Simulated Annealing with Parameter Tuning for Wind Turbine Placement Optimization

Daniel Lückehe[1], Oliver Kramer[2], and Manfred Weisensee[3]

[1] Department of Geoinformation, Jade University of Applied Sciences, Oldenburg, Germany daniel.lueckehe@uni-oldenburg.de
[2] Department of Computing Science, University of Oldenburg, Oldenburg, Germany oliver.kramer@uni-oldenburg.de
[3] Department of Geoinformation, Jade University of Applied Sciences, Oldenburg, Germany weisensee@jade-hs.de

Abstract. Because of wake effects and geographical constraints, the search for optimal positions of wind turbines has an important part to play for their efficiency. The determination of their positions can be treated as optimization problem and can be solved by various methods. In this paper, we propose optimization approaches based on Simulated Annealing (SA) to improve solutions for the wind turbines placement problem. We define neighborhoods of solutions and analyze the influence of specific parameters, e.g., neighborhood distance and SA temperature in experimental studies. The experiments are based on a real-world scenario with a wind model, wind data from a meteorological service, and geographical constraints. Inspired by adaptive step size control applied in Evolutionary Strategies (ES), we propose an approach using an adaptive neighborhood distance and compare the results to optimization runs with a constant neighborhood distance. Also the best and worst optimization run and the corresponding placement results are shown and compared.

1 Introduction

Planning and optimization of renewable energy resources is an important part of today's activities towards an ecologically friendly smart grid. As the environment of wind turbines is significant for their efficiency, the determination of their locations should be handled carefully and with consideration of various aspects. In this paper, we use a wind model based on wind distributions using data from the German Weather Service and take geographical constraints into account. We apply different optimization approaches using SA to the turbine placement problem with the objective to maximize the power output. SA is often used for combinatorial problems, but can also be applied to continuous solution spaces. For this, we define neighborhoods in the continuous solution space of turbine positions. We also propose an adaptive variant of neighborhoods for SA inspired by an adaptive method from the field of ES, as the definition of neighborhood corresponds to the step size of ES.

This paper is structured as follows. In Section 2, we give an overview to related work. The wind model is explained in Section 3 and includes the definition of the employed scenario. In Section 4, we introduce the optimzation approaches, followed by the experimental results in Section 5. In Section 6, conclusions are drawn.

2 Related Work

The wind turbine placement problem is widely known and there are a lot of different models and approaches to solve it [5]. We observe a trend towards more realistic representations of the optimization problem, e.g., Kusiak and Song [8] are using Weibull [18] distribution and the Jensen [13] wake model to describe the behavior of the wind. Their model is able to compute the power output of turbines on a continuous map considering wake effects. They solved the optimization problem with a simple ES. Further works exploit more complex approaches like the covariance matrix adaptation evolution strategy (CMA-ES) [4] to solve the optimization problem [17]. There are also approaches that include geographical information from map services to the turbine placement problem [9].

To solve the turbine placement problem, we employ stochastic search algorithms [12]. In particular, we aim for an approach that exploits SA. For a comprehensive overview of SA, we refer to [6]. In this work, the basic idea of SA is explained, critically analyzed, and different variants of SA are experimentally considered. Nourani and Andresen [14] put a focus on the cooling schedules for SA. In their work, constant thermodynamic speed, exponential, logarithmic, and linear cooling schedules are analyzed. Rivas et al. [15] published an approach to solve the turbine placement problem for large offshore wind farms by SA. Their algorithm employs three types of local search operations: add, move, and remove. The operations are performed recursively and each has its own temperature. Based on the experimental results, in the conclusion of this work SA is called a suitable method for the wind turbine placement optimization problem.

3 Wind Setting

In this section, we introduce the wind turbine model that we use in the experimental part of this work. This model computes the produced energy of a wind farm. The description is followed by the specification of the real-world scenario used in this paper.

3.1 Wind Turbine Model

As the objective in this work is to maximize the power output of a wind farm, we apply a wind turbine model to calculate the produced energy of the turbines.

The wind turbine model f exploits a scenario that consists of wind turbines and their power curves based on the Enercon E101, wind data from the German Weather Service, wake effects computation using the Jensen wake model [13], and geographical constraints based on data from OpenStreetMap [3]. The COSMO-DE [2] wind data from the German Weather Service are used to calculate Weibull distributions [18] for every position and wind direction. With the model from Kusiak and Song [8] the power output E is calculated. For one wind turbine with position $\mathbf{t}_i$, it applies:

$$E(\mathbf{t}_i) = \int_0^{360} p_\theta(\mathbf{t}_i, \theta) \cdot E_\theta(\mathbf{t}_i, \theta) d\theta \tag{1}$$

with the power output $E_\theta(\mathbf{t}_i, \theta)$ for one wind direction:

$$E_\theta(\mathbf{t}_i, \theta) = \int_0^{\infty} \beta_i(v) \cdot p_v(v, k(\mathbf{t}_i, \theta), c(\mathbf{t}_i, \theta)) dv. \tag{2}$$

The distribution of wind angles is described in $p_\theta(\mathbf{t}_i, \theta)$, the function $\beta_i(v)$ specifies the power curve of the used wind turbine, and the Weibull distribution $p_v(v, k(\mathbf{t}_i, \theta), c(\mathbf{t}_i, \theta))$ represents the wind speed distribution. In the evolutionary optimization process, the geographical constraints are modeled by a variant of death penalty that is similar to the approach used by Morales and Quezada [11]. For a detailed description of the wind model, we refer to our depiction in [10].

The model f computes the power output of a solution $\mathbf{x}$ that describes the positions of multiple wind turbines for a defined scenario, i.e., a *scenario* specifies the map section and the wind distributions. Thereby, the optimization objective is to maximize the sum of the power output E of all turbines $\mathbf{t}$: $f(\mathbf{x}) = \sum_{i=1}^{N/2} E(\mathbf{t}_i)$. The solution $\mathbf{x}$ is a vector of elements $\mathbf{x} = (x_1, x_2, \dots, x_N)^T$ with the length N coding the x- and y-coordinate of every turbine x^t and y^t, i.e.:

$$\mathbf{x} = (x_1^t, y_1^t, x_2^t, y_2^t, \dots, x_{N/2}^t, y_{N/2}^t).$$

With $\mathbf{t}_i = (x_i^t, y_i^t)$ the solution vector can be written as:

$$\mathbf{x} = (\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_{N/2}).$$

We denote the position $\mathbf{t}_i$ of a specific turbines of solution $\mathbf{x}_j$ as $\mathbf{t}_i^{\mathbf{x}_j}$. This notation is required for the definition of neighborhoods in Section 4.1.

3.2 Scenario

To specify a scenario, we keep in mind the objective to model realistic settings for Lower Saxony, Germany. There are more than $5,000$ wind turbines in Lower Saxony. Most of them are grouped in wind farms smaller than 30 turbines [16]. We define an onshore scenario with 22 turbines in an area of 5 km x 5 km, which leads to a 44-dimensional solution space. To take into account the constraints outside the feasible area, we also consider the geographical information within

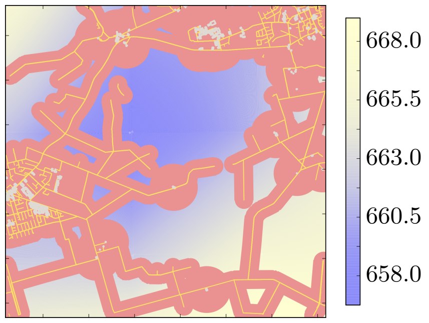


Fig. 1. Visualization of the scenario.

a distance of 1 km beyond each border. The coordinates of the scenario are 53.3925° - 53.45648°, 7.7395° - 7.84304° in decimal degrees. Figure 1 shows the scenario with red constrains and yellow to blue potential map. It consists of 311 buildings and 355 streets consisting of 1987 parts modeled in OpenStreetMap. The potential for a turbine without wake effect by other turbines is about 660 kW becoming lower in the middle of the map. Figure 2 shows an exemplary wind rose in our scenario. Most wind is coming from south-west in this scenario.

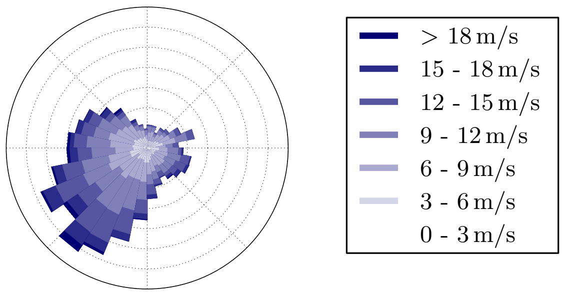


Fig. 2. Wind rose at location of the scenario.

4 Simulated Annealing

In the following, we introduce the SA variants that are compared in the experimental study. First, the concept of SA is introduced. Then, we define the neighborhood of two different solutions $\mathbf{x}_i$ and $\mathbf{x}_j$ using a neighborhood distance d_n, followed by the introduction of two optimization algorithms. The first one is an SA approach with a deterministic cooling schedule. In the second approach, we propose an adaptive control of the neighborhood distance d_n, which is inspired by the step size control used by ES.

SA is based on the cooling process of metallic elements that reduce defects in crystals. The algorithm starts with an initial solution $\mathbf{x}_0$. In our work, we start with a feasible randomly created solution. Then, the optimization algorithm creates a neighbor $\mathbf{x}'$ of the actual solution $\mathbf{x}$. The neighboring solution $\mathbf{x}'$ is analyzed w. r. t. fitness function f. If $f(\mathbf{x}') > f(\mathbf{x})$ the solution $\mathbf{x}'$ replaces $\mathbf{x}$ in the following iteration. For a better chance to leave local optima, SA can also accept worse solutions $\mathbf{x}'$ with a fitness function value $f(\mathbf{x}') < f(\mathbf{x})$. The probability M that describes the chance to accept a worse solution is depending on the difference of the fitness function values $f(\mathbf{x}')$ and $f(\mathbf{x})$ and on a parameter T that is called temperature, i.e.:

$$M(\mathbf{x}, \mathbf{x}', T) = e^{-\frac{f(\mathbf{x})-f(\mathbf{x}')}{T}} \tag{3}$$

The temperature T is variable during an optimization run with SA. Equation 3 shows how the temperature T affects the probability M to accept a worse solution: The higher the temperature T is, the higher is the probability M.

4.1 Turbine-Oriented Neighborhood

To define the neighborhood between two solutions $\mathbf{x}_i$ and $\mathbf{x}_j$, we first specify a neighborhood distance d_n. Solution $\mathbf{x}_j$ is in the neighborhood of solution $\mathbf{x}_i$, if one turbine $\mathbf{t}_k$ with $k \in \{1, \ldots, N/2\}$ is different in both solutions but within the distance d_n:

$$\max\left(\mathbf{t}_k^{\mathbf{x}_i} - \mathbf{t}_k^{\mathbf{x}_j}\right) \leq d_n \tag{4}$$

and all other turbine positions from both solutions are equal:

$$\forall \mathbf{t}_l : \mathbf{t}_l^{\mathbf{x}_i} = \mathbf{t}_l^{\mathbf{x}_j} \tag{5}$$

with $l \in \{1, \ldots, N/2\}$ and $l \neq k$.

4.2 Deterministic Cooling Schedule

In SA, the cooling process starts with a high probability M at the beginning of the optimization process, while M is reduced in the course of the optimization. Various options to reduce M have been introduced in the past, mainly by decreasing temperature T. A common rule to cool down is a deterministic temperature control with $T' = \alpha \cdot T$. Defining i as iteration number and T_0 as starting temperature, the cooling process can be described by:

$$T_i = \alpha^i \cdot T_0 \tag{6}$$

with $0 < \alpha < 1$.

4.3 Approach with Constant Neighborhood Distance

To implement an optimization algorithm using SA with a deterministic cooling schedule, we have to determine the neighborhood distance d_n, the initial temperature T_0, and the cooling factor α. In this work, we use a small and a large neighborhood distance d_n. The small distance d_n^- is set to 50 m, which means a turbine $\mathbf{t}$ can be shifted up to 50 m per iteration in both dimensions. With this distance the solution can reach the next local optimum, but is not able to shift a turbine $\mathbf{t}$ over a constraint like a street. Therefore, we define the large distance $d_n^+ = 500\,\mathrm{m}$ which makes it possible to shift a turbine over a street but the fine-tuning is more difficult.

As we can see in Equation 3, the temperature T must be chosen depending on the scale of the difference $\varDelta f = f(\mathbf{x}) - f(\mathbf{x}')$. Preliminary experiments show that the difference $\varDelta f$ depends on the neighborhood distance d_n. Using d_n^-, in the first 100 iterations with a new feasible solution $\mathbf{x}'$, it applies for the difference $\varDelta f$ that its mean value and standard deviation are $\varDelta f \approx 0 \pm 6$ with $\max(\varDelta f) \approx 20$. It means better and worse solutions are equally distributed, in about 70% of the iterations the difference is smaller than 6, and the maximum value is approximately 20. Using d_n^+, it applies for the first 100 iterations with a new feasible solution $\mathbf{x}'$: $\varDelta f \approx 10 \pm 30$ with $\max(\varDelta f) \approx 120$. We define two different initial temperatures using this information. A high temperature T_0^h has the objective that at the beginning of the optimization process in 10% of the cases a worse solution is accepted. As we are not focusing on extreme values, we use the standard deviation. It applies for d_n^-:

$$M = 0.1/0.7 = e^{-\frac{6}{T_0^h}} \Rightarrow T_0^h = -\frac{6}{\ln(0.1/0.7)} \approx 3.08 \tag{7}$$

And a low temperature T_0^l with the objective to accept 1% of the worse solutions at the beginning of the optimization process:

$$T_0^l = -\frac{6}{\ln(0.01/0.7)} \approx 0.71 \tag{8}$$

For d_n^+, we are using $T_0^h = 15.4$ and $T_0^l = 3.55$. To determine α, we define that the temperature T should be decreased by 10% every 100 iterations. So it applies:

$$\alpha = \sqrt[100]{0.9} \approx 0.99895 \tag{9}$$

4.4 Approach with Adaptive Neighborhood Distance

In this section, we propose an approach based on the algorithm using the deterministic cooling schedule from the last section extending it with an adaptive technique. In the field of ES, an adaptive step size control is a common tool to improve optimization results. The idea is that the solution space conditions change during an optimization run and therefore the optimal step size is not the same during the whole run. At the beginning, a large step size allows the exploration of the solution space, while at the end, a small step size allows the fine-tuning of solutions. A well-known example is Rechenberg's step size control [1]. In our approach using SA, the neighborhood distance d_n play a similar role like the step size, i.e., controlling the exploration characteristics of the algorithm. According to Rechenberg's step size control, we count the number of improved solutions $\mathbf{x}'$ with $f(\mathbf{x}') > f(\mathbf{x})$ and compute ratio of improved solutions w.r.t. all solutions. If more than 1/5th of the new solutions have been improved, the neighborhood distance d_n is increased. Otherwise, it is decreased. To become independent from short-term fluctuations, we evaluate the ratio after 100 new solutions. To increase or decrease, the neighborhood distance d_n is modified with factor $\tau = 1.1$ representing a change of 10%.

Algorithm 1 Adaptive Neighborhood Distance

```
Require: $d_n$, $T$, $\alpha$
  $\mathbf{x} \leftarrow \mathbf{x}_0$, $i \leftarrow 1$, $o \leftarrow 0$
  while $i \leq I$ do
    Create neighbor $\mathbf{x}'$ from $\mathbf{x}$
    if $f(\mathbf{x}') > f(\mathbf{x})$ then
      $\mathbf{x} \leftarrow \mathbf{x}'$
      $o \leftarrow o + 1$
    else if $U \sim \mathcal{U}[0,1] > e^{-\frac{f(\mathbf{x})-f(\mathbf{x}')}{T}}$ then
      $\mathbf{x} \leftarrow \mathbf{x}'$
      $d_n \leftarrow d_n \cdot 2.0$
    end if
    if $i \mod 100 = 0$ then
      if $o \geq 20$ then
        $d_n \leftarrow d_n \cdot 1.1$
      else
        $d_n \leftarrow d_n / 1.1$
      end if
      $o \leftarrow 0$
    end if
    $T \leftarrow \alpha \cdot T$, $i \leftarrow i + 1$
  end while
  return $\mathbf{x}$
```

Preliminary experiments show that the acceptance of a worse solution can change the optimization process. The adaptive control may result in an inappropriate neighborhood distance d_n. To prevent this, we implement a neighborhood distance boost b_{d_n} in case of accepting a worse solution. Hence, the optimization process can explore a larger area after taking a worse solution and thus it is less sensitive to changes in the solution space. We set this neighborhood distance boost to $b_{d_n} = 2.0$, which turned out to be reasonable in our experimental studies. The neighborhood distance d_n is doubled after accepting a worse solution.

Algorithm 1 shows the pseudocode of the optimization approach with the total number of iterations I and a random value $U \sim \mathcal{U}[0, 1]$.

5 Experimental Results

Every optimization is run for $10,000$ iterations. As we use SA which is a heuristic optimization approach and also apply random initializations, we repeat every experiment 100 times and interpret the mean value and standard deviation. Additionally, we test the significance of the experimental results with a Wilcoxon signed rank-sum test [7].

5.1 Comparison of the Configurations

We use eight different configurations to test the capability of SA for the wind turbine placement problem with geographical constraints. The optimization runs are separated into two categories. First, runs that use the deterministic cooling schedule with a constant neighborhood distance and second, runs that use the adaptive neighborhood distance. In each category, we test the initial neighborhood distances d_n^- and d_n^+ and the starting temperatures T_0^h and T_0^l.

Table 1. Experimental comparison between constant and adaptive neighborhood distance control.

Algorithm	Mean	$\pm$	Std	Max
	P in kW	$\pm$	P in kW	P in kW
Without Optimization	12 923.81	$\pm$	186.64	13 397.48
Constant Neighborhood Distance	13 233.57	$\pm$	183.49	13 598.25
Adaptive Neighborhood Control	13 384.33	$\pm$	121.81	13 654.61

Table 1 shows the comparison of the experimental results for the two different categories including various configurations. The values specify the average power production from the wind turbines in kilowatts. Both approaches are able to clearly improve the initial solution. The approach with adaptive neighborhood control performs significantly better than the approach with a constant neighborhood distance, confirmed by a Wilcoxon signed rank-sum test with a p-value $= 6.39 \cdot 10^{-30}$. It should also be noted, that the approach with the adaptive neighborhood control is able to reduce the standard deviation considerably, which means that the approach is more reliable. Also the best solution is created by the approach with adaptive neighborhood control.

In Table 2 concentrates on a detailed comparison between different configurations for start temperatures and neighborhood sizes. First, we can observe that the random initialization leads to slightly different initial values. But clearly confirmed by a Wilcoxon signed rank-sum test with a p-value $= 0.761$, there is no significant difference between the initial solutions. With a constant neighborhood distance, the approaches using T_0^l perform better than the approaches using

Table 2. Experimental comparison with various start temperatures and neighborhood sizes.

	Constant Neighborhood Distance		Adaptive Neighborhood Distance	
Algorithm	Mean ± Std P in kW ± P in kW	Max P in kW	Mean ± Std P in kW ± P in kW	Max P in kW
Without optimization	12 925.07 ± 182.41	13 309.67	12 922.55 ± 190.78	13 397.48
SA with T_0^l & d_n^-	13 340.32 ± 123.44	13 598.25	13 401.58 ± 132.44	13 624.78
SA with T_0^h & d_n^-	13 173.90 ± 140.71	13 418.70	13 402.24 ± 101.37	13 633.63
SA with T_0^l & d_n^+	13 370.38 ± 111.74	13 565.51	13 395.63 ± 123.92	13 643.72
SA with T_0^h & d_n^+	13 049.68 ± 140.22	13 306.36	13 337.86 ± 115.20	13 654.61

T_0^h. This is probably due to the fact, that a higher temperature increases the number of accepted worse solutions. As we have a highly complex 44-dimensional solution space, too many accepted worse solutions can reduce the quality of the optimization. This effect is increased by the use of a larger neighborhood distance, as we can see comparing the results of SA with T_0^h & d_n^- and SA with T_0^h & d_n^+ using a constant neighborhood distance.

The adaptive neighborhood distance adapts itself to the solution space and can counteract this effect. But also here, we can observe that the worst mean value is achieved by SA with T_0^h & d_n^+. Interestingly, this configuration created the best overall solution. Although the higher acceptance rate of worse solutions decreases the optimization results in mean, there is a small chance that the algorithm chooses exactly the worst solutions which helps to leave local optima. A run with *good* choices is able to create the best solution with this configuration. This indicates that it might be promising future research to further analyze the highly complex solution space. This may allow better predictions, if accepting worse solutions may help to leave local optima. Between the approaches with the best mean values SA with T_0^l & d_n^-, SA with T_0^h & d_n^-, and SA with T_0^l & d_n^+ using the adaptive neighborhood distance is not significant difference. With carefully chosen parameters, the adaptive neighborhood control is able to operate well.

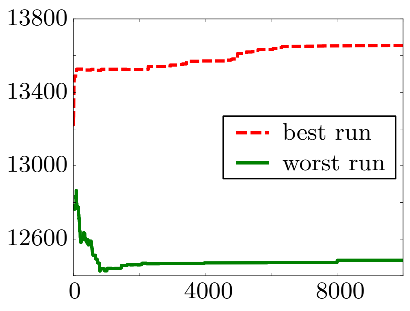


Fig. 3. Comparison of the best and worst optimization run.

5.2 Optimization Runs

Figure 3 shows the dynamic of the best with and the worst run. Both runs are achieved by SA with T_0^h & d_n^+ but the best run uses the adaptive neighborhood distance while the worst run uses the constant neighborhood distance. The important parts of the fitness development are enlarged and visualized in Figure 4. The y-scale in Figure 4(a) and 4(b) is different by factor 5 because the changes in the worst run are larger than the changes in the best run. In the best run, we can see the acceptance of worse solutions, but only with few deteriorations of the fitness function. This observation is different in case of the worst solution. Clearly worse solutions are accepted and the optimization process is unable to improve the fitness function to the starting level. This also explains, why it is possible that the best solution optimized with SA with T_0^h & d_n^+ is worse than the best initial solution, see Table 2.

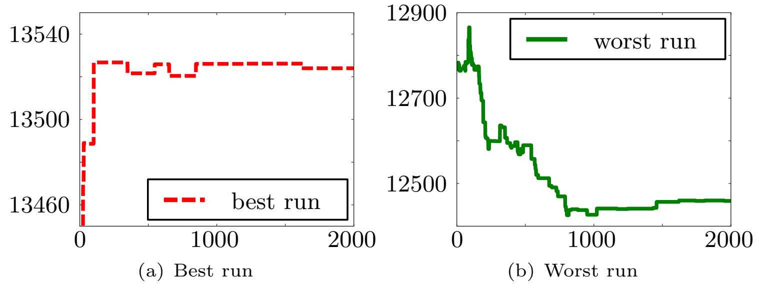
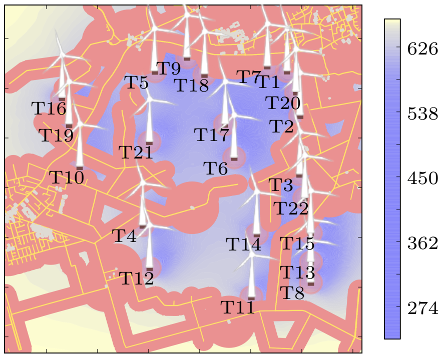


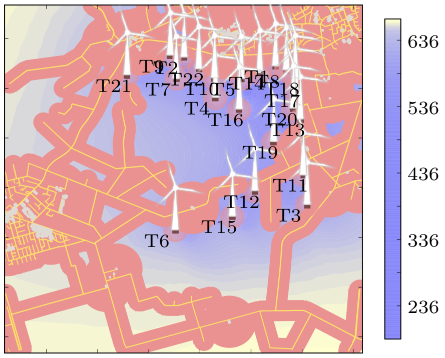
Fig. 4. Zoomed visualization of best and worst optimization run.

5.3 Placement Result

In the last experimental section, we show the placement results of the best solution, see Figure 5(a), and the worst solution, see Figure 5(b). The best solution maximizes the distances between the turbines w. r. t. the wind rose, see Figure 2. We can observe four lines of turbines. The largest line has been placed on the right, because most of the wind comes from direction south-west. This line of turbines does no cause wake effects for other turbines. The second line on the right is curved, also reducing the wake effects, e.g., for Turbine $T9$ and $T18$, as the wind comes from direction south-west. The geographical constraints are considered, e.g., on the left of the area with Turbines $T10$, $T16$, $T19$ placed in the free areas between the constraints. The placement of the worst solution is interesting. The turbines are pulled to the upper right corner. The distances between the turbines are small, so major wake effects reduce the power output. We can also see this behavior in other solutions created with a high temperature. Accepting to many worse solutions can lead to deadlock situations in the



(a) Best placement



(b) Worst placement

Fig. 5. Placement results.

optimization process, where multiple turbines blockade each other, i.e., increasing the distance between two turbines will decrease the distance to a different turbine. An aggravation of this effect can be the geographical constraints. SA has serious difficulties to resolve deadlock situations. Again, this confirms the need to analyze in detail, if accepting worse solutions in highly complex solution spaces may help to leave local optima and avoiding deadlock situations.

6 Conclusions

Finding optimal turbine locations is important for the power output of wind turbines. The optimization process to improve the positions of turbines induces a highly complex solution space. SA is able to optimize the solutions, but the choice of appropriate parameters for neighborhood distance and temperature is important. Our approach using an adaptive neighborhood distance is more reliable than the variant with constant neighborhood distance and can significantly improve the results.

Our experiments show that an intelligent choice of accepted worse solutions can be a promising field for further research. Further, the integration of additional techniques from ES to SA may be an interesting research direction, e.g., the employment of a population like in parallel SA. The success of ES in turbine placement is an indicator that a smart combination of both fields could improve the results.

References

1. H. Beyer and H. Schwefel. Evolution strategies - A comprehensive introduction. *Natural Computing*, 1(1):3–52, 2002.
2. German Weather Service. COSMO-DE: numerical weather prediction model for Germany, 2012. http://tinyurl.com/dwd-cosmo-de.

3. M. M. Haklay and P. Weber. Openstreetmap: User-generated street maps. *IEEE Pervasive Computing*, 7(4):12–18, Oct. 2008.
4. N. Hansen. The CMA evolution strategy: a comparing review. In *Towards a new evolutionary computation. Advances in estimation of distribution algorithms*, pages 75–102. Springer, 2006.
5. J. F. Herbert-Acero, O. Probst, P.-E. Rethore, G. C. Larsen, and K. K. Castillo-Villar. A review of methodological approaches for the design and optimization of wind farms. *Energies*, 7(11):6930–7016, 2014.
6. L. Ingber. Simulated annealing: Practice versus theory. *Mathl. Comput. Modelling*, 18:29–57, 1993.
7. G. Kanji. *100 Statistical Tests.* SAGE Publications, London, 1993.
8. A. Kusiak and Z. Song. Design of wind farm layout for maximum wind energy capture. *Renewable Energy*, 35(3):685–694, 2010.
9. D. Lückehe, O. Kramer, and M. Weisensee. An evolutionary approach to geo-planning of renewable energies. In *28th International Conference on Informatics for Environmental Protection: ICT for Energy Effieciency (EnviroInfo)*, pages 501–508, 2014.
10. D. Lückehe, M. Wagner, and O. Kramer. On evolutionary approaches to wind turbine placement with geo-constraints. In *Genetic and Evolutionary Computation Conference, GECCO '15, Madrid, Spain, July 11-15, 2015*, pages 1223–1230, 2015.
11. A. K. Morales and C. V. Quezada. A universal eclectic genetic algorithm for constrained optimization. In *In: Proceedings 6th European Congress on Intelligent Techniques and Soft Computing (EUFIT)*, pages 518–522. Verlag Mainz, 1998.
12. F. Neumann and C. Witt. *Bioinspired Computation in Combinatorial Optimization: Algorithms and Their Computational Complexity.* Springer-Verlag New York, Inc., New York, NY, USA, 1st edition, 2010.
13. H. Neustadter. Method for evaluating wind turbine wake effects on wind farm performance. *Journal of Solar Energy Engineering*, pages 107–240, 1985.
14. Y. Nourani and B. Andresen. A comparison of simulated annealing cooling strategies. *Journal of Physics A: Mathematical and General*, 31:8373, 1998.
15. R. Rivas, J. Clausen, K. Hansen, and L. Jensen. Solving the turbine positioning problem for large offshore wind farms by simulated annealing. *Wind Engineering*, 33:287–297, 2009.
16. The Wind Power. Wind farms in Lower Saxony, Germany, 2015. http://tinyurl.com/parks-lower-saxony.
17. M. Wagner, K. Veeramachaneni, F. Neumann, and U.-M. O'Reilly. Optimizing the layout of 1000 wind turbines. In *European Wind Energy Association Annual Event*, 2011.
18. W. Weibull. A statistical distribution function of wide applicability. *Journal Applied Mechanics - Transactions of ASME*, 3(18):293–297, 1951.