The 6th International Conference on Renewable Power Generation (RPG)

19–20 October 2017

Heuristic algorithm for the problem of vessel routing optimisation for offshore wind farms

Rafael Dawid, David McMillan, Matthew Revie

University of Strathclyde, Electronic & Electrical Engineering Department, Glasgow, UK
E-mail: rafael.dawid@strath.ac.uk

Published in *The Journal of Engineering*; Received on 11th October 2017; Accepted on 2nd November 2017

Abstract: A new heuristic method is proposed for the problem of vessel routing optimisation for offshore wind farms. Turbines requiring a maintenance action are arranged into clusters, each associated with a vessel and a value for repairing the turbines. The clusters with the highest value are used to produce offspring, which is selected from the remaining high-value clusters, provided the constraints are met. The process is repeated until vessels available or turbines requiring maintenance are exhausted. To test the performance of the proposed approach, the same problem was formulated as integer linear programming problem and benchmarked against the IBM CPLEX commercial solver. The proposed method was shown to consistently produce close-to-optimal policies within seconds, even in problems with 15–20 turbines requiring a maintenance action. Although the proposed method only outperformed the commercial solver in one instance, its benefits include short and consistent computational times and the fact that the users can easily understand, implement and adapt the algorithm to suit their needs.

1 Introduction

The offshore wind industry has grown significantly in the last 5 years, with 6 GW of new capacity installed in Europe between 2013 and 2015 [1]. However, the cost of energy remains higher than other renewables such as onshore wind and solar [2]. O&M costs constitute up to a third of the total cost of energy from offshore wind turbines [3]; one of the ways to reduce this proportion, and therefore, the overall cost of energy is to optimise the use of vessels and resources when carrying out maintenance activities.

The largest operational offshore wind farm in the world, London Array, is made up of 175 wind turbines; managing the Operations & Maintenance (O&M) actions on this scale is a complex problem. Interviews with wind farms operators have revealed that this problem is solved manually by experienced maintenance planners. However, given the complexity of the problem and the number of constraints which need to be considered, human ability of consistently producing close-to-optimal policies when faced with billions of possible combinations of assignment of vessels to turbines and order in which they are visited has to be questioned. Operators would benefit from effective algorithms, which can be transformed into commercial decision support tools.

In this paper, the cluster matching algorithm (CMA) is proposed; a heuristic method which can contribute to the development of decision support tools for offshore wind, which will aid O&M cost reductions.

1.1 Related work

A comprehensive overview of different heuristic approaches in the context of vehicle routing problems (VRPs) was provided by Laporte [4], who summarised the 50 years of research on vehicle routing. One of his conclusions was that 'several of the most successful metaheuristics are over engineered' and that researchers should attempt to produce simpler and more flexible algorithms which can handle a wide range of constraints. It is the lack of flexibility of the available metaheuristic methods, which led the authors of this paper to develop a heuristic method specifically for the problem at hand.

Many VRPs can be formulated as integer linear programming problems, which can be solved effectively with commercial solvers such as IBM CPLEX Optimizer, which use branch-and-bound and branch-and-cut algorithms. Branch-and-bound methods [5] are widely used for solving a variety of VRPs; they intelligently search the space of all possible solutions. Branches that contain no solutions which are better than the current best are discarded without enumeration. Branch-and-bound methods have been used in the context of VRPs and O&M optimisation in [6, 7]. Branch and cut is an extension of the branch-and-bound method, in which cutting planes are used to tighten the linear programming relaxations [8]. Use of branch and cut in the VRP domain includes [9–11].

Within the offshore wind domain, the problem of vessel routing for O&M actions has been tackled by a number of researchers [12–15]. Rolling horizon heuristic proposed by Raknes [12] takes a long time (up to 2 h) to compute for large problems and does not guarantee a feasible result. Stahlane [13] used path and arc flow heuristics, while Dai [14] and Irawan [15] used commercial solvers (Xpress Optimizer and IBM CPLEX Optimizer, respectively), capable of providing the optimal solution in a short computational time. Naturally, the downside of using commercial solvers is that, in most cases, there is a fee associated with using them and the user has limited or no ability to influence the solver.

We proposed a model for optimising the vessel routing for offshore wind farms described in Dawid *et al.* [16]. However, it was only capable of solving problems with up to ten failed wind turbines, as exhaustive search was used to evaluate all possible combinations of assignment of vessels to turbines. In this paper, a heuristic method is proposed for solving the master problem (as defined by Irawan [15] and Dawid [16]) only. IBM CPLEX Optimizer, a commercial solver, was chosen as the benchmark to evaluate the proposed methodology against.

2 Methodology

The master problem involves generating multiple clusters of turbines, assigning a vessel to each cluster and matching individual

clusters into a policy. Given a large set of clusters of turbines, the objective is to choose a set of clusters which maximises the total rewards (or minimises costs), while satisfying the constraints. Each cluster is defined by four parameters:

(i) Turbines visited
(ii) Vessel used
(iii) Number of technicians required to carry out repairs
(iv) Value of repairing turbines visited (rewards – costs)

Examples of the clusters, which are an input to the heuristic model, are shown in Section 3. Parameters 3 and 4 are calculated in the inner section of the general vessel routing problem, and are inputs to the outer problem. More information on how these parameters are calculated is available in [16].

The number of clusters depends on the size of the problem (number of turbines and vessels) and on the maximum number of turbines in a cluster (η). It was shown that setting η to 4 and enumerating all possible combinations of turbines, yields, in most cases, answers within 97% of the optimal solution [15]. Limiting η decreases the computational time significantly, by reducing the number of possible combinations when matching individual clusters into a policy. This limit reflects the fact that the number of turbines a single vessel can maintain on one day will be limited by its capacity and the time available. Assuming a problem with a number of turbines requiring a maintenance action T, a set of available vessels V, the number of clusters can be calculated from:

$$\text{Number of clusters} = V * \sum_{n=1}^{n=N} \binom{T}{n} \tag{1}$$

Multiplying the sum of binomial coefficients by the number of vessels is necessary if the vessels have different properties; as some clusters may only be successfully serviced by the more capable vessels due to time constraints.

In the next step, all clusters are sorted by value in a descending order. Alternatively, value per technician in a cluster can be used to sort all clusters in scenarios with a shortage of technicians. The first policy is generated by choosing the highest value cluster, and matching it with the next highest value cluster, which:

(i) Does not visit the same turbines
(ii) Does not use the same vessel
(iii) Does not exceed the total number of technicians available on the day

These constraints ensure that no turbine is visited by more than one vessel (although it does not guarantee that all turbines will be visited). The above is repeated until vessels available or turbines requiring maintenance are exhausted, resulting in the creation of a single policy. Note that this procedure is detailed in the Appendix. The user can specify the number of policies which are to be generated, depending on the time available for the decision to be made, by defining the amount of 'mothers' in each tier (with 'mothers' being individual clusters which are selected to be matched with multiple 'children' clusters from the following tiers). For example, in a case with three vessels, defining:

$$\text{Tier 1 limit} = a$$
$$\text{Tier 2 limit} = b$$
$$\text{Tier 3 limit} = c$$

would result in the algorithm producing an amount of policies equal to $a*b*c$, which consist of up to three clusters (as there are three tiers), each containing up to η turbines. Setting a equal to 100 as an example, means that 100 of the highest value clusters will be used to produce offspring (tier 1). Setting $b = 10$, as an example, ensures that each 'mother' cluster selected in tier 1 will be matched with ten highest value 'children' clusters, which are compatible with its 'mother'. Clusters available for selection in tiers below the top one will vary depending on the clusters chosen in the tiers above it. The value of each policy is calculated by adding the values of individual clusters, with the highest value policy selected and displayed to the user as suggested vessel routing plan. The entire solving procedure is detailed in the Appendix.

This methodology ensures that the highest value clusters are used to produce offspring. Many policies can be generated in a relatively short computational time as the next section demonstrates and close-to-optimal policies are generated in fractions of a second.

3 Computational studies

To assess the performance of the algorithm proposed in this paper, its outputs were compared to the results obtained from IBM CPLEX Optimizer software. In total, 30 cluster matching problems were solved, 10 for each of the cases are outlined in Table 1. The simulations were run on a computer with an i7 3.4 GHz processor and 8 GB RAM using MATLAB and CPLEX for MATLAB software.

Tests were run based on a wind farm with 100 turbines arranged on a 10-by-10 grid. In all, 10, 15 and 20 out of 100 turbines required a maintenance action in Cases A, B and C, respectively. The complexity of the problem increases significantly as the number of failed turbines is doubled between Cases A and C; the number of clusters to choose from is over 26 times larger (1155 for Case A and 30,975 for Case 3 from (1), using η equal to 4).

The number of policies generated by CMA in each of the cases (Table 1) was set at a relatively low level to ensure short computational time and enable meaningful comparison with CPLEX optimiser. The only difference between the instances in each of the cases was the turbine locations and type of repairs. In some cases, it will not be possible to repair all turbines with a fault, due to the constraint on the number of technicians. As the number of technicians available to the operator in Case B was set at a lower level (in terms of technicians per failed turbine) compared to Cases A and C, the former is expected to feature a larger proportion of heavily constrained problems.

Table 2 illustrates a sample of the key input to the model – the clusters of turbines and the parameters associated with each cluster.

Table 1 Comparison of inputs to Cases A–C

	Case A	Case B	Case C
turbines failed	10	15	20
technicians available	25	32	45
vessels available	3	4	5
number of policies generated by CMA	15,000	36,000	51,840
clusters to choose from (1)	1155	7760	30,975

Table 2 Sample of 4 out of 1155 clusters which are input to the heuristic model in Case A, instance #1

Cluster ID	Turbines visited	Technicians required	Vessel no. used	Value
1	T1 T2 T3 T4	9	CTV[a] 1	£32,207
2	T1 T2 T3 T5	9	CTV 1	£32,239
3	T1 T2 T3 T6	10	CTV 1	£34,090
4	T1 T2 T3 T7	10	CTV 1	£33,977

[a]CTV – crew transfer vessel.

doi: 10.1049/joe.2017.0511

Table 3 Results for Case A: 10 turbines. The number in bracket is the difference between CMA and CPLEX results

Instance	CMA value	CPLEX value (difference)	CMA CPU time, s	CPLEX CPU time, s
#1	£88,183	£88,183 (0%)	2	0.6
#2	£86,240	£86,244 (0.004%)	2	0.5
#3	£88,063	£88,063 (0%)	2	4.7
#4	£88,204	£88,204 (0%)	2	0.5
#5	£90,198	£90,198 (0%)	2	0.5
#6	£88,061	£88,061 (0%)	2	0.5
#7	£92,134	£92,134 (0%)	2	0.4
#8	£90,135	£90,135 (0%)	2	0.5
#9	£94,144	£94,144 (0%)	2	0.5
#10	£89,993	£89,993 (0%)	2	0.5

4 Results

Table 3 compares the performance of the CMA to IBM CPLEX Optimizer in test case A. The former matched the policies of the latter in nine out of ten instances, with a minute difference in the other instance (£4). The computational time was marginally shorter for CPLEX solver, with the exception of instance #3.

The results of simulations for Case B are shown in Table 4. Increased complexity of the problem led to longer computational times for the CPLEX solver. The values achieved by the CMA are very close to values produced by CPLEX, the difference between the two was never higher than £173 or 0.17%. The CMA managed to match CPLEX value in four out of ten instances, outperforming the commercial optimiser in instance #4, as the optimal result was achieved in a shorter computational time. It is also worth noting that the CPLEX computational time in instance #1 was over ten times longer than CMA.

Table 5 shows the results of Case 3 tests. The average computational time for both methods has increased. While in Cases A and B, the number of turbines repaired using CMA- and CPLEX-generated policies were the same, in Case C the CPLEX managed to repair an additional turbine in instances #2, #5 and #9, which is reflected in the higher difference in value.

Table 4 Results for Case A: 15 turbines

Instance	CMA value	CPLEX value (difference)	CMA CPU time, s	CPLEX CPU time, s
#1	£119,083	£119,104 (0.02%)	32.1	438.2
#2	£116,107	£116,107 (0%)	34.5	5.3
#3	£126,907	£126,951 (0.04%)	33.7	2.2
#4	£116,034	£116,034 (0%)	35.2	53.5
#5	£116,932	£117.133 (0.17%)	32.6	5.1
#6	£114,946	£115,134 (0.16%)	34.1	27.8
#7	£114,093	£114,093 (0%)	30.3	1.5
#8	£124,722	£124, 945 (0.17%)	33.4	1.8
#9	£114,871	£115,036 (0.14%)	35	27.4
#10	£113,862	£113,862 (0%)	30.4	3.3

Table 5 Results for Case C: 20 turbines. The second result for instance #5 (in brackets) indicates that improved solution was found by using a different sorting technique

Instance	CMA value	CPLEX value (difference)	CMA CPU time, s	CPLEX CPU time, s
#1	£146,952	£147,112 (0.11%)	47	13.6
#2	£143,126	£150,897 (5.43%)	40.4	7.6
#3	£155,034	£155, 269 (0.14%)	48.3	8.5
#4	£150,902	£151,010 (0.07%)	44.2	8.3
#5	£145,214 (£152,883)	£153,173 (5.48%, 0.19%)	43.4 (47)	7.9
#6	£154,972	£155,192 (0.14%)	48.2	8.7
#7	£145,027	£145, 200 (0.12%)	51.5	8.9
#8	£136,157	£140,995 (3.52%)	38.7	16.3
#9	£139,309	£147,147 (5.63%)	44.1	19.5
#10	£151,163,	£151,308 (0.1%)	47.2	8.2

In the CMA method, high-value policies are encouraged by sorting all clusters by their value and using the best of them to generate offspring. However, in heavily constrained scenarios, an alternative approach can be adopted; the user can choose to sort all clusters by the value they deliver divided by the number of technicians required to repair all turbines in a cluster. This encourages policies which achieve high value while utilising few technicians. Given that the number of technicians is one of the key constraints, this can lead to a better performance of the CMA in heavily constrained scenarios.

Applying the aforementioned approach to instances in which CPLEX outperformed the CMA by repairing an additional turbine (Case C, instances #2, #5 and #9) led to a significant improvement in instance #5: an additional turbine was repaired, as shown in Table 5.

In some of the scenarios across Cases A–C, neither CPLEX nor CMA managed to find a policy which resulted in all turbines being repaired. This was caused by a higher proportion of turbines requiring complex repairs and more technicians to complete, taking the number of technicians required beyond the amount available to the operators on the day. These problems can be considered as being heavily constrained. It is worth noting that across all cases, there were six instances, in which the CPLEX computational time exceeded 15 s, five of them were heavily constrained (cases with a shortage of resources and time leading to failure in repairing all turbines). These constituted only 40% of all test instances (12/30). It can be concluded that CPLEX takes longer to compute heavily constrained problems. This does not affect the CMA method; its consistency can be seen as an advantage for applications where short computational time is crucial. A comparison of the computational time of each method versus the value of the best policy generated is shown in Fig. 1.

From Fig. 1, it can be seen that the CMA produces a very good-policy (within 0.34% of optimal) in a fraction of a second. The rate at which the solution is improved is high initially, but reduces with computational time. It is likely that given a longer computational time, the CMA result would match the value achieved by CPLEX.

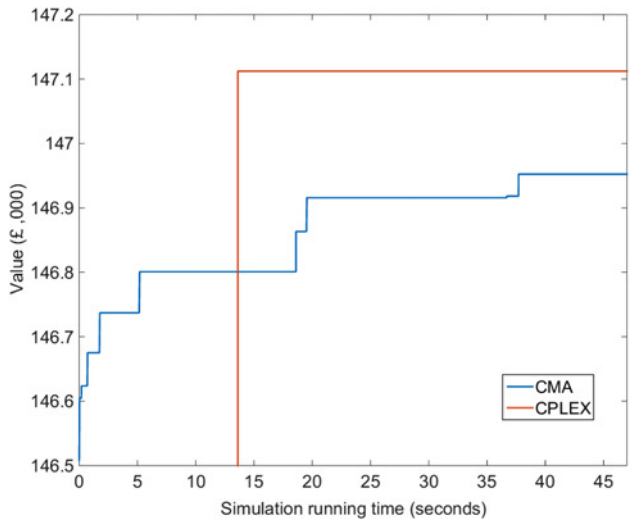

Fig. 1 *Comparison of computational time and performance of CMA and CPLEX for Case C instance #1*

```
Sort all clusters according to value, in descending order
Define tier limits a and b (c=1 as 3 vessels available)
i=1
For n=1:a
  Select nth best value cluster
  If all turbines have been assigned a vessel
    Store the policy properties based on cluster n
    PolicyValue(i)=Value(Cluster n)
    i=i+1
  Else
    Remove clusters using the same vessel or visiting the
    same turbines as cluster n
    For m=1:b
      Select mth best value cluster-vessel pairing
      If all turbines have been assigned a vessel
        Store policy properties based on clusters n and m
        PolicyValue(i)=Value(Cluster n)+Value(Cluster m)
        i=i+1
      Else
        Remove clusters which are incompatible with cluster
        m
        For p=1
          If Cluster(p) exists
            Select the highest value cluster-vessel pairing
            Store the policy properties based on clusters n, m
            and p
            PolicyValue(i)=Value(Cluster n) +
            Value(Cluster m)+Value(Cluster p)
            i=i+1
          Else
            Store the policy properties based on clusters n
            and m
            PolicyValue(i)=Value(Cluster n) +
            Value(Cluster m)
            i=i+1
          End (If)
        End (p)
      End (If)
    End (m)
  End (If)
End (n)
Select the highest PolicyValue
```

Fig. 2 *Solving procedure for a problem with three vessels available*

5 Conclusions

The proposed CMA methodology is capable of providing close-to-optimal solutions in under a minute and performs particularly well in scenarios with fewer than 15 turbines. Although the use of commercial solvers can be advantageous in complex problems with over 20 failed wind turbines, having such a high number of issues on an average offshore wind farm is very rare.

In comparison to the CPLEX Optimizer, the CMA computational times were more consistent. It was found that, unlike CPLEX, the CMA computational time was not increased when solving heavily constrained problems.

Interviews with O&M planners have revealed that practitioners value being able to understand how the algorithm calculates the answer and being able to change or influence it. The CMA is easy to understand and it allows the user to modify its parameters, such as the value used to sort all clusters. The number of iterations can also be defined by the user depending on the time available for computation. Increasing the number of iterations leads to better quality results at a cost of increased computational time. The proposed method is easy to implement in any programming language and provides quality solutions to complex problems without the need for purchasing a commercial solver.

This heuristic method could be improved further by using a local search or a large neighbourhood search, to enhance the quality of results produced in complex scenarios.

6 Acknowledgments

This work was funded by Engineering Physical Sciences Research Council via the University of Strathclyde's Wind Energy Systems Centre for Doctoral Training, grant number EP/G037728/1.

7 References

[1] Corbetta G., Mbistroba A., Ho A.: 'Wind in power 2015 european statistics', 2016. Available at: https://windeurope.org/wp-content/uploads/files/about-wind/statistics/EWEA-Annual-Statistics-2015.pdf, accessed 12 June 2017.

[2] World Energy Council: 'World energy perspective – cost of energy technologies', 2013, p.48. Available at: http://www.worldenergy.org/wp-content/uploads/2013/09/WEC_J1143_CostofTECHNOLOGIES_021013_WEB_Final.pdf, accessed 12 June 2017.

[3] Scheu M., Matha D., Hofmann M., *ET AL.*: 'Maintenance strategies for large offshore wind farms', *Energy Proc.*, 2012, **24**, pp. 281–288

[4] Laporte G.: 'Fifty years of vehicle routing', *J. Transp. Sci.*, 2009, **43**, pp. 408–416

[5] Wood E.L., Lawler D.E.: 'Branch-and-bound methods: a survey', *Oper. Res.*, 1966, **14**, (4), pp. 699–719

[6] Kovacs A., Erdos G., Monostori L., *ET AL.*: 'Scheduling the maintenance of wind farms for minimizing production loss'. 18th IFAC World Congress Proc., 2011, vol. **44**, no. 1

[7] Fischetti M., Toth P., Vigo D.: 'A branch-and-bound algorithm for the capacitated vehicle routing problem on directed graphs', *Oper. Res.*, 1994, **42**, (5), pp. 846–859

[8] Mitchell J.E.: 'Branch and cut', (James J. Cochran (Ed.)), 'Wiley encyclopedia of operations research and management science' (John Wiley & Sons, Inc., Troy, New York, 2011) doi: 10.1002/9780470400531.eorms0117

[9] Ropke S., Cordeau J.F., Laporte G.: *Models Branch-and-Cut Algorithms Pickup Deliv. Probl. with Time Windows*, 2007

[10] Kenyon A.S., Morton D.P.: 'Stochastic vehicle routing with random travel times', *Transp. Sci.*, 2017, **2003**, pp. 74–80

[11] Bianchessi N., Irnich S.: 'Branch-and-price-and-cut for the split delivery vehicle routing problem with time windows'. Technical Report, LM-2016-07, 2016

[12] Raknes N.T., Ødeskaug K., Stålhane M., *ET AL.*: 'Scheduling of maintenance tasks and routing of a joint vessel fleet for multiple offshore wind farms', *J. Mar. Sci. Eng.*, 2017, **5**, (11)

[13] Stålhane M., Hvattum L.M., Skaar V.: 'Optimization of routing and scheduling of vessels to perform maintenance at offshore wind farms', *Energy Proc.*, 2015, **80**, (1876), pp. 92–99

[14] Dai L., Stålhane M., Utne I.B.: 'Routing and scheduling of maintenance fleet for offshore wind farms', *Wind Eng.*, 2014, **39**, (1), pp. 15–30
[15] Irawan C.A., Ouelhadj D., Jones D., *ET AL.*: 'Optimisation of maintenance routing and scheduling for offshore wind farms', *Eur. J. Oper. Res.*, 2017, **256**, (1), pp. 76–89
[16] Dawid R., Mcmillan D., Revie M.: 'Development of an O&M tool for short term decision making applied to offshore wind farms'. WindEurope Summit, Online Proc., 2016, available at: https://windeurope.org/summit2016/conference/submit-an-abstract/pdf/1801328683846.pdf

8 Appendix

The solving procedure for a problem with three vessels available is shown in Fig. 2