# PhD proposal: "Optimizing over Learned Objective functions"

**Advisors** Emiliano Traversi, LIPN (emiliano.traversi@lipn.univ-paris13.fr)
Roberto Wolfler Calvo, LIPN (roberto.wolfler@lipn.univ-paris13.fr)
**Co-advisor** Pegah Alizadeh, Ericsson Research Center (pegah.alizadeh@ericsson.com)

**Abstract**

In this thesis, we propose to solve a series of real-world applications using optimization models and algorithms defined by machine learning techniques. We propose to obtain (i) mathematical programming problems closer to reality by meta-modeling techniques and (ii) efficient heuristic algorithms based on imitating exact combinatorial optimization algorithms.

## Context

Training a neural network can be viewed as merely interpolating a complex function. In many applications, describing the objective function or the constraints with a mathematical formulation is impossible. In these situations, resorting to tools from the vast black-box optimization field is necessary. A well-established technique in black box optimization is the so-called meta-modeling. Metamodeling forms a subfield of Machine Learning that represents a complex model $f$ with a simpler surrogate model $\hat{f}$ at the price of some approximation. One of the advantages of using the surrogate model is to enable a faster computation of new values of $\hat{f}$, as simulations are usually expensive to compute. For a more in-depth analysis of Surrogate Modeling, we refer the reader to [1, 2].

In [3], we propose to use a new objective function for a Dial-a-Ride Problem (DaRP) that considers the history of requests and the policy used by the online advertisement algorithm. Learning the objective function is only part of the job. The natural next step is to try to optimize it. In [3], we use a local search heuristic algorithm developed by a transportation company to optimize over the learned meta-model. The idea of not stopping at the *learning* phase by adding a subsequent *optimization* phase, using the learned function, is quite promising and has gathered some attention in recent years in the optimization community. For example, some works aiming at solving Neural Networks with generic MILP solvers are already available [4, 5, 6].

In [7], we propose to learn the function that returns its expected solving time for a given clique decomposition. The results obtained, even if preliminary, are promising. They show that it is possible to learn such a function, and it can be used to choose among different decompositions the most promising. From an optimizer point of view, the idea of optimizing "by enumeration" over a limited set of options is only the first (and unsatisfactory) step. The actual gain will come if we optimize the learned objective function over the whole set of feasible decompositions (in a similar way to that done in [3], where to find an optimal set of routes, we used a learned objective function). This could be done, for example, with an ad-hoc local search heuristic. An alternative is to use a Reinforcement Learning approach, where an agent is trained by learning from an expert, represented by the learned objective function. ML can be used as a fast heuristic to obtain good solutions by learning from an effective (but slow) optimization algorithm. This approach can also be viewed as a kind of imitation learning [8], where the expert is an optimization algorithm. In [9, 10], a Graph Neural Network is used to estimate which variables in a MIP will be part of an optimal solution. This allows to reduce the size of the problem by fixing a part of the variables and solving the reduced problem quickly, without losing the quality of the solution.

## Applications

In this section, we present three examples of potential applications that can be used to test the techniques mentioned.

**Dial-a-Ride Problem with a Neural Network as objective function** As a follow-up of the work in [3], we propose to use as an objective function for a Dial-a-Ride Problem (DaRP) a Neural Network using rectified linear unit (ReLU) as an activation function. We, therefore, propose to solve it with an exact algorithm, instead of a local search heuristic algorithm, as performed in [3]. It is possible to show that a Neural Network using rectified linear unit (ReLU) as an activation function can be modeled as a 0-1 Mixed Integer Linear Program [4]. An exciting research direction based on this idea is to investigate the polyhedral properties of such a model [6]. More precisely, we propose to investigate the convex hull of the intersection of the constraints representing the ReLU activation function with the constraints used to describe the Dial-a-Ride Problem. This would allow us to speed up the exact resolution of the newly defined model.

**Neural architecture search (NAS) for aerial drones** Neural architecture search (NAS) is a cutting-edge technique in the field of deep learning that automates the process of designing and optimizing neural network architectures. The traditional approach to designing neural networks involves manually selecting the network architecture, followed by a trial-and-error process of tweaking the hyperparameters to achieve the desired performance. This process can be time-consuming and highly dependent on the researcher's expertise. On the other hand, NAS searches through a vast space of potential neural network architectures and selects the most optimal one for a given task. This approach not only saves time and effort but also has the potential to discover novel and highly efficient network architectures that might have yet to be discovered through manual design.

NAS can be applied to various deep learning tasks such as image classification, object detection, speech recognition, and natural language processing. It has the potential to significantly improve the performance of these tasks by finding the best neural network architecture for the given data and problem. NAS also applies to scenarios with limited computational resources, such as mobile devices and edge computing, as it can identify lightweight and efficient neural network architectures that can run on such devices.

We propose to identify the most effective design for a neural network to control an aerial drone. By "effective", we mean the design that can provide the best accuracy, defined through an appropriately defined loss function. The idea is to start with a supernet that resumes all the possible choices that can be taken with respect to the architecture and subsequently simplify it. We propose to learn a surrogate function able to capture not only the contribution of an architectural choice but also the interactions between two choices. This can be achieved by using a surrogate function that is quadratic.

**Unmanned Aerial Vehicles as additional wireless network** Unmanned Aerial Vehicles (UAVs) can be used as aerial base stations and exploit their flexible and quick deployment to enhance the wireless network performance. In this context, it is crucial to have efficient algorithms able to identify both the placement of the drones and the associated resource allocation in a UAV-aided RAN slicing system. In the first work, we propose to use an algorithm based on Reinforcement Learning to identify the correct placement and resource allocation. We also propose to use a quadratic model to identify the optimal solution to the same problem. The solutions provided by the exact model are better than the ones obtained with the Reinforcement Learning approach. We therefore plan to use an imitation learning approach where the optimal placement and slicing are learned from the exact mathematical model. In this way we can achieve high-quality solutions quickly enough to use the algorithm in real time.

## Required skills

The Ph.D. student should have a master's degree in combinatorial optimization or operational research and knowledge in Machine Learning or Statistical Physics. Programming skills in C++ and/or Python and/or Julia with a Machine Learning framework (Pytorch, JAX...) will be much appreciated.

## References

[1] R. Alizadeh, J. K. Allen, and F. Mistree, "Managing computational complexity using surrogate models: a critical review," *Research in Engineering Design*, vol. 31, no. 3, 2020.

[2] S. Amaran, N. V. Sahinidis, B. Sharda, and S. J. Bury, "Simulation optimization: a review of algorithms and applications," *Annals of Operations Research*, vol. 240, no. 1, 2016.

[3] L. Zigrand, P. Alizadeh, E. Traversi, and R. Wolfler Calvo, "Machine learning guided optimization for demand responsive transport systems," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 420–436, Springer, 2021.

[4] M. Fischetti and J. Jo, "Deep neural networks and mixed integer linear optimization," *Constraints*, vol. 23, no. 3, pp. 296–309, 2018.

[5] T. P. Papalexopoulos, C. Tjandraatmadja, R. Anderson, J. P. Vielma, and D. Belanger, "Constrained discrete black-box optimization using mixed-integer programming," in *International Conference on Machine Learning*, pp. 17295–17322, PMLR, 2022.

[6] R. Anderson, J. Huchette, W. Ma, C. Tjandraatmadja, and J. P. Vielma, "Strong mixed-integer programming formulations for trained neural networks," *Mathematical Programming*, vol. 183, no. 1, pp. 3–39, 2020.

[7] C. Alizadeh, P. Alizadeh, M. F. Anjos, L. Létocart, and E. Traversi, "How to learn the optimal clique decompositions in solving semidefinite relaxations for opf," in *2022 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, IEEE, 2022.

[8] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Computing Surveys (CSUR)*, vol. 50, no. 2, pp. 1–35, 2017.

[9] M. Gasse, D. Chételat, N. Ferroni, L. Charlin, and A. Lodi, "Exact combinatorial optimization with graph convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[10] V. Nair, S. Bartunov, F. Gimeno, I. von Glehn, P. Lichocki, I. Lobov, B. O'Donoghue, N. Sonnerat, C. Tjandraatmadja, P. Wang, *et al.*, "Solving mixed integer programs using neural networks," *arXiv preprint arXiv:2012.13349*, 2020.