robots 'grow'. The use of such 'random trees' for planning in robotics was introduced in [12]. In contrast to our approach these random trees are only virtual and are centrally controlled. As we are growing several trees or clusters at the same time this approach is also connected to diffusion-limited cluster-cluster aggregation [11]. However, the clusters in the present work are static.

In the following section we define the EST problem and give a short survey of the related work. In section 3 we present a robot control algorithm generating a collective problem solving system for the EST problem. Thereafter, some results and validations are given in section 4 which are discussed in section 5.

2 The Euclidean Steiner Tree Problem

The EST problem is named after the swiss mathematician Jakob Steiner and is defined as follows: A given set Z of N points or terminals in a plane has to be connected by lines of minimal length and, in contrast to the related minimal spanning tree problem, it is allowed to add a set of extra points S, called *Steiner points* (for an example see Fig. 2(d)). The resulting network is a graph $G = (V, E)$ with nodes $V = Z \cup S$ and edges E accordingly defined. The probably better known instance of the Steiner tree problem class is defined on graphs where Steiner points can be picked from a finite set of points instead of placing them anywhere in a plane.
There are many applications to this problem in circuit design, mining, network design, and routing in ad hoc networks. Computing an optimal EST is NP-hard, i.e. no efficient algorithm is known and is unlikely to be found, and the discretized variant is NP-complete [5]. It is even hard to find an approximation within 95/94 of the optimum [3]. A lot of work has been done to find both better exact algorithms as well as polynomial time heuristics [10, 15]. The best known heuristic was presented in [19] usually yielding solutions close to the optimum within a few seconds at least for $N < 1000$. A software to compute exact EST is the GeoSteiner package [18].
An amusing anecdote reported by Aaronson [1] gives a reason to suspect that anyway not too much intelligence might be of need for approximations of lower quality: "Yet a well-known piece of computer science folklore maintains that, if two glass plates with pegs between them are dipped into soapy water, then the soap bubbles will rapidly form a Steiner tree connecting the pegs, this being the minimum-energy configuration." Aaronson even experimented with real soap bubbles. He observed correct solutions but also cycles, incomplete trees, entirely different trees for the same configuration, and a relaxation process of several seconds leading to better soap bubble configurations. He follows that soap bubbles do not solve NP-complete problems in polynomial time. However, we note that (very) alternative approaches might be quite productive, too.

3 Growing Random Trees

3.1 Preconditions

We restrict ourselves to a two-dimensional setting but the proposed algorithm would work in the same way in three dimensions. We assume objects called *seeds* being placed in a bounded plane. Each seed represents a terminal out of Z of a considered Steiner tree problem. Furthermore we need some kind of robots that can move in the given environment. Whether these robots drive, crawl, fly, swim, or submerge is not relevant as long as they are able to move and to remain with sufficient accuracy at one spot. They have to be equipped with sensors allowing them to perceive other robots and seeds within a very short range compared to the dimensions of the bounded plane. Furthermore, they should be able to communicate to and measure an approximate bearing of other robots in their

neighborhood. Their control mechanism suffice to be reactive. A large group of such mobile robots is positioned uniformly distributed over the whole plane. This could be done by the swarm autonomously in a previous phase.

3.2 Algorithm

In the following we explain the control algorithm of the robots. See Fig. 1 for a schematic overview. In general the robots move forward, try to find seeds, and listen for pings by robots being already connected to a seed. When a robot finds a seed (not shown in the schematic) and does not receive any pings it stops next to it. The robot generates a tree identification number (tree ID), that is with high probability unique, for example, by using a big random number. Then it starts to ping. When another robot receives this signal it checks back with the sender, if there is still an open slot (only 3 connections per robot are allowed). If so it maximizes the angles between itself, the sender, and the sender's neighbors. In case of 1 neighbor it forms a straight line. In case of 2 neighbors they try to reach a configuration with 120°, but only if this requires the movement of no more than 2 robots (local optimization). This can be achieved by some communication overhead and relying on relative angles only (see [7] for details). This restriction of 3 connections per robot and the consequent angle of 120° is not arbitrarily chosen, as the optimal solution of a Steiner tree problem always consists of Steiner points of degree 3 and angles of 120° only. Additionally, we want to cover as much space in the plane as possible with a minimal number of stopped robots. The optimal solution to this tiling problem is provably the hexagon as it is found in honey-combs [6]. However, the robots form only partial hexagons because the result should be a tree, i.e. a cycle-free graph.
Using this control algorithm the robots perform a process similar to diffusion-limited aggregation [17]. Provided with a sufficient number of robots and time a tree will grow at each seed. At some time, 2 trees will be connected. This is the case if a robot approaches 2 aggregated robots of different trees virtually at the same time. By communicating their tree ID they ensure not to form a loop and agree upon a new tree ID, which is propagated through the new tree. Later, more trees will join. When only one huge tree is left, cf. Fig. 2(a), or at an assigned time (reduction condition), a new process is started: All robots being connected to only one other robot, i.e. they are leafs, will cut this connection and leave. In a chain reaction all unnecessary robots cut their connections and a tree consisting of a relatively small number of robots is left, cf. Fig. 2(b). After this reduction robots being connected to 3 other robots represent Steiner points. By straightening the connections between the seeds and the Steiner points, i.e. releasing surplus robots in between (for details see [7]), we get a first approximation to the optimal Steiner tree, cf. Fig. 2(c). Note, the positions of the Steiner points were determined dynamically during the tree growth process. Interpreting this in the sense of swarm intelligence: the Steiner point set S is the result of a collective decision that emerges from the numerous agent-agent interactions. An additional improvement is achieved by locally rearranging the Steiner points towards their optimal position (an approximation to the Fermat-Torricelli point).

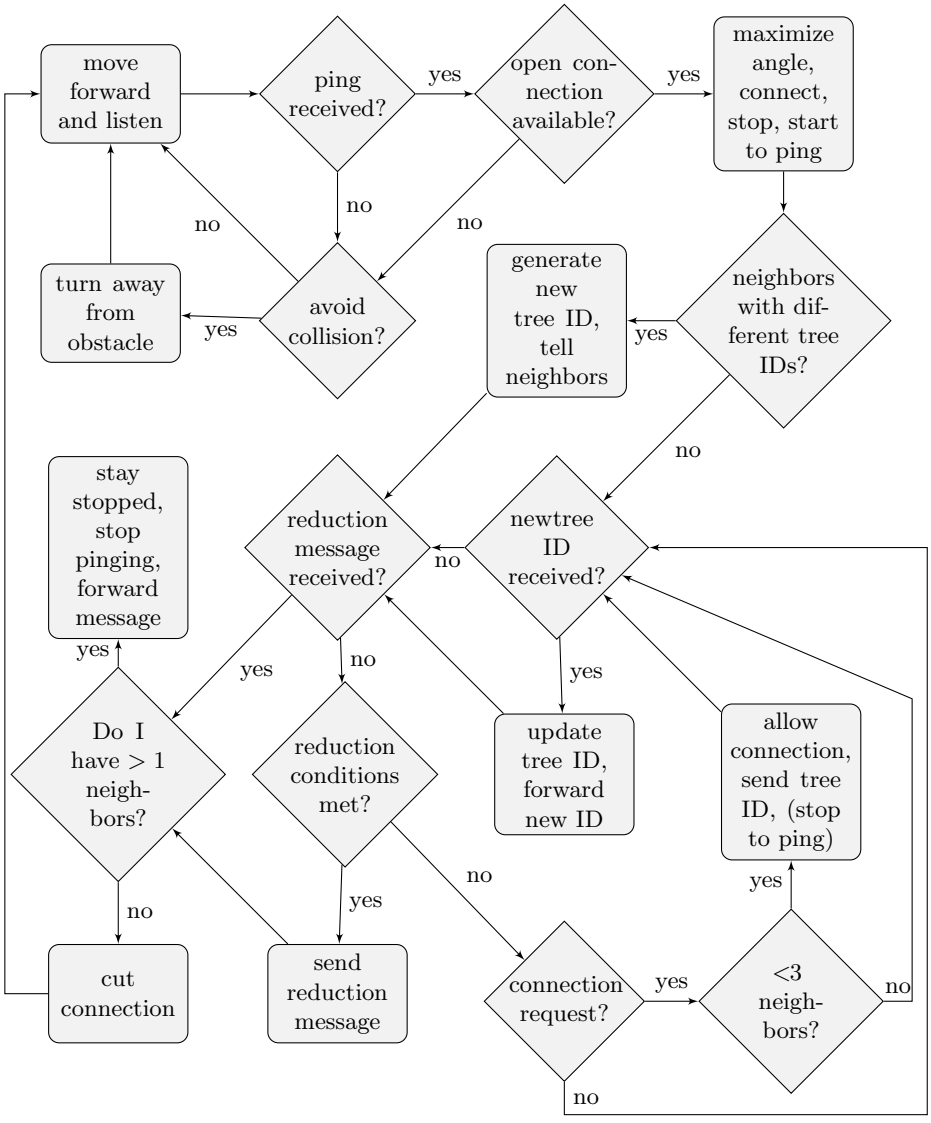


Fig. 1. Schema of the robot controller for the random tree growth algorithm.

This is achieved by moving the robots at the Steiner point towards the direction of the smallest angle. This might result in the optimal solution as shown in Fig. 2(d) but in general the optimal configuration of the Steiner points is only achievable by global optimization.
Keeping a reasonable amount of redundant robots in the lines this heuristic is robust to breakdowns of single robots although it might seem very inefficient. In addition, it is scalable because of its totally local approach. Whether this method can be a fast way of approximating a Steiner tree is a question of the reaction times and speed of the robots. At the time mass production of such devices will become possible, this scenario might actually be feasible [16].

4 Results

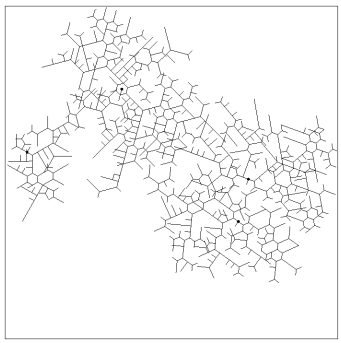
The following results were obtained using a first-order geometric simulation with continuous space. Our emphasis is on the general behavior of the agent system which we claim to be covered by this kind of simulation. A more complex simulation would have multiplied the computational complexity. However, already this abstract simulation kept a personal computer busy for days computing thousands of runs using 10^5 and more robots. As we do not yet have such quantities of robots or other computational devices available we had to simulate our massively parallel heuristic serially.
We compare the results of our heuristic to the optimal solution and the minimal spanning tree—the typical benchmark problem for Steiner tree heuristics. For this purpose we compare the reduction in percent r of the (suboptimal) Steiner tree length L_{steiner} to the minimal spanning tree length L_{spanning}

$$r = \frac{L_{\text{spanning}} - L_{\text{steiner}}}{L_{\text{spanning}}} \cdot 100\% \tag{1}$$

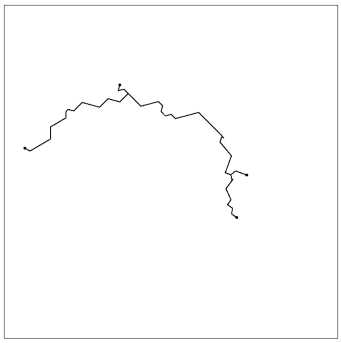
with

$$L = \sum_{(\mu,\nu)\in E} \|\mu - \nu\|_2, \tag{2}$$

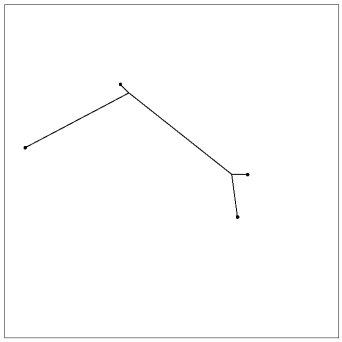
$\mu, \nu \in \mathcal{R}^2$ are node positions and $\|\cdot\|_2$ denotes the Euclidean norm.
For the optimal Steiner tree this value ranges from instance to instance between $r_{\min} = 0\%$ (minimal Steiner and spanning tree identical) and $r_{\max} > 10\%$. For the minimal Steiner tree the reduction averaged over many instances converges almost independently of the terminal number N to $\overline{r}_{\text{opt}} \approx 3.1\%$.
Before we compare the actual performance we have a look at the number of Steiner points generated by the random tree heuristic. In principal this number ranges from 0 (spanning tree) to $N-2$. However, it turns out that the heuristic generates reasonable numbers of Steiner points. See Fig. 3 showing the situation of a single problem instance with a mean that overestimates the optimal number of Steiner points by 5.8%.
Now we compare our heuristic to the exact solution focusing on the governing parameter, the robot number. We omit a time analysis since the time consumption can be kept constant with increasing robot number (due to strictly local



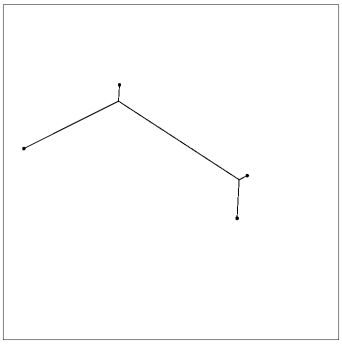
(a) Aggregated robots.



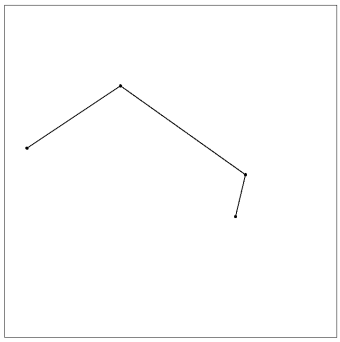
(b) Reduced to connected robots.



(c) Straightened lines.



(d) Local optimization (identical to optimum).



(e) Minimal spanning tree.

Fig. 2. Phases of the heuristic and minimal spanning tree.

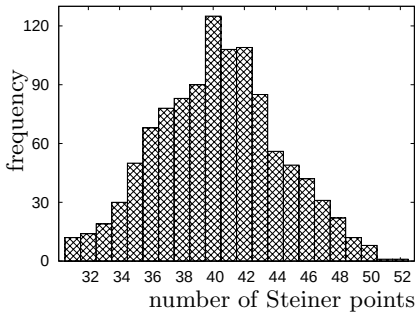
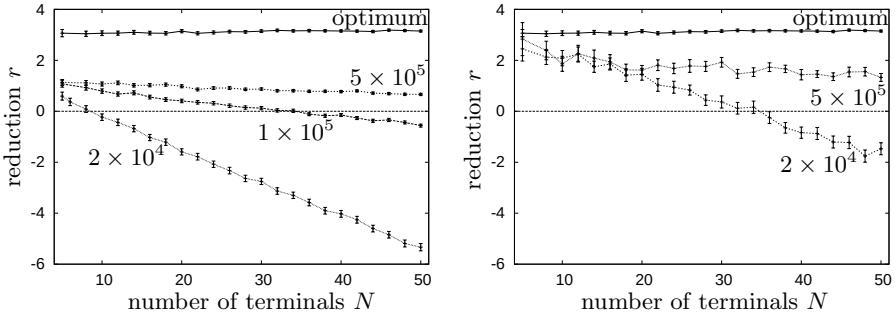


Fig. 3. Histogram of Steiner points generated in 550 samples by the heuristic for a problem instance of 100 terminals showing a mean of about 40.2, optimal are 38 Steiner points, and maximally possible are 98.



(a) 2×10^4, 1×10^5, and 5×10^5 robots, 500 samples for each point.

(b) Taking only the best out of 50 runs using 2×10^4 (80 samples per point) and 5×10^5 (60 samples per point) robots.

Fig. 4. Comparing the reduction of the heuristic for different robot numbers and varied problem size N to the optimal solution; error bars are 95% confidence intervals.

actions of each robot). Only the serialized simulation of the heuristic suffered from the complexity of high robot numbers. The solutions become better the more robots are used as shown in Fig. 4(a). The reduction decreases linearly in problem size. Due to the probabilistic characteristic of the proposed heuristic the average performance is improved by repeated runs. In Fig. 4(b) we show the performance achieved by selecting the best solution out of 50 runs. Especially for bigger instances $N > 50$ the performance could be furthermore improved by increasing the number of robots.

While our heuristic is inferior to the state-of-the-art heuristics in the performance we identify its advantage in its adaptivity and due to the decentralized approach also in its robustness. We test the adaptivity by replacing a terminal after 40 time steps. The terminal configuration shown in Fig. 5 and 10^4 robots were used. Only 8.2% were irregular approximations (not all terminals connected)

evaluating 500 samples. The average reduction dropped by 42% compared to the heuristic started with the final terminal configuration.

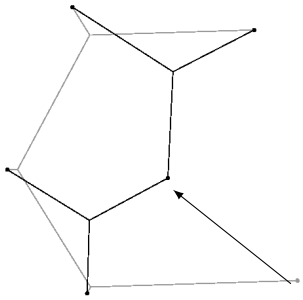


Fig. 5. Superimposed optimal solutions for the beginning set of terminals (gray) and the final set of the adaptivity test scenario. The arrow indicates the rearranged terminal.

5 Discussion and Conclusion

The heuristic proposed in this paper is definitely not superior to the state-of-the-art heuristics concerning quality and computing time (the best known heuristic typically delivers approximations within about 4% from optimum [19]). However, it shows a reasonable degree of adaptivity. Furthermore, it is supposed to be quite robust as there is no single point of failure. Comparing the fault tolerance of our method to the classical approach corresponds to answering the question: How wrong can wrong be? One might argue that this comparison is unfair because the difference is only due to different output methods: explicit and physical (motion, positions, angles) compared to symbolic (numbers, calculations). However, this actually is the fundamental difference in the method of information processing between these approaches. The classical computer processes abstract symbols while the agent system processes physical positions. Therefore, we consider a comparison to be fair and the difference in the wrongness is big. A single error might cause almost infinitely high deviations using a symbol-based approach as there are virtually no limitations for what could happen to a symbol in the CPU or the memory, e.g. a single bit shift might lead to negative distances. This is in contrast to our agent system, where a single error might break two subtrees but arbitrary deviations are impossible. This is one advantage of the strictly bounded operating range of the robots, limiting not only their possibilities but also the consequences of errors. An analysis of the proposed algorithm would obviously be very hard due to its properties that might be considered 'emergent'. For example, the growth of a random tree is dependent on its relative position to other trees. Any model describing the tree growth independently would have little explanatory power. Including all trees into the model would increase its complexity significantly.

Acknowledgments

Hamann is supported by the German Research Foundation (DFG) within the Research Training Group GRK 1194 Self-organizing Sensor-Actuator Networks.

References

1. S. Aaronson. NP-complete problems and physical reality. *ACM SIGACT News*, 36(1):30–52, March 2005.
2. E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford Univ. Press, 1999.
3. M. Chlebík and J. Chlebíková. Approximation hardness of the Steiner Tree problem on graphs. In *Proc. 8th Scandinavian Workshop on Algorithm Theory (SWAT)*, volume 2368 of *LNCS*, pages 170–179. Springer, 2002.
4. M. Dorigo and G. D. Caro. Ant Colony Optimization: A new meta-heuristic. In P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, and A. Zalzala, editors, *Proceedings of the 1999 Congress on Evolutionary Computation (CEC'99)*, pages 1470–1477, Piscataway, NJ, 1999. IEEE Press.
5. M. R. Garey, R. L. Graham, and D. S. Johnson. Some NP-complete geometric problems. In *Annual ACM Symp. on Theory of Computing*, pages 10–22, 1976.
6. T. C. Hales. The honeycomb conjecture. *Discrete and Computational Geometry*, 25(1):1–22, 2001.
7. H. Hamann. Modeling and investigation of robot swarms. Master's thesis, University of Stuttgart, Germany, 2006.
8. H. Hamann and H. Wörn. Embodied computation. *Parallel Processing Letters*, 17(3):287–298, September 2007.
9. D. R. Hofstadter. *Gödel, Escher, Bach*. Basic Books, 1979.
10. F. K. Hwang, D. S. Richards, and P. Winter. *The Steiner Tree Problem*. North-Holland, 1992.
11. M. Kolb and H. J. Herrmann. The sol-gel transition modelled by irreversible aggregation of clusters. *J. Physics A*, 18(8):L435–L441, June 1985.
12. S. M. LaValle and J. J. Kuffner. Rapidly-exploring random trees: Progress and prospects. In B. R. Donald, K. M. Lynch, and D. Rus, editors, *Algorithmic and Computational Robotics*, pages 293–308, Wellesley, MA, USA, 2001. A. K. Peters.
13. Y. Litus, P. Zebrowski, and R. Vaughan. Energy-efficient multi-robot rendezvous: Parallel solutions by embodied approximation. In *Workshop on Algorithmic equivalencies between biological and robotic swarms, Atlanta, USA*, June 2007.
14. D. Payton, M. Daily, R. Estowski, M. Howard, and C. Lee. Pheromone robotics. *Autonomous Robots*, 11(3):319–324, November 2001.
15. G. Robins and A. Zelikovsky. Improved Steiner Tree approximation in graphs. In *11th ACM-SIAM Symposium on Discrete Algorithms*, pages 770–779, 2000.
16. J. Seyfried, M. Szymanski, N. Bender, R. Estaña, M. Thiel, and H. Wörn. The I-SWARM project. In E. Şahin and W. M. Spears, editors, *Swarm Robotics Workshop*, pages 70–83, Berlin Heidelberg New York, 2005. Springer.
17. J. Tom A. Witten and L. M. Sander. Diffusion-limited aggregation, a kinetic critical phenomenon. *Phys. Rev. Lett.*, 19:1400 – 1403, 1981.
18. D. Warme, P. Winter, and M. Zachariasen. Geosteiner homepage. http://www.diku.dk/geosteiner/.
19. M. Zachariasen and P. Winter. Concatenation-based greedy heuristics for the Steiner tree problem in the Euclidean plane. *Algorithmica*, 25:418–437, 1999.