Optimizing Energy Output and Layout Costs for Large Wind Farms using Particle Swarm Optimization

Kalyan Veeramachaneni*, Markus Wagner†, Una-May O'Reilly*, and Frank Neumann†
*Massachusetts Institute of Technology, Cambridge, MA, USA
[kalyan, unamay]@csail.mit.edu
†School of Computer Science, University of Adelaide, Australia
[markus.wagner, frank.neumann]@adelaide.edu.au

***Abstract*—The design of a wind farm involves several complex optimization problems. We consider the multi-objective optimization problem of maximizing the energy output under the consideration of wake effects and minimizing the cost of the turbines and land area used for the wind farm. We present an efficient particle swarm optimization algorithm that computes a set of trade-off solutions for the given task. Our algorithm can be easily integrated into the layout process for developing wind farms and gives designers new insights into the trade-off between energy output and land area.**

***Index Terms*—Particle Swarm Optimization, repair strategies, renewable energy.**

I. Introduction

Evolutionary algorithms are good problem solvers for a variety of complex optimization problems. This paper contributes to the application of these techniques to the production of renewable energy. We are pursuing optimization challenges arising in wind power generation. The problem we study here arises during the preliminary phase of a wind turbine farm project. The "farm layout" problem entails the process of planning the placement of turbines (and supporting equipment) on a potential wind farm site, while at the same time engaging in multiple dialogues with stakeholders (1) to address their objectives, and (2) to gather their input towards the design of an "optimal" layout. In this layout, the cost of energy forecast to be delivered by the farm is to be minimized, when taking all constraints, costs of generation, and energy capture into account. Examples of supporting equipment are cables, service modules and internal roads. The site itself can be on land or water. Stakeholders include representatives of the community, financiers, developers and turbine manufacturers. Constraints include a minimum turbine spacing based upon rotor diameter, topographical land features. The energy capture must account for loss of energy due to wake effects. The creation of a farm layout involves the invocation (usually more than once) of a software optimization module which attempts to efficiently place the turbines while adhering to the constraints and optimizing the stated objectives. Often, this module is embedded within a specialized tool provided by wind power consultants such as Garrad Hassan or AWS TruePower, who offer a product such as OpenWind [1].

Wan et al, [5], [6], [7] used a cell based approach and compared different bio-inspired algorithms including evolution strategies and particle swarm optimization on the same set of wind farm models and parameters. They used successively more expressive layout representations (and algorithms) and relaxed the positions where in a cell a turbine can be located. The options for positioning a turbine where strictly in the middle of a cell, anywhere in a cell, or anywhere in a cell subject to proximity constraints with neighbouring turbines.

Kusiak et al, [2], exploited an alternative approach to the cell placement. Here each turbine's location is a decision variable pair of real-valued, spatial (x,y) coordinates. With this representation, many more layouts are possible. In [2], a multi-objective evolutionary strategy is used, but only settings for up to 6 turbines are considered. They used as a second objective the minimization of turbine proximity constraint violations. They confine the model farm area to a $500m$ radius and cannot identify even one feasible solution in it for a larger number of turbines. Wagner et al [4] studied a more powerful evolution strategy, called CMA-ES, for the placement of wind turbines such that they achieve a maximum amount of energy. The approach in [4] allows the effective optimization of huge layouts for up to 1000 turbines. However, in this process, the number of turbines to be placed, and the available area have to be set in advance.

In this paper we describe the design of a multi-objective particle swarm optimization (PSO) algorithm to act as a farm layout optimization module. An innovative contribution of this work is that it handles two conflicting objectives: the maximization of energy capture and the minimization of layout costs. The former objective is straightforward. It expresses how much electric power the farm can be expected to generate. The latter objective reflects the reality of costs in the form of land, number of turbines, and fixed and variable costs that are incurred.

Our new algorithm can efficiently optimize the layout of hundreds of turbines. This allows it to accommodate the emerging requirements of larger farms. For example, the Horse Hollow Wind Energy Center in Texas, USA operates with 735.5 megawatt (MW) capacity and consists of more than

300 turbines spread over nearly 47,000 acres (190 km^2). By using PSO we intend to develop a lighter weight algorithm that is sufficiently adept at generating layouts that can be integrated flexibly into the farm layout process which may result in changes to them anyway. There are two complexities arising from placing high quantities of turbines. The first is the increasing frequency of infeasible layouts as the number of turbines increases. Infeasibility can arise for a number of reasons. In this contribution, we present two layout repair strategies that deal with possible infeasible layouts and compare them. One is a First-Come-First-Removed strategy. When a turbine is placed in the layout area by reading its Cartesian spatial coordinates from the particle vector, it immediately removes that turbine if it is found to be too close to one already placed in the layout area. Our alternate strategy is called "Worst-First Removal". This strategy first places all the turbines out on the layout area (by reading the particle vector) then counts, for each turbine, the number of turbines to which it is too close. It then removes the turbine with the most violations, re-calculates the violation counts and iterates until no more violations exist.

The second complexity is the scaling cost of modeling wake effect when estimating energy capture for increasing numbers of turbines. To estimate the energy capture of a layout the optimization module models the free stream wind flowing through the site in and out of the turbines. Some degree of non-linear wind turbulence occurs at the outflow of a turbine and affects the inflow to turbines close enough behind it. Modeling this effect is necessary because wake has a great effect on wind resource but the modeling is computationally expensive: as the number of turbines increases, the cost of modeling wake effects for a given layout increases quadratically. We will explain the wake effect modeling in technical detail later.

We proceed as follows: Section II provides a description of the layout optimization problem. In Section III, we describe the multi-objective PSO algorithm and two repair strategies. The results of our experimental investigations are reported in Section IV. Section V summarizes our findings and mentions future work.

II. Layout Optimization

We now proceed to formulate the farm layout module optimization problem. Let $X = \{x_1, \dots, x_n\}$ and $Y = \{y_1, \dots, y_n\}$ be the x and y coordinates of the n turbines. Based on these coordinates our objectives are to maximize energy output of the whole wind farm and to minimize the costs associated with land area used for the placement of the turbines and the number of turbines. At this level, the goal of the optimizer is defined by

$$\max\{E^{farm}\}, \min\{C^{farm}\} \tag{1}$$

where E^{farm} is the energy capture achieved by the farm, and C^{farm} is the costs associated with the farm. In the following subsections we present the mechanisms to evaluate these two objectives.

A. Energy Output

Depending on the chosen coordinates the overall energy output of the wind farm varies as we have to take the wake effects into account. Table I provides a reference of the symbols we use.

We consider the Park wake model. In this model the wake effects on a turbine i change the wind resource available to it along different directions by reducing the *scale* parameter c of the Weibull distribution estimated for the entire farm, which is also called the freestream wind resource. This is dependent on its location and the location of the rest of the turbines. Hence, we have a parameter c_i for each turbine i which involves a complex computation. We refer the reader to [2] for a detailed presentation on the computation of this parameter when considering wake effects in the Park wake model. The energy output of the whole wind farm is given by

$$E^{farm}[\eta] = \sum_i \int_\theta P(\theta) \int_v p(v(\theta), c_i(\theta, X, Y), k(\theta))\beta^i(v). \tag{2}$$

In this equation v is the wind speed, and the function $\beta^i(v)$ is the power curve for turbine i. Wind speed v however is a random variable with a Weibull distribution, $p(v(\theta), c_i(\theta), k(\theta))$, which is estimated from wind resource data and considers the wake effect. This distribution is also function of the wind direction, θ which varies from $0^0 - 360^0$. Additionally, wind flows from a certain direction with some probability $P(\theta)$.

B. Cost of a Layout

For a more realistic scenario a layout has associated costs. The simplest case incorporates the costs associated with the land area and the number of turbines in the layout. In this contribution we associate these two costs with the layout. We form a second objective that is a sum of these costs. The second objective is given by

$$C^{farm} = Area + N, \tag{3}$$

where $Area$ is in km^2 and N is the number of turbines. This implies that one turbine contributes to the layout cost in the same way as 1 km^2 of land.

The objective of the optimization problem is to find layouts that maximize Equation 2 and minimize Equation 3.

In the following subsection, we present the constraints and assumptions we made for the optimization problem.

C. Constraints and Assumptions

We have the following constraints placed on our optimization function:

- Upper bound on the area of the farm: This constraint ensures that we can only place a turbine within a certain area, which is a realistic constraint for most layout problems. For a circular farm with radius r and the origin as the center, this constraint is satisfied *iff* $sqrt(x_i^2 + y_i^2) \leq r, \forall i$. For a rectangular farm with

TABLE I
SYMBOL DEFINITIONS

Number of turbines	N
Wind velocity	v
Wind direction	$0^0 < \theta < 360^0$
Farm radius	r
Rotor diameter	R
Weibull distribution for wind speed	$p(v,k,c) = k/c(v/c)^{k-1}e^{-(v/c)^k}$
Weibull shape parameter	k
Weibull scale parameter	c
Wind direction distribution	$P(\theta)$
Expected power of a single turbine	$E^i[\eta]$
Piecewise power curve of turbine	$\beta(v) = \begin{cases} 0 & v < v_{cut\_in} \\ \lambda v + \gamma & v_{cut\_in} \leq v \leq v_{rated} \\ P_{rated} & v_{rated} < v < v_{cut\_out} \end{cases}$

length l and width w this constraint is satisfied *iff* $0 \leq x_i \leq l$ & $0 \leq y_i \leq w, \forall i$.

- Proximity constraint: This constraint ensures a minimal distance between any pair of turbines. The constraint is satisfied *iff* $\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \geq MR, \forall i \forall j$ where R is the rotor radius and M is a proximity factor usually decided ahead of the optimization based on the make and model of the turbines used. We use $M = 8$ based on the industry standard.

In addition to the above constraints, we assume that all turbines have the same power curves (approximated as piecewise linear functions) and that the same wind resource spans the entire farm.[1] The assumptions can be revised in a very straight forward manner to generate more realistic scenarios.

III. MULTI-OBJECTIVE PARTICLE SWARM OPTIMIZATION FOR THE WIND FARM LAYOUT PROBLEM

We design particle swarm optimization algorithm for the wind layout problem. Each *particle* is a potential solution to the problem, i.e. a layout. Therefore, a particle is a vector of N (x, y) coordinates implying *2N* search variables. The search space is continuous and of dimension *2N*. The i^{th} particle in the swarm is represented as a vector $Z_i = (z_{i1}, z_{i2} \dots z_{i2N})$. The algorithm starts out by initializing each particle randomly. Each particle maintains in its memory the personal best particle P_i it has found so far. Furthermore the global best particle P_g which represents the best solution found so far is stored as a vector P_g. Each particle is evaluated using the objective function and adapted using velocity and position update equations. The velocity update equation for the velocity i^{th} particle is given by

$$\begin{aligned} V_i^{(t+1)} &= \omega V_i^{(t)} + \psi_1(P_i^{(t)} - Z_i^{(t)})U[0,1] + \\ & \quad \psi_2(P_g^{(t)} - Z_i^{(t)})U[0,1] \end{aligned} \tag{4}$$

[1]For additional accuracy, these resources can be estimated for different parts in the farm.

where ψ_1 and ψ_2 are parameters determining the influence of the personal best and the global best particle. Furthermore, $U[0,1]$ is a real value that is chosen in every velocity uniformly from the interval $[0,1]$.

The position of particle i in the next iteration is given by

$$Z_i^{(t+1)} = Z_i^{(t)} + V_i^{(t+1)}. \tag{5}$$

Once all particles are updated using the above two equations, they are again evaluated using the objective function and a particle's $P_i$vector is updated if it is the best layout the particle has yet found. The best performing layout of the swarm P_g is appropriately updated too. The entire process iterates until a termination condition is fulfilled.

A. Multi-Objective PSO

We want to apply PSO to multi-objective wind farm layout problems. The multi-objective particle swarm algorithm (MOPSO) for turbine layout is described in Algorithm 2 and relies on the general approach of multi-objective particle swarm optimization presented in [3]. To deal with multi-objective problems, we change the way the P_i particles are updated and the way P_g is selected. Our goal is to ensure that the particles are forced to progress towards each objective during the run of the algorithm. To update the P_g usually a crowding distance measurement or a niche count measurement is used ensuring a diverse set of solutions in the objective space. However, in this paper we use the P_g as the particle with the highest energy capture. This works well with the techniques we designed in this paper to generate layouts from particles while satisfying all the layout constraints. Algorithm 1 is used to update the personal best position P_i of particle i.

B. Constraint Handling

One constraint of the layout optimization is that no turbine can be less than 4 rotor diameters from any other. To check whether the constraint is violated we first construct a matrix D

Algorithm 2: Multi-objective particle swarm optimizer for Wind Farm Layout Optimization

```
1  Initialize the particles $Z_i^{(0)}$, $1 \leq i \leq n$, uniformly at random within the range for each dimension.;
2  Set personal best particle $P_i^{(0)} = Z_i^{(0)}$ and initial velocity $V_i^{(0)} = 0$, $1 \leq i \leq n$;
3  Initialize parameters of PSO, $\omega = 0.8$, $\psi_1 = 1$, $\psi_2 = 1$;
4  Randomly initialize $P_g^{(0)}$ and $g$;
5  for $i = 1$ to $n$ do
6      Evaluate $E^{farm}$ and $C^{farm}$ for $Z_i$;
7  for $t = 0$ to $maxiter - 1$ do
8      for $i = 1$ to $n$ do
9          $V_i^{(t+1)} = \omega V_i^{(t)} + \psi_1(P_i^{(t)} - Z_i^{(t)})U[0,1] + \psi_2(P_g^{(t)} - Z_i^{(t)})U[0,1]$;
10         $Z_i^{(t+1)} = Z_i^{(t)} + V_i^{(t+1)}$;
11         Evaluate $E^{farm}$ and $C^{farm}$ for $Z_i^{(t+1)}$;
12     Compute the set $\bar{P}^{(t+1)} = \{P_i^{(t+1)} \mid 1 \leq i \leq n\}$, using Algorithm 1;
13     Identify $P_g^{(t+1)}$: particle with maximum energy capture found so far;
14     Store $P_g^{(t+1)}$ for iteration $t + 1$ ;
15     $t \leftarrow t + 1$ ;
16 Output $P_g^{(maxiter)}$;
```

Algorithm 1: Computation of the set of personal best particles $\bar{P}^{(t+1)} = \{P_i^{(t+1)} \mid 1 \leq i \leq n\}$

```
1) Initialize $\bar{P}^{t+1}$ to be empty.
2) Let $\bar{ZC} = \bar{Z} \cup \bar{P}$ where $\bar{Z} = \{Z_i^{(t)} \mid 1 \leq i \leq n\}$, and
   $\bar{P} = \{P_i^{(t)} \mid 1 \leq i \leq n\}$.
3) Identify the set of non-dominated solutions $\bar{ND}$ in $\bar{ZC}$.
4) Add $\bar{ND}$ to $\bar{P}^{t+1}$.
5) If $|\bar{P}^{t+1}| < n$, remove $\bar{ND}$ from $\bar{ZC}$ and goto Step 3).
```

in which d_{kj} represents the euclidean distance between turbine k and turbine j. Then

$$c = \begin{cases} \text{true} & \text{if } \sum_{\forall k,j,k\neq j} u(d_{k,j} - \delta) \geq 1 \\ \text{false} & \text{otherwise} \end{cases} \tag{6}$$

where u is the unit step function and δ is the minimum distance constraint imposed by the designer. We have designed two strategies that repair an infeasible layout where this constraint is violated.

1) First-Come-First-Removed: We generate a layout, L_i by scanning the particle, Z_i from left to right ($j = i \ldots N$) and incrementally placing a turbine in the layout area according to its Cartesian coordinates ($z_{ij} = (x_{ij}, y_{ij})$). We check whether this turbine is too close to any other turbine already in the layout area. If it is, it is removed from the layout and the number of turbines goes down by one. This removal algorithm is described in Algorithm 3.

2) Worst-First Removal: Like First-Come-First-Removed, we remove turbines that cause violations. In this method, however, we rank the turbines based on their number of proximity violations. Let L_i be the i^{th} particle and is equivalent to

$$L_i = \{T_1, \ldots T_n\} \tag{7}$$

where T_k is the k^{th} turbine located at $= (x_k, y_k)$. Let $D = \{d_{k,j}\}$, $\forall j, j \neq k$ be the set of its Euclidean distances from all the other turbines. The rank of the turbine k is given by

$$\rho_k = \sum_{j,j\neq i} u(d_{k,j} - \delta), \tag{8}$$

where δ is the minimum distance constraint imposed by the designer and is MR as mentioned in Section II-C. u is a unit step function. We then remove the turbine with highest rank and re-iterate calculation of violations and rankings with the new smaller (by one) set of turbines. Ranking removes turbines from the most crowded regions in the layout. The stepwise details of this method are presented in Algorithm 4.

The constraint on the maximum area the layout can occupy is addressed by expressing area in terms of its cost within the layout cost then minimizing layout cost as the second objective of the optimization algorithm.

IV. Experimental Investigations

To evaluate our alternate repair methods and the optimization, we set up two scenarios with the wind resources defined as in [2] (see Table II).

The wind direction is binned in 15^0 intervals. Scenario 1 has the same *scale* and *shape* parameters for the Weibull distribution for all bins. Scenario 2 has the same *shape* parameter for the bins, but different *scale* parameters. The *shape* parameter k increases the spread of the Weibull distribution as it gets

TABLE II
WIND SCENARIO 1 AND SCENARIO 2

l	θ^l	θ^{l+1}	Scenario 1			Scenario 2			l	θ^l	θ^{l+1}	Scenario 1			Scenario 2		
			k	c	$P(\theta)$	k	c	$P(\theta)$				k	c	$P(\theta)$	k	c	$P(\theta)$
0	0	15	2	13	0	2	7	0.0002	12	180	195	2	13	0.01	2	10	0.1839
1	15	30	2	13	0.01	2	5	0.008	13	195	210	2	13	0.01	2	8.5	0.1115
2	30	45	2	13	0.01	2	5	0.0227	14	210	225	2	13	0.01	2	8.5	0.0765
3	45	60	2	13	0.01	2	5	0.0242	15	225	240	2	13	0.01	2	6.5	0.008
4	60	75	2	13	0.01	2	5	0.0225	16	240	255	2	13	0.01	2	4.6	0.0051
5	75	90	2	13	0.2	2	4	0.0339	17	255	270	2	13	0.01	2	2.6	0.0019
6	90	105	2	13	0.6	2	5	0.0423	18	270	285	2	13	0.01	2	8	0.0012
7	105	120	2	13	0.01	2	6	0.029	19	285	300	2	13	0.01	2	5	0.001
8	120	135	2	13	0.01	2	7	0.0617	20	300	315	2	13	0.01	2	6.4	0.0017
9	135	150	2	13	0.01	2	7	0.0813	21	315	330	2	13	0.01	2	5.2	0.0031
10	150	165	2	13	0.01	2	7	0.0994	22	330	345	2	13	0.01	2	4.5	0.0097
11	165	180	2	13	0.01	2	9.5	0.1394	23	345	360	2	13	0	2	3.9	0.0317

Algorithm 3: First-Found-First-Removed Repair of a layout

```
1 $L_i = \{\}$;
2 for $j = 1$ to $N$ do
3     Add turbine $T_j$ to $L_i$;
4     $c = Check(L_i)$ for constraint violations;
5     if $c$ then
6         Remove turbine $T_j$ from $L_i$;
7 Return $L_i$;
```

Algorithm 4: Worst-First Removal Repair of a layout

```
1 $L_i = i^{th}$ particle ;
2 while $g$=$1$ do
3     $\overline{\rho} = Rank(L_i)$;
4     Remove turbine $T_j$ where $j = \max of\ \overline{\rho}$ ;
5     $c = Check(L_i)$ for constraint violations;
6     if $c = 0$ then
7         g=2;
8 return $L_i$;
```

larger. In Scenario 1, the dominant wind directions are 75^0 - 90^0 (with $P(\theta) = 0.2$) and 90^0- 105^0 (with $P(\theta) = 0.6$). There is no wind coming from 0^0- 15^0, and 345^0- 360^0. For the rest of the bins the $P(\theta) = 0.01$.

Scenario 2 is more complex and realistic. The *shape* parameter is the same for all bins, however, the *scale* parameter is different for different bins and ranges from 4 - 10. Similarly, $P(\theta)$ also varies over the range 0.001 to 0.1839. It is more difficult to nominally identify competent layouts as there is no prominent wind direction. In Scenario 1 one can optimize for the prominent directions and not lose significant efficiency. In Scenario 2, one has to optimize the layout to work with minimum wake loss along all the wind directions.

A. Algorithm settings

To successfully run a particle swarm optimization on this constrained optimization problem we have to specify a variety of settings including initialization.

- Initialization of particles: In our layout optimization problem we are attempting to search for turbine positions and at the same time shrink the search area and reduce the number of search variables while still not violating the constraints. Due to this complexity, initialization of the particles plays an important role. We tried different initialization strategies including a grid based initilization. However our results were best when turbines were placed randomly. Initializing in a grid provided a local optimal solution and the particles could not move without breaking a constraint. We chose two different initialization cases. In *Case 1* we initialize 200 turbines in a $20km \times 20km$ grid. In *Case 2* the grid is sized $10km \times 10km$. No other restrictions were placed on the location of turbines.

- Maximum velocity: Since there is a possibility that particles can attempt to move large distances, an upper and lower bound on the maximum velocity is placed. If the velocity update and hence position update is not bounded, particles can quickly become infeasible due to large displacements in turbine positions. We used $[-38.5, 38.5]$ as the limits for the velocity (and hence the position update). This value is equivalent to the rotor radius.

- Algorithm parameters: Both social and cognitive learning factors, ψ_1 and ψ_2 were set equal to 1. We used a population size of 100 and ran the swarm for 200 iterations giving us 20000 fitness evaluations. A inertia weight of 0.8 was used.

In the following we compare the two different constraint violation repair strategies: First-Come-First-Removed and Worst-First Removal on Case 1. Afterwards, we use

Worst-First Removal repair and explore both initialization cases.

B. Comparison of Repair Strategies

Here we compare the two different constraint violation repair methods: First-Come-First-Removed and Worst-First Removal on Case 1. We collected the results in terms of the Pareto front plots from each run and then constructed what we call a super Pareto front plot for each method. A super Pareto Front is the non-dominated solutions of the union of all final iteration pareto fronts of the runs. Figure 1 shows them. The Worst-First Removal approach works slightly better than the First-Come-First-Removed approach. For the rest of the experiments we focused on the Worst-First Removal approach.

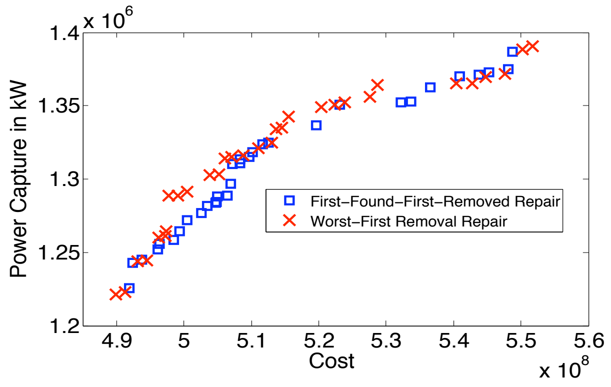

Fig. 1. Super Pareto plots generated for the two repair methods. Worst-First Removal approach performs slightly better than the First-Come-First-Removed approach.

Figure 2 shows the progress of the Pareto front estimate from the 1st iteration, to the 50th and final one of 200 iterations, when using Wind Scenario 2. We notice that majority of the improvements are achieved in the first 50 iterations, more so for Case 1. This is because Case 1 initializes the swarm in a larger area and this makes it easier to navigate and make good progress. However, the progress stagnates after 100 iterations. For Case 2, the smaller land area initialization, the MOPSO still achieve improvements but to a lesser degree. In Figure 3 we present the super Pareto front plots aggregated over 5 independent runs of the algorithm. Because the algorithm stagnated when initialized with the larger area the resulting layouts used a higher area for the same power capture versus initialization with the small area (Case 2). An expert layout designer would prefer the results from Case 2 because they have lower layout cost for approximately the same energy capture. From this we conclude that initialization plays an important role for the multi-objective layout optimization problem. This stems from the fact that the second objective works to shrink the layout area during optimization.

For the Wind Scenario 1, we use Worst-First Removal repair and explore the smaller initialization case (an area only $20X10km^2$). The results are shown in Figure 4. We are able to achieve higher power than in Scenario 2 because the wind has two dominant directions and turbines can be placed in an position ideal for the majority of the time.

V. Conclusions

The design of a wind farm involves a lot of (possible conflicting) optimization problems. In this paper, we have considered the goal of maximizing the energy output under the consideration of wake effects as well as minimizing the layout costs incurred due to the turbines and the land area used. We designed strategies to overcome the infeasible solutions in the search space and developed a particle swarm optimization algorithm that is able to deal with this complex multi-objective optimization problem. Our approach was tested on a 200 turbine layout problem. The results give new insights into the trade-off of these two optimization goals and present multiple competent layouts. Multiple competent layouts are useful in wind farm design since often many constraints and objectives are not expressed prior to the optimization. Our algorithm can be easily used as a tool for helping designers to trade-off these conflicting goals. Further studies should take into account other relevant optimization goals such as cable lengths, electrical subsystems and infrastructure costs.

References

[1] AWS openwind www.awsopenwind.org.

[2] A. Kusiak and Z. Song. Design of wind farm layout for maximum wind energy capture. *Renewable Energy*, 35(3):685 – 694, 2010.

[3] X. Li. A non-dominated sorting particle swarm optimizer for multi-objective optimization. In *Lecture Notes in Computer Science*, volume 2723/2003, page 198. Springer, 2003.

[4] 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.

[5] C. Wan, J. Wang, G. Yang, X. Li, and X. Zhang. Optimal micro-siting of wind turbines by genetic algorithms based on improved wind and turbine models. In *Decision and Control*, pages 5092–5096, 2009.

[6] C. Wan, J. Wang, G. Yang, X. Li, and X. Zhang. Optimal siting of wind turbines using real coded genetic algorithms. *Proceedings of European Wind Energy Association Conference and Exhibition*, 2009.

[7] C. Wan, J. Wang, G. Yang, and X. Zhang. Optimal micro-siting of wind farms by particle swarm optimization. In *Advances in Swarm Intelligence*, volume 6145 of *Lecture Notes in Computer Science*, pages 198–205. Springer, 2010.

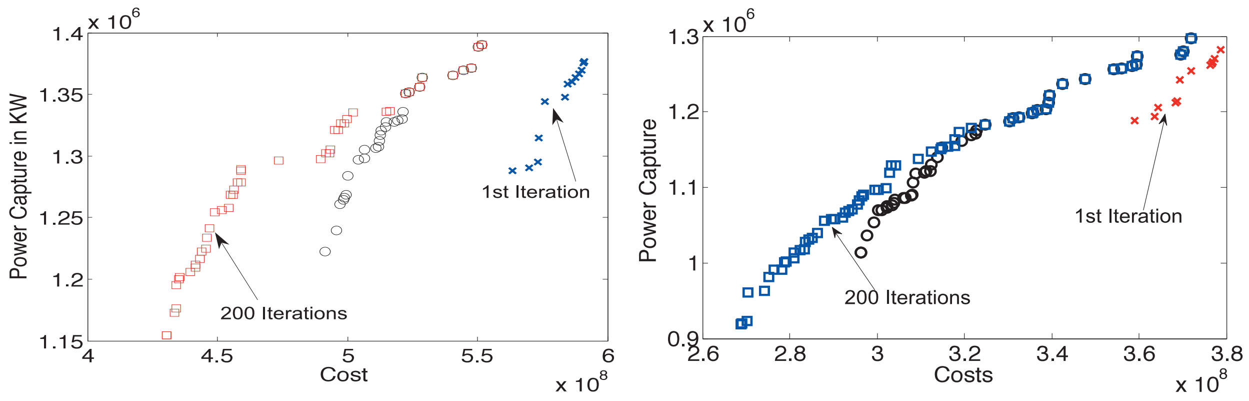

Fig. 2. Wind scenario 2: Estimated pareto front for Case 1 (left) and Case 2 (right). Notice the significant reduction in costs for the same energy capture as iterations progress.

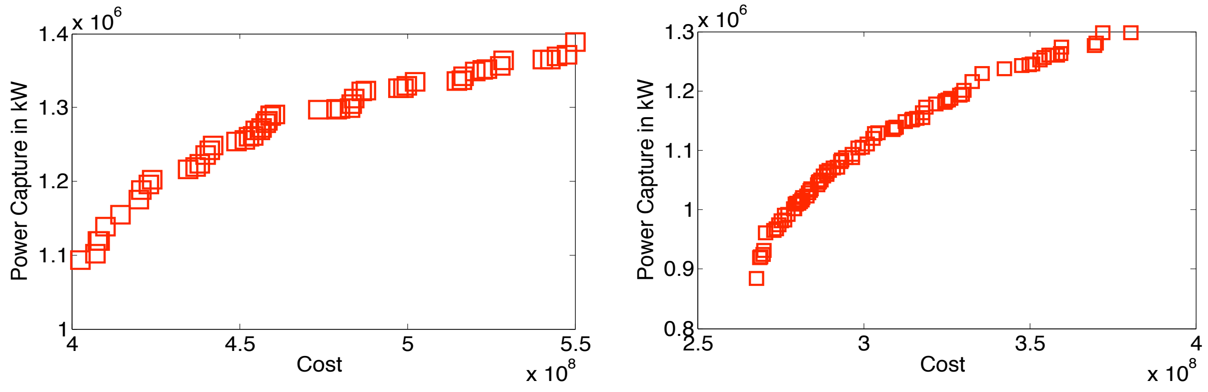

Fig. 3. Wind scenario 2: Super Pareto front aggregrated from 5 independent runs for Cases 1(l) and 2(r).

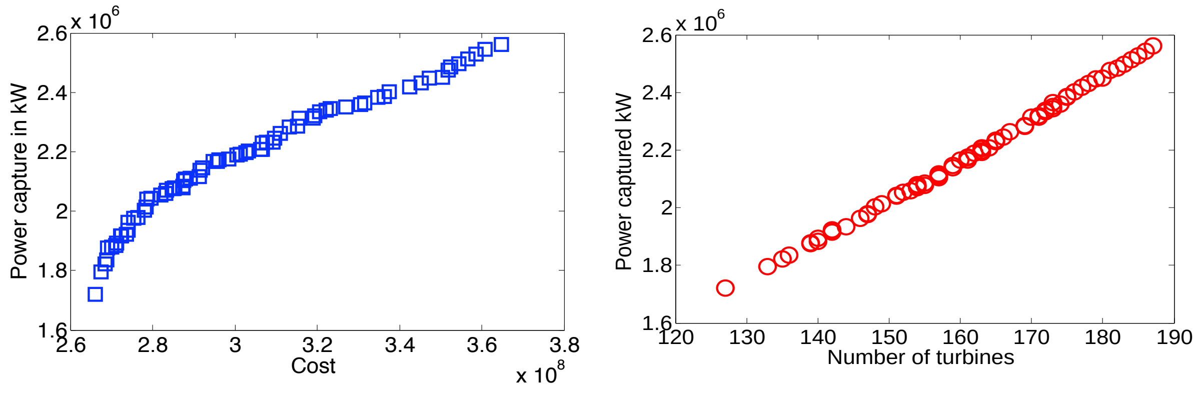

Fig. 4. Results achieved for Wind Scenario 1: Case 1(l) and 2(r).