MULTIDISCIPLINARY DESIGN OPTIMIZATION OF A MOBILE MINER USING THE OPENMDAO PLATFORM

Vidner, Olle (1);
Pettersson, Robert (2);
Persson, Johan A (1);
Ölvander, Johan (1)

1: Linköping University;
2: Epiroc Rock Drills AB

ABSTRACT

This paper proposes an optimization framework based on the OpenMDAO software library intended for engineer-to-order products and applies it to the conceptual design of a Mobile Miner. A Mobile Miner is a complex machine and a flexible alternative to Tunnel Boring Machines for small-scale tunneling and mining applications. The proposed framework is intended for use in early design and quotation stages with the objective to get fast estimates of important product characteristics, such as excavation rate and cutter lifetime. The ability to respond fast to customer requests is vital when offering customized products for specific applications and thereby to stay competitive on the global market. This is true for most engineer-to-order products and especially for mining equipment where each construction project is unique with different tunnel geometries and rock properties. The presented framework is applied to a specific use-case where the design of the miner's cutter wheel is in focus and a set of Pareto optimal designs are obtained. Furthermore, the framework extends the capabilities of OpenMDAO by including support for mixed-variable formulations and it supports an exploratory approach to design optimization.

Keywords: Optimisation, Multi- / Cross- / Trans-disciplinary processes, Large-scale engineering systems, Mobile Miners

Contact:
Vidner, Olle
Linköping University
Division of Product Realisation
Sweden
olle.vidner@liu.se

Cite this article: Vidner, O., Pettersson, R., Persson, J. A., Ölvander, J. (2021) 'Multidisciplinary Design Optimization of a Mobile Miner Using the OpenMDAO Platform', in *Proceedings of the International Conference on Engineering Design (ICED21),* Gothenburg, Sweden, 16-20 August 2021. DOI:10.1017/pds.2021.482

1 INTRODUCTION

Small scale mining equipment are important tools for excavation of resources and creating tunnels. Among solutions for this, a Mobile Miner is a flexible and highly competitive option to Tunnel Boring Machines (TBM) and traditional drill-and-blast operations (Lyly, Hartwig, and Nord, 2018). However, a Mobile Miner is a specialized engineer-to-order product that is tailored for the tunnel geometry and the ground properties at the mining site. Engineer-to-order products are characterized by the addition of engineering to the product lead time and uncertain product properties at the start of the project (Wortmann, 1983). This is true for the Mobile Miners as well, where accurate estimations of the performance and cost of the product are needed during the ordering process since detailed design is started when the contract is won. Reducing engineering time is therefore crucial for commercial success in signing and delivering Mobile Miner orders.

An important part of the preliminary design process for a Mobile Miner, is determining a number of system-level parameters that control the overall mechanical layout and the excavation process. While these parameters have a large impact on the engineering process, system behavior and product performance, they are also interrelated in non-trivial ways and finding the optimal combination of these parameters is a difficult task. Consequently, there is a need for methods that can 1) make accurate and fast predictions of the performance of the finished product; and 2) speed up the development process to reduce the engineering lead time.

A method that fulfills both criteria is multidisciplinary design optimization (MDO), as defined by Schmit, 1960 and Simpson and Martins, 2011. MDO is used to holistically explore the design space to find optimal designs given multiple disciplines and analyses (Martins and Lambe, 2013). Important aspects are how to control the data flow between the different models to enable fast optimization, which optimization algorithm to use and how to handle and present the data.

A flexible MDO framework should enable quick estimations regarding the cost and performance of a new product variant based on the customer requirements. The results should also be used as a basis for the continued engineering work. Consequently, there is a need for a flexible framework where appropriate models can be connected depending on the case at hand. The flexibility sought after here refers to being able to exchange the models in use, without having to reconfigure the rest of the formulation considerably. An example of this would be that one disciplinary model could be replaced without any modification of another model or of the solver configuration.

Solving the challenges associated with these formulations is part of the motivation behind the OpenMDAO software library (Gray et al., 2019). While OpenMDAO is gaining interest in the context of high-fidelity modelling and optimization (Jasa et al., 2020; Hendricks and Gray, 2019; Sgueglia et al., 2020), it has not been widely used for exploratory design optimization using multi-objective optimization (or meta-heuristic) techniques, such as genetic algorithms. Even though the basic technical support is in place for this kind of optimization, some components and properties are however missing.

The contributions presented in this paper are twofold. First, a number of extensions to the OpenMDAO platform are presented, to support its use in early-stage design of engineer-to-order products and other exploratory optimization problems. Second, the Mobile Miner design problem is addressed by implementing and solving it using OpenMDAO along with these extensions, to provide an example of their applicability on an actual problem as well as the utility of an optimization-driven approach for engineer-to-order products.

The remainder of the paper is organized as follows. Section 2 contains a background where previous work and key methods are presented. Section 3 describes the extensions to OpenMDAO that have been implemented to enable and improve the optimization platform. This is followed by a section that presents the optimization study of the Mobile Miner. Finally, the paper ends with a discussion and conclusion section.

2 FRAME OF REFERENCE

2.1 Mobile Miners

A Mobile Miner is a complex machine intended for *small-scale* mining and tunneling applications (Robbins, 2000, Lyly, Hartwig, and Nord, 2018). The Mobile Miner is a large piece of machinery with a cross-section of at least 2-4 meters in each direction and a length of up to 20 meters, see figure 1.

https://doi.org/10.1017/pds.2021.482 Published online by Cambridge University Press

Figure 1. An example of a mobile miner. The cutter wheel can be seen to the right in the figure. Image reproduced from Epiroc, 2021 with permission, © Epiroc.

The miner is self-propelled using wheels or crawler-tracks to move forward, and it has excavating cutters placed on a cutter wheel in the front of the machine. The cutter wheel is pressed towards the rock wall as it rotates in order to excavate the rock. Furthermore, the cutter wheel could be maneuvered both horizontally and vertically using a hydraulic actuation mechanism to control the shape of the excavated tunnel. The miner also has a system that transports the excavated rock material to the end of the machine from where it could be transported off-site using other machines such as dumpers or a conveyor system. Hence the miner is a self-contained mining system. The main focus of the design process proposed in this paper is the design of the cutter wheel, and particularly the design and placement of the cutters on the wheel. This is where the machine interacts with the rock, and hence the properties of the complete system is strongly dependent on the design and control of the cutter wheel.
The rotating motion of the cutter wheel means that the individual cutters follow a circular trajectory. This cutting motion, the kerf, is what fractures the rock and consequently also imposes loads to the cutter wheel. Multiple cutters can work along each kerf. The cutters must be placed in a physically feasible manner and so that certain damaging static and dynamic loads are avoided.

2.2 Optimization

Optimization is the mathematical process of identifying the point that yields the maximum or minimum value of a defined function. In engineering, optimization is often characterized by the presence of multiple (and contradicting) objective functions, and problems where the functions are not known as explicit mathematical equations, but more often by numerical simulation models from different engineering disciplines (Andersson, 2001).
An MDO problem includes several system and sub-system models from different disciplines which need to be interconnected and solved in a numerical process to obtain overall optimal solutions. The problem is decomposed in what is referred to as an architecture, and according to the definition of Martins and Lambe, 2013, "architectures define how to organize the disciplinary analysis models, the approximation functions (if any), and the optimization software in concert with the problem formulation so that an optimal design can be achieved". The most common architectures for MDO are the All-at-Once (AAO) approach, Individual Discipline Feasible (IDF), Multidisciplinary Feasible (MDF) and Collaborative Optimization (CO).
In Sun, X. Wang, L. Wang, et al., 2016 a TBM is optimized using an AAO MDO formulation in order to determine control parameters and excavation strategy, considering aspects such as tool wear and excavation rate. In Sun, X.Wang, Shi, et al., 2018 the authors improve the strategy further and variables such as the number of cutters and their placement are added as optimization variables, and the problem is formulated as a CO MDO problem. In this paper, we perform a similar optimization study as in Sun, X.Wang, Shi, et al., 2018 using an MDF MDO formulation, applied to a Mobile Miner instead of a TBM.
There are numerous options available for the definition and execution of MDO formulations, including both commercial and open-source software packages. Some commercial offerings include *modeFRONTIER* and *HEEDS MDO*, while some open-source alternatives include *Dakota*, *RCE* and *OpenMDAO*. One substantial benefit with the open-source alternatives is that they can be freely distributed, extended and scaled out due to their open license models. On the other hand, the commercial offerings generally include data visualization tools and integrations with third-party tools.

https://doi.org/10.1017/pds.2021.482 Published online by Cambridge University Press

2.2.1 OpenMDAO

OpenMDAO is an open-source optimization platform implemented in Python, whose development is led by NASA Glenn Research Center (Gray et al., 2019). Its main focus is an efficient handling of optimization problems where analytical gradients of the models are available. However, it also enables creation of MDO frameworks with MDF solvers and has the capability to manage the data flow between the different models and optimization algorithms. Furthermore, it can be called as a software library from other applications, meaning that it can be integrated in custom user interfaces and workflows.

2.2.2 NSGA-II

The multi-objective evolutionary algorithm Non-Dominated Sorting GA (NSGA-II) was developed by Deb, Pratap, et al., 2002 and has been used in many engineering optimization projects. Its main characteristics is the ranking procedure for the population, which ranks the designs belonging to the same Pareto rank. The Pareto optimal points are ranked according to how far they are from their neighbours in the design variable space. Consequently, the design with the best rank is the Pareto optimal design that is farthest from the other Pareto optimal designs. This ensures diversity in the genes of the population which reduces the probability of premature convergence and ensures a good spread of solutions on the Pareto front.

3 DEVELOPED EXTENSIONS TO OPENMDAO

After examining the existing implementations of genetic algorithms for OpenMDAO, the following criteria were found to be missing:

- simultaneous usage of continuous, integer, discrete and categorical variables
- ability to handle Pareto-optimization of multi-objective formulations
- inequality constraint handling without extra parameters

As part of realizing this, an enhanced NSGA-II (Deb, Pratap, et al., 2002) implementation for OpenMDAO was implemented, featuring multiple variable types and parameter-less constraint handling.
The optimization and data management tools developed as a part of this work are available in the *OpenMDAO-Bridge-MATLAB* (Vidner, 2020a), *OpenMDAO-NSGA* (Vidner, 2020b) and *Scop* (Vidner, 2020c) packages, respectively.

3.1 Discrete and categorical values

Unlike continuous and integer variables, discrete and categorical variables have the special property that their values might not be numeric. But, variational operators (such as crossover and mutation) usually assume this. To circumvent this conflict, we exploit the assumption that each one of the discrete and categorical variables can only take a finite number of enumerable values. For each variable, the possible values are thus enumerated and given an integer index, meaning that the possibly non-numeric values used within the models are represented by integers within the optimization. The optimization thus only has to handle real-coded continuous and integer values.
However, while categorical values can be represented by a series of integers, the ordering of these integers are not significant in contrast to *native* integer variables. Thus, the different types of variables need different variational operators (such as mutation and crossover) that can take the different variables' properties in consideration.

3.2 Crossover and mutation operators

Different types of variables require different crossover and mutation operators, respectively. To solve this problem, the population is split into different parts, each part using different strategies, as represented in figure 2. The different sub-populations are mated and mutated using the operators described below before being merged into a unified offspring population.
The *simulated binary crossover* (Deb and Ram Bhushan, 1995) and polynomial mutation operators are used for continuous and integer variables, truncated to integer values for integer variables. Uniform crossover and uniform mutation operators are meanwhile used for the categorical and discrete variables. All mating/mutation methods respect variable bounds and use the implementations offered by the DEAP software package (Fortin et al., 2012).

https://doi.org/10.1017/pds.2021.482 Published online by Cambridge University Press

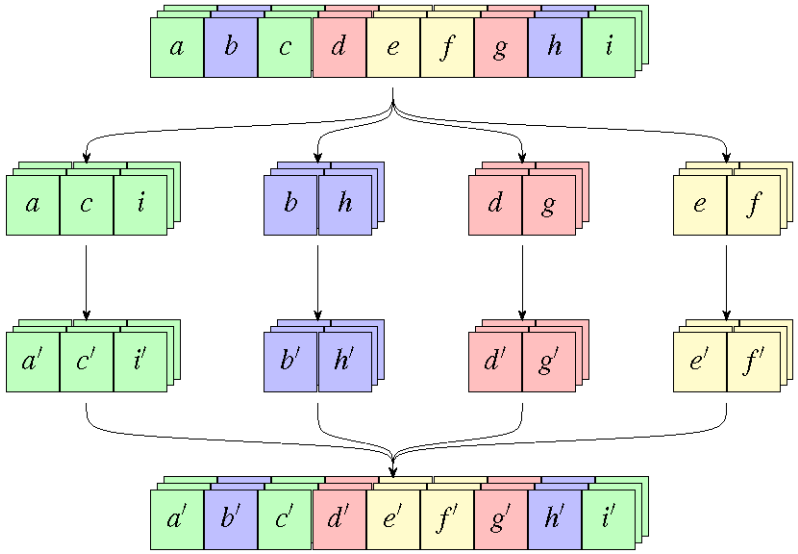

Figure 2. Schematic view of the GA variational procedure, where an example (parent) population is split into multiple sub-populations, based on the represented variables' properties. Each sub-population is mated and mutated accordingly, and thereafter merged into the complete offspring population.

The choice of operators for discrete and categorical variables have not been extensively explored, and is considered a subject for future studies. For problems with only continuous variables, the algorithm behaves just as the reference NSGA-II implementation as described by Deb, Pratap, et al., 2002.

3.3 Constraint handling

As part of implementing the reference NSGA-II algorithm, its constraint handling method (as described by Deb, Pratap, et al., 2002), *constraint-domination*, have also been implemented. One benefit of this constraint handling method is that it does not require any extra parameters, in contrast to e.g. penalizing constraint methods. This is viewed as an attractive property, as the optimization strategy is aimed towards exploration and that the number of parameters adjustments necessary is to be kept small in order to maintain engineer focus on the application and ease of use for ordinary engineers.

3.4 Data management

The NSGA-II algorithm used (and many other meta-heuristics) cannot guarantee feasibility nor optimality of each and every evaluated point in the design space. This means that infeasible, dominated or otherwise non-interesting points will be evaluated during the optimization. As the algorithm will also need to evaluate a very large number of points, the amount of (possibly uninteresting) data generated will be immense. Thus, implementing and running an optimization formulation like this must also include managing data during and after execution. While OpenMDAO offers some utilities for this, they are somewhat restricted to continuous variables and single-objective optimization, and do not directly help with common post-processing activities, such as filtering and ranking of designs. Additionally, these built-in tools use a custom, procedural data structure that does not directly support an exploratory workflow nor enable integration with other tools in the Python scientific ecosystem.

As a complement to OpenMDAO's built-in utilities for data management, another approach is proposed, using the *Dataset* and *DataArray* data structures from the *xarray* package (Hoyer and Hamman, 2017). Instead of examining only one design at a given time, here the complete dataset can be conceptualized as one big array containing all variable data gathered during the execution. Alongside the actual variable values, metadata is also contained within the dataset object, storing properties such as units, constraint bounds, objective directions, etc.. Thus, the dataset can be efficiently indexed and examined along any axis using shorthand syntax. Furthermore, generic utility functions are also provided that operate on the dataset object, for performing a number of ordinary post-processing activities such as calculating constraint violation, discarding designs that violate any constraint or discarding Pareto-dominated designs.

https://doi.org/10.1017/pds.2021.482 Published online by Cambridge University Press

The important visualization of optimization results (Simpson and Martins, 2011) also becomes available "for free" by utilizing a widely-used data structure, through the Python ecosystem.

4 OPTIMIZATION OF THE MOBILE MINER

The developed methods and tools have been applied to create an optimization framework for a Mobile Miner concept. Taking user input in the form of tunnel geometry specifications and rock properties, the framework will evaluate numerous designs to identify a set of Pareto-optimal designs for the Mobile Miner.

4.1 Overview of the framework

An overview of the framework is shown in figure 3, where the models and functions are denoted with grey squares while the yellow squares represent optimizations and the blue squares design variables and environmental variables, such as rock properties. The latter are specified by the user before the optimization and are used as constants.

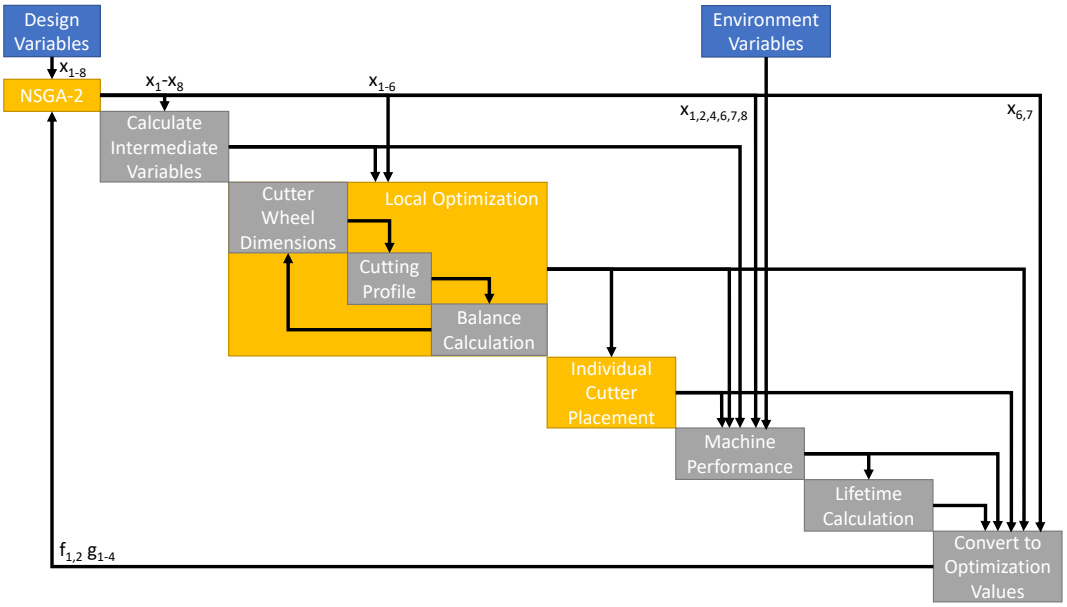

Figure 3. A simplified flowchart for the framework. Design variables, x_i, and environmental variables are marked with blue squares, optimization processes with yellow squares and models with grey squares.

First, the dimensions of the cutter wheel are calculated according to the design variables to fulfill the requirements of the customer. This is followed by placement of each individual cutter on the cutter wheel. With the cutter wheel designed, its impact on the rest of the machine is calculated. This results in estimations of overall entities such as power, torque, thrust, stiffness and excavation rate. These entities are used to calculate the loads on the cutters, which yields the lifetime of each individual cutter.
Different versions of some of the models have been developed to handle customer requests for different characteristics. Consequently, the optimization framework needs to automatically select the proper models for each case, which complicates the framework architecture. Hence, the optimization problem changes depending on the case, which makes it more difficult to tune the optimization algorithm as it is expected to solve all cases in an efficient manner and return Pareto optimal designs.
All models are considered as black boxes from the perspective of the calling entity, meaning that only zero-order information is available at each design point. The models are implemented in Python and Matlab, respectively. A generic Matlab-interfacing component for OpenMDAO has therefore been developed as part of this work.

https://doi.org/10.1017/pds.2021.482 Published online by Cambridge University Press

4.2 Problem formulation

In this framework, the requirements are considered as constants from the perspective of the optimization formulation. The design variables consist of other geometric and operational parameters and are summarized in table 1 together with the constraints and objectives. Variables x_1 through x_6 controls the actual shape of the cutter wheel, whereas x_7 and x_8 controls the excavating motion of the wheel. Other intermediate variables and constants are excluded for the sake of brevity.

Table 1. An overview of the design variables and responses defined in the optimization model.

Notation	Description	Value set	Units
x_1	Cutters per kerf in face section	$\mathbb{Z} \cap [1,3]$	—
x_2	Cutters per kerf in gage section	$\mathbb{Z} \cap [1,2]$	—
x_3	Number of kerfs in face section	$\mathbb{Z} \cap [3,24]$	—
x_4	Number of vertical strokes	$\mathbb{Z} \cap [2,5]$	—
x_5	Cutter wheel width	$\mathbb{R} \cap [1200,1800]$	mm
x_6	Cutter wheel diameter	$\mathbb{R} \cap [3200,4000]$	mm
x_7	Nominal rotational speed of cutter wheel	$\mathbb{R} \cap [9,15]$	min^{-1}
x_8	Special excavation mode	{True, False}	—
g_1	Minimum distance between any pair of cutters	$\mathbb{R} \cap [1,\infty)$	mm
g_2	Minimum load on any cutter at any time	$\mathbb{R} \cap [150,\infty)$	kN
g_3	Kerf spacing-penetration ratio	$\mathbb{R} \cap [5,15]$	—
g_4	Velocity of cutter wheel perifery	$\mathbb{R} \cap [2,2.75]$	ms^{-1}
f_1	Excavation rate	—	$\text{m}^3\text{min}^{-1}$
f_2	Minimum excavation lifetime of any cutter	—	m^3

This results in the problem formulation shown in eq. 1 where both the excavation rate (f_1) and the shortest lifetime (f_2) should be maximized to ensure as fast excavation process as possible. Since replacement of worn-out cutters means that the machine is standing by, the lifetime of all the cutters should be as long as possible, hence maximizing the shortest lifetime. The constraints ensure that the cutter configuration is physically feasible and that the required power is realistic.

$$\begin{array}{rlll} \text{maximize} & f_1(\vec{x}), & f_2(\vec{x}) & \\ \text{with respect to} & \vec{x} & & \\ \text{subject to} & 1 & \leq g_1(\vec{x}) & \leq \infty \\ & 150 & \leq g_2(\vec{x}) & \leq \infty \\ & 5 & \leq g_3(\vec{x}) & \leq 15 \\ & 2 & \leq g_4(\vec{x}) & \leq 2.75 \end{array} \tag{1}$$

4.3 Optimization execution and results

Following the procedure described above, the optimization problem has been solved using the presented NSGA-II implementation, with 76 individuals evaluated for 75 generations. All 5700 designs were sequentially evaluated over the course of approximately 20 minutes on an HP ZBook 15v G5 laptop, equipped with an Intel Core i7-8850H CPU and 16 GB of physical memory, running Windows 10 (build 18363).

The progression of the optimization solver can be followed through the evolution of each evolutionary generation's hypervolume (Zitzler and Thiele, 1998), as represented in figure 4. It can be seen that the hypervolume is initially *decreasing*, probably due to the elimination of infeasible solutions. Thereafter, it takes on a positive trend up until approximately generation 45 before reaching a plateau with minor fluctuations.

After filtering the complete design set from unfeasible solutions and subsequently selecting the non-dominated solutions, a Pareto-frontier of 68 points (represented in figure 5) is achieved. It is immediately visible that the Pareto-frontier can be divided into three main areas of interest. When examining the design variables behind these designs, it can be subjectively concluded that these areas (or clusters) coincide with the value of x_1. This is also reflected in the figure through the coloring of scatter points.

https://doi.org/10.1017/pds.2021.482 Published online by Cambridge University Press

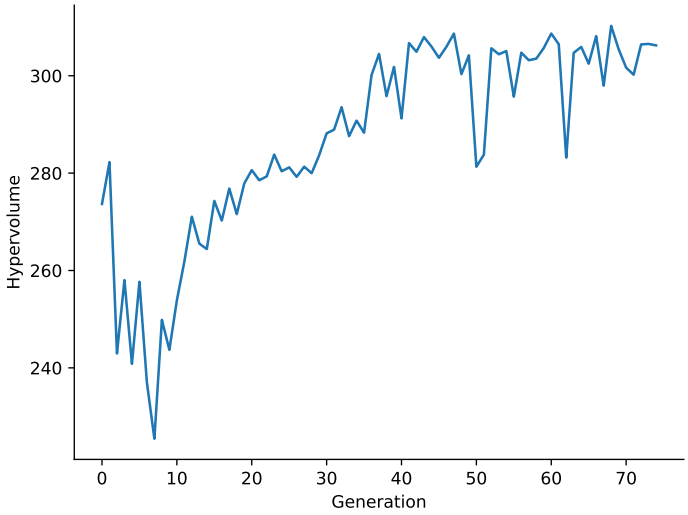

Figure 4. Hypervolume per generation.

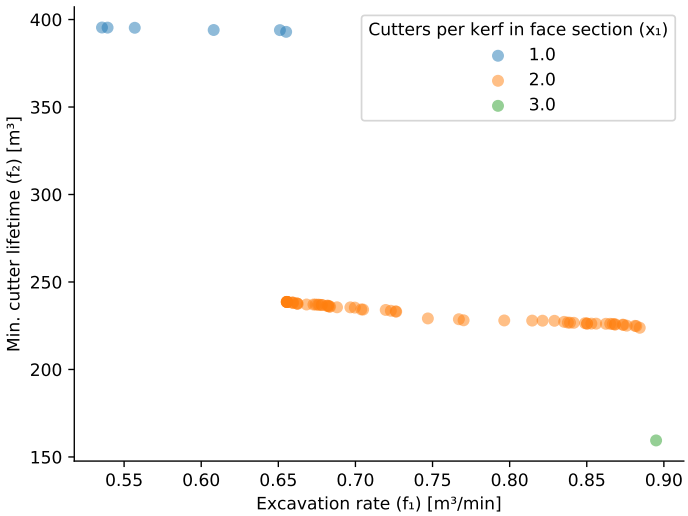

Figure 5. Scatter plot of the Pareto-optimal set of feasible solutions.

A scatter matrix with the two objectives and the eight design variables is presented in figure 6. Here, it can be observed that a number of design variable values are consistent across *all* designs in the obtained Pareto-optimal set: $x_2 = 1$, $x_4 = 3$ and $x_8 = \text{False}$.
While all constraints are not directly active (meaning that the constraint value would be equal to or close to a boundary) for the Pareto set, it has been observed that deactivating these constraints would yield solutions outside the allowed boundaries. Consequently, the constraints are needed to guide the optimization towards feasible regions.

5 DISCUSSION AND CONCLUSION

In this paper, a number of contributions that extend the OpenMDAO software library has been proposed, with the goal of broadening its area of applicability to include early-stage, exploratory design optimization with mixed variables. OpenMDAO was selected as the base platform for this work due to its currently strong support, both in terms of development and community engagement. By giving the means to structure the problem into separated, interchangeable and extensible parts, it can be a useful tool in building flexible and maintainable optimization applications.

https://doi.org/10.1017/pds.2021.482 Published online by Cambridge University Press

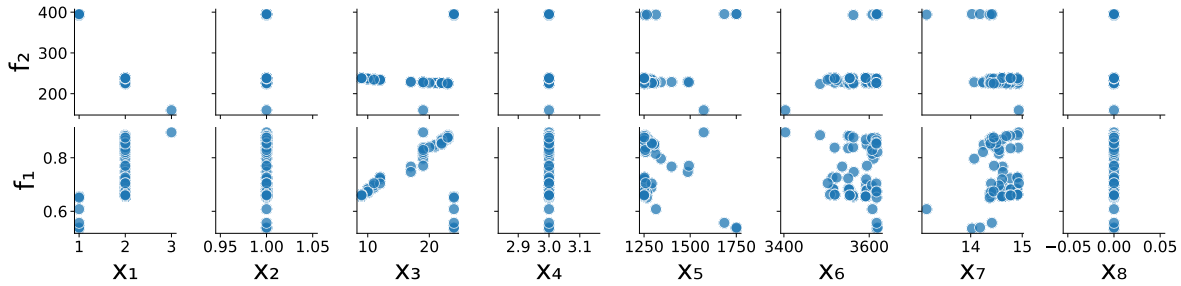

Figure 6. Paired scatter plots of Pareto-optimal set with design variables on the horizontal axis and objectives on the vertical axis.

To demonstrate the utility of these contributions, it has been shown that they can be used to assemble an optimization framework that is capable of optimizing a Mobile Miner on a conceptual level. The framework is intended for use in early design and quotation stages with the objective to get an estimate of important machine characteristics such as excavation rate and cutter lifetime. These characteristics are among the most important properties for the customer as they determine the tunneling speed to a great extent, which could be directly translated to cost for the client and contractor. The optimization parameters are focused around the cutter wheel (such as number of cutters, cutter placement, cutter wheel speed etc.) as these are decisive for the studied objectives. It is evident that the number of cutters strongly affects the shape of the Pareto front, and hence this parameter is essential to get right early in the design and quotation stage. Also, it is apparent that the problem is strongly characterized by the presence of a mix of continuous and discrete variables and hence the proposed extension of the OpenMDAO platform is essential to handle this type of problems in an efficient manner.

The MDO architecture used for the Mobile Miner example was selected due to its non-intrusive character, meaning that it does not in itself require any adaptations of the disciplinary models or the *native* optimization problem. Conversely, a CO approach would probably require case-specific modifications of the models, increasing the logical coupling between them. This would imply a decreased degree of flexibility in the framework, but could possibly render performance improvements on the other hand.

Directions for future work include higher fidelity models, in order to improve the prediction capabilities of the framework, and to link to further product development activities. While the Mobile Miner models have been approved by a professional in the field, their validity should also be investigated further. One way could be through the integration of alternative models in the same framework, providing a means to compare and contrast their results.

Finally, the Mobile Miner is a good example of a complex engineer-to-order product which requires a lot of engineering efforts for each new order. Here the concept of product configurators should be explored in future studies, and the optimization framework is developed with this in mind. By minor changes to the framework it would be possible to automatically reconfigure the optimization problem formulation to fit in a configurator setting. This configurator approach should provide visualization tools and should facilitate reuse of both models and problem formulations, and also the ability to include self-contained software packages so that proprietary models and knowledge could be handled efficiently.

ACKNOWLEDGEMENTS

Financial support from *Stiftelsen tekn.dr. Erik Johnssons stipendiefond* for the work presented in this paper is gratefully acknowledged.

REFERENCES

Andersson, Johan (2001). “Multiobjective optimization in engineering design: applications to fluid power systems”. PhD thesis. Linköping University. 201 pp.

Deb, Kalyanmoy, Amrit Pratap, et al. (2002). “A fast and elitist multiobjective genetic algorithm: NSGA-II”. *IEEE Transactions on Evolutionary Computation* 6.2, pp. 182–197. DOI: 10.1109/4235.996017.

Deb, Kalyanmoy and Agrawal Ram Bhushan (1995). “Simulated Binary Crossover for Continuous Search Space”. *Complex systems* 9.2, pp. 115–148.

https://doi.org/10.1017/pds.2021.482 Published online by Cambridge University Press

Epiroc (2021). *Mobile Miner 22H 3D render front/left side view*. Epiroc Media Gallery. URL: https://www.media.epiroc.com/en/search-results/file-detail.html/content/dam/epiroc/undergroundmining-and-tunneling/mre/mobile-miner/mobile-miner-22h/Mobile%20Miner%2022H_Mining_Tech02.tif.html (visited on 03/15/2021).
Fortin, Félix-Antoine et al. (2012). "DEAP: Evolutionary algorithms made easy". *The Journal of Machine Learning Research* 13.1, pp. 2171–2175. DOI: 10.5555/2503308.2503311.
Gray, Justin S. et al. (2019). "OpenMDAO: An open-source framework for multidisciplinary design, analysis, and optimization". *Structural and Multidisciplinary Optimization* 59.4, pp. 1075–1104. DOI: 10.1007/s00158-019-02211-z.
Hendricks, Eric S. and Justin S. Gray (2019). "pyCycle: A Tool for Efficient Optimization of Gas Turbine Engine Cycles". *Aerospace* 6.8, p. 87. DOI: 10.3390/aerospace6080087.
Hoyer, Stephan and Joseph J. Hamman (2017). "xarray: N-D labeled Arrays and Datasets in Python". *Journal of Open Research Software* 5, p. 10. DOI: 10.5334/jors.148.
Jasa, John P. et al. (2020). "Large-Scale Path-Dependent Optimization of Supersonic Aircraft". *Aerospace* 7.10, p. 152. DOI: 10.3390/aerospace7100152.
Lyly, Johnny, Sverker Hartwig, and Gunnar Nord (2018). "Epiroc Mobile Miner: hard rock cutting is now a reality". Bergdagarna 2018. Svenska Bergteknikföreningen, pp. 277–290.
Martins, Joaquim R. R. A. and Andrew B. Lambe (2013). "Multidisciplinary Design Optimization: A Survey of Architectures". *AIAA Journal* 51.9, pp. 2049–2075. DOI: 10.2514/1.J051895.
Robbins, R.J (2000). "Mechanization of underground mining: a quick look backward and forward". *International Journal of Rock Mechanics and Mining Sciences* 37.1, pp. 413–421. DOI: 10.1016/S1365-1609(99)00116-1.
Schmit, Lucien A (1960). "Structural design by systematic synthesis". *Proceedings of the 2nd Conference on Electronic Computation. 2nd Conference on Electronic Computation*. Pittsburgh: American Society of Civil Engineers, pp. 105–132.
Sgueglia, Alessandro et al. (2020). "Multidisciplinary Design Optimization Framework with Coupled Derivative Computation for Hybrid Aircraft". *Journal of Aircraft* 57.4, pp. 715–729. DOI: 10.2514/1.C035509.
Simpson, TimothyW. and Joaquim R. R. A. Martins (2011). "Multidisciplinary Design Optimization for Complex Engineered Systems: Report From a National Science Foundation Workshop". *Journal of Mechanical Design* 133.10, p. 101002. DOI: 10.1115/1.4004465.
Sun, Wei, Xiaobang Wang, Maolin Shi, et al. (2018). "Multidisciplinary design optimization of hard rock tunnel boring machine using collaborative optimization". *Advances in Mechanical Engineering* 10.1, p. 168781401875472. DOI: 10.1177/1687814018754726.
Sun, Wei, Xiaobang Wang, Lintao Wang, et al. (2016). "Multidisciplinary design optimization of tunnel boring machine considering both structure and control parameters under complex geological conditions". *Structural and Multidisciplinary Optimization* 54.4, pp. 1073–1092. DOI: 10.1007/s00158-016-1455-9.
Vidner, Olle (2020a). *OpenMDAO-Bridge-MATLAB*. Version v0.1. DOI: 10.5281/ZENODO.4316483.
— (2020b). *OpenMDAO-NSGA: NSGA-II and NSGA-III implementations for OpenMDAO*. Version v0.1. DOI: 10.5281/ZENODO.4279483.
— (2020c). *Scop: an opinionated, self-contained data format and post-optimization toolchain*. Version v0.3. DOI: 10.5281/ZENODO.4263728.
Wortmann, J. C. (1983). "A Classification Scheme for Master Production Scheduling". *Efficiency of Manufacturing Systems*. Ed. by B. Wilson, C. C. Berg, and D. French. Boston, MA: Springer US, pp. 101–109. DOI: 10.1007/978-1-4684-4475-9_10.
Zitzler, Eckart and Lothar Thiele (1998). "Multiobjective optimization using evolutionary algorithms— A comparative case study". *Parallel Problem Solving from Nature — PPSN V.* Ed. by Agoston E. Eiben et al. Red. by G. Goos, J. Hartmanis, and J. van Leeuwen. Vol. 1498. Berlin, Heidelberg: Springer, pp. 292–301. DOI: 10.1007/BFb0056872.

https://doi.org/10.1017/pds.2021.482 Published online by Cambridge University Press