Progress report Wind Farm Layout Optimization with Particle Swarm Optimization

L. Nies (s4136748) & G. Zuidhof(s4160703)

19 May 2015

Abstract

This progress report is a draft of our final report. Many sections are not complete as of yet. Progress report-only notes are *italicized.*

In this report we describe our particle swarm optimization (PSO) approach that was created as an entry for the GECCO 2015 Wind Farm Layout Optimization competition. Wind farm layout design is a complex problem that is hard to optimize due to discontinuities caused by wake effects and practical constraints. Many previous approaches only consider subsets of wind turbines from a grid in which turbines are placed as close together as possible. Our approach does not limit solutions in such a way and involves searching in a more continuous space.

1 Introduction

With an increasing need for alternative forms of energy, wind energy is increasing in popularity. Huge wind farms are being build to accommodate for this need. An issue in with these wind farms is the placement of the wind turbines in such a way that is maximizes the cost/energy trade off, which is an non-trivial problem. Several factors must be taken into account: wind direction: wake interactions between wind turbines, land availability, etc. Several methods have been applied to optimize this layout, but these is still some room for improvement [8]. We try to contribute to this problem by using Swarm Particle Optimization to find a (close to) optimal solution. We will submit the resulting algorithm to the *Wind Farm Layout Optimization Competition*[4].

2 Problem

The Wind Farm Layout Optimization problem involves optimizing the layout of windmills to maximize the ratio between energy/cost. *To be extended.*

2.1 Wind wake

One of the problems of finding a good layout is not only the large search space, but also the actual calculation of the wind wake model: if windmills are standing a straight line next to each other, say from the north to the south, and the wind comes from the north, the most northern windmill will have the most wind, with each windmill further in the row having less and less wind. If the wind comes from the east or west this is not a problem at all. To calculate this effect from each possible wind direction is computationally incredible expensive. So finding a method that quickly converges to a good solution is desired since approaches which needs a lot of evaluations will simply take too long to calculate. *To be extended.*

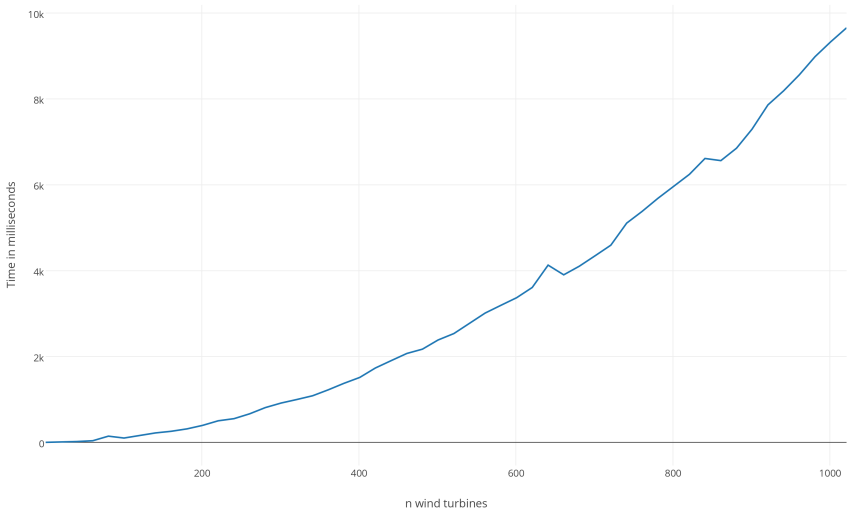


Figure 1: Computation time of competition evaluation function (see next section)

2.2 Wind Farm Layout Optimization Competition

The 2nd Edition of the *Wind Farm Layout Optimization Competition*[4] is part of the collection of competitions hosted for the 24th *GECCO (Genetic and evolutionary computation)* conference[3].

The goal of this challenge is to optimize the wind farm layout of 5 unknown scenarios which differ in wind forces, layout shapes, ect. The fitness of the layout is determined by the function given in figure 2. This cost function is based on the production price of a kilowatt, so it provides a realistic representation of the actually fitness of a real wind farm.

Large layouts will be rewarded with a better cost, but optimizing the placement of the windmills becomes more complex, so that the algorithm solving this problem needs to find a good balance between these two criteria. Another difficulty is the problem of obstacles: this years' competition introduces obstacles to the layout so that is it discontinuous and the windmills cannot be placed everywhere.

2.3 Evaluation

Layouts are evaluated by the function below. Farm energy output is determined by the energy output algorithm developed by Kusiak et al[7].. In this algorithm the energy output of the layout is calculated for twelve wind directions and added.

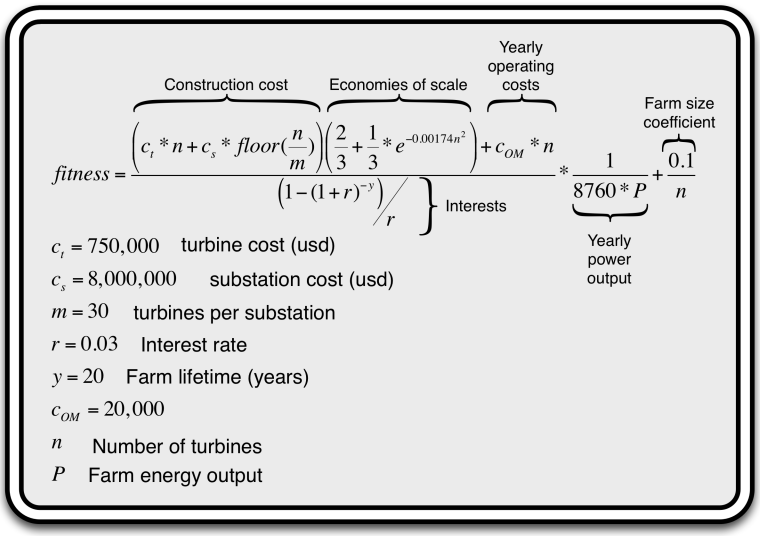


Figure 2: The fitness function of a layout

Submissions to the competition are tested on five previously unpublished scenarios, and for each participant a total of 10,000 layout evaluations are allowed. This limitation makes for an additional challenge regarding computational resource efficiency.

2.4 Technicalities

APIs and Interfaces An API is provided in C++, Java and MATLAB for interacting with an open source wind evaluation model, which can be found on GitHub[1]. The evaluation part of the API is to be called with a n-by-2 matrix, where n is the amount of matrix and the columns are the coordinates of the

turbines. Returned is the fitness of the entire layout (the so called *Wake free ratio* and some other evaluated variables.

Competition deadline Submission deadline is the 15th of June, and the results will be published during GECCO 2015 (July 11-15).

3 Related research

Rašuo et. al. state that there is little work that has tackled the Wind Farm optimisation problem. They suggest that there is a lot of potential in finding improvements on existing solutions. With the rising need for alternative forms of energy the need for more efficient wind farms is obvious [8].

Chowdhury et al. [5] employed a particle swarm algorithm to determine the key factors which influence the power generation of wind farm layouts. This did not investigate the layout as much as it did the individual characteristics of the windmills (such as the rotor diameter), it does however give insights into the influence of the number of turbines on the cost per kilowatt of power produced.

Samorani [9] provides a survey of the existing approaches and applications, and provides some overview of the practical issues that are typical for specific layouts.

Non-PSO, natural computing approaches The following approaches are not particle swarm algorithm approaches, but may still yield valuable insights into how to best approach the problem. Eroglu et al.[6] used an Ant Colony Optimization approach to maximize the energy output of a wind farm layout. Rašuo et al.[8] used an evolutionary algorithm approach. *To be extended*

4 Method

In our approach, each particle depicts a wind turbine and moves around with a certain velocity throughout a 2-dimensional space. The velocity of each particle is affected by it's score on that position and the score of it's neighbouring particles: The better the score of a particle, the "heavier" it becomes, so it is reluctant to move from it's (good) position. Particles with a high score not only prefer to stay where they are, they also apply more force on their neighboring particles, forcing them away if they are not heavy enough (e.g. the have a low score). This way, wind tribunes with a good score stay where they are and turbines with a low score keep traveling, searching for a place to settle.

We are also considering using memory: each particle has a memory of it's best position, which also influences it's velocity (it tries to go back to that spot), but since all the other particles keep moving, what used to be a good position might later on be terrible with other wind turbines catching all the wind. Another option using memory, is remembering the best layout so far. Each particle is

influence in such a way that if there are no other forces applied, that layout is recreated. We are still figuring that part out.

4.1 Continuous search space

In many of the previously described approaches from recent literature the search space is made discrete by only considering wind turbine placement on certain locations on a grid. The cells in this grid are the minimal distance between turbines apart (which is part of the problem description). This bitfield is often used as the genome in genetic algorithm approaches.

Our PSO approach does not limit the search space to these points on the grid only, and instead wind turbines can be placed in any location which does not break constraints (such as being too close to another turbine, or in an area where it can not be built).

5 Particle implementation

Since each particle is moving around with a certain velocity it is bound to hit the boundaries of the wind farm, or an obstacle at some point. Each particle also has to keep a certain distance to the other particles, or the layout is invalid. This distance is given as $8 * R$ in which R is a constant given by the scenario. It denotes the radius of the wind turbines blades. Currently, this constant is the same for all given scenarios, namely 38.5. The minimal distance between each windmill is then 308 (To get a sense of how large this is: each scenario is 7000 by 14000 units large).

5.1 *dyn4j* physics engine

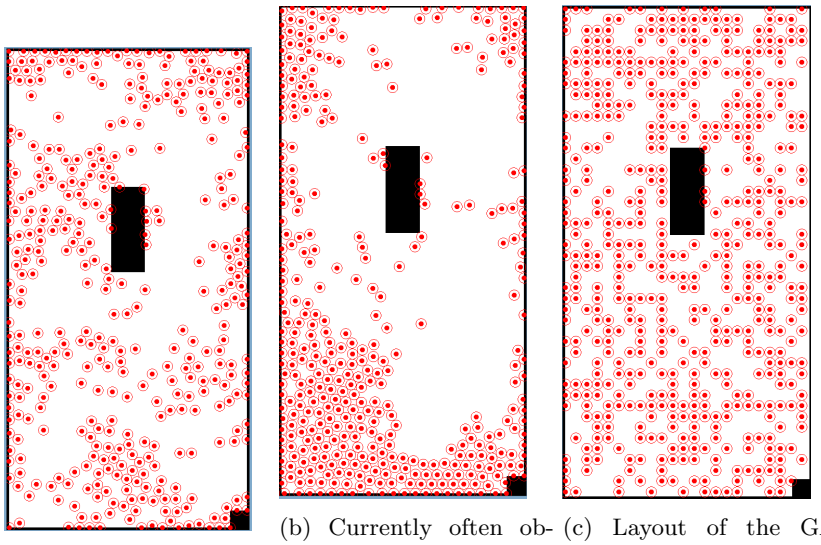
To model not only the trajectory of each particle, but also model the collisions of particles with other particles, objects and the boundaries we use *dyn4j*[2], a collision detection and physics engine for Java (which is often used for game development). This engine is a very useful tool to easily model each particle and their trajectory.

It also allows for modelling their inter-distance constraints by attaching rigid bodies to each of the particles with of half this radius. The boundaries of the region, as well as the obstacles, are also modelled by infinite mass rectangle rigid bodies.

5.2 Visualization

Without visualization it is hard to get a grasp of what is going on during the particle swarm simulation. To solve this we implemented a graphic visualization of a layout which is updated as the wind turbine particles move around. This was implemented using Java AWT and Graphics2D APIs.

In the above figure the center of the red dots denotes the placement of a wind mill, and the circle around it indicates the size of the rigid collision body



(a) Unconverged layout

(b) Currently often observed behavior

(c) Layout of the GA benchmark

Figure 3: Visualization of wind turbine layouts *To do: same dimensions, use final algorithm*

described in the previous section. The black areas are obstacles where no wind turbines can be built.

5.3 *Current particle behaviour*

In our current version each particle moves with a certain velocity through a 2-dimensional space. When it collides with an obstacle, particle or boundary, it bounces back, losing some of it's kinetic energy. Our next step is to apply swarm like behavior to each particle by changing it's velocity based on certain factors, which was explained in section 3

5.4 Pseudocode

Pseudocode to be added to relevant sections

5.5 Benchmarks

If we want to be able to make a claim about how well our approach performs on the given problem, we need to have results we can use for the comparison. Since the results of the competition are published a month after the deadline of this course and there is no open leaderboard with the performance of other teams we need to find a suitable benchmark. Luckily, the API also provides a simple genetic algorithm which we will use as a benchmark. We will also make two other benchmarks, one which produces a random wind farm layout, and one

which places as many windmills in the farm as possible. The results of these three benchmarks will be used to validate our approach.

6 Results

7 Discussion

References

[1] https://github.com/d9w/WindFLO. WindFLO API.

[2] Dyn4j, a java collision detection and physics engine. http://www.dyn4j.org/. Accessed: 2015-05-18.

[3] Genetic and evolutionary computation conference. http://www.sigevo.org/gecco-2015/. Accessed: 2015-04-20.

[4] Wind farm layout competition. http://www.irit.fr/wind-competition/. Accessed: 2015-04-20.

[5] Souma Chowdhury, Jie Zhang, Achille Messac, and Luciano Castillo. Unrestricted wind farm layout optimization (uwflo): Investigating key factors influencing the maximum power generation. *Renewable Energy*, 38(1):16–30, 2012.

[6] Yunus Eroğlu and Serap Ulusam Seçkiner. Design of wind farm layout using ant colony algorithm. *Renewable Energy*, 44:53–62, 2012.

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

[8] Boško P Rašuo and Aleksandar Č Bengin. Optimization of wind farm layout. *FME Transactions*, 38(3):107–114, 2010.

[9] Michele Samorani. The wind farm layout optimization problem. In Panos M. Pardalos, Steffen Rebennack, Mario V. F. Pereira, Niko A. Iliadis, and Vijay Pappu, editors, *Handbook of Wind Power Systems*, Energy Systems, pages 21–38. Springer Berlin Heidelberg, 2013.