Multi-objective optimization of wind farm layouts
Complexity, constraint handling and scalability

Fragoso Rodrigues, Silvio; Bauer, Pavol; Bosman, Peter A.N.

DOI
10.1016/j.rser.2016.07.021

Publication date
2016

Document Version
Final published version

Published in
Renewable & Sustainable Energy Reviews

Renewable and Sustainable Energy Reviews 65 (2016) 587–609

Contents lists available at ScienceDirect

Renewable and Sustainable Energy Reviews

journal homepage: www.elsevier.com/locate/rser

Multi-objective optimization of wind farm layouts – Complexity, constraint handling and scalability

S. Rodrigues [a,*], P. Bauer [a], Peter A.N. Bosman [b]

[a] DC systems, Energy conversion & Storage Group, Delft University of Technology, Delft, The Netherlands
[b] Centrum Wiskunde & Informatica, Amsterdam, The Netherlands

ARTICLE INFO

Article history:
Received 15 April 2015
Received in revised form
1 February 2016
Accepted 4 July 2016
Available online 22 July 2016

Keywords:
Wind farm layout design
Offshore wind farms
MO optimization
Constraint-handling techniques
MOGOMEA
NSGA-II
Scalability

ABSTRACT

Currently, Offshore Wind Farms (OWFs) are designed to achieve high turbine density so as to reduce costs. However, due to wake interferences, densely packing turbines reduces energy production. Having insight into optimized trade-offs between energy production, capital investment and operational costs would be valuable to OWFs designers. To obtain this insight, the design of OWFs should be formulated as a multi-objective optimization problem. How to best solve a Multi-Objective Wind Farm Layout Optimization Problem (MOWFLOP) is however still largely an open question. It is however known that evolutionary algorithms (EAs) are among the state-of-the-art for solving multi-objective optimization problems. This work studies the different features that an MO Evolutionary Algorithm (MOEA) should have and which Constraint-Handling Techniques (CHTs) are suitable for solving MOWFLOP. We also investigate the relation between problem dimensionality/complexity and the degrees of freedom offered by different turbine-placement grid resolutions. Finally, the influence of problem size on algorithm performance is studied. The performance of two variants of the recently introduced Multi-Objective Gene-pool Optimal Mixing Evolutionary Algorithm (MOGOMEA) is compared with a traditional and a novel version of the Nondominated Sorting Genetic Algorithm II (NSGA-II). Five CHTs were used to assess which technique provides the best results. Results on a case study with different OWF areas demonstrate that one variant of MOGOMEA outperforms the NSGA-II for all tested problem sizes and CHTs.

Contents

* Corresponding author.
E-mail address: S.M.FragosoRodrigues@tudelft.nl (S. Rodrigues).

http://dx.doi.org/10.1016/j.rser.2016.07.021

1. Introduction

In 2007, the European Union (EU) targeted to generate 20% of its energy consumption through renewable sources and to improve the energy efficiency by 20% compared to 1990 levels, by 2020 [1]. Renewable energy sources are anticipated to help Europe meet these challenging targets. Among other renewable sources, such as hydro, solar and onshore wind, the northern European countries have been investing in Offshore Wind Farms (OWFs) for more than two decades due to higher and steadier mean wind speeds offshore compared to onshore and lower visual impact [2,3].

The EU and the European Wind Energy Association (EWEA) estimated that the joint installed capacity of European OWFs will be 40 GW by 2020 and 150 GW by 2030 [1,4,5]. These predictions require a yearly increase rate of the offshore installed capacity of 29.6% and 19.1% to be satisfied, respectively [6]. Fig. 1 shows that these predictions may represent plausible scenarios since the

Nomenclature

AEP	Annual Energy Production
CAPEX	Capital Expenditure
CFD	Computational Fluid Dynamics
c-NSGA	clustering NSGA
D	turbine rotor Diameter
DCO	Development Consent Order
EA	Evolutionary Algorithm
EU	European Union
EWEA	European Wind Energy Association
FEED	Front-End Engineering and Design
FI	Forced Improvement
GA	Genetic Algorithm
GOMEA	Gene-pool Optimal Mixing
IBEA	Indicator Based EA
LES	Large Eddy Simulations
LT	Linkage Tree
MR	Multi-Resolution
MI	Mutual Information
MO	Multi Objective
NIS	No Improvement Stretch
NSGA	Nondominated Sorting Genetic Algorithm
OM	Optimal Mixing
o-MOGOMEA	offline MOGOMEA
OPEX	Operational Expenditure
OPF	Optimal Pareto Front
OWF	Offshore Wind Farm
SPEA	Strength Pareto Evolutionary Algorithm
PSO	Particle Swarm Optimization
UPGMA	Unweighted Pair Grouping Method Arithmetic
WFLOP	Wind Farm Layout Optimization Problem

required growths are below the average European industry growth (36.1%) since 2002, when the first large-scale OWF was built [7,8].

The European OWFs have become larger throughout the years, with the average area reaching almost 60 km^2 in 2012 (see Fig. 2a). Similarly, also the number of turbines per OWF has been increasing, as can be seen in Fig. 2b. Projects commissioned from 2002 until 2011 had on average 39 turbines, whereas between 2012 and 2015, that value increased to approximately 72 turbines.

Despite the important lessons learned by developers and technological advances achieved in recent years, the cost of energy generated offshore is not yet competitive. In fact, without encouragements and incentives from governments, the industry would probably not consider offshore wind. Since 2012, OWFs have been very capital intensive, costing on average EUR 1 billion. The Gwynt y Môr project, with an installed capacity of 576 MW and commissioned in 2015, is the second-largest OWF, costing EUR 2 billion (its layout is shown in Fig. 3). It is a complex project due to the challenging seabed conditions and human-made constraints (a pre-existing pipeline crosses the project area, separating it into two zones). These high costs are mainly due to the larger seabed areas, large distance to shore, large water depth and large number of turbines [8,7,9].

One possible strategy to decrease the cost of energy of an OWF, is to increase its energy yield [13]. OWFs are usually designed with a high turbine density due to limitations on space and to reduce Capital Expenditure (CAPEX) on, for example, cables to interconnect the turbines [14]. However, turbines induce wake interferences on other turbines. For example, the energy produced at the Danish OWF, Horns Rev 1, is 89% of what the same turbines together would produce if installed solitarily [15]. Therefore, wake effects are considered to be the strongest economical driver and designers create layouts that maximize the turbines' exposure to the prevailing wind direction to increase energy production [16,17].

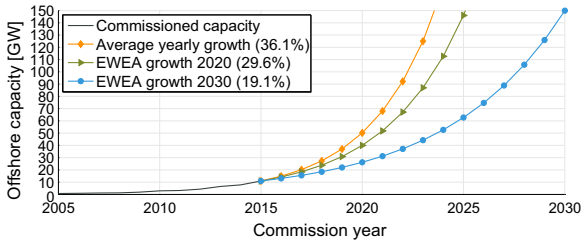

Fig. 1. Joint installed capacity of offshore wind farms at different growth rates.

Designing the initial OWFs was simple because of the low number of turbines and homogeneity of depth and seabed soil properties. The layout was primarily defined by placing the turbines in regular structures with greater distances in the prevailing wind direction [18]. Optimizing the design of recent OWFs is a much more challenging task because it requires the analysis of sophisticated trade-offs between conflicting goals, most notably Annual Energy Production (AEP), CAPEX and Operational Expenses (OPEX). As previously noted, state-of-the-art OWFs are composed of a larger number of turbines and have more challenging seabed areas with water-depth and soil conditions that vary across the

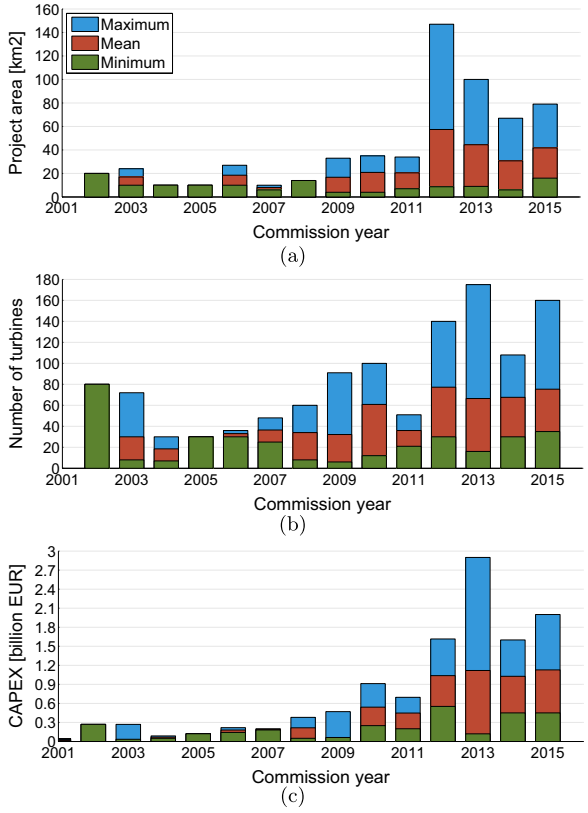

Fig. 2. Yearly statistics of area, number of turbines and CAPEX of European OWFs with five or more turbines commissioned and under construction since 2002 [8,7,9–11].

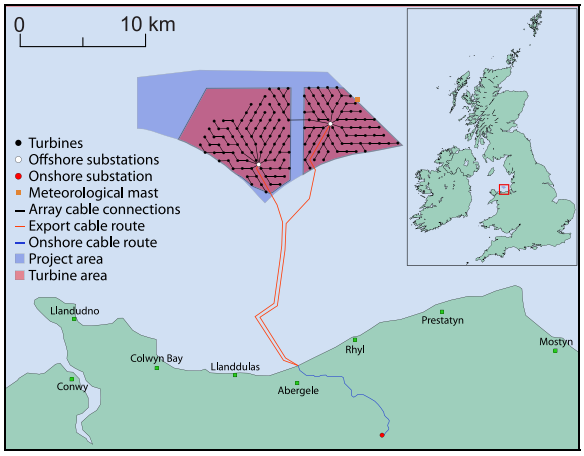

Fig. 3. Layout of the British Gwynt y Môr OWF. Reproduced from [12].

site [16]. Approximately 4% of the CAPEX of an OWF is allocated to the development phase. In this phase, in addition to the layout, all the components and technologies to be used, have to be decided [19]. For such vast projects, automated optimization tools are crucial to ensure that optimized, i.e. efficient, layouts are designed in this phase [20].

The Wind Farm Layout Optimization Problem (WFLOP) is a very difficult problem to solve [21]. It is non-linear, multi-modal, non-differentiable, non-convex, discontinuous and cannot be described analytically (without making vast simplifying assumptions) [18]. Moreover, due to various characteristics of the WFLOP, calculus-based approaches such as linear programming and gradient methods are not suitable to solve the problem [18,22,23]. The problem becomes even more complex if partial turbine wake shadowing is considered [21]. Although different modeling techniques have been introduced in the literature to reduce the computational burden of the WFLOP [24–27], it remains a very computationally demanding problem. Optimal solutions to the WFLOP can be confirmed only for small instances [21].

A solution that is widely used in academia, is to the solve the WFLOP with Evolutionary Algorithms (EAs), a type of optimization algorithm that belongs to the class of metaheuristics [21]. EAs are moreover among the most suitable methods to solve Multi-Objective (MO) problems [28]. Contrary to most (heuristic) optimization algorithms, these methods use a set of multiple solutions (also called a population) during optimization. EAs are moreover relatively simple to apply to solve optimization problems because only require a way to evaluate the objective functions of interest for them to work [28]. Several reviews of the different approaches for solving the WFLOP have appeared in literature [21,29,18,30–32]. The first work that used a metaheuristic, namely the classic Genetic Algorithm (GA), to solve the WFLOP was carried out in 1994 [33]. Several metaheuristics have been applied to the WFLOP and although EAs, like virtually all metaheuristics, do not guarantee to find the global optimum given a certain computational budget, EAs remain the most used type of optimization algorithm to solve the WFLOP [21].

The objective of this article is to address the following open research questions:

1.1. What characteristics should an optimization algorithm have to present optimized layouts?

Despite the fact that multiple studies on the (MO)WFLOP have appeared in literature, no fair comparison of different algorithms has been performed, i.e. employing the same settings for computational budget and using the same problem formulations and problem instances [21]. Rather, the different algorithms have been used in different scenarios, using different constraints and objectives. This limits the conclusions about the performance of algorithms that can be drawn from previously published work [21,31]. Furthermore, no analysis has been carried out to understand what characteristics are required of an optimization algorithm to efficiently solve the MOWFLOP. Lastly, to correctly assess the optimization performance of EAs, multiple runs have to be done as these algorithms are stochastic in their operation. Nonetheless, statistics such as averages and variances over multiple runs have not presented in previous literature [21].

1.2. What is the best constraint-handling technique to ensure feasibility of the OWF layouts?

In WFLOP, if the formulation of the optimization problems allows placement of turbines closer to each other than the minimum distance required between turbines, the final design layouts may be infeasible [34]. For real-world problems, constraints that identify when a solution is feasible often have a key contribution to a problem's difficulty. Being able to handle these constraints efficiently and effectively to ensure that the final outcome of optimization is indeed a feasible solution, is therefore important. In metaheuristic optimization, such techniques are called Constraint-Handling Techniques (CHTs). It is known that different CHTs affect the performance of a MOEA differently (see, e.g. [35]). Hence, studying the performance under different CHTs is important and should be considered as an intrinsic part of the EA. Although several CHTs have been employed in the literature for solving the (MO)WFLOP, their impact on the performance of (MO)EAs has not yet been investigated [36].

1.3. How does the problem complexity scale with the number of design variables?

The performance of metaheuristics, typically measured in the time required to reach an optimal solution or a solution of a certain quality level relative to the optimum (e.g. 95% of the optimum), is highly correlated with a problem's dimensionality (that is, assuming the problem itself can be scaled to larger dimensionalities) [28,37]. Previous publications on (MO)WFLOP have not discussed the impact of increasing the number of design variables (which depends on the number of turbines, the number of grid positions, or both) on the performance of the EAs used [18].

1.4. What is the relation between problem dimensionality/complexity and the degrees of freedom offered by different turbine-placement grid resolutions?

In general, two approaches can be identified to model the positioning of turbines. Either the coordinates of a fixed number of turbines is given, or a grid is defined over the available area and turbines may be placed at grid points. One advantage of the latter approach is that it does not necessarily require specifying a prior the number of turbines to be placed. Instead, this number can be optimized together with the types of turbines to be placed and where to place them on the grid by making each grid point a decision variable. Because this flexibility on the number of turbines is a desirable design property, we adhere to the latter choice in this article. No work has however previously been published that studies the trade-off between problem dimensionality and the degrees of freedom offered by different grid resolutions according to which turbines can be placed. The usual strategy is to keep the number of optimization variables as low as possible in order to

require less evaluations of the objective functions [38]. This would imply using a coarse grid. On the other hand, using a fine grid for the OWF area increases the design freedom since it is possible to obtain a larger number of different OWF layouts [34]. Using a finer grid however increases the complexity of the problem, even though it is still covering the same area, because the resolution of the grid is directly linked to the number of optimization variables. This in turn affects the rate of convergence and optimization performance of metaheuristic algorithms, especially when considering a fixed budget of computation time [18,26].

The remainder of this article is organized as follows: Section 2 introduces the MOWFLOP in more detail. Section 3 describes the algorithms selected in this work as well as their main characteristics. The performance of these algorithms combined with different CHTs is tested on OWFs with different dimensions and grid-step sizes in Section 4. A discussion of the results is given in Section 5, while final conclusions, recommendations and future work are presented in Section 6.

2. Multi-objective wind farm layout optimization problem

The layout of an OWF is designed and optimized during the Front End Engineering Design (FEED) phase of the project. The FEED phase is performed after initial feasibility studies and before investment decisions. The design options remain relatively flexible during the FEED phase. For example the number, model and location of the turbines is still not fixed [39,40]. Furthermore, wind farm developers have to make a pre-selection of economically viable design concepts and associated key components during the FEED phase [13,19]. The number of turbines and their locations have a strong impact on the overall efficiency of the project and hence, they may be considered to be one of the most important optimization variables in the WFLOP.

One of the most used objective functions used to formulate the WFLOP is the Net Present Value (NPV) [21], which may be calculated as:

$$NPV = (AED \cdot p_{kWh} - OPEX)a - CAPEX \quad (1)$$

where a is the annuity factor $\left(a = \left(1 - (1+r)^{-n}\right)/r\right)$, r is the interest rate, n is the project lifetime and p_{kWh} is market energy price.

The NPV requires *a priori* economic values for the interest rate, wind farm lifetime and market energy price. If these values are changed, it is not guaranteed that the OWF layout that leads to the minimum NPV remains the same. Thus, if the designers wish to obtain a new layout for different economic parameters, another optimization run has to be performed, which, depending on the complexity of the models and the computational power available, may require a considerable amount of time. Using functions that depend on a priori determined economic values can be seen as converting a problem that is actually inherently multi-objective to a single-objective (SO) problem by the use of weighting factors. In doing so, developers will gain only limited insight into the problem and options for designing layouts because they are only given single layouts each time (one for every combination of economic values) instead of immediately being informed of all solutions that correspond to efficient trade-offs between the key aspects that are of importance, such as AEP, CAPEX and OPEX.

Although more than 150 publications may be found in literature that have dealt with the WFLOP [21], only a few studies have investigated the trade-offs that emerge while designing a wind farm using a multi-objective formulation of the problem [30]. Table 1 presents the characteristics of the MOWFLOP as considered by relevant studies. The study carried out in [41] optimized the AEP with the problem constraints being considered as a second objective function. The AEP and the noise generated by the turbines were optimized in [43,44]. Similarly, [46] optimized the AEP as a first objective and the sum of the wind farm area and the number of turbines as second objective. Three simultaneous optimization goals were used in [48]: AEP, area used and collection system length.

The great majority of existing approaches, for both the WFLOP and its MO variant, have assumed either a fixed or a maximum number of turbines, despite the fact that the number of turbines is an important real-world optimization objective during the FEED phase [21]. An example of an exception to this is the work presented in [51], in which both the locations and number of turbines were used as design parameters to optimize the AEP and the sum of CAPEX and OPEX.

The energy production was used as an objective in all approaches that studied the MOWFLOP (see Table 1). In fact, the energy production is the most common objective function used both in academia and in commercial software [21]. In order to realistically approximate the amount of energy produced by an OWF it is necessary to consider wake effects due to the close proximity of turbines. Next, an overview of wake losses modeling is given and the model that we chose to employ, is described.

2.1. Wake losses

Currently, there is a wide variety of models to calculate, with different accuracy levels, the wind deficits due to wake losses in wind farms [52–56]. Examples of low-fidelity engineering models to describe wake losses include the Katic-Jensen model [57,58], the Eddy viscosity model [59], the Frandsen et al. model [60], the deep-array wake model [61] and the Larsen model [62]. These models, due to their simplified wake-speed deficit approach, can be evaluated in only little computation time and can provide a preliminary description of the far wake regime [63].

Other models were built to provide medium-fidelity results,

Table 1
Existing approaches for the MOWFLOP.

References	Optimization variables	Design objectives	Wake model	Constraint handling	Variables domain	MOEA
Kusiak et al. [41]	Turbine locations	Energy generation, Problem constraints	Katic/Jensen	Extra Objective	Continuous	SPEA [42]
Zhang et al. [43,44]	Turbine locations	Energy generation, Noise level	Jensen	–	Continuous	NSGA-II [45]
Veeramachaneni et al. [46]	Turbine locations	Energy generation, Cost	Katic/Jensen	Repair	Continuous	MO-PSO [47]
Tran et al. [48]	Turbine locations	Energy generation, Collection system length, Wind farm area	Katic/Jensen	–	Continuous	NSGA-II [45], SPEA2 [49], IBEA [50]
Sisbot et al. [51]	Turbine locations and quantity	Energy generation, Cost	Katic/Jensen	Repair/Penalty	Discrete	MOGA [28]

such as the Dynamic Wake Meandering model [64] and several other approaches based on the actuator disk model [65,66]. At the high-fidelity end, Computational Fluid Dynamics (CFD) models are found [54]. The highest of fidelity is obtained using models based on Large Eddy Simulations (LES). Although of value in their own right, one evaluation of such a model may take several weeks to complete [67].

CFD and LES models may be used for detailed studies such as: interactions between a turbulent flow and a rotor blade; the interaction between multiple wakes; or validation and calibration of simpler models [15,67]. However, for wind farm layout optimization, the huge computational requirements of these models make them prohibitive to use because during optimization many evaluations of these models are typically required, especially in case of large OWFs [68].

For this reason, a commonly adopted design methodology (see, e.g. [69,70,38]) is to tackle the WFLOP with simplified and computationally light models. The solutions obtained by doing so provide a first, and often already quite insightful, assessment of the potential solutions to the problem and their quality because although the simpler models may not be as accurate as the ones with the highest fidelity, they do differentiate the performance of one wind farm layout versus another. A designer may then choose a few solutions from the optimized set and further evaluate them with more detailed models to get more accurate figures for the expected performance of these solutions. In this work, the Katic-Jensen model was employed during the optimization experiments. A description of the model is given next.

2.1.1. Katic-Jensen wake model

The Jensen model, originally proposed in 1983, is a simplified and fast manner of calculating the wind speed inside the wake of a turbine [57]. The model, further developed by Katic et al. [58], has been widely adopted in wind farm modeling due to its ease of implementation and low computational requirements [29,21,18,71–73]. All the MO approaches presented in Table 1 used the Katic-Jensen model (with the exception of [43,44] which used the original Jensen model). According to the Katic-Jensen model, the wind speed seen by the *j-th* turbine positioned in the wake of one or more turbines, is given by:

$$U_j = U_0\left(1 - deficit\right) \tag{2}$$

where U_0 is the ambient wind speed and *deficit* is the velocity decrease caused by shadowing effects.

The wake expansion is considered to be linear [57,58]:

$$R_{kw} = R_k + \alpha d_{kj} \tag{3}$$

where R_{kw} is the wake front radius, R_k is the turbine rotor radius, α is the momentum entrainment or wake decay coefficient and d_{kj} is the distance between the turbines (displayed in Fig. 4).

The value of α may be calculated according to [74]:

$$\alpha = \frac{A}{\log\left(\frac{h_{hub}}{z_0}\right)} \tag{4}$$

where A is a constant (0.5), h_{hub} is the turbine hub height and z_0 is the surface roughness height, which, for offshore environments, is usually considered to be 0.0005 [75].

The interference caused by an upstream *k*-th turbine to the *j*-th turbine may be calculated as [58]:

$$U_{kj} = \frac{1 - \sqrt{1 - C_{T_k}}\,\frac{A_{kj}}{A_j}}{\left(1 + \frac{\alpha d_{kj}}{R_j}\right)^2} \tag{5}$$

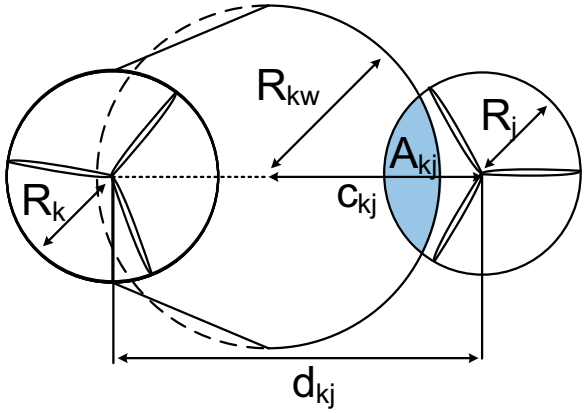

Fig. 4. Turbine partially affected by the wake of an upstream turbine.

where C_{T_k} is the *k*-th turbine thrust coefficient at a given wind speed, A_j is the *j-th* turbine rotor area and A_{kj} is the *j-th* turbine rotor area influenced by the upstream turbine *k*.

If the wake front affects entirely the *j-th* turbine, $A_{kj} = \pi R_j^2$ whereas if the wake front does not impact the *j-th* turbine, $A_{kj} = 0$. If the wake wave affects partially the turbine rotor sweep area (Fig. 4), A_{kj}, is given by [24,25,76]:

$$A_{kj} = \frac{1}{2}\left(R_{kw}^2\left(2\arccos\left(\frac{R_{kw}^2 + c_{kj}^2 - R_j^2}{2R_{kw}c_{kj}}\right) - \sin\left(2\arccos\left(\frac{R_{kw}^2 + c_{kj}^2 - R_j^2}{2R_{kw}c_{kj}}\right)\right)\right)\right) + \frac{1}{2}\left(R_j^2\left(2\arccos\left(\frac{R_j^2 + c_{kj}^2 - R_{kw}^2}{2R_jc_{kj}}\right) - \sin\left(2\arccos\left(\frac{R_j^2 + c_{kj}^2 - R_{kw}^2}{2R_jc_{kj}}\right)\right)\right)\right) \tag{6}$$

To account for multiple interferences from upstream turbines (see Fig. 5), the deficit term is calculated as [58]:

$$deficit = \sqrt{\sum_{k=1}^{n} U_{kj}^2} \tag{7}$$

2.1.2. Assumptions

The Katic-Jensen wake model assumes, among other things [21], that the wind speed inside the wake front is axisymmetric, decreases linearly, and that the wake front starts to expand after the turbine rotor.

The fact that the wake front is assumed to be axisymmetric and that the wind speed reduction is assumed to be linear means that different layouts will have similar performance according to the model. For example, the turbines inside the wake front of Fig. 5 have the same energy production according to the Katic-Jensen model (the ones at the same distance from the first turbine).

2.2. Constraint-handling

Similar to most real-world problems, the WFLOP has constraints. To obtain feasible wind farm layouts different constraints have to be respected [18]:

- *wind farm boundaries*: the turbines have to be placed inside the wind farm area [77];
- *infeasible areas*: there may be some areas that are not available to install turbines due to human-imposed constraints (e.g. unexploded ordnances, pre-existing cables, shipwrecks) or nature-imposed constraints (e.g., inappropriate type of soil material, cliff areas, soil areas with insufficient bearing capacity, large seabed gradients) [17];

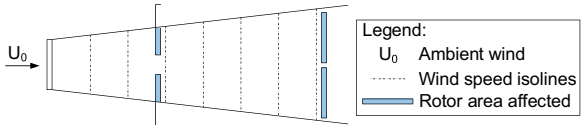

Fig. 5. Wake growth and wind speed decrease according to the Katic-Jensen model.

- *minimum clearing distance*: a minimum distance between neighboring turbines is required to guarantee proper functioning and to safeguard their structural integrity [73];
- *number of turbines*: the number of turbines that may be installed has to be within the range specified in the Development Consent Order (DCO) [39].

In the following, several CHTs, that have previously been used in literature for the WFLOP, are described.

2.2.1. No constraints

Various approaches avoid the use of CHTs:

- In the first work that used an EA for the wind farm layout design problem, the wind farm area was divided using a grid with a step size large enough to allow turbines to be placed on grid points without ever being able to violate proximity constraints [33]. Similar approaches have been used in recent literature [78,38,79,80]. This approach has a further benefit that the search space of the problem is relatively small because of the relatively large grid spacing, which may lead to faster convergence speeds of optimization algorithms. A big drawback of the approach however is that it oversimplifies the problem, prohibiting finding of better solutions that require more intricate layouts;
- In [48,81] new variation operators were developed that guarantee that the new layouts remain feasible;
- The proximity constraints were not explicitly used in [43,44]. Instead, the authors expected the algorithm to converge to the feasible space because of inherent characteristics of the problem, e.g. two turbines placed in close vicinity will generate less energy. Although generally true, this does not guarantee that the required minimum distance, which may differ from the minimum efficient distance in terms of energy production, will be respected.

2.2.2. Resample

In [68] the wind farm layout was replaced by an entirely new feasible, but randomly generated, layout whenever a turbine violated the proximity constraint. This strategy has the drawback that it disrupts the progressive evolution of knowledge acquired by the optimization algorithm. Consequently, this may mislead algorithms that employ variation operators that span multiple generations to learn about the problem structure and exploit this information [35,82]. Furthermore, this approach fails, or at least takes a long time, when it is hard to obtain a feasible solution through a random process [83].

2.2.3. Penalty term

The WFLOP has also been transformed into an unconstrained problem by computing a measure of how badly constraints have been violated, and by adding this measure to the objective function [51,84]. This method, known as the penalty-function method, transforms the search space of the problem, which may increase the roughness of the search space. This may lead the optimization algorithm to local minima in the new search space [85], which may still correspond to infeasible solutions. Furthermore, it is difficult to create a generic penalty function that is optimal for all problems and that does not over- nor under-penalize the infeasible solutions [85,83,86]. On the other hand, if the penalty term is properly designed, the penalty-function method may work quite well, allowing the optimization algorithm to also explore the infeasible search space, which may act as a bridge between different feasible areas of the search space [85].

2.2.4. Constraint domination

In several publications, problem constraints have been tackled by the use of the so-called constraint-domination technique [24,25,87–89,38]. Similar to the penalty-function method, a measure of how badly constraints are violated, is computed. When comparing wind farm layouts, the one with the lowest measure of constraint violation is then always preferred. The main drawback of this approach is that it may be difficult for an algorithm to achieve layouts that are located close to infeasible areas of the search space [83]. Furthermore the method negatively effects diversity in the population, which is important in EAs [86].

2.2.5. Repair mechanism

Repair mechanisms ensure that solutions are always in the feasible space by repairing any solution that has been generated and is infeasible [83]. In several studies of the WFLOP, turbine removal has been used [51,90,14,91,26]. Two different approaches were introduced in [46]: removal of the first turbine that is too close to another turbine and removal of the turbine that has the most conflicts with its neighbors. Although the second strategy presented slightly better results, it did not outperform by far the former approach [46].

2.2.6. Extra optimization goal

The sum over all constraints of the measure of constraint violation was considered as a separate minimization goal in [41]. Although effective, this approach increases the complexity of the problem by adding another objective goal. Moreover, it does not guarantee that the algorithm will return any feasible solution [83].

2.3. Domain of optimization variables

Different representations of solutions to the WFLOP problem have been considered, including real-valued variables, discrete variables and combinations of both. Next, each of these representations is described.

2.3.1. Real-coded

A real-valued, also referred to as continuous, representation of wind farm layouts, encoding the coordinates of turbines, was considered for the first time in [84]. Thereafter, many recent works have considered a real-valued representation (see Table 1 for MO examples). However, the wake models used actually do not offer sufficient resolution for the high precision of a real-valued representation to be of added value (see Section 2.1). Moreover, as stated previously, the number of turbines is an important optimization parameter. However, it is difficult to simultaneously optimize the number of turbines with a real-coded optimization algorithm since the turbine number is discrete. Furthermore, given the same number of problem variables and the same inherent underlying problem complexity, continuous optimization problems typically take longer to solve than discrete problems because of the larger variable-domain size of continuous variables.

2.3.2. Mixed-integer

A possible solution is to tackle the WFLOP as a mixed-integer optimization problem, in which both discrete variables, e.g. the number of turbines, and continuous variables, e.g. turbine positions, may be employed. However, tackling a mixed-integer

problem is very hard, presenting many optimization challenges which do not arise in purely real or discrete optimization problems [92]. Moreover, straightforward encodings with many turbine locations and integer variables describing how many and/or which of these turbines to actually use, are highly redundant, which have a negative effect on optimization performance. Furthermore, simple wake models would still be required for computational efficiency reasons and hence the insufficient fidelity problem of the wake model also arises with this approach. The WFLOP has also previously been transformed into a mixed-integer *linear* optimization problem [93,94]. Differently from heuristic methods, algorithms that solve optimization problems formulated as so-called linear programs guarantee that the optimal solution will be found [94]. However, the WFLOP needs to be linearized to this end, which strongly diminishes the extent to which the problem, and consequently its solutions, relate to the real-world scenario.

2.3.3. Discrete

Representations using discrete variables have also been considered. Typically, a grid is placed over the designated area and grid points correspond to (variables encoding) potential locations for turbines. This makes it straightforward to optimize both the number and locations of turbines. All the constraints presented in Section 2.2 may be automatically respected if the positions are encoded as discrete variables [38]. Furthermore, the location of a turbine may be described by a single parameter in a discrete approach, whereas two coordinates have to be used in a continuous domain. It is furthermore known that if a hexagonal or a regularly spaced packing is optimal, optimization based on a discrete grid may yield a solution that is close to the continuous optimum for a bounded area [95]. Finally, the general trend of the offshore wind industry has been to place the turbines in regular structures, with greater distance turbine separation in the prevailing wind direction. Even in current state-of-the-art OWPs, turbines were placed in a grid-based layout (see Fig. 3). For this reason, it is likely that the use of discrete turbine locations leads to layouts that may be more easily accepted by the offshore wind industry. A potential drawback is that a grid has to be defined a priori [21,96].

3. Optimization algorithms for the multi-objective wind farm layout optimization problem

Although only few published works have dealt with the MOWFLOP, different MOEAs have already been used in these works (see Table 1). The Strength Pareto Evolutionary Algorithm (SPEA) [42] was used in [41], whereas in [46] the Multi-Objective version of the Particle Swarm Optimization (MO-PSO) algorithm was employed [47]. The well-known Nondominated Sorting Genetic Algorithm II (NSGA-II) algorithm has been used in several of the existing works [43,44,48]. Finally, only [48] has performed a basic comparison between three algorithms: NSGA-II, SPEA2 and Indicator Based EA (IBEA). It was concluded that IBEA was the most adequate algorithm for the MOWFLOP even though the NSGA-II and SPEA2 found Pareto fronts with a greater spread [48].

3.1. Definitions for MO optimization

We assume to have m objective functions $f_i(\boldsymbol{x})$, $i \in \{1, 2, \ldots, m\}$ that, without loss of generality, all need to be *minimized*. A solution $\boldsymbol{x}^1$ is said to (Pareto) *dominate* a solution $\boldsymbol{x}^2$ (denoted $\boldsymbol{x}^1 \succ \boldsymbol{x}^2$) if and only if $f_i(\boldsymbol{x}^1) \leq f_i(\boldsymbol{x}^2)$ holds for all $i \in \{1, 2, \ldots, m\}$ and $f_i(\boldsymbol{x}^1) < f_i(\boldsymbol{x}^2)$ holds for at least one $i \in \{1, 2, \ldots, m\}$. A *Pareto set* of size n is a set of solutions $\boldsymbol{x}^j$, $j \in \{1, 2, \ldots, n\}$ for which no solution dominates any other solution, i.e. there are no $j, k \in \{1, 2, \ldots, n\}$ such that $\boldsymbol{x}^j \succ \boldsymbol{x}^k$ holds. A *Pareto front* corresponding to a Pareto set is the set of all m-dimensional objective function values corresponding to the solutions, i.e. the set of all $\boldsymbol{f}(\boldsymbol{x}^j)$, $j \in \{1, 2, \ldots, n\}$. A solution $\boldsymbol{x}^1$ is *Pareto optimal* if and only if there exists no other $\boldsymbol{x}^2$ such that $\boldsymbol{x}^2 \succ \boldsymbol{x}^1$ holds. Further, the *optimal Pareto set* is the set of all optimal Pareto solutions and the *Optimal Pareto Front* (OPF) is the Pareto front that corresponds to the optimal Pareto set.

3.2. Characteristics

In the following we will describe key features that a MOEA should have to adequately solve the MOWFLOP.

3.2.1. Clustering

Standard MOEAs steer the population towards the OPF while trying to preserve diversity in the population through different mechanisms, e.g. use of the crowding distance in NSGA-II [45], use of the environmental selection in SPEA2 [49] or use of the hypervolume in MO-CMA [82]. However, it has been shown that these mechanisms are insufficient to achieve good scalability [97]. Furthermore, selection based on the domination criterion tries to exploit all objectives simultaneously, thus reducing the pressure in the direction of the OPF [97]. For this reason, handling different parts of the objective space differently is of major importance. Furthermore solutions along the OPF are typically very different, especially for the extreme regions of the front. The use of clusters to divide the objective space into smaller areas allows the MOEA to specialize variation to meet the specific requirements to find improvements in narrower areas of the search space, leading to better results, for both continuous and discrete problems [97,37,35].

3.2.2. Single-objective optimization

Having a mechanism that puts extra pressure on exploiting individual objectives can be highly beneficial because nondominated selection may not provide enough pressure to find the extreme solutions (i.e. solutions that optimize a single objective) [97]. This is of special importance if one of the optimization goals is much harder to solve than the others since the algorithm may converge prematurely and discover only a small subset of the OPF [35]. Moreover, in some problems the number of available solutions in the extreme regions can be much smaller than in the middle regions of the front [98]. The benefit of adding SO optimizers to specifically obtain solutions in extreme regions of the front has been shown to improve the performance of MOEAs [99,35,37].

3.2.3. Problem structure exploitation

A key property of EAs is their ability to juxtapose partial solutions or substructures from different solutions to create improved solutions. This mixing is only efficient when key substructures are not disrupted too often by the variation operators [100]. In GAs, subsets of variables of two solutions are exchanged. However, it has been shown that without detecting and exploiting the dependencies between problem variables, EAs cannot solve some decomposable problems efficiently [37]. When enough knowledge of the problem is available, mixing can be made efficient by designing the variation operation in an appropriate way. However, for black-box optimization problems nothing is known of the problem structure a prior. For such problems, this knowledge has to be inferred from the population of solutions by identifying groups of variables that together make an important contribution to the quality of a solution [100]. Such information is commonly referred to as *linkage information*.

A general linkage model that can be used to capture the interactions between the l optimization variables is, called the Family Of Subset (FOS) [100]. A FOS $\mathcal{F}$ consists of subsets that

contain identifiers of decision variables, i.e. $\mathcal{F} = \{\boldsymbol{F}^0, \boldsymbol{F}^1, \ldots, \boldsymbol{F}^{|\mathcal{F}|-1}\}$ where $\boldsymbol{F}^i \subseteq \{0, 1, \ldots, l-1\}, i \in \{0, 1, \ldots, |\mathcal{F}|-1\}$. Every subset $\boldsymbol{F}^i$ is called a linkage subset and represents a group of decision variables which exhibit some degree of joint dependency and hence should be copied together during variation.

In this article, two algorithms are used: the well-known NSGA-II [45] and a variant of the recently introduced Multi-Objective Gene-pool Optimal Mixing Evolutionary Algorithm (MOGOMEA) [37], which was designed by combining the Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA) [100] with a MO framework [99]. Both algorithms as well as two slightly modified variants are presented in the following subsections.

3.3. MOGOMEA

A flowchart of our variant of the MOGOMEA is shown in Fig. 6 and a description of each step of the algorithm is given next.

3.3.1. Population initialization

The n_i initial wind farm layouts are created with a nearest neighbor heuristic to selects turbine locations from the available locations that are spread as well as possible. Specifically, first, the initial number of turbines, m, is randomly generated between one and the maximum number of turbines that may be installed for the given area. The location of the initial turbine is randomly chosen from all the possible locations. The distance of the remaining locations is computed to the first turbine and the most distant one is chosen for the second turbine. The distances for the remaining locations are updated by checking whether the distance to the new turbine is smaller than the currently stored distance (the shorter distance is kept). The procedure is repeated m times or until a turbine violates the proximity constraint. In this way, it is guaranteed that feasible wind farm layouts are generated, similarly to several previous works [68,81,90].

3.3.2. k-leaders

The same nearest-neighbor heuristic as in the previous step is used to select k solutions, also called cluster leaders, from the nondominated solutions of the population that are spread as well as possible. This is done to bias the leaders towards the best solutions of the population. Differently from the previous step, a solution with a minimum value for a randomly chosen objective is chosen to be the first leader to increase the probability of having leaders at the extremes of the Pareto front. The distances are measured in the objective space. The heuristic is then repeated until all the necessary cluster leaders are selected (see Fig. 7).

3.3.3. Clustering

Next, the c closest solutions (including the leader solutions themselves) to each leader are clustered together. Because the assignment is done independently for each cluster, some solutions may be assigned to multiple clusters while other solutions may not be clustered. To reduce the probability of this happening we increased the probability of clusters overlap by setting $c = 2\lfloor \tau n \rfloor$ as proposed in [101]. This increases the probability of finding a good, uniform spread of solutions with a multi-objective EA faster [101]. The clustering is performed on *normalized* objective values to remove the influence of differently scaled objectives.

3.3.4. Linkage learning

A linkage model $\mathcal{F}_j$ is learned for each cluster C_j to distinguish different regions along the Pareto Front and allow a different, objective-space region-specific, exploitation bias to be formed for each of them. Although different FOS structures may be used [102,103], the Linkage Tree (LT) structure is used here since it has been demonstrated to result in the best and most reliable

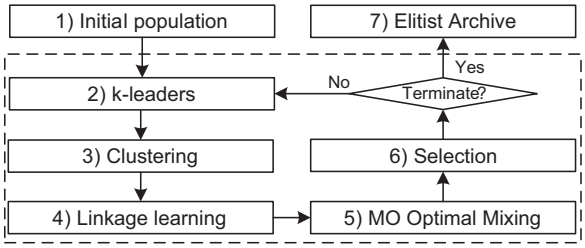

Fig. 6. Flowchart of the MOGOMEA variant used in this article.

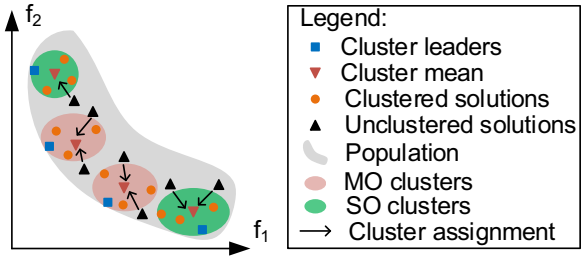

Fig. 7. Selection of k leaders and cluster assignment.

performance of the GOMEA framework on several benchmark functions [100,102,104].

An LT captures all decision variables as being fully independent in singleton (leaf nodes) subsets ($\boldsymbol{F}^i = \{i\}, i \in \{0, 1, \ldots, l-1\}$). Furthermore, it organizes combinations of variables in a tree-like fashion. A branch node of the LT is a multivariate subset $\boldsymbol{F}^i$, which is the combination of two subsets $\boldsymbol{F}^j$ and $\boldsymbol{F}^k$ such that $\boldsymbol{F}^j \cap \boldsymbol{F}^k = \emptyset$, $|\boldsymbol{F}^j| < |\boldsymbol{F}^i|$, $|\boldsymbol{F}^k| < |\boldsymbol{F}^i|$ and $\boldsymbol{F}^j \cup \boldsymbol{F}^k = \boldsymbol{F}^i$. The LT FOS has $2l-2$ linkage subsets because the root node is discarded since it contains all the variable indices and hence it does not generate a different offspring solution. Note that a variable can be part of multiple subsets of the LT. Therefore, any two variables may be dependent according to some subsets, but independent according to others [105].

The LT is constructed using a pairwise measure of distance between sets of variables known as the Unweighted Pair Grouping Method with Arithmetic-mean (UPGMA [106]). Constructing an LT can be conceptually considered to start from the leaf nodes and creating branch nodes by consecutively combining two closest groups until the root node is obtained. An efficient implementation that takes a slightly different approach, but results in the same LT, has a computational complexity of only $O(cl^2)$ [106] where c is the cluster size. To compute similarity between two variables as a foundation for the UPGMA method, various measures may be used. In this article, we use Mutual Information (MI), since it has demonstrated to lead to the most efficient performance on several benchmark problems [107]. MI is a dimensionless quantity and can be thought of as the reduction in uncertainty about one random variable given knowledge of another. A high MI value represents a large reduction in uncertainty, a lower value constitutes a low reduction of uncertainty and a null MI value means that the two variables are independent.

To illustrate the notion of a linkage tree, Fig. 8a shows the LT that resulted from clustering the turbine positions of the wind farm area shown in Fig. 12b. The clusters of turbine positions, which show that neighboring positions are clustered together first, are also indicated in Fig. 8b.

3.3.5. MO Gene-pool Optimal Mixing

The main operator of variation in MOGOMEA is called Gene-pool Optimal Mixing (GOM). GOM is applied to every solution in the population. For this reason, it is necessary to determine which

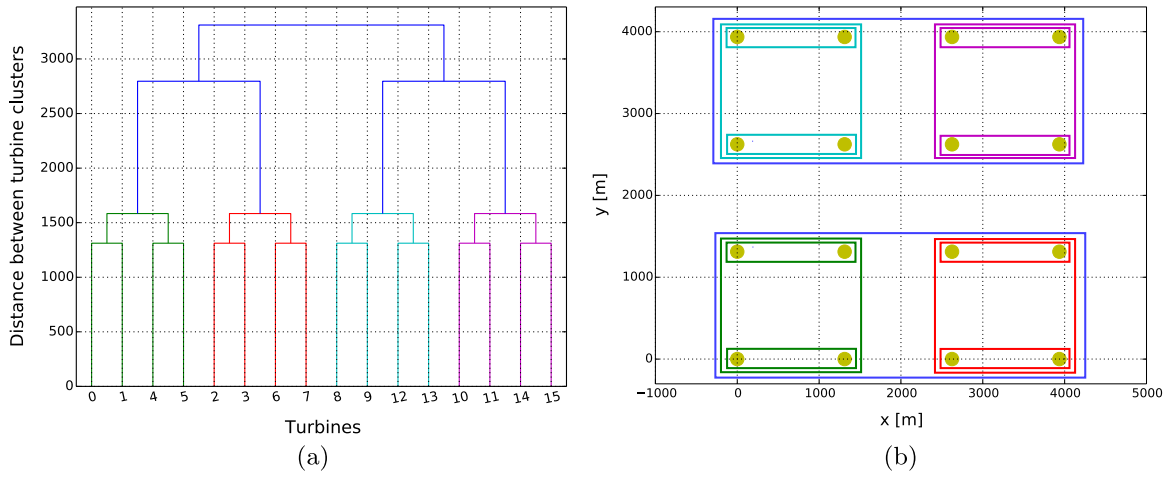

Fig. 8. Figure (a) shows the LT learned offline based on the distances between the possible locations for a 8D grid step size. The turbines are numbered from left to right and from bottom to top (see Fig. 12b). Figure (b) shows the respective clustering of the positions in the wind farm.

cluster a solution belongs to since a separate LT is learned for each cluster. Solutions that have not been assigned to a cluster by the clustering algorithm are now assigned to the cluster with the nearest mean value (see Fig. 7). In case of multiple cluster assignments, ties are broken randomly. Thereafter, every solution $\boldsymbol{x}$, also called the parent solutions, is incrementally changed into an offspring solution. Firstly, the offspring solution $\boldsymbol{o}$ and a backup $\boldsymbol{b}$ are created by cloning $\boldsymbol{x}$. The linkage groups in FOS $\mathcal{F}_j$ are then enumerated in a random order. For every $\boldsymbol{F}^i \in \mathcal{F}_j$, a donor solution $\boldsymbol{d}$ is randomly chosen from the same cluster C_j. The optimization variables whose indices are indicated by the linkage group $\mathcal{F}_i$ are copied from the donor $\boldsymbol{d}$ to $\boldsymbol{o}$. Recent literature has demonstrated the need for mutation to reliably solve certain types of problems [37]. Therefore, in our version of MOGOMEA we use a simple bit-flipping mutation with probability $1/l$. Hence, when the copy occurs, the variables may be altered via mutation. If the copy generated a new offspring solution, the objective values of $\boldsymbol{o}$ are evaluated and compared with the backed-up solution $\boldsymbol{b}$. The changes are accepted and $\boldsymbol{b}$ is updated if: $\boldsymbol{o}$ dominates $\boldsymbol{b}$ ($\boldsymbol{o} \succ \boldsymbol{b}$); is equally good ($\boldsymbol{f}(\boldsymbol{o}) = \boldsymbol{f}(\boldsymbol{b})$); is a side step, i.e. $\boldsymbol{o}$ does not dominate $\boldsymbol{b}$ but it is also not dominated by any elitist solution $\mathcal{A} \not\succ \boldsymbol{o}$ (see Section 3.3.7). Otherwise, the changes are rejected and $\boldsymbol{o}$ is reverted to the backed-up state $\boldsymbol{b}$. Pseudo-code is given in Fig. 9.

The number of consecutive generations that the elitist archive (see Section 3.3.7) has remained unaltered is called the No-Improvement Stretch (NIS). A routine called Forced Improvement (FI), which was introduced in previous literature [37,102], is triggered when the NIS exceeds a threshold of $1 + \lfloor \log_{10}(n) \rfloor$. FI is a second round of OM in which the donor solutions are randomly chosen from the elitist archive. Furthermore, in the FI phase, a mixing step is only accepted if the offspring dominates its parent (i.e. $\boldsymbol{o} \succ \boldsymbol{b}$) or if it is a new nondominated solution ($\mathcal{A}^t \not\succ \boldsymbol{o} \wedge \boldsymbol{f}(\boldsymbol{o}) \notin \boldsymbol{f}(\mathcal{A}^t)$). FI for a solution is stopped with the first accepted change [102]. If $\boldsymbol{o}$ is still unchanged after FI, it is replaced by a random solution from the elitist archive.

Lastly, in every generation, in the cluster that has the largest mean value in objective i, the original SO version of GOM is used to perform variation [103]. Ties in terms of multiple cluster assignment being randomly broken (see Fig. 7).

3.3.6. Survivor selection and automated population sizing

At the end of each generation, a selection procedure is used to create the next population. If the current elitist archive (introduced in the next step) is larger than the population size, n leaders are chosen from the archive. If the elitist archive is smaller than required, the solutions from the population and archive are combined, with duplicated solutions being discarded. If the size of this new combined population is still smaller than the population size, the remaining spots are filled with new randomly-generated solutions (see step one). If the size is larger than required, solutions from the best nondominated fronts are chosen. With this strategy one avoids the inherent problem of MOGOMEA that the new population may actually not be well spread over the currently known Pareto front since the OM procedure only keeps the last modified version of a solution in the population, whereas all the solutions that were allowed into the elitist archive do not necessarily remain in the population.

In every generation, the population is increased by its initial size, n_i, by adding new randomly-generated solutions (see step one) to the population.

3.3.7. Elitist archive

The algorithm is equipped with a secondary population, called the elitist archive, for storing the nondominated solutions found during the search [108]. The use of such an archive is extremely useful since the primary population may be smaller than the number of Pareto front solutions and therefore nondominated solutions may be rejected during the selection procedure [109]. Every new and feasible solution, which dominates or is nondominated compared to its parent, is checked to see if it can be added into the archive. If the new solution is dominated by any archive member, it is discarded. If it is a new nondominated solution, it is added to the archive and the archive members that are dominated by it are removed. In the case that there exists an archive member with the same objective values, the previously archived solution is replaced by the new one if such replacement results in a diversity improvement for the archive in the decision-variable space. The solution which has a greater Hamming distance to its nearest archive neighbor is chosen [37].

3.3.7.1. New aspects. The MOGOMEA used in this article is based on previous work [37]. However a few alterations were implemented:

- The k-means algorithm is not used because the clusters means tend to drift "inwards", leading to reduced search effort in the vicinity of the Pareto extremes. Instead, the clusters are grown directly around the leaders chosen from the selection set [35].
- Previously, the LTs were learned on selection sets that were obtained using tournament selection with tournament size 2. In the new approach, the LTs are learned with all the solutions from the clusters, which is expected to increase the gene

```
MOGenepoolOptimalMixing(x, C, F, A^t)
 1  b ← o ← x;  f(b) ← f(o) ← f(x);  changed ← false
 2  for i ∈ {0, 1, ..., |F| − 1} do
 3     d ← Random(C)
 4     o_{F^i} ← MUT(d_{F^i})
 5     if o_{F^i} ≠ b_{F^i} then
 6        f(o) ← EvaluateFitness(o)
 7        A^t ← UpdateElitistArchive(o)
 8        if (o ≻ b) or (f(o) = f(b)) or (A^t ⊁ o) then
 9           b_{F^i} ← o_{F^i};  f(b) ← f(o);  changed ← true
10        else
11           o_{F^i} ← b_{F^i};  f(o) ← f(b)
12  if ¬changed or t^{NIS} > 1 + ⌊log10(n)⌋ then
13     changed ← false
14     for i ∈ {0, 1, ..., |F| − 1} do
15        d ← Random(A^t)
16        o_{F^i} ← d_{F^i}
17        if o_{F^i} ≠ b_{F^i} then
18           f(o) ← EvaluateFitness(o)
19           A^t ← UpdateElitistArchive(o)
20           if (o ≻ b) or (A^t ⊁ o and f(o) ∉ f(A^t)) then
21              b_{F^i} ← o_{F^i};  f(b) ← f(o);  changed ← true
22           else
23              o_{F^i} ← b_{F^i};  f(o) ← f(b)
24        if changed then breakfor
25  if ¬changed then
26     d ← Random(A^t); o ← p;  f(o) ← f(p)
```

Fig. 9. Pseudo code for MO Gene-pool Optimal Mixing [37].

diversity and decrease the fitness bias.

- The population size is a very important internal parameter of EAs that should be adjusted according to the instance of the WFLOP being solved [18,26]. If the population size is too small, there may not be enough genetic variation available to reach parts of the OPF [37]. In this article, a new population-growing scheme is introduced (Section 3.3.6). This makes the algorithm more robust because if a larger population size is needed than what is used initially, the algorithm will eventually reach this population size.

3.4. o-MOGOMEA

In each generation and for every cluster, a linkage model is learned in MOGOMEA by building a hierarchical cluster tree [104]. This is a key feature of the algorithm that makes it especially efficient for problems which have an exploitable linkage structure [37].

The MOWFLOP however is not a fully black-box optimization problem since it is known that turbines influence the energy production of neighboring turbines and that this influence might be considered negligible for turbines situated far enough apart [25,26]. This knowledge about the underlying problem structure suggests that a slightly different variant of the MOGOMEA may be designed that does not require structure learning. Although learning a linkage tree is relatively efficient compared to various alternative models in literature, especially for large problems this is still a potentially time-consuming part of the algorithm. For this reason, we consider a version of MOGOMEA in which the linkage tree is predetermined and kept fixed. To build this linkage tree, the geographical distance between the potential turbine positions is used as a distance metric. The underlying problem structure is thus learned offline. Hence, we refer to it as offline MOGOMEA (*o*-MOGOMEA).

3.5. NSGA-II

A flowchart of the NSGA-II is shown in Fig. 10 and a description of the algorithm is given next.

3.5.1. Population initialization

The population initialization scheme used for the MOGOMEA is also employed in the NSGA-II (see Section 3.3.1).

3.5.2. Ranking and crowding

Initially, the solutions of the population are ranked. Rank one is assigned to all solutions that are not dominated by any other solution. Rank two is given to the solutions which are only dominated by rank one solutions. The procedure is then repeated until all solutions are ranked.

The crowding distance is used to compare solutions in the same rank (see Fig. 11) and acts as a diversity operator. It measures the cuboid size defined by the locations of the closest neighbors (from the same rank) of a solution in the objective space, as shown in Fig. 11. Larger values for the cuboid are preferred as this indicates that the solutions are located in areas of the search space that are not crowded.

3.5.3. Parents selection

Two solutions are randomly chosen from the population and compared. The one with the lowest rank is selected. If they have the same rank, the one with the largest cuboid is chosen. The procedure is repeated until n parents are chosen.

3.5.4. Sampling

In this step, an offspring population is created. To do so, two parents are taken from the parent population, mixed via a classic crossover operator and mutated to generate two new solutions. The procedure is then repeated until all offspring are created.

3.5.5. Ranking and crowding

The same procedure is used as in step two. However, this is now performed in a set composed of the parent and offspring solutions.

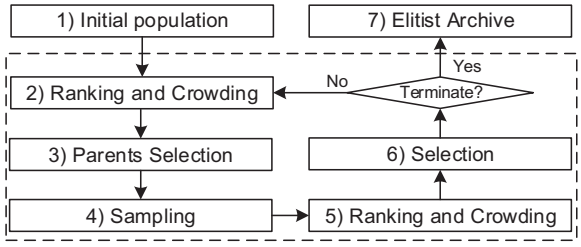

Fig. 10. Flowchart of the NSGA-II.

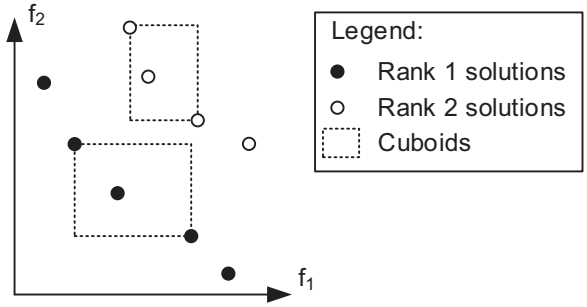

Fig. 11. Nondominated ranking and crowding distance [45].

3.5.6. Selection

The selection procedure of the MOGOMEA is used to create the new population (see Section 3.3.6). However, for the sake of making a fair comparison, the population size was not increased at the end of all generations because MOGOMEA uses far more fitness evaluations per generation than the NSGA-II (see Section 3.3.5). Hence, the population size of the NSGA-II-based algorithms was only increased after a number of generations that corresponds to an equivalent number of evaluations as would have been performed in MOGOMEA after one generation.

3.5.7. Elitist archive

Although the original NSGA-II did not make use of an elitist archive [45], it has been shown that its performance is enhanced if it is equipped with one [110,111,37]. Therefore, in this work the NSGA-II uses a similar elitist archive as the one used in MOGOMEA.

3.6. c-NSGA-II

In the default version of the NSGA-II the solutions selected for mating are randomly chosen from the entire population. To test the influence of using the clustering strategy for the MOWFLOP, we designed a new variant of NSGA-II: clustering NSGA-II (c-NSGA-II). The same principle as used in MOGOMEA is employed: solutions are recombined solely if they are in the same cluster. The remainder of the algorithm is the same as the standard NSGA-II.

3.7. Overview of the algorithms

Table 2 provides a comparison of the characteristics of the algorithms under study. The original implementation of NSGA-II does not have any of the characteristics that are being investigated, whereas the c-NSGA-II, due to clustering, differentiates variation along the Pareto front. The o-MOGOMEA includes inherent SO optimization in its extreme clusters and uses information of the WFLOP to learn the FOS offline. Finally, the MOGOMEA has all the characteristics and also learns the FOS structure throughout the optimization run.

4. Case study

This section provides the details of a case study. Specifically, the turbine and wind resource, wind farms, optimization goals, algorithm parameter settings, CHTs and performance indicators used are described in the following.

4.1. Turbine and wind resource

The selected wind turbine was the Vestas 8 MW [112] whose power and thrust curves are given in Table 3. The turbine has an 164 m rotor Diameter (D) and a hug height of 107 m. The power and thrust curves were linearly interpolated and all turbines were considered to be similar.

The wind resource used (displayed in Table 4) is based on measurement data collected in the North Sea [113]. The wind behavior may be characterized by a Weibull distribution [18,41]. Nonetheless, a discrete distribution was used during the optimization routine, in a similar fashion to other literature [87,38,33,78,84,90,91]. In this way, the computational cost to evaluate the energy production is low and, furthermore, the wake loss model used does not provide high-fidelity results, as stated previously (see Section 2.1). The wind resource was linearly interpolated and used at the turbine hub height.

Table 2
Characteristics of the MOEAs under study.

Algorithm	Clustering	SO optimizers	Problem structure
NSGA-II	No	No	No
c-NSGA-II	Yes	No	No
o-MOGOMEA	Yes	Yes	Offline
MOGOMEA	Yes	Yes	Online

Table 3
Turbine power and thrust values.

Wind speed [m/s]	Power production [kW]	Thrust value
4	100	0.700000000
5	570	0.722386304
6	1103	0.773588333
7	1835	0.773285946
8	2858	0.767899317
9	4089	0.732727569
10	5571	0.688896343
11	7105	0.623028669
12	7873	0.500046699
13	7986	0.373661747
14	8008	0.293230676
15	8008	0.238407400
16	8008	0.196441644
17	8008	0.163774674
18	8008	0.137967245
19	8008	0.117309371
20	8008	0.100578122
21	8008	0.086883163
22	8008	0.075565832
23	8008	0.066131748
24	8008	0.058204932
25	8008	0.051495998

Table 4
Wind resource: average speed and annual frequency of occurrence.

Direction [deg.]	Mean wind speed [m/s]	Frequency [%]
0	9.77	6.3
30	8.34	5.9
60	7.93	5.5
90	10.18	7.8
120	8.14	8.3
150	8.24	6.5
180	9.05	11.4
210	11.59	14.6
240	12.11	12.1
270	11.90	8.5
300	10.38	6.4
330	8.14	6.7

4.2. Wind farms

We designed four different areas (see Fig. 12a). All wind farms have a square area which is suitable for locations in which there is a predominant wind direction [34]. For each wind farm, three grid step sizes (2, 4 and 8D) were used to define possible turbine locations. The characteristics of the different wind farm areas are listed in Table 5. Note: the number of possible layouts include infeasible layouts.

The maximum number of wind turbines, n_{pack}^{max}, is the number of turbine positions available using an 8D grid step size. It should be noted that in our setup, the hexagonal packaging, the densest circle packing in a plane [114], is not possible. In fact, a grid step size of eight, or multiples of eight, times smaller than the minimum separation distance is recommended to allow a hexagonal packaging [95].

4.3. Optimization goals

The two design objectives chosen for our case study, namely AEP and efficiency, are important design goals for wind farm designers. The optimization goals are described and the motivation for selecting them, is provided in the following.

4.3.1. Energy production

Energy production is one of the most commonly used optimization goals both in academia (all previous MO approaches for the WFLOP used energy production as the first optimization goal (see Table 1)) and in commercial software [21]. It is described by:

$$E_{wf} = \frac{8760 \sum_{i=1}^{\theta} \sum_{j=1}^{turbs} P_j\left(\overline{wind}_i\right) \cdot f_i}{8760 \sum_{i=1}^{\theta} n_{pack}^{max} \cdot P_{turb}^{ideal}\left(\overline{wind}_i\right) \cdot f_i} \tag{8}$$

where f_i is the wind frequency of occurrence for direction i, $\overline{wind}$ is mean wind speed for the i-th direction, $P_j\left(\overline{wind}_i\right)$ is the power production of the j-th turbine for the wind speed $\overline{wind}_i$, n_{pack}^{max} is the maximum packing of turbines for the wind farm area, 8760 is the number of hours in one year and $P_{turb}{}^{ideal}$ is the energy production of a turbine without wake losses.

4.3.2. Efficiency

Efficiency is targeted at the maximization of the usage of the installed turbines. A common measure for this is the coefficient of utilization, which is the fraction of the year required for the wind farm to produce its annual production if it produced energy at its full power all the time. This measure is also captured by the maximization of the wind farm efficiency [88,24,25,73]. The ideal wind farm production is the power production of a single wind turbine (without wake losses) multiplied by the number of turbines in the wind farm. The wind farm efficiency is calculated by:

$$\eta_{wf} = \frac{\sum_{i=1}^{\theta} \sum_{j=1}^{turbs} P_j\left(\overline{wind}_i\right) \cdot f_i}{n_{turbs}^{wf} \cdot P_{turb}^{ideal}} \tag{9}$$

where η_{wf} is the wind farm efficiency and n_{turbs}^{wf} is the total number of turbines in the wind farm.

Wind farm energy production and efficiency are conflicting design goals. The wind farm energy production is maximized by placing more turbines in the wind farm area. However, both the energy production and wake effects increase with the number of turbines. Only for very large wind farms it could happen that the energy generated by an extra turbine would not be enough to compensate for the additional wake losses. The second design goal is maximized by reducing the wake effects between the turbines. However since the wake effects increase with the turbine number, this objective is maximized for layouts with only a few turbines that are placed far apart.

4.3.2.1. Motivation. The selected optimization goals solely depend on the energy production and wake losses. A more detailed WFLOP would need to use nested algorithms, e.g. an heuristic to design the collection system [115–118]. Although this is of importance to the final application, the use of nested algorithms also substantially increases runtime while it does not provide additional insight into the topics under study in this article. Therefore, this is omitted here.

4.4. Constraint-handling approaches

We use a minimum separation of 8D between neighboring wind turbines [48,89]. Hence, the algorithms are required to

Table 5
Characteristics of the wind farm areas under study.

Wind farm	Area (km^2)	Maximum packing	Step size (D)	Number of variables	Number of possible layouts
A	15.49	16	8	16 (4^2)	65,536
			4	49(7^2)	5.6×10^{14}
			2	169(13^2)	7.5×10^{50}
B	61.97	49	8	49(7^2)	5.6×10^{14}
			4	169(13^2)	7.5×10^{50}
			2	625(25^2)	1.4×10^{188}
C	139.43	100	8	100(10^2)	1.3×10^{30}
			4	361(19^2)	4.7×10^{108}
			2	1369(37^2)	1.3×10^{412}
D	247.89	169	8	169(13^2)	7.5×10^{50}
			4	625(25^2)	1.4×10^{188}
			2	2401(49^2)	5.9×10^{722}

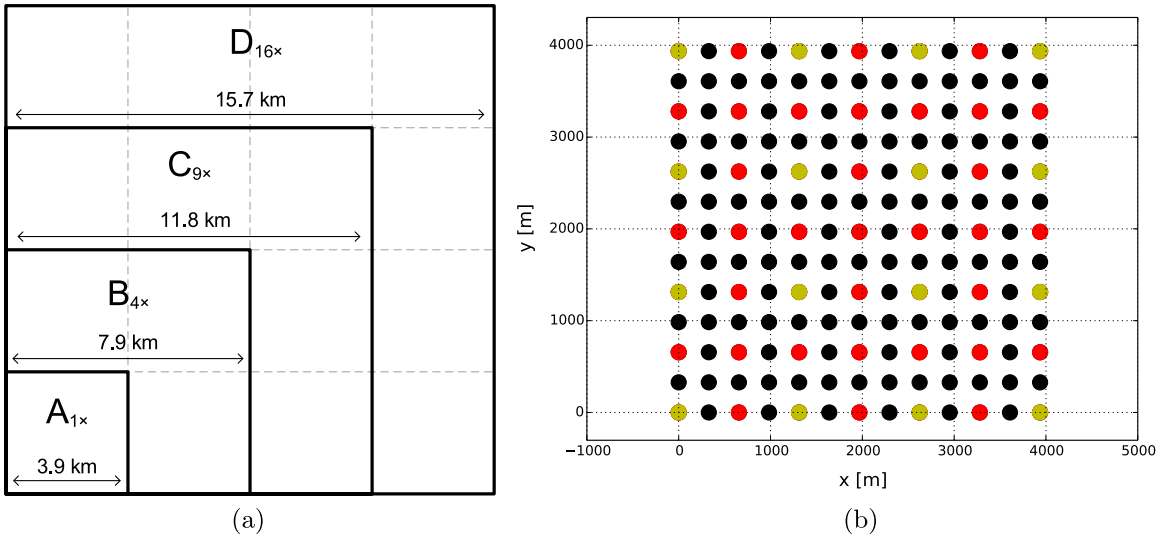

Fig. 12. Figure (a) shows the dimensions of the wind farm areas considered in the case study. Figure (b) shows the turbine locations for wind farm A and three different grid steps: 2D spacing - all locations; 4D - red and yellow (every second location); 8D - yellow (every forth location). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article).

handle this proximity constraint for layouts with 2D and 4D grid step sizes. All CHTs used (except for the extra optimization goal strategy) were implemented to assess their impact on the performance of the algorithms under study.

4.4.1. No constraints

The grid step size was selected as the minimum separation required for the turbines (8D). In this way, the algorithms may place turbines in any of the locations while guaranteeing that the wind farm layout remains feasible.

4.4.2. Constraint domination

The constraint violation value is the sum of the number of pairs of turbines which are placed closer than the minimum distance. During variation in MOGOMEA, a newly generated solution is directly compared with its parent (see Fig. 9). Therefore, if a solution violates constraints, it is discarded before its fitness is evaluated. Since the initial population is guaranteed to be feasible (see Section 3.3.1), MOGOMEA does not cross into the infeasible part of the search space. In a similar fashion, the NSGA-II does not evaluate newly generated infeasible layouts and they are not present in the offspring population.

4.4.3. Penalty term

A penalty term was constructed as follows: for each pair of turbines that violate the proximity constraint, the energy production was reduced by what one turbine would produce if it were installed alone (P_{turb}^{ideal}). In this way, it is guaranteed that the solutions are not under-penalized since a turbine installed alone would not have any wake losses. This approach is called minimum penalty rule since the penalty is just above the limit where infeasible solutions are preferred [86]. The penalization weight increases with the number of turbine pairs that violate the proximity constraint because the penalization grows linearly, whereas the energy generation does not grow proportionally due to the increase of wake losses. Hence, the penalty term behaves dynamically and is not only based on the number of proximity constraints which are violated, which has demonstrated to provide better results for benchmark functions [119]. Using a penalty term, all algorithms evaluate infeasible layouts. Although this might not be possible for some optimization problems, the wake losses model used is able to provide output. Nonetheless, these evaluations are not accurate since the Katic-Jensen model is a far-wake model [63].

4.4.4. Repair mechanism

To repair infeasible solutions, a turbine is randomly chosen to be removed from a pair of turbines that are too close. This strategy is expected to remove, half of the time, the turbine which most degraded the performance of the wind farm. The procedure is repeated until the layout becomes feasible.

4.4.5. Resample

With this approach, new layouts are resampled until a feasible one is created. For NSGA-II variants, the parent solutions are directly carried to the next population if the newly generated layouts are still infeasible after 100 trials. For MOGOMEA variants, a new donor is randomly chosen every time an infeasible solution is generated. If no feasible layout is created after 100 trials, the solution remains unaltered and the algorithm moves on to the next FOS subset.

4.5. MOEAs

4.5.1. MOGOMEA and o-MOGOMEA

Following recommendations from recent literature, we used five clusters [37]. Hence, two clusters are used to extend the Pareto front towards the two individual optimization goals while the remaining three clusters are responsible for finding good solutions in the center part of the OPF.

For cluster sizing, we initially set each cluster size to four, which is the minimal size with which MOGOMEA robustly solves the one-max zero-max problem [37]. This leads to an initial population of size 20, regardless of the dimensionality of the problem. This is however compensated by the population-increment mechanism which increases the population size at the end of each generation (see Section 3.3.6).

4.5.2. NSGA-II and c-NSGA-II

For variation, two-point crossover was used with probability 0.9 [37] and bit-flipping mutation with probability $1/l$, as originally proposed [45]. Five clusters were used in the c-NSGA-II in a similar fashion to the MOGOMEA variants.

4.6. Measuring performance

As previously noted, optimal solutions for the WFLOP can only be obtained for wind farms with a low number of potential turbine locations, given a limited budget of evaluations. Therefore, comparisons between EAs, for larger scenarios, may only be done on a relative basis [21]. We consider the elitist archive upon termination to be the approximation set (Figs. 6 and 10). The elitist archive size was set large enough to accept all nondominated layouts for all cases, preventing oscillatory convergence of the archive [120].

To compare approximation sets, the hypervolume indicator was chosen since it is a commonly adopted useful indicator for evaluating the performance of algorithms on problems for which the OPF is unknown [121]. This performance indicator computes the search space covered by an approximation set (using a reference point). A higher hypervolume value is indicative of better performance. The hypervolume indicator captures both the diversity (even if all solutions are on the OPF, the indicator is not maximized unless the solutions are also spread out) and the proximity of the approximation set to the OPF.

The algorithms were given one million function evaluations (per objective) for all case study instances. However, if the hypervolume did not improve by at least 10^{-5} after $2 \times$ NIS generations, the algorithms were stopped. This means that further improvements might have been found if the algorithms were run longer but the convergence was deemed to be too slow [96]. Both optimization goals were normalized and the point $(0, 0)$ was used as reference point for computing hypervolume. As a result, the maximum hypervolume value is one. Note that this value cannot be achieved since it requires a layout with the maximum turbine packing and no wake losses.

5. Results

The experiments were run using Python implementations of the algorithms on a server computer with 32 cores (Intel Xeon ES-2690@2.9 GHz) running the 64-bit version of Ubuntu 12.04. The results were averaged over 10 independent runs. Next, we will address the open research questions presented in the introduction of the article.

5.1. What characteristics should an optimization algorithm have to present optimized layouts?

Although it is very hard to identify which feature has led to an improvement of the results, we try to breakdown and analyze each one of the characteristics under study.

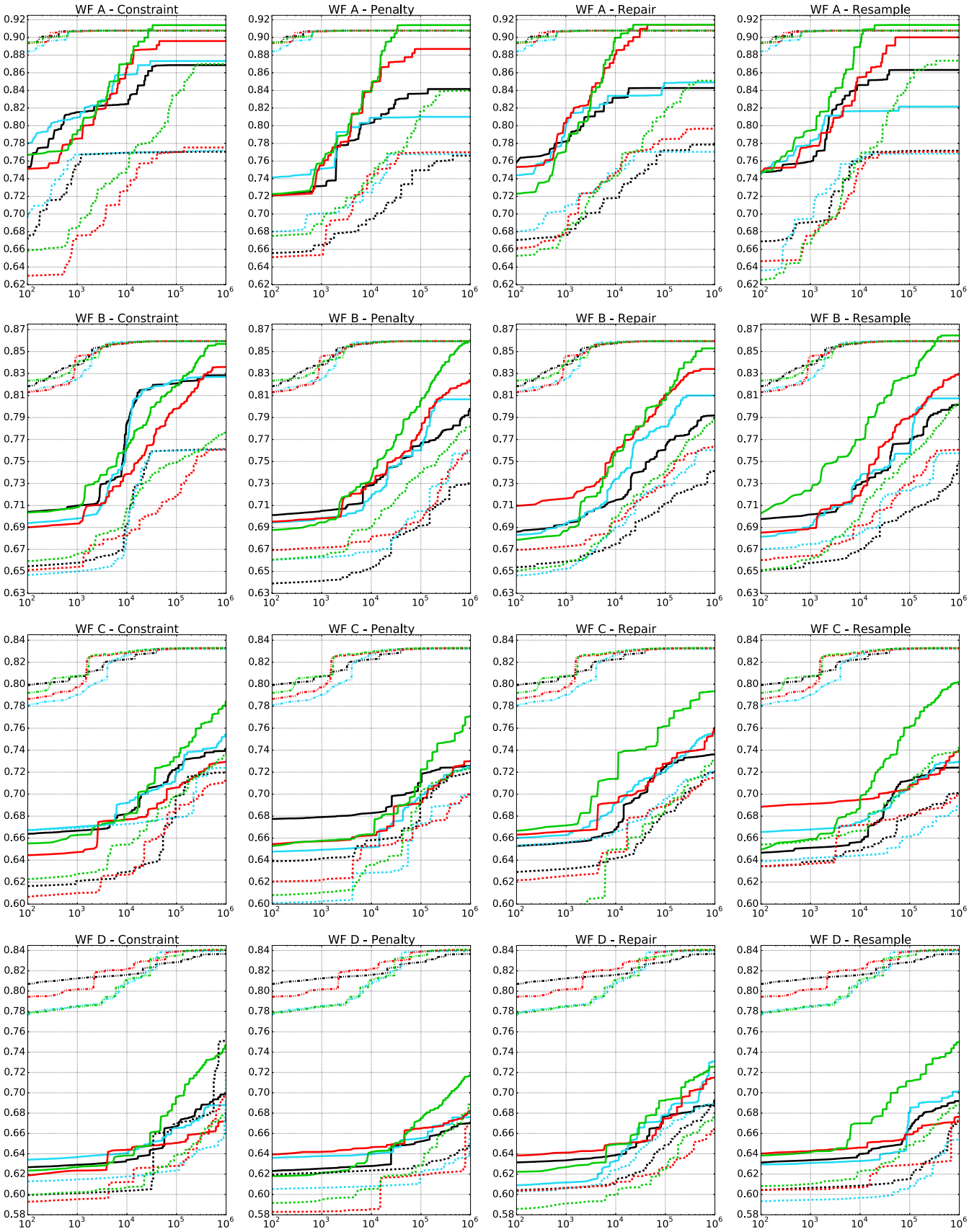

Fig. 13. Averaged hypervolume obtained with the different algorithms, CHTs and grid step sizes. The *x*-axis shows the number of fitness valuations and the y-axis displays the hypervolume. MOGOMEA - red; o-MOGOMEA - green; NSGA-II - black; c-NSGA-II - blue. 8D grid spacing - dash-dotted lines; 4D - solid lines; 2D - dotted lines. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

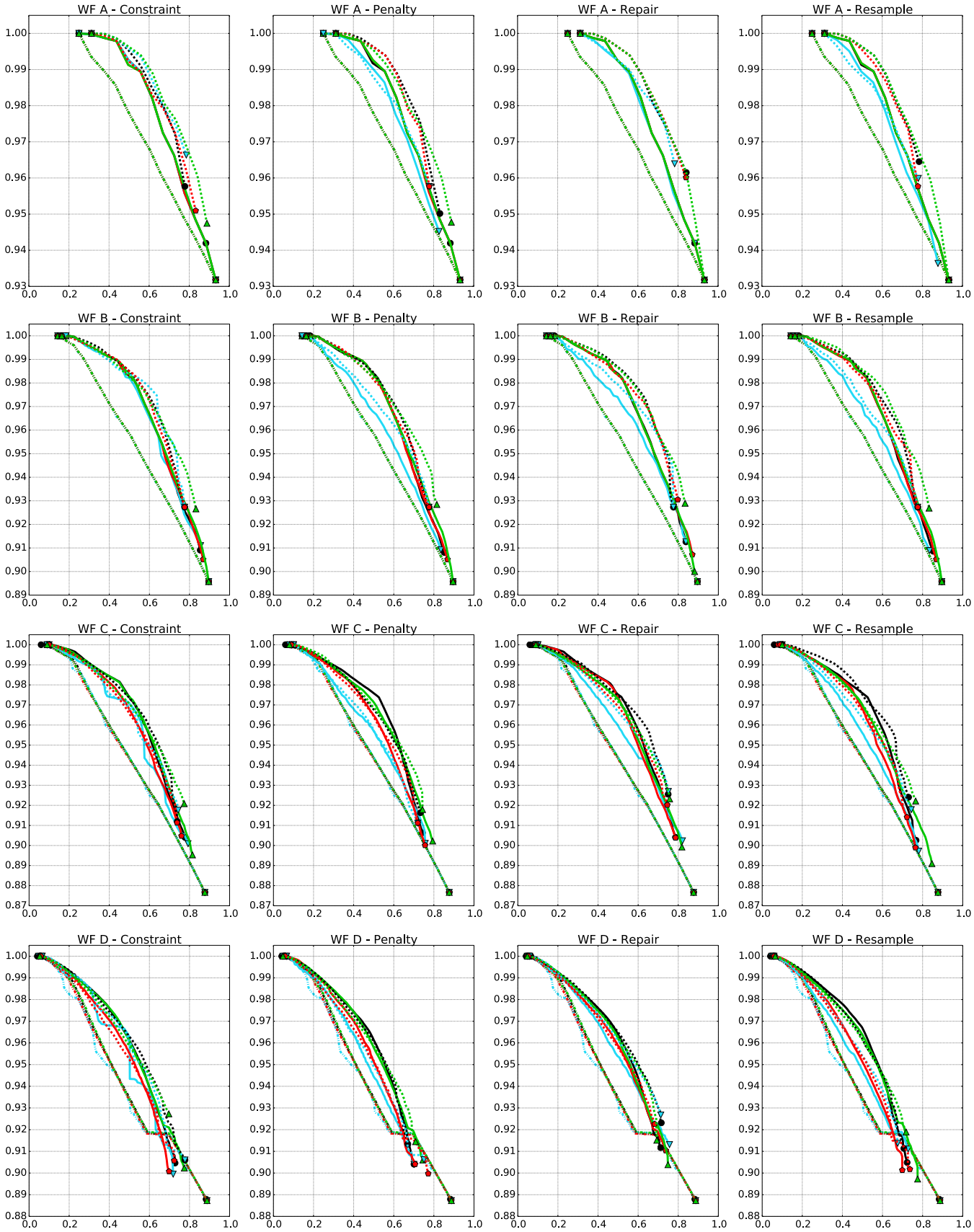

Fig. 14. Aggregated Pareto fronts obtained with the different algorithms and CHTs. The *x*- and *y*-axes show the optimization goals, respectively. MOGOMEA - red; o-MOGOMEA - green; NSGA-II - black; c-NSGA-II - blue. 8D grid spacing - dash-dotted lines; 4D - solid lines; 2D - dotted lines. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

5.1.1. Clustering

The results shown in Fig. 13 demonstrate that the clustering scheme did not always improve the performance of the NSGA-II. Overall, both variants of the NSGA-II had similar average performance.

5.1.2. SO Optimization

Fig. 14 shows that, in general, NSGA-II variants did not extend the front as much as MOGOMEA variants did towards the highest energy production goal. A reason for this is that NSGA-II variants did not have any mechanism that puts extra pressure on individual objectives. As previously noted, these mechanisms can be highly beneficial if one of the goals is much harder to solve. In fact, finding the layout with the highest density pack of turbines in a pre-determined area is equivalent to the circle packing problem, which is a NP-complete problem [122,123]. Furthermore, the number of available layouts in this extreme region of the OPF is much lower than in the middle section due to the lower number of available positions.

5.1.3. Problem internal structure

Using information on problem structure demonstrated to be beneficial since the MOGOMEA variants, on average, perform better than the NSGA-II variants as shown in Fig. 13.

In previous literature, it was shown that an online approach to learning linkage outperforms the offline approach for problems with an intricate interaction structure (and conversely in case of a straightforward structure) [104]. On the problems tested here, the *a priori* fixed model used in the o-MOGOMEA resulted in improved performance over MOGOMEA. It is important to point out that the same maximum number of fitness evaluations (one million, for each objective) was used for all wind farm areas and that the number of fitness evaluations, per solution, during the OM procedure has a ceiling of $2^l - 2$. For this reason, less generations were available to the MOGOMEA to learn linkage information online. A further reason for the improved performance of the offline learning variant is that the linkage structure in the MOWFLOP is largely dominated by the physical locations of the turbines, suggesting that the linkage structure of the MOWFLOP is not highly intricate and thus can well be exploited using an offline-learned structure.

5.2. What is the best constraint-handling technique to ensure feasibility of the OWF layouts?

The MOGOMEA presented, on average, higher hypervolumes when equipped with the repair mechanism, especially in case of a grid resolution of 4D. The o-MOGOMEA exhibited superior performance when equipped with the resample approach. Both variants of NSGA-II showed the best average performances when using the constraint domination technique. Overall, no CHT has demonstrated to lead to superior performance in all problem instances.

5.3. How does the problem complexity scale with the number of design variables?

We note that in this article we do not adhere to the common definition of scalability in experimental algorithmic design (i.e. how an algorithms' requirements such as memory, computing time or number of evaluations to reach the optimum increase as the problem size increases), but rather the impact of increasing the size of the problem on the performance of an algorithm under a predefined fixed budget of evaluations. As a final remark, although we are assessing the impact of the number of variables of the MOWFLOP on the computational time, the design phase of a wind farm is a process that takes several months [124] and hence, it is an optimization problem that is not highly time-bounded in reality [96]. Therefore, the results obtained in this article should be mostly used as a guideline for the expected run times of the algorithms when equipped with different CHTs and applied to the MOWFLOP with different number of variables and wind farm sizes. Fig. 15 shows the average run times of the algorithms.

First, it should be noted that algorithms may have been stopped before reaching the maximum number of evaluations. Second, there are multiple sources of influence on the time requirements. As a problem gets bigger (smaller grid spacing), the time requirements for model building (MOGOMEA) increase. However, for an algorithm to reach solutions of a similar quality on a larger problem, a higher number of evaluations is typically required. Since we have fixed the budget of function evaluations, final results are expected to be of lower quality for larger problems. Because evaluating the objectives involves simulating wind farms, these evaluations are thus less costly for the larger problems, within our budget of evaluations. Because the evaluation time is substantial for our application, the observed time requirement does not always increase, especially for the biggest problem, wind farm D. Consequently, we can safely say that wind farm D was too big for interesting results to be expected within our allowed time budget.

The results further show that the model building of the MOGOMEA is not a major extra computational burden. In fact, the offline learned linkage tree allowed o-MOGOMEA to find better results, i.e. more densely packed wind farm layouts, which almost always leads to requiring more computation time than does the fact that MOGOMEA has to learn a linkage tree online every generation. Only for the biggest problem, Case D, requiring almost 2500 variables for a 2D grid resolution, a clear increase in required computation time for MOGOMEA versus o-MOGOMEA can be seen. On the other hand, it is important to realize that with longer runs, more densely packed wind farms would be found, ultimately increasing again the time spent performing function evaluations, potentially surpassing the time spent building linkage models.

Differently from the MOGOMEA, the run-time requirement of all remaining algorithms does not increase substantially with the number of optimization variables. This can be easily seen for the run times of case D. All the algorithms presented shorter average runs when a 2D step size was employed. Once again, this is explained by the fact that the most packed wind farms were not achieved.

Regarding the impact of the CHTs, the use of the penalty term resulted in longer running times, for 4D and 2D grid resolutions, since infeasible layouts were evaluated that were packed with more turbines.

Consequently, the main influence on the running time, if running the algorithms sufficiently long (as would typically be the case in a real-world scenario), the main influence on computation time is the evaluation of the problem, which becomes especially expensive if the layouts become dense. Ultimately, given a fixed time-budget rather than a number-of-evaluations budget (as would be likely in a real-world scenario), a big advantage could be the use of CHTs that do not allow evaluating infeasible layouts because the number of evaluations per second would thereby increase, thus allowing the optimization algorithms to progress further.

5.4. What is the relation between problem dimensionality/complexity and the degrees of freedom offered by different turbine-placement grid resolutions?

All algorithms were able to find similar Pareto fronts for all wind farm areas and CHTs when the 8D grid step size was used. The MOWFLOP with such a coarse grid becomes much easier to solve since it becomes an unconstrained problem with only few

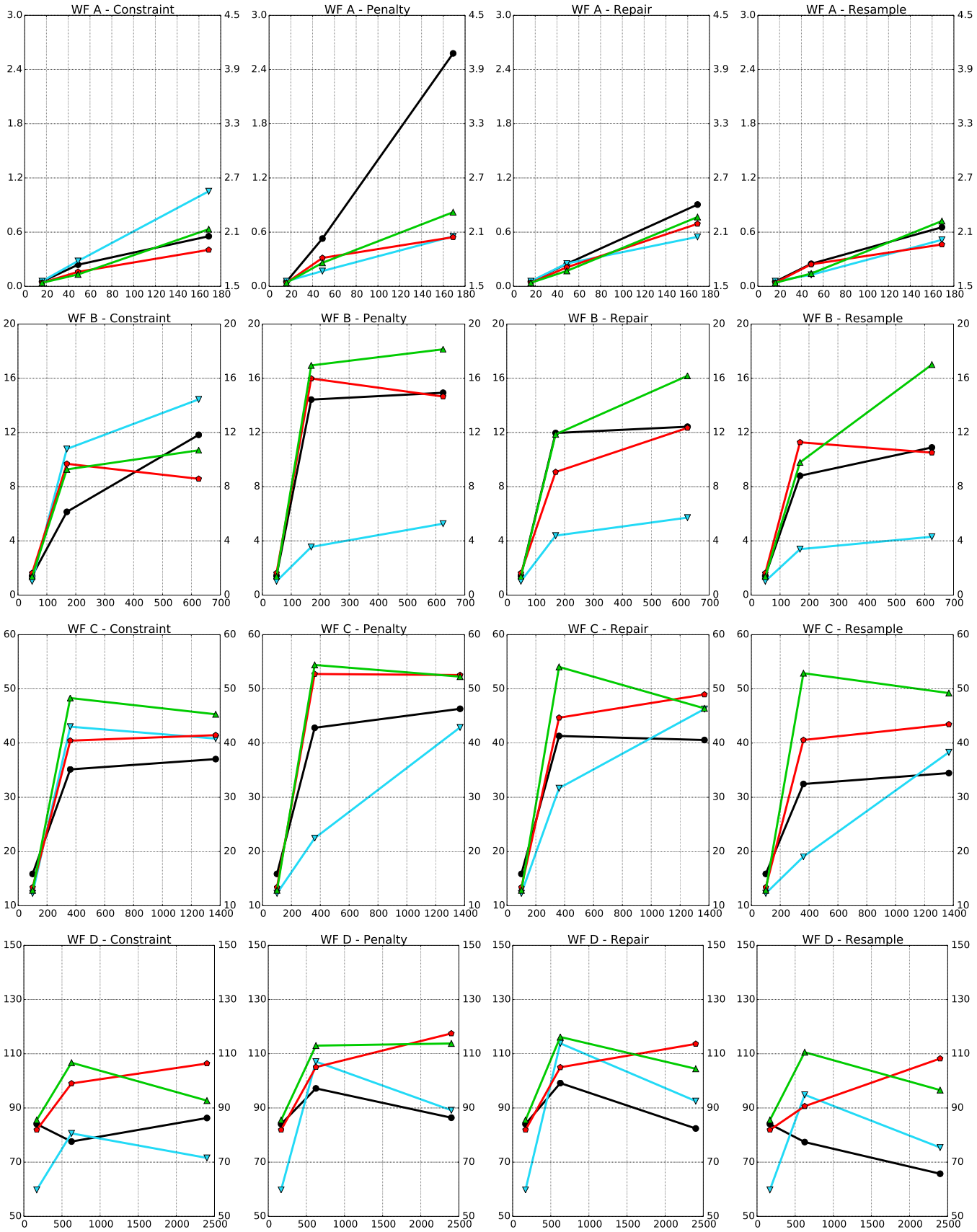

Fig. 15. Time needed by each optimization algorithm to converge for the different constraint handling techniques. The x-axis shows the number of variables and the y-axis displays the average time, in hours, needed for termination. MOGOMEA - red; o-MOGOMEA - green; NSGA-II - black; c-NSGA-II - blue. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

variables. All algorithms converged before one million evaluations for all instances of the 8D step size. Although relatively fast to compute, the real-world applicability is low because of the limited design freedom.

Smaller grid step sizes allow to find better wind farm layouts in the middle section of the Pareto front for problem instances A and B. However, for larger areas (C and D) the advantage of using the finest step size (2D) decreased within our limited budget of evaluations, since the problem complexity is highly increased (see Table 5), which is in line with previous literature [34].

Therefore, if wind farm developers aim at highly dense wind farms they should either use large grid spacings, manually create initial layouts to help the algorithms find good solutions in that area of the search space, let the algorithms run longer or start with a larger initial population size. The results obtained also indicate that it could be beneficial to use a Multi-Resolution (MR) scheme in which a large grid step is employed in the first optimization iteration to allow the algorithms to find densely packed wind farms with more ease [38]. Thereafter, these solutions would serve as an initial population for a second round of optimization in which a smaller grid step is used. In order to validate this assumption, we implemented an MR approach, which is introduced next.

5.5. Multi-resolution

The algorithms start optimization with the 8D grid step and advance to the next smaller step size if the hypervolume did not increase at least 10^{-5} after NIS generations. The elitist archive is then mapped onto the finer grid and the new population is created through the same procedure used at the end of each optimization run (see Section 3.3.6). When the algorithms advance to a finer grid resolution, all the new variables are set to zero since they were not mapped before. However, all algorithms create new offspring by exchanging subsets of variables between solutions. In this way, the only sources of new genetic material are the random solutions added at the end of each run and the mutation operator. To increase the impact of the latter, the mutation probability for the 8D step was kept constant, meaning that the rate is higher than previously for the 4D and 2D step sizes.

Fig. 17 shows the results for the hypervolume and aggregated Pareto fronts for wind farm area B. The left plot shows the hypervolume of the MR approach for 4D and 8D step sizes. It can be seen that initially all the algorithms present similar hypervolume values since the same 8D step size is being used. Once the algorithms converge, they advance to the next grid resolution. This is accompanied with a higher design freedom and hence, better layouts are found. Both MOGOMEAs presented higher average hypervolumes with the MR scheme.

The right plot of Fig. 17 shows the aggregated best Pareto fronts for all step sizes and the aggregated Pareto front found with the MR scheme. The MR approach allowed the algorithms to find Pareto fronts which cover the entire spectrum of wind farm densities. Both variants of MOGOMEA outperformed the NSGA-II implementations. Nonetheless, some parts of the new fronts are dominated by solutions found with the previous fixed grid schemes. The fact that the shown fronts are aggregated over multiple runs may explain this result.

5.6. Wind farm layouts

The Pareto front obtained for wind farm D with an 8D step size in the grid has a peculiarity: in the middle section there are several wind farm layouts with similar efficiency but with different energy production. If the number of turbines was fixed before optimization, wind farm designers would not find out that there are wind farm layouts with more turbines, and hence, more energy production, and similar wake losses. This demonstrates the advantage of using multi-objective optimization for the design of wind farms.

The results also show that the heuristic used for the initial populations (see Section 3.3.1) did not create solutions representing highly-packed layouts for the larger wind farm areas when the 4D and 2D grid spacings were used. Therefore, the optimization algorithms are required to find the more packed wind farm layouts, differently from previous literature in which a feasible solution with the maximum number of turbines was used as an initial layout [96]. Nonetheless, the heuristic had some effect in the performance of the algorithms. Similar to what was found in previous works [96], if the heuristic was able to find a layout with a high turbine density this could help the algorithms to extend the Pareto front in the first optimization goal.

Fig. 13 shows that the hypervolume was lower for problem instances with larger wind farm areas. Such areas allowed layouts with more wind turbines and, hence, lower efficiencies. Furthermore, the impact of extending the front is much larger in the hypervolume indicator than improving the solutions in the middle of the front. This explains why the layouts with an 8D grid step size had, in general, a larger hypervolume indicator, despite their lower design freedom.

Fig. 16 compares several wind farm layouts, composed of the same number of turbines, obtained with the o-MOGOMEA with a grid resolution of 8D and with the MR scheme. The algorithm, with an 8D step, optimized the layouts by placing more turbines at the edges of the farm and leaving wider spaces in the middle region. In this way, there is more free space for wake recovery and therefore, turbines in the wake of other turbines will receive higher mean wind speeds. On the other hand, with the MR scheme the free areas are evenly distributed throughout the area and spread the turbines in the area while preventing rows or columns from being formed. In fact, alignment of turbines only happened once it could not be avoided due to space limitations (see last plot of Fig. 16). The wind farm layouts obtained with the MR scheme go against what has been done for most of the existing wind farms, in which the turbines were installed in grid-based layouts Fig. 17.

6. Conclusions

The main objective in this article was to address several important open questions in solving the WFLOP and more specifically, in its MO variant, which is arguably more relevant to real-world applications. The NSGA-II, the MOGOMEA and two variants of these algorithms were applied to solve the MOWFLOP, while considering different CHTs. Wind farms with different areas, discretized using three grid step sizes to identify potential turbine locations, were used to assess the performance of the MOEAs.

The two variants of the MOGOMEA obtained, on average, better wind farm layouts as well as areas with higher density of turbines and hence higher energy production. This result indicates that algorithmic characteristics specific to the MOGOMEA are important factors to obtain good trade-off solutions. The MOGOMEA combines wind farm layouts that are roughly similar (in objective space) to create new layouts. The MOGOMEA also actively searches for wind farm layouts that represent extreme trade-offs, i.e., that optimize only a SO. Finally, the MOGOMEA tries to identify and exploit the underlying variable-dependency structure of the problem.

No specific CHT demonstrated to have clear a advantage over the others for all algorithms. The resample approach was the best CHT for the o-MOGOMEA, whereas the repair mechanism was the most suited for the MOGOMEA. Finally, the constraint domination

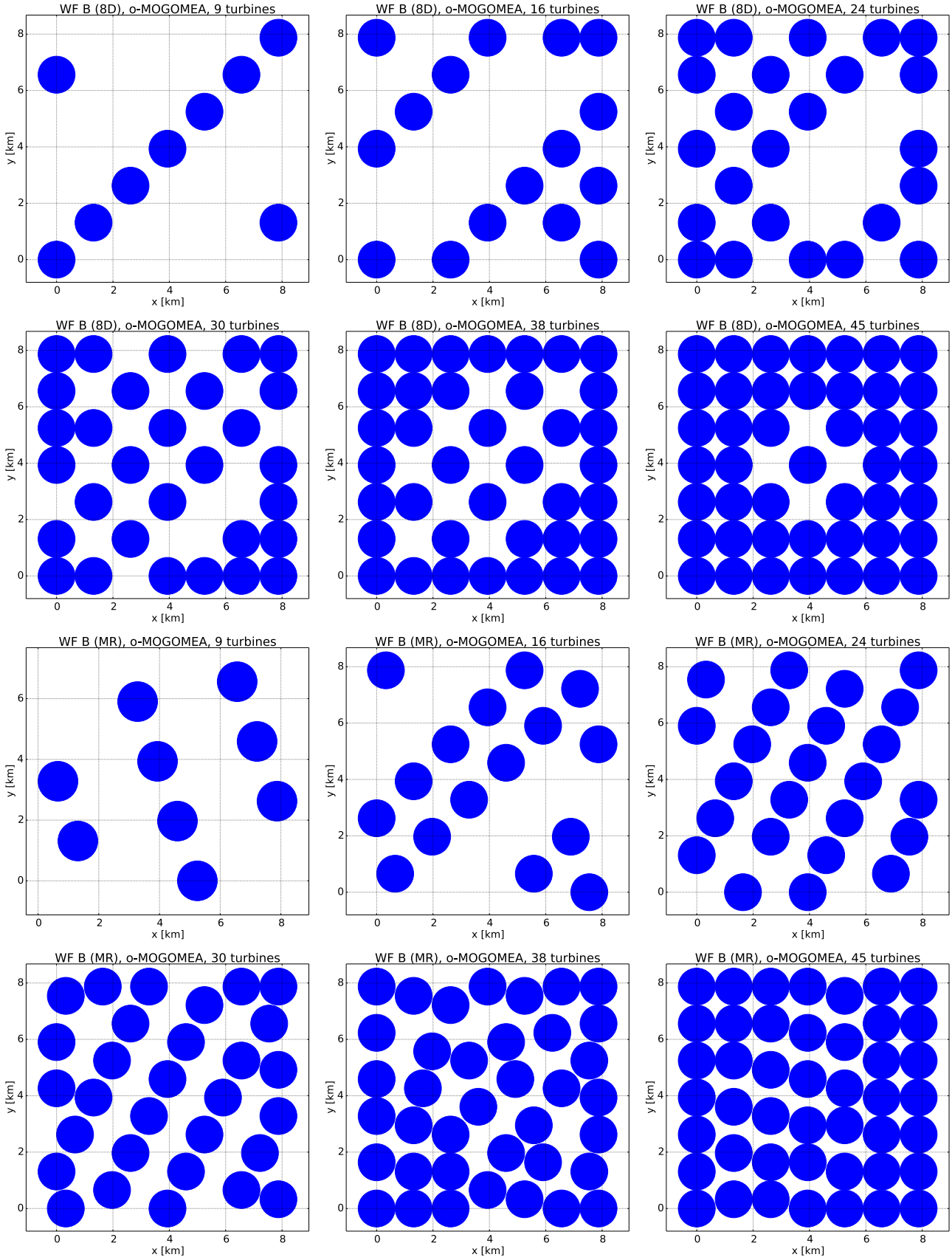

Fig. 16. Layouts obtained with the o-MOGOMEA for an 8D step size and using the multi-resolution scheme. The turbines are located at the center of the circles, which present the minimum separation between neighboring turbines.

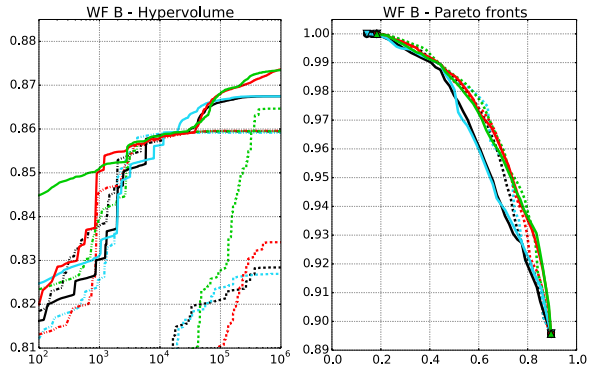

Fig. 17. The left plot shows the average hypervolume obtained with the different algorithms with 8D (dash-dotted), 4D (dotted) and with the multi-step (solid) approach for wind farm area B. The *x*-axis shows the number of fitness valuations and the *y*-axis displays the hypervolume. The right plot shows the aggregated Pareto fronts between all the different step sizes (dotted) and the aggregated Pareto fronts obtained with the multi-resolution scheme (solid). The *x*- and *y*-axes show the optimization goals, respectively. For both plots: MOGOMEA - red; o-MOGOMEA - green; NSGA-II - black; c-NSGA-II - blue. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

technique presented better results for both variants of the NSGA-II. When used for real-world planning however, CHTs that first check constraints and prevent infeasible solutions from being evaluated, may present a potential advantage because they effectively increase the number of evaluations that can be performed per second.

Regarding the trade-off between problem dimensionality/complexity and design freedom, the results showed that, for all algorithms and small wind farm areas, finer grid step sizes allowed to find better wind farm layouts in the middle section of the Pareto front due to the larger degree of freedom in designing layouts. On the other hand, for larger areas, the advantage of using fine step sizes was reduced within the time we allowed the algorithms to run, due to increase of the problem complexity.

The results demonstrated that there is a trade-off between the computational time required for model building of the MOGOMEA and the evaluation time required to calculate the power production of the wind farm. The increase of the number of variables leads to higher computational requirements for model building. For very fine grid resolutions the time to build and exploit linkage models may overtake the time to evaluate the efficiency of wind farm layouts. However, for a problem such as MOWFLOP, we demonstrated that very good results can be obtained by fixing the linkage model to one that is learned a priori, i.e., offline, based on potential turbine locations since these locations are expected to have the most impact on the dependencies between problem variables in the MOWFLOP. This fully removes the potentially time-consuming overhead of model building as the problem size increases due to the use of finer grid resolutions.

Finally, designing wind farm layouts based on a multi-resolution approach whereby wind farms are first designed using a coarse grid, and over time are adjusted using finer grids, demonstrated to offer the best results when considering the same budget of function evaluations. The reason for this is that a wide trade-off is fairly easily obtained using a coarse grid resolution, which is then further improved along the entire front of trade-offs using finer grid resolutions.

For future work it would be interesting to assess the influence of smaller grid steps and wind farms with larger areas. It would also be interesting to test instances of the MOWFLOP with more than two optimization goals. Recent works introduced strategies to filter hierarchical linkage relations from the LT that may be superfluous leading to more concise linkage models without negatively effecting the performance [103]. Hence, it could be interesting to evaluate these filtering strategies in the MOWFLOP. The proposed population-sizing-free scheme has its drawbacks and other approaches may still work better, which is interesting to study further. Furthermore, the multi-step approach demonstrated to provide good results and hence, an in-depth study into the competences and a true (experimental) scalability of this approach that designates the required resources such as runtime are required to reach the optimal Pareto front, should be carried out in future work.

Acknowledgements

The work has partly been performed within the project “Far and Large Offshore Wind (FLOW)”. The project is supported by the Ministry of Economic Affairs, Agriculture and Innovation of the Netherlands within the EOS-LT program of Agentschap-NL (P201101-005-ECN). The opinion expressed by the authors do not necessarily reflect the position of the Ministry of Economic affairs, nor does it involve any responsibility on its part.

The authors would like to thank the Het Collectief company (http://www.collectief.org) for providing the computational resources used to obtain the results presented in this paper.

References

[1] European Commission, [link]. URL ⟨http://www.eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:52008DC0768:EN:HTML⟩ [Last accessed 01.03.15.].
[2] Green R, Vasilakos N. The economics of offshore wind. Energy Policy 2011;39(2):496–502.
[3] EWEA, The european offshore wind industry - key trends and statistics 2013, Tech. Rep.; 2014.
[4] EWEA, Oceans of Opportunity - Harnessing Europe's largest domestic energy resource, Tech. Rep. September, EWEA; 2009.
[5] EWEA, 2030 Targets Bringing Certainty, Tech rep; 2013.
[6] Rodrigues S, Restrepo C, Kontos E, Pinto RT, Bauer P. Trends of offshore wind projects. Renew Sustain Energy Rev 2015;49:1114–35.
[7] 4C Offshore. [link]. URL ⟨http://www.4coffshore.com/windfarms⟩ [Last accessed 01.03.15.].
[8] LORC. [link]. URL ⟨http://www.lorc.dk/offshore-wind-farms-map/list⟩ [Last accessed 01.03.15.].
[9] The Wind Power; 2014. [link]. URL ⟨www.thewindpower.net/windfarms_offshore_en.php⟩ [Last accessed 01.03.15.].
[10] Renewables Map. [link]. URL ⟨http://www.renewables-map.co.uk/windfarm.asp⟩ [Last accessed 1.03.15.].
[11] Bilgili M, Yasar A, Simsek E. Offshore wind power development in europe and its comparison with onshore counterpart. Renew Sustain Energy Rev 2011;15(2):905–15.
[12] RWE. [link]. URL ⟨http://www.rwe.com/web/cms/en/1252456/rwe-innogy/sites/wind-offshore/under-construction/gwynt-y-mr/tech-and-spec/⟩ [Last accessed 31.03.15.].
[13] The Crown Estate, Offshore Wind Cost Reduction - Pathways Study, Tech Rep; 2012.
[14] Serrano-Gonzalez J, Gonzalez-Rodriguez AG, Castro-Mora J, Burgos-Payan M, Riquelme-Santos J. Overall design optimization of wind farms. Renew Energy 2011;36(7):1973–82.
[15] Sorensen T, Thogersen ML, Nielsen P. Adapting and calibration of existing wake models to meet the conditions inside offshore wind farms, Tech Rep, EMD International A/S; 2008.
[16] Anholt. [link]. URL ⟨http://www.anholt-windfarm.com/en/the-project/project-site-and-scope⟩ [Last accessed 31.03.15.].
[17] The Crown Estate, Submarine cables and offshore renewable energy installations - proximity study, Tech Rep; 2012.
[18] Gonzalez JS, Payan MB, Santos JMR, Gonzalez-Longatt F. A review and recent developments in the optimal wind-turbine micro-siting problem. Renew Sustain Energy Rev 2014;30(0):133–44.
[19] BVG Associates, Value breakdown for the offshore wind sector, Tech rep; 2010.
[20] Wind Energy, Wind Energy - The Facts Part I Technology, Tech Rep; 2009.
[21] Herbert-Acero JF, Probst O, Rethore P-E, Larsen GC, Castillo-Villar KK. A review of methodological approaches for the design and optimization of wind farms. Energies 2014;7(11):6930–7016.
[22] Tesauro A, Rethore P-E, Larsen G. State of the art of wind farm optimization. Eur Wind Energy Assoc 2012.
[23] Elkinton CN, Manwell JF, McGowan JG. Algorithms for offshore wind farm

layout optimization. Wind Eng 2008;32:67–84.
[24] Rodrigues S, Bauer P, Pierik J. Modular approach for the optimal wind turbine micro siting problem through cma-es algorithm. In: Proceeding of the fifteenth annual conference companion on Genetic and evolutionary computation conference companion, GECCO '13 Companion. New York, NY, USA: ACM; 2013. p. 1561–68.
[25] Rodrigues S, Bauer P, Pierik J. A clustering approach for the wind turbine micro siting problem through genetic algorithm. In: 39th annual conference IECON, 2013.
[26] Gonzalez JS, Payan MBurgos, Santos JRiquelme. A new and efficient method for optimal design of large offshore wind power plants. IEEE Trans Power Syst 2013;28(3):3075–84.
[27] Gonzalez-Rodriguez AG, Burgos-Payan M, Riquelme-Santos J, Serrano-Gonzalez J. Reducing computational effort in the calculation of annual energy produced in wind farms. Renew Sustain Energy Rev 2015;43(0):656–65.
[28] Deb K. Multi-Objective Optimization using Evolutionary Algorithms. 1st edition.. Wiltshire, England: John Wiley & Sons, Ltd,; 2001.
[29] Salcedo-Sanz S, Saavedra-Moreno B, Paniagua-Tineo A, Prieto L, Portilla-Figueras A. A review of recent evolutionary computation-based techniques in wind turbines layout optimization problems. Central Eur J Comput Sci 2011;1 (1):101–7.
[30] Khan S, Rehman S. Computational intelligence techniques for placement of wind turbines: a brief plan of research in saudi arabian perspective. In: energy conference and exhibition (EnergyCon), 2010 IEEE international; 2010. p. 519–23.
[31] Khan SA, Rehman S. Iterative non-deterministic algorithms in on-shore wind farm design: a brief survey. Renew Sustain Energy Rev 2013;19(0):370–84.
[32] Lumbreras S, Ramos A. Offshore wind farm electrical design: a review. Wind Energy 2013;16(3):459–73.
[33] Mosetti G, Poloni C, Diviacco B. Optimization of wind turbine positioning in large windfarms by means of a genetic algorithm. J Wind Eng Ind Aerodyn 1994;51(1):105–16.
[34] Wang F, Liu D, Zeng L. Study on computational grids in placement of wind turbines using genetic algorithm. In: Proceedings of the world non-grid-connected wind power and energy conference. WNWEC 2009; 2009. p. 1–4.
[35] Rodrigues S, Bauer P, Bosman P. A novel population-based multi-objective cma-es and the impact of different constraint handling techniques. In: Proceedings of the sixteenth annual conference companion on Genetic and evolutionary computation conference, GECCO14, ACM, New York, NY, USA; 2014.
[36] Pardalos P, Rebennack S, Pereira M, Iliadis N, Pappu V. Handbook of Wind Power Systems. no. IX, Springer; 2013.
[37] Luong NH, La Poutré H, Bosman PA. Multi-objective gene-pool optimal mixing evolutionary algorithms. In: Proceedings of the 2014 conference on genetic and evolutionary computation; 2014. p. 357–64.
[38] Rethore P-E, Fuglsang P, Larsen G, Buhl T, Larsen T, Madsen H. Topfarm: Multi-fidelity optimization of offshore wind farm. In: ISOPE conference, Hawaii, USA; 2011.
[39] RWE, Galloper wind farm project environmental statement - chapter 5 project details, Tech Rep; 2011.
[40] RWE, Gwynt y mor offshore wind farm - environmental statement (non technical summary). Tech Rep; 2005.
[41] Kusiak A, Song Z. Design of wind farm layout for maximum wind energy capture. Renew Energy 2010;35:685–94.
[42] Zitzler E, Thiele L. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. IEEE Trans Evol Comput 1999;3 (4):257–71.
[43] Zhang PY. Topics in wind farm layout optimization: analytical wake models, noise propagation, and energy production [Master's thesis]. Toronto, Canada: University of Toronto; 2013.
[44] Kwong WY, Zhang PY, Romero D, Moran J, Morgenroth M, Amon C. Wind farm layout optimization considering energy generation and noise propagation. In: Proceedings of the ASME 2012 international design engineering technical conferences & computers and information in engineering conference, Chicago, USA; 2012.
[45] Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective genetic algorithm: Nsga-ii. IEEE Trans Evol Comput 2002;6(2):182–97. http://dx.doi.org/10.1109/4235.996017.
[46] Veeramachaneni K, Wagner M, O'Reilly U-M, Neumann F. Optimizing energy output and layout costs for large wind farms using particle swarm optimization. In: Proceedings of the IEEE congress on evolutionary computation; 2012. p. 1–7.
[47] Li X. A nondominated sorting particle swarm optimizer for multiobjective optimization. In: lecture notes in computer science, Proceedings of genetic and evolutionary computation GECCO 2003, Vol. 2723, Part I; 2003. p. 37–48.
[48] Tran R, Wu J, Denison C, Ackling T, Wagner M, Neumann F. Fast and effective multi-objective optimisation of wind turbine placement. In: Proceedings of the 15th annual conference on genetic and evolutionary computation; 2013. p. 1381–8.
[49] Zitzler E, Laumanns M, Thiele L. Spea2: improving the strength pareto evolutionary algorithm. Tech Rep; 2001.
[50] Zitzler E, Kunzli S. Indicator-based selection in multiobjective search. In: parallel problem solving from nature - PPSN VIII, Vol. 3242 of lecture notes in computer science, Springer Berlin Heidelberg; 2004, pp. 832–42.
[51] Sisbot S, Turgut O, Tunc M, Camdali U. Optimal positioning of wind turbines on gokceada using multi-objective genetic algorithm. Wind Energy 2010;13 (4):297–306.
[52] Andersen SJ, Sorensen JN, Ivanell S, Mikkelsen RF. Comparison of Engineering Wake Models with CFD Simulations. J Phys Conf Ser 2014;524:012161.
[53] Barthelmie RJ, Larsen GC, Frandsen ST, Folkerts L, Rados K, Pryor SC, et al. Comparison of wake model simulations with offshore wind turbine wake profiles measured by Sodar. J Atmos Ocean Technol 2006;23(7):888–901.
[54] Sanderse B. Aerodynamics of wind turbine wakes - literature review. Tech Rep, Energy Research Centre of the Netherlands (ECN); 2009.
[55] Crespo A, Hernandez J, Frandsen S. Survey of modelling methods for wind turbine wakes and wind farms. Wind Energy 1999;2(1):1–24.
[56] Vermeer L, Sorensen J, Crespo A. Wind turbine wake aerodynamics. Prog Aeros Sci 2003;39:467–510.
[57] Jensen N. A note on wind generator interaction. Tech Rep, Riso-M-2411, Riso National Laboratory; 1983.
[58] Katic I, Hojstrup J, Jensen N. A simple model for cluster efficiency. In: EWEC'86. Proceedings. Vol. 1, 1986, pp. 407–10.
[59] Ainslie FF. Calculating the flowfield in the wake of wind turbines. J Wind Eng Ind Aerodyn 1988;27(1):213–24.
[60] Frandsen S, Barthelmie R, Pryor S, Rathmann O, Larsen S, Hojstrup J, et al. Analytical modelling of wind speed deficit in large offshore wind farms. Wind Energy 2006;9(12):39–53.
[61] Brower MC, Robinson NM. The openwind deep-array wake model - development and validation. Tech Rep; 2012.
[62] LGC. A simple stationary semi-analytical wake model. Tech Rep Riso-R-1713 (EN), Riso National Laboratory, Roskilde; 2009.
[63] Annoni J, Seiler P, Johnson K, Fleming P, Gebraad P. Evaluating wake models for wind farm control. In: Proceedings of the American control conference; 2014. p. 2517–23.
[64] Larsen GC, Madsen HA, Bingl F, Mann J, Ott S, Sorensen JN, Okulov V, Troldborg N, Nielsen M, Thomsen, Larsen TJ, Mikkelsen R. Dynamic wake meandering modeling. Tech. Rep. Riso-R-1607(EN), Riso National Laboratory, Roskilde; 2007.
[65] Mikkelsen R. Actuator disc methods applied to wind turbines, pH.D. thesis, Technical University of Denmark; 2003.
[66] Torres P, van Wingerden J-W, Verhaegen M. Modeling of the flow in wind farms for total power optimization. In: International Conference on Control and Automation (ICCA); 2011, p. 963–68.
[67] Ott S, Berg J, Nielsen M. Linearised cfd models for wakes. Tech Rep, Riso National Laboratory; 2011.
[68] Wagner M, Veeramachaneni K, Neumann F, O'Reilly U-M. Optimizing the layout of 1000 wind turbines. In: European wind energy association annual event, european wind energy association, Brussels, Belgium; 2011. p. 1–10.
[69] Kumar P. A framework for multi-objective optimization and multi-criteria decision making for design of electrical drives [Ph.D. thesis].Delft, The Netherlands: Delft University of Technology; 2008.
[70] Zhou Y, Kumar P, Bauer P, Ferreira J. Optimization of a hybrid wind park through a design approach - progressive design methodology. In: Proceedings of the IEEE 6th international conference on power electronics and motion control, 2009. IPEMC '09; 2009. p 1092–98.
[71] Eroglu Y, Seckiner SU. Wind farm layout optimization using particle filtering approach. Renew Energy 2013;58(0):95–107.
[72] Turner S, Romero D, Zhang P, Amon C, Chan T. A new mathematical programming approach to optimize wind farm layouts. Renew Energy 2014;63 (0):674–80.
[73] Rodrigues S, Pinto RT, Soleimanzadeh M, Bosman PA, Bauer P. Wake losses optimization on offshore wind farms with moveable floating wind turbines. Energy Convers Manag 2015;89:933–41.
[74] Frandsen S. On the wind speed reduction in the center of large clusters of wind turbines. J Wind Eng Ind Aerodyn 1992;39:251–65.
[75] Elkinton CN. Offshore wind farm layout optimization [Ph.D. thesis].Amherst: University of Massachusetts; 2007.
[76] Serrano-Gonzalez J, Gonzalez-Rodriguez AG, Mora JC, Riquelme-Santos J, Burgos-Payan M. Optimization of wind farm turbines layout using an evolutive algorithm. Renew Energy 2010;35(8):1671–81.
[77] Gu H, Wang J. Irregular-shape wind farm micro-siting optimization. Energy 2013;57(0):535–44.
[78] Grady S, Hussaini M, Abdullah M. Placement of wind turbines using genetic algorithms. Renew Energy 2005;30(2):259–70.
[79] Wilson D, Awa E, Cussat-Blanc S, Veeramachaneni K, O'Reilly U-M. On learning to generate wind farm layouts. In: Proceedings of the 15th annual conference on genetic and evolutionary computation, GECCO '13. New York, NY, USA: ACM; 2013. p 767–774. http://dx.doi.org/10.1145/2463372.2463462.
[80] Emami A, Noghreh P. New approach on optimization in placement of wind turbines within wind farm by genetic algorithms. Renew Energy 2010;35 (7):1559–64.
[81] Wagner M, Day J, Neumann F. A fast and effective local search algorithm for optimizing the placement of wind turbines. Renew Energy 2013;51(0):64–70.
[82] Igel C, Hansen N, Roth S. Covariance matrix adaptation for multi-objective optimization. Evol Comput 2007;15(1):1–28.
[83] Harada K, Sakuma J, Ono I, Kobayashi S. Constraint-handling method for multi-objective function optimization: Pareto descent repair operator. In: Evolutionary multi-criterion optimization, Vol. 4403; 2007.
[84] Wan C, Wang J, Yang G, Zhang X. Optimal micro-siting of wind farms by particle swarm optimization. In: Tan Y, Shi Y, Tan K, editors, Advances in swarm intelligence, Vol. 6145 of lecture notes in computer science, Berlin

Heidelberg: Springer ; 2010. p. 198–205.
[85] Runarsson TP, Yao X. Constrained evolutionary optimization - the penalty function approach; 2002.
[86] Coello CAC. A survey of constraint handling techniques used with evolutionary algorithms, TechRep, Laboratorio Nacional de Informatica Avanzada; 1999.
[87] Chowdhury S, Zhang J, Messac A, Castillo L. Optimizing the arrangement and the selection of turbines for wind farms subject to varying wind conditions. Renew Energy 2013;52(0):273–82.
[88] Chowdhury S, Zhang J, Messac A, Castillo L. Unrestricted wind farm layout optimization (UWFLO): investigating key factors influencing the maximum power generation. Renew Energy 2012;38(1):16–30.
[89] Eroglu Y, Seckiner SU. Design of wind farm layout using ant colony algorithm. Renew Energy 2012;44(0):53–62.
[90] Saavedra-Moreno B, Salcedo-Sanz S, Paniagua-Tineo A, Prieto L, Portilla-Figueras A. Seeding evolutionary algorithms with heuristics for optimal wind turbines positioning in wind farms. Renew Energy 2011;36(11):2838–44.
[91] Chen Y, Li H, Jin K, Song Q. Wind farm layout optimization using genetic algorithm with different hub height wind turbines. Energy Convers Manag 2013;70(0):56–65.
[92] Sadowski K, Thierens D, Bosman PA. Combining model-based eas for mixed-integer problems. In: Parallel Problem Solving from Nature – PPSN XIII, Vol. 8672, Springer International Publishing; 2014. p 342–51.
[93] Zhang P, Romero D, Beck J, Amon C. Solving wind farm layout optimization with mixed integer programmig and constraint programmig. In: Gomes C, Sellmann M, editors, integration of AI and techniques in constraint programming for combinatorial optimization problems, Vol. 7874 of lecture notes in computer science. Berlin Heidelberg:Springer; 2013, pp. 284–99.
[94] Archer R, Nates G, Donovan S, Waterer H. Wind turbine interference in a wind farm layout optimization mixed integer linear programming model. Wind Eng 2011;35:165–75.
[95] Fagerfjall P. Optimizing wind farm layout - more bang for the buck using mixed integer linear programming [Master's thesis].Gothenburg: Gothenburg University; 2010.
[96] Ozturk UA, Norman BA. Heuristic methods for wind energy conversion system positioning. Electr Power Syst Res 2004;70(3):179–85.
[97] Pelikan M, Sastry K, Goldberg DE. Multiobjective hboa, clustering, and scalability. In: ACM, editor, Procedings of the genetic and evolutionary computational conference (GECCO); 2005. p 663–70.
[98] Sastry K, Pelikan M, Goldberg DE. Decomposable problems, niching, and scalability of multiobjective estimation of distribution algorithms. Tech Rep 2005004, University of Illinois; 2005.
[99] Bosman P. The anticipated mean shift and cluster registration in mixture-based edas for multi-objective optimization. In: Proceedings of the genetic and evolutionary computation conference - GECCO-2010. New York: ACM Press; 2010. p. 351–58.
[100] Thierens D, Bosman PA. Optimal mixing evolutionary algorithms. In: Proceedings of the 13th annual conference on genetic and evolutionary computation; 2011. p. 617–24.
[101] Bosman P, Alderliesten T. Incremental gaussian model-building in multi-objective edas with an application to deformable image registration. In: Proceedings of the genetic and evolutionary computation conference - GECCO-2012. New York: ACM Press; 2012. p. 241–48.
[102] Bosman PA, Thierens D. Linkage neighbors, optimal mixing and forced improvements in genetic algorithms. In: Proceedings of the 14th annual conference on genetic and evolutionary computation, GECCO '12. New York, NY, USA: ACM; 2012. p. 585–92.
[103] Bosman PA, Thierens D. More concise and robust linkage learning by filtering and combining linkage hierarchies. In: Proceedings of the 15th annual conference on genetic and evolutionary computation, GECCO '13. New York, NY, USA: ACM; 2013. p. 359–66.
[104] Thierens D, Bosman P. Predetermined versus learned linkage models. In: Proceedings of the 14th annual conference on genetic and evolutionary computation, GECCO '12. New York, NY, USA: ACM; 2012. p. 289–96.
[105] Thierens D, Bosman P. Evolvability analysis of the linkage tree genetic algorithm. In: Parallel problem solving from nature - PPSN XII, Vol. 7491 of lecture notes in computer science; 2012. p. 286–95.
[106] Gronau I, Moran S. Optimal implementations of UPGMA and other common clustering algorithms. Inf Proces Lett 2007;104(6):205–10.
[107] Bosman P, Thierens D. On measures to build linkage trees in LTGA. In: parallel problem solving from nature - PPSN XII, Vol. 7491; 2012. p. 276–85.
[108] Luong N, Bosman P. Elitist archiving for multi-objective evolutionary algorithms: to adapt or not to adapt. In: parallel problem solving from nature (PPSN XII), Springer-Verlag, Berlin; 2012. p. 72–81.
[109] Knowles J, Corne D. Properties of an adaptive archiving algorithm for storing nondominated vectors. IEEE Trans Evol Comput 2003;7(2):100–16.
[110] Zitzler E, Deb K, Thiele L. Comparison of multiobjective evolutionary algorithms: empirical results. Evol Comput 2000;8(2):173–95.
[111] Bosman P, Thierens D. The balance between proximity and diversity in multiobjective evolutionary algorithms. IEEE Trans Evol Comput 2003;7(2):174–88.
[112] Vestas Wind Systems A/S, Offshore v164–8.0 mw v112–3.3 mw, Tech. rep. (2013). URL ⟨http://www.nozebra.ipapercms.dk/Vestas/Communication/ProductbrochureOffshoreProductBrochure/OffshoreProductBrochure/⟩.
[113] Berge E, Byrkjedal O, Ydersbond Y, Kindler D. Modelling of offshore wind resources. Comparison of a mesoscale model and measurements from FINO 1 and North Sea oil rigs. In: Proceedings of the European wind energy conference and exhibition (EWEC); 2009.
[114] Chang H-C, Wang L-C. A simple proof of thue's theorem on circle packing. Tech Rep; 2010. URL arXiv:1009.4322.pdf.
[115] Bauer J, Lysgaard J. The offshore wind farm array cable layout problem: a planar open vehicle routing problem. J Oper Res Soc 2014;66:360–8.
[116] Rodrigues S, Bosman P, Bauer P. Collection network cable routing and wake losses optimization in offshore wind farms. In: Proceedings of the XIII symposium of specialists in electric operational and expansion planning, Brazil; 2014.
[117] Gonzalez-Longatt F, Wall P, Regulski P, Terzija V. Optimal electric network design for a large offshore wind farm based on a modified genetic algorithm approach. IEEE Syst J 2012;6(1):164–72.
[118] Berzan C, Veeramachaneni K, McDermott J, O'Reilly U-M. Algorithms for cable network design on large-scale wind farms. Tech Rep, Tufts University.
[119] Richardson JT, Palmer MR, Liepins GE, Hilliard M. Some guidelines for genetic algorithms with penalty functions. In: Proceedings of the third international conference on genetic algorithms, San Francisco, CA, USA; 1989. p. 191–97.
[120] Fieldsend J, Everson R, Singh S. Using unconstrained elite archives for multiobjective optimization 7, 3; 2003. p. 305–23.
[121] Zitzler E, Thiele L. Multiobjective optimization using evolutionary algorithms - a comparative case study. In: Parallel Problem Solving from Nature (PPSN-V), Vol. 1498, Springer; 1998. p. 292–01.
[122] Hifi M, Paschos VT, Zissimopoulos V. A simulated annealing approach for the circular cutting problem. Eur J Oper Res 2004;159(2):430–48.
[123] Castillo I, Kampas FJ, Pinter JD. Solving circle packing problems by global optimization: numerical results and industrial applications. Eur J Oper Res 2008;191(3):786–802.
[124] Thomsen KE. Chapter three - project planning. In: Thomsen KE, editor, Offshore Wind, Elsevier, Boston; 2012. p. 27–49.