Hybrid heuristic for the optimal design of photovoltaic installations considering mismatch loss effects

Matteo Salani[a,*], Gianluca Corbellini[b], Giorgio Corani[a]

[a] *Dalle Molle Institute for Artificial Intelligence (IDSIA), USI/DTI-SUPSI, Switzerland*
[b] *Institute for Applied Sustainability to the Built Environment (ISAAC), DACD-SUPSI, Switzerland*

Abstract

We consider the Photovoltaic Installation Design Problem (PIDP). In this problem, photovoltaic modules must be connected in "strings" and wired to a set of electronic components. The aim is to minimize installation costs and maximize power production, which is affected by "mismatch losses" caused by non-uniform irradiation (shading) and is also directly related to design decisions. We relate the problem to the known class of location routing problems and we design a route-first/cluster-second heuristic. We propose an efficient machine learning approach to evaluate Photovoltaic (PV) string performances accounting for mismatch losses. We prove that our approach is effective in real-world instances provided by our industrial partner.

Keywords: Hybrid Heuristic, Machine Learning, Photovoltaic Installation Design

1. Introduction

Recent progress in Photovoltaic (PV) technology has led to a lower levelized cost of energy, mostly thanks to rising efficiencies and to falling price of components [13]. Most research focuses primarily on improving the efficiency and reliability of modules, while less attention is devoted to the system design process itself. A sub-optimal configuration represents a bottleneck for the realization

*Corresponding author
Email address: matteo.salani@idsia.ch (Matteo Salani)

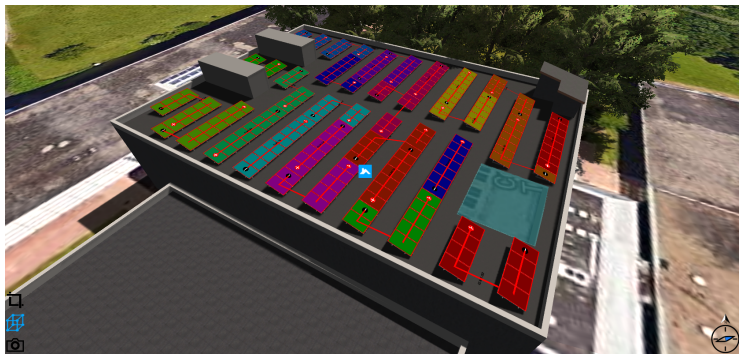


Figure 1: Computer simulation of irradiation and shading effects on a photovoltaic plant.

and capitalization of an efficient PV installation [12]. When using the design support tools available on the market today, it is often difficult to evaluate the cost-effectiveness of design decisions related to a specific installation, especially when photovoltaic systems are installed in the built environment (mainly on rooftops).

One of the disregarded aspects concerns the electrical performance (voltage and current) of PV installations. Mismatch loss effects are due to different level of irradiation between modules due to partial shading. These can cause additional losses up to 25% on an optimal configuration [16].

Computationally intensive simulations, exploiting high-resolution information of shading effects, are necessary to perform accurate electrical performance analyses. In fig. 1 we show an example of a simulated photovoltaic plant where module shading is computed for a given position of the sun. In order to produce a performance analysis for a given configuration, the simulation must be repeated for many conditions. This excessive computational cost renders the photovoltaic installation design problem, including electrical performance, intractable.

In the context of this applied project, financed by the Swiss Federal Commission for Innovation and Technology, we collaborated with an industrial partner: InSun SA, an IT company based in Lugano, which offers web-based configuration and simulation software for designing any kind of PV system. Thanks

to the simulation of shading effects provided by an advanced 3D graphics engine, InSun's technology is suited for accurately estimate the performance of PV installations in the built environment.

The goal of our work is to design a computationally efficient solution approach for the design of a PV installation exploiting AI and Optimization algorithms. The contributions are the following: we define the Photovoltaic Installation Design Problem (PIDP) and relate it to location routing problems; we propose an efficient machine learning approach to evaluate PV string performances accounting for mismatch losses; we design a multi-objective optimization approach for the problem and perform an empirical computational study on real-world instances.

In section 2 we describe the physical model underlying the mismatch loss estimation and we detail its approximation by machine learning in section 3. We describe the PIDP problem in section 4 and an optimization approach in section 5. Finally, in section 6 we report on computational experiments.

2. Physical model of the mismatch loss

Figure 2 illustrates a PV installation. The PV modules are organized in strings (in series) and the strings are assembled in parallel. Figure 2 illustrates 44 PV modules connected in four strings, strings S1 and S2 and strings S3 and S4 are assembled in parallel.

PV strings can operate at different voltage set points. The Maximum Power Point Tracker (MPPT) is the electronic component that sets the optimal operational point of PV strings for current and voltage in order to maximize their power production. In figure 2 there are two MPPTs. In any installation, all strings connected to a MPPT must be composed of an equal number of modules.

Strings are connected in parallel in junction boxes (JB) and then connected to specialized cables so as to reach MPPTs which are physically located within *Inverters* these being devices used to convert the energy from direct current to alternate current. Every inverter can host one or more MPPTs. Finally, inverters are connected to the power grid.

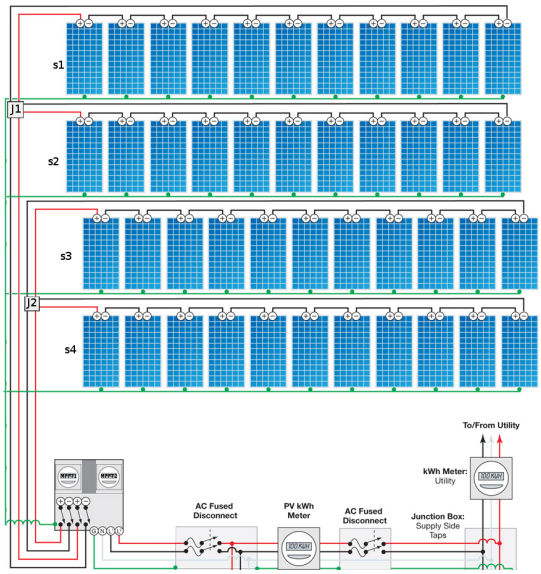


Figure 2: Photovoltaic installation with four strings. S1 and S2 are connected in parallel into junction J1. S3 and S4 are connected in parallel into junction J2. The inverter is equipped with two MPPTs (illustration from www.homepower.com).

In photovoltaic systems the power output depends on the irradiance incident on the modules and shading can cause significant energy losses. Shading conditions change during the day and vary according to the time of year. When modules are facing the same environmental conditions (irradiation and temperature), every module is working at its own best set-point. Modules under differing conditions, due mostly to partial shading by surrounding objects, lead to some of them not working at their best set-point. This reduces the (common) current flowing through all the modules connected to the same MPPT eventually reducing the power output of other, non-shaded, modules. This phenomenon is known as mismatch loss effect.

A simulation is required to compute the mismatch loss of a given plant configuration. The simulation uses the single diode model (Fig. 3) and the implicit equation [2] providing the current (I) voltage (V) characteristic of the module for a given irradiance condition ([3]).

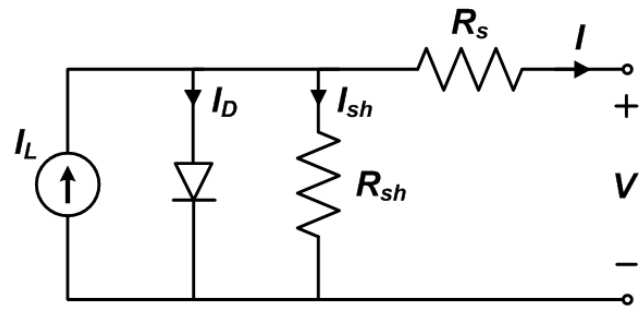


Figure 3: The single diode model for photovoltaic cells

$$I = I_L - I_0 \left[exp \frac{q(V + IR_S)}{nV_T} - 1 \right] - \frac{(V + IR_S)}{R_{SH}}$$

where I_L represents the light-generated current in the cell, I_0 the diode reverse saturation current, R_S the series resistance , R_{SH} the shunt resistance and n the diode i1deality factor ([11]).

For every sun position when partial shading is occurring, the software computes the I-V characteristic of each solar cell of the PV field (which can easily reach several thousands) using equation 2 and then computing the I-V curve resulting from the connection in series and in parallel. For medium and large photovoltaic installations (from 50kWp to several MWp), this computation can be very time consuming, even for a single configuration of modules rendering intractable the evaluation of several different layout configurations.

3. Approximating the physical model with an estimator

We use a meta-model approach, approximating with a statistical model the mismatch loss computed by the physical model. Thus for each different number of PV strings s, we build a synthetic data set by simulating the physical model in a range of different conditions and we train a statistical predictor to approximate the mismatch loss computed by the physical model.

The inputs provided to both the physical model and the statistical model are:

- the ratio between diffuse irradiation and global irradiation,
- the proportion of shadow referring to each string, which yelds s variables.

There are hence $s+1$ independent variables, on the basis of which we predict the mismatch loss computed by the physical model. For each different number of strings s, we train a random forest model [10, Chap.15] as a statistical predictor. The random forest algorithm is widely recognized as one the most accurate machine learning approaches for function approximations [9], given its ability in capturing interaction between features and non-linearities.

	Number of strings						
	1	2	3	4	5	6	7
	Characteristic of the data						
Number of data points	200	2300	17,700	20,000	20000	20,000	20,000
mean mismatch loss	0.11	0.13	0.14	0.15	0.16	0.16	0.16
	Modelling accuracy						
mean absolute error	0.01	0.007	0.004	0.005	0.005	0.006	0.007
correlation true/ predicted	0.97	0.99	1.00	1.00	1.00	1.00	1.00

Table 1: The random forest model reliably approximates the mismatch loss computed by the physical model on the different data sets.

We validate the model via 10-folds cross-validation. As reported in Table 1, we obtained very good approximations in all data sets. The mean absolute error is generally *two* orders of magnitude smaller than the variable to be predicted (mismatch loss), and the correlation between true and predicted values is generally $>.99$. The most difficult case is the data set with s=1; however the performance is also good in this case (the absolute error is smaller by one order of magnitude than the mismatch loss and the correlation is around .97). The non-linearity of the problem is demonstrated by the fact that on the same data set a linear regression yields a true/predicted correlation of only 0.53.

The results are robust also with respect to the sample size of the training data where we checked how the results vary with the sample size, using training

sets of size 200 and 2000. For instance, in the case where s=3, the correlation is 0.98 for n=200 and 0.99 for n=2000. Similar results are obtained also with the other values of s. Thus the results are robust with respect to the size of the training data.

Once trained, the random forest model yields very fast predictions, being capable of producing hundreds of predictions per seconds. Thus it can be used for quickly computing the mismatch loss within the optimization problem.

Good accuracy when approximating a physical model with a statistical one has already been observed in literature; see for instance [4]. Indeed, synthetic data might contain strong non-linearities but they are much less noisy than the real data; hence a powerful statistical estimator can approximate such data with a high level of accuracy.

4. Photovoltaic installation design problem

The photovoltaic installation design problem (PIDP) is stated as follows: given a set of homogeneous modules, an installation field (i.e., a surface with physical constraints), a set of electronic components (i.e., MPPTs and inverters) and a set of sun irradiation samples, determine the installation layout so that the overall cabling cost and the overall mismatch loss are minimized.

4.1. Input data

We formally define the data needed to describe the decision problem: given a set M of homogeneous PV modules, let p_m be the nominal power of module $m \in M$, and c_m its position, represented by a point in a bi-dimentional space. Given a set T of MPPTs, let nm_t and ns_t be the maximum number of modules and the maximum number of strings that can be connected to tracker $t \in T$, respectively. Additionally, each string attached to the tracker $t \in T$ must be composed of no less than lm_t modules and no more than um_t modules. Given a set I of inverters, let nm_i be the maximum number of modules that can be connected to inverter $i \in I$. For each inverter $i \in I$ we also know the subset of MPPTs belonging to it, $T_i \subseteq T$. Given a set S of irradiance samples, let di_s and

Notation	Description
M	Set of homogeneous modules
p_m	Nominal power of module $m \in M$ expressed in Watts
c_m	Coordinates of module $m \in M$ expressed as pair (x_m, y_m)
T	Set of trackers (MPPTs)
nm_t	Maximum number of modules per tracker $t \in T$
lm_t	Minimum number of modules per string connected to tracker $t \in T$
um_t	Maximum number of modules per string connected to tracker $t \in T$
I	Set of inverters
nm_i	Maximum number of modules per inverter $i \in I$
$T_i \subseteq T$	Sub-set of trackers belonging to inverter $i \in I$
S	Set of sun irradiation samples
di_s	Direct irradiance for sample $s \in S$
ii_s	Diffuse irradiance for sample $s \in S$
w_s	Relative weight of sample $s \in S$
$a_{m,s} \in (0,1)$	Shading status of module $m \in M$ in sample $s \in S$

Table 2: Notation summary

ii_s the direct irradiance and the diffuse irradiance of sample $s \in S$, respectively. We are also given a relative weight w_s of a sample $s \in S$ as the fraction of that time of year when the corresponding irradiance sample is expected (note that $\sum_{s \in S} w_s = 1$) . For each module $m \in M$ and irradiation sample $s \in S$, we are given the shading status of the module $a_{m,s} \in (0,1)$. It represents the power fraction of the module with respect to the nominal power we are expecting at irradiation sample s. Notation is summarized in Table 2.

4.2. Decisions

The decisions associated with the PIDP concern the selection of inverters and MPPTs to be used for the solution and the design of the cabling layout to organize PV modules in strings and connect them to the selected MPPTs. Therefore, for each MPPT, we have to decide the number and length of PV strings. Finally, we have to design the location of the junction box where a given set of PV strings are connected in parallel.

4.3. Objectives

The PIDP has multiple objectives. The first objective (z_1) requires the maximization of the number of modules in the solution. As feasible MPPT configurations are limited, there may be problem instances in which not all modules can be connected.

The second objective (z_2) requires minimizing the overall length installation cables. The cables required to assemble the installation are comprised of those necessary to form each PV string, plus the cables needed to connect strings in parallel and the cables required to connect the strings to the MPPTs. The length of all cables is computed using the L1 norm as in general they are secured to the hardware which is organized in a sort of grid. The third objective (z_3) requires minimizing production losses due to mismatch effects computed on the given sun irradiation samples.

The first objective is much more important then the others. Therefore, given any two solutions r_1 and r_2, r_1 dominates r_2 if

$$z_1(r_1) > z_1(r_2) \vee (z_1(r_1) = z_1(r_2) \wedge z_2(r_1) < z_2(r_2) \wedge z_3(r_1) < z_3(r_2))$$

4.4. Multi-depot location routing problem

We observe that the PIDP can be conveniently cast to a multi-depot location routing problem with some problem specific constraints. Location-routing problems (LPRs) are vehicle routing problems (VRPs) in which the number and the position of the depots are decided at the same time as the routes the vehicles will take so that the overall cost is minimized [14]. About a hundred scientific contributions have been published on the LRP so far (see [15] and [20]). Among those, we focus on contributions related to the *capacitated* LRP (CLRP), i.e., the problem in which both depots and vehicles have a finite capacity.

There are few contributions on exact algorithms, we mention Contardo et al. [6], based on branch-and-cut-and-price and able to solve instances with 50 customers and 5 to 10 potential depots and Farham et al. [8] who addressed the problem with time windows. More contributions on heuristic methods exist.

Contardo et al. [5], presented a GRASP + ILP matheuristic. It is a population based heuristic which shows good quality performances but relatively poor computational performances (thousands of seconds on instances with up to 200 customers). More recently, Peng et al. [17] proposed a particle swarm algorithm consistently faster that solves instances with up to 200 customers with an average gap of 3%. We mention some variants and applications of the CLRP, even if a full review is out of the scope of this contribution (see [7]). Ahn et al. [1] addressed the LRP with profits, i.e. the problem in which some of the customers can be left unserved, and Toro et al. [21] addressed the LRP with multiple objectives.

The CLRP is defined on a complete, undirected and weighted graph. Vehicles are homogeneous, i.e., all vehicles have the same limited capacity. A solution to the CLRP consists of a subset of depots to be opened, assigning customers to one opened depot and a set of vehicle routes starting at opened depots and visiting all customers. Constraints state that the demand of customers assigned to one depot must not exceed its capacity, and the demand of customers in a route must not exceed the vehicle's capacity; routes begin and end at the same depot; each vehicle undertakes at most one trip; each customer is served by a single vehicle. The objective is to minimize the sum of the fixed costs of opening depots, the fixed costs of using vehicles and the total cost of the routes used. The CLRP is NP-hard.

We now describe how an instance of the PIDP can be seen as an instance of the CLRP and outline the few differences among the two problems. Conceptually, each PV string in PIDP is modeled as a vehicle route in CLRP. In CLRP, the depots are the locations where vehicles' routes start and end. In PIDP, depots are the location where PV strings are joined together (i.e. junction boxes). The set of customers is composed by the set of modules. The set of potential depot nodes is composed of a copy of the modules set. Indeed, PV strings belonging to a MPPT can be conveniently connected together in the proximity of a module as the space is usually sufficient. Alternatively, the set of potential positions for junction boxes can be provided as input data.

In PIDP, the customer demand is unitary, the number of vehicles is unbounded, and the vehicles' capacities are represented by the upper bound on the number of modules per string um_t. In PIDP, the number of modules per string is also bounded from below by lm_t and this represents one of the differences between the PIDP and the CLRP. The cost matrix of the CLRP can conveniently be described by the assembly distance between two modules. Adjacent modules that can be connected directly by their cables have distance 0.

The second difference between the PIDP and the CLRP is that all PV strings connected to a MPPT must contain the same number of modules, in CLRP terms, the number of customers associated with routes belonging to the same depot must be the same. This peculiarity is not addressed in any of the contributions we reviewed.

Finally, we observe that the objectives detailed in subsection 4.3 make our problem closer to the CLRP described in [1] and [21]. Indeed, considering objective z_1, there is the chance that not all modules are used in the solution (i.e., not all customers are serviced by a vehicle). However, once [1] the benefit of visiting customers are set against transportation costs, our solution considers the connection of a module (i.e. visiting a customer) with higher priority.

5. Hybrid heuristic

The many similarities between the PIDP and the CLRP allow us to exploit knowledge of the CLRP to design a suitable algorithm to solve the PIDP. We decided not to consider exact methods to solve our problem as our instances of the PIDP consist of several hundreds of modules, and the computational time available to produce a solution is limited to a few minutes. We focused on methods based on the route-first/cluster-second principle introduced by Beasley [2] and succesfully applied to routing problems (e.g. Prins et al., [19]) which consists of computing a Hamiltonian cycle (called "giant" TSP tour), i.e. a vehicle tour visiting all nodes of the graph, and then splitting the cycle into node subsets that satisfy capacity constraints.

In our application, the route-first/cluster-second approach has some advantages with respect to other heuristic solutions, namely:

- The computational effort related to the routing component is concentrated on computing one or just a few Hamiltonian cycles once. The position of each PV module is known before the designer takes decisions on the set of electronics to be used in the final installation. Therefore, the route-first component of the algorithm can be executed in parallel while the design choices are made. This enables the computation costs to be split with advantages in terms of responsiveness of the solution.

- Solving a Hamiltonian cycle problem on a planar graph (as is the case in our application) guarantees that no two edges cross in any optimal solution. No edge crossing means fewer cabling issues during installation. The same result is not guaranteed by other methods, in particular when burdens on computational time are imposed.

- The PIDP is a multi-objective problem. After the routing component provides a good solution in terms of cabling costs, several good candidate solutions can very rapidly be computed along with the clustering component.

In order to solve the PIDP, we designed a hybrid heuristic. Each objective is treated separately by a different component of the heuristic, figure 4 illustrates how the different components interact.

In particular, the main objective z_1, is addressed by an iterative partitioning algorithm that produces a set of MPPT configuration schemes. The minimization of total cabling z_2, is addressed by the routing component of the algorithm by using different metrics, being able to compute different Hamiltonian cycles. Finally, the clustering components produce a set of non dominated solutions exploiting the meta-model presented in section 2 to compute objective z_3.

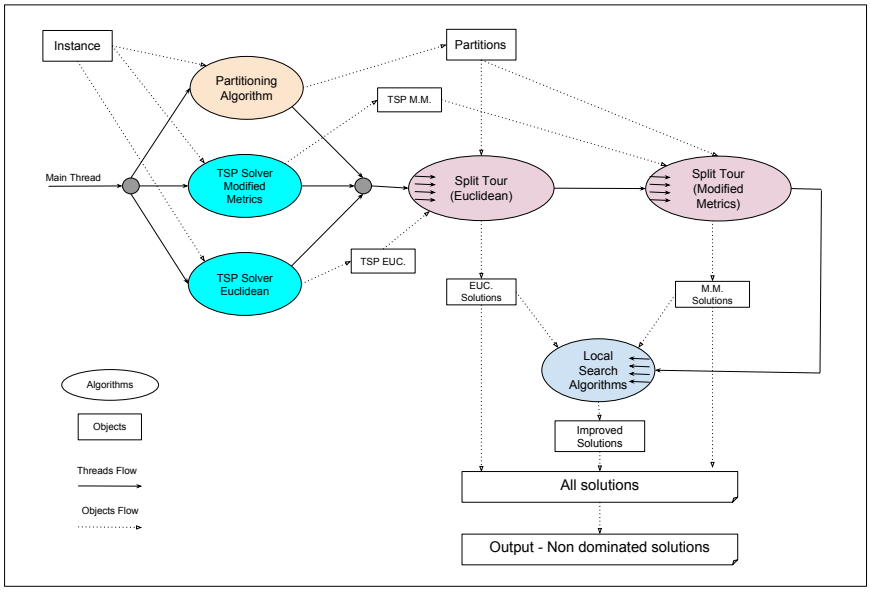


Figure 4: Algorithm architecture: algorithms are in ellipsoids, data is in squares, execution flow follows solid arrows, data flow follows dashed arrows, parallel arrows means parallel execution.

5.1. Optimal MPPT configurations

The purpose of this component of the heuristic is to determine a set of MPPT configurations for the PIDP.

A configuration is defined by a subset of selected trackers $T' \subseteq T$ and, for each selected tracker $t \in T'$, by a feasible tracker setup. In order to be feasible, a tracker setup must not exceed the maximum number of modules, nm_t and must be composed by no more than ns_t strings all formed by at most um_t and at least lm_t modules. The set of all feasible tracker setups, C_t, can be easily computed by an enumeration algorithm in a preprocessing phase. To solve the configuration problem, we designed two approaches: the first based on Integer Programming and solution cuts, the second on a recursive enumeration scheme.

Assume for simplicity that all trackers are identical, we therefore have a unique set of feasible setups C. In the case of non identical trackers, the approach is slightly more complicated and not reported here the sake of com-

pactness. Let m_c be the number of modules connected in setup $c \in C$. Let $x_c \in \mathbb{Z}^+$, represent the number of times the setup $c \in C$ is selected. The optimal configuration problem (OC) can be computed solving the following integer problem:

$$z_1 = \max \sum_{c \in C} m_c \cdot x_c \tag{1}$$

$$s.t. \sum_{c \in C} m_c \cdot x_c \leq |M| \tag{2}$$

$$\sum_{c \in C} x_c \leq |T| \tag{3}$$

$$x_c \in \mathbb{Z}^+ \qquad \forall c \in C \tag{4}$$

where (1) maximizes the total number of modules, which cannot be more than the available ones (2). We cannot select more than $|T|$ setups, (3). As the purpose of this component of the heuristic is to identify several MPPT configurations, we designed an iterative approach in which, at each iteration, the optimal solution of the OC problem is discarded by a specific solution cut. Let us consider the binary encoding of the problem. We obtain this encoding considering the number of bits necessary to represent all possible values of variables x_c, that is $B = 0 \cdots \lceil \log(|T|) \rceil$. Then, the OC can be transformed to its binary version OCB as follows:

$$\begin{aligned}
z_1 = & \max \sum_{c \in C} m_c \cdot \sum_{b \in B} 2^b \cdot x_c^b \\
s.t. & \sum_{c \in C} m_c \cdot \sum_{b \in B} 2^b \cdot x_c^b \leq |M| \\
& \sum_{c \in C} \sum_{b \in B} 2^b \cdot x_c^b \leq |T| \\
& x_c^b \in \{0, 1\} \qquad \forall b \in B, \forall c \in C
\end{aligned}$$

Let $x^*(i)$ be the optimal solution of OCB at iteration i and let P the indices of non-zero variables in $x^*(i)$. Then the following solution cut renders infeasible

the current optimal solution:

$$\sum_{p \in P} x_p \leq |P| - 1$$

The iterative procedure adds solution cuts until at a given iteration j, $z_1(j) < z_1(1)$ and it guarantees finding all MPPT configurations.

Even if the OCB is small in terms of variables and its solution is very fast (about 15-20 milliseconds in our tests), the number of optimal configurations can be quite large (order of several thousands). Therefore, the approach is acceptable for small-sized instances but cumbersome for bigger instances. In our experiments we used the exact approach with instances up to 100 modules in order to maintain the computational time in order of seconds.

To overcome the issue of dimensionality of the exact approach, we designed a greedy recursive enumeration scheme. For a given number of modules n and number of tracker k, we compute the average number of modules per tracker $a = n/k$ and we discard all tracker configurations with less than a modules. For each tracker, we consider ordered configuration iteratively. For a given selected configuration for tracker t, we ignore all configurations for tracker $t + 1$ with more modules than those associated with t. We therefore obtain all simple combinations and we avoid full enumeration of all possible configuration and avoid redundancy. Once a setup has been selected, say with i modules, the algorithm recursively solves the configuration with $k - 1$ trackers and $n - i$ modules. If no solution is found with n modules, the process is repeated iteratively with $n - 1$ modules and so on. At the end of the process, a non redundant subset of MPPT configurations is obtained. The pseudo code of the procedure is reported in algorithms 1 and 2.

In order to explore some structurally different solutions, once the set of MPPT configurations has been designed, say set G, we sample a subset of configurations with differing variances in the number of assigned modules per tracker. The configuration's variance for each configuration g is computed as

follows:

$$v_g = \frac{1}{|T'|} \cdot \sum_{t \in T'} (m_t - \overline{m})^2$$

where m_t is the number of modules in configuration g for tracker t and $\overline{m}$ is the average number of modules per tracker.

In our experiments, we observed that there exist good solutions in which the configuration of trackers has low variance (all trackers are similarly configured), and good solutions in which the configuration of trackers has high variance (some trackers with many modules and some with few modules). Different types of MPPT configurations are therefore worth exploring.

Algorithm 1: Recursive configuration algorithm

```
Data: M ← Total modules in instance
Data: MinT ← Minimum amount of trackers required
Data: MaxT ← Maximum amount of trackers required
Data: MaxModules ← Maximum modules per tracker
Data: TConfigs ← Tracker Configurations
Result: Configurations
while Configurations is empty do
    for i = MinT to MaxT do
        Configurations.Add(generateConfs(M, T, MaxModules,
          TConfigs))
    end
    M ← M-1
end
return Configurations;
```

5.2. Routing component

PV modules are assembled in an installation field, i.e. a bi-dimensional space with some physical constraints. In particular, as installation fields may be on rooftops, straight connection between any two modules is sometime impossible as rooftops may be non-convex or contain obstacles along the way. Still, there may be a viable path that a cable can be placed along to connect any two pairs of modules. To this end, we developed a pre-processing component with which we compute the cabling distance between any pair of modules in an installation field defined as a possibly non-convex polygon and a set of possibly non-convex

Algorithm 2: generateConfs() - Recursive call

```
Data: M ← Modules
Data: T ← Tracker Amount
Data: TConfigs ← Tracker Configurations
Result: Configurations
for i in TConfigs do
    if i ≤ (M/T) then
        continue;
    end
    MaxModules ← (M/T)
    TmpConfs ←
     generateConfs(M − i, T − 1, Min(MaxModules, i), TConfigs)
    for TC in TmpConfs do
        TC.Add(i)
        Configurations.Add(TC)
    end
end
return Configurations;
```

obstacles. Details are omitted for brevity. This pre-processing component computes two distance matrices for two different norms: Euclidean norm and a modified norm.

The modified norm has been designed so that some module connections are preferred over others according to practical issues related to installation. In particular, any photovoltaic installation is composed of a series of longitudinal tracks where modules are assembled and cables can be effectively secured. Therefore, given any two modules m_1 and m_2, the modified norm is computed as $d_{m_1,m_2} = |y_{m_1} - y_{m_2}| + \alpha \cdot \sqrt{(y_{m_1} - y_{m_2})^2 + (x_{m_1} - x_{m_2})^2}$, that is the ordinate distance of modules is the main factor of the norm, this is perturbed with a component proportional to the Euclidian norm. In our tests, the parameter α is set to 0.1.

Given a set of modules M and a set of distance matrices L, according to the route-first/cluster-second methodology (see Prins et al. [19]), we compute a set F of Hamiltonian cycles.

In our implementation, we exploit existing Large Neighbourhood Search algorithms for the Traveling Salesman Problem (e.g., Pisinger & Ropke [18]).

Efficient implementations are available off the shelf: the well known Concorde library (`http://www.math.uwaterloo.ca/tsp/concorde/index.html`) or the routing library in Google's OR tools (`https://developers.google.com/optimization/`).

5.3. Clustering component

Given a MPPT configuration $g \in G$ and a TSP tour $f \in F$, we build a solution of the PIDP splitting the tour into as many strings as configuration g prescribes. Furthermore, we assign strings to trackers and decide on the location of the junction point to connect the strings.

We designed an iterated local search that leads to an initial set of solutions with a constructive heuristic and then improves it with local exchanges.

In the construction phase, we select k pairs of consecutive modules (m_1, m_2) on tour f connected with the k-longest distances. Then, for each pair, we construct a solution by splitting the tour starting from module m_2, following the tour according to configuration g, assigning trackers to strings in the order they appear in configuration g.

Given tracker t, and given a set of strings connected to it, we compute the average point of all strings' heads and tails. The location of the junction point corresponds to the module that is closest to the average point. Once strings are associated with a tracker the mismatch loss can be estimated exploiting the meta-model presented in section 2.

The described procedure generates a population of $k \cdot |G| \cdot |F|$ solutions. Each solution in the population is then improved by a local search aiming at the minimization of objectives z_2 and z_3. The neighbourhood is computed by exchanging any pair of strings of the same length belonging to two different trackers until no further improvement can be obtained. While the cabling component of the strings does not change, z_2 can still improve as the junction point can be further optimized. Moreover, objective z_3 can also be improved on to the different configuration of each tracker.

Only non-dominated solutions, according to definition in subsection 4.3, belong to the final set of solutions for the PIDP.

Instance	$\|M\|$	$\|I\|$	$\|T\|$	ns_t	lm_t	um_t	$\lceil \|M\|/\|T\| \rceil$
60_3_6	60	3	6	1	7	18	10
68_3_6	68	3	6	1	9	24	12
210_5_10L	210	5	10	1	14	24	21
210_5_10H	210	5	10	1	14	24	21
577_6_12	577	6	12	3	13	24	49
631_3_3	631	3	3	11	21	24	211
674_6_12	674	6	12	3	15	24	57
903_14_14	903	14	14	3	22	24	65
1218_4_4	1218	4	4	13	18	27	305

Table 3: Test cases

6. Computational results

We performed experiments for real-world instances proposed by our industrial partner. In this section, we compare solutions obtained with our methodology with those obtained through the company's best practices, consisting of a manual configuration phase in which designers selects the MPPT configuration (number of strings and number of modules per string) and an automated phase in which modules are associated with the MPPTs with a greedy constructive heuristic.

We have at hand 9 real-world instances described in Table 3. All instances have homogeneous trackers. We report which instance we are referring to, the number of modules, $|M|$, the number of inverters, $|I|$, the number of trackers, $|T|$, the maximum number of strings, ns_t, the minimum and maximum number of modules per string, lm_t and um_t, respectively. Finally we report the average number of modules per tracker, $\lceil |M|/|T| \rceil$. The smallest instance has 60 modules and an average of 10 modules per tracker, the largest instance has 1218 modules and an average of 305 modules per tracker.

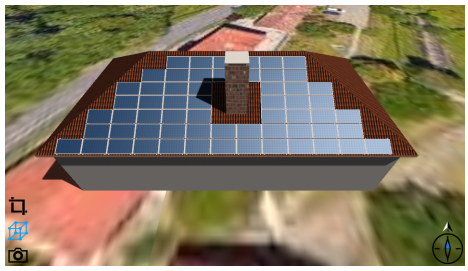
Table 4 reports on computational results. For each instance we report the reference solution provided by the industrial partner and three solutions obtained by the method described in this paper. The computational time to obtain all so-

lutions is reported in the last column. The first solution, marked "Best MML", records the best solution in terms of mismatch loss (z_3), the second solution, marked "Best Length", records the best solution in terms of total cable length (z_2), and the third solution, marked "Best Improvement", records the solution with the best overall positive percentage improvement over the benchmark. In the following columns, "Assigned modules" records the number of modules in the solution, "Total Length" (z_2) records the overall cabling necessary to install the PV plant in meters and "MML" records on the overall mismatch losses (z_3). Percentage improvements are also reported with respect to the reference solution. In all instances but one, the number of connected modules (objective z_1) is maximum, i.e. all modules are connected. Savings related to cabling can be up to 31.76% in the largest instance and mismatch losses are reduced up to 45.20%. In some cases, the best solution with respect to one objective (e.g., z_2) has a worse performance on the corresponding value of the other objective (e.g., z_3) and this is expected. For example, in instance 631_3_3 the solution with the best mismatch loss performance of 5.95 corresponds to a increase of cable length by 3.27%. Anyway, we observe that the method is able to provide several solutions to the decision maker. Indeed, we observe that the proposed method always produces for each instance a solution that simultaneously improves both objective functions. Finally, we remark that for the largest instance, the proposed method enables all 1218 modules to be connected while, for the reference solution, one module less is connected and still the overall cabling and mismatch loss are largely reduced.

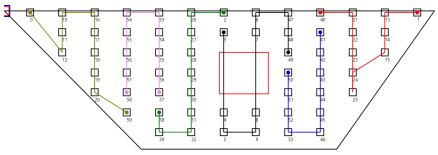
We illustrate some solutions in order to visually compare benchmark solutions with those obtained with our method. Figure 5 illustrates the solutions of instance 60_3_6. It is the smallest instance of the set defined for a convex surface (the rooftop of a villa) with one obstacle. Figure 5a illustrates the 3D rendering of the instance, while figures 5b and 5c illustrate the schematic connection of modules in strings in the provided solution and in the "Best improvement" solution. In the pictures, the connection points of modules are represented by black squares. We observe that in the solution provided strings are organized

Instance	Solution Type	Assigned Modules	Total Length	(%)	MML	(%)	Time(s)
instance_60_3_6	Reference	60	176.6		4.19		
	Best MML	60	200.9	-13.75%	**3.74**	**10.74%**	1.8
	Best Length	60	**142.1**	**19.55%**	4.40	-5.01%	
	Best Improvement	60	151.6	14.16%	4.04	3.58%	
instance_68_3_6	Reference	68	227.2		6.46		
	Best MML	68	226.1	0.50%	**6.11**	**5.42%**	2.1
	Best Length	68	**158.4**	**30.30%**	6.75	-4.49%	
	Best Improvement	68	185.2	18.49%	6.43	0.46%	
instance_210_5_10H	Reference	210	665.9		3.52		
	Best MML	210	582.7	12.50%	**2.96**	**15.91%**	6.9
	Best Length	210	**548.4**	**17.65%**	3.46	1.70%	
	Best Improvement	210	582.7	12.50%	**2.96**	**15.91%**	
instance_210_5_10L	Reference	210	665.9		2.21		
	Best MML	210	628.0	5.70%	**1.99**	**9.95%**	5.0
	Best Length	210	**548.4**	**17.65%**	2.19	0.90%	
	Best Improvement	210	555.2	16.62%	2.14	3.17%	
instance_577_6_12	Reference	577	2243.8		4.48		
	Best MML	577	2034.0	9.35%	**3.49**	**22.10%**	123.9
	Best Length	577	**1765.1**	**21.33%**	3.94	12.05%	
	Best Improvement	577	1785.5	20.43%	3.87	13.62%	
instance_631_3_3	Reference	631	2254.4		7.86		
	Best MML	631	2328.2	-3.27%	**5.95**	**24.30%**	79.9
	Best Length	631	**1825.3**	**19.03%**	7.52	4.33%	
	Best Improvement	631	2026.3	10.12%	6.54	16.79%	
instance_674_6_12	Reference	674	1884.2		2.97		
	Best MML	674	2251.0	-19.47%	**2.60**	**12.46%**	98.9
	Best Length	674	**1719.1**	**8.76%**	3.14	-5.72%	
	Best Improvement	674	1771.7	5.97%	2.82	5.05%	
instance_903_14_14	Reference	903	1979.3		3.48		
	Best MML	903	2072.6	-4.71%	**2.87**	**17.53%**	102.9
	Best Length	903	**1736.3**	**12.28%**	4.84	-39.08%	
	Best Improvement	903	1869.0	5.57%	3.02	13.22%	
instance_1218_4_4	Reference	1217	4957.2		10.93		
	Best MML	**1218**	3824.9	22.84%	**5.99**	**45.20%**	192.5
	Best Length	**1218**	**3382.9**	**31.76%**	9.45	13.54%	
	Best Improvement	**1218**	3592.0	27.54%	6.15	43.73%	

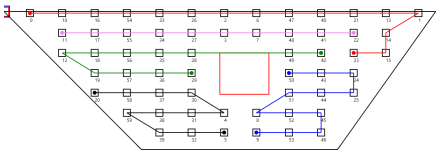
Table 4: Computational results



(a) Rendering



(b) Provided solution



(c) Our solution

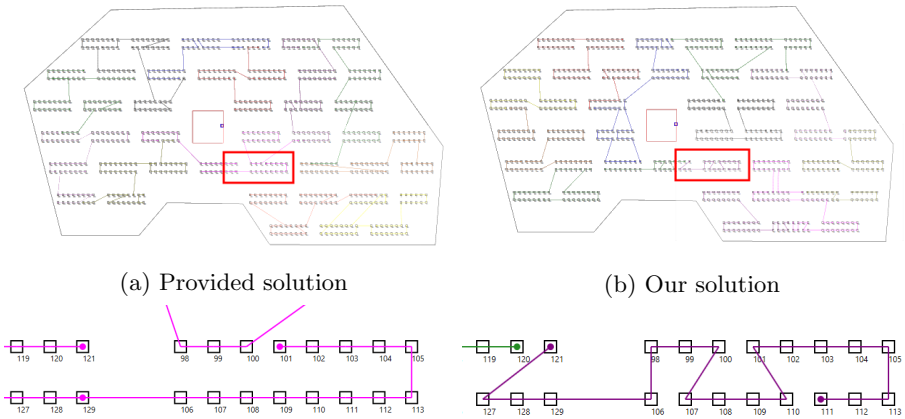
Figure 5: Instance 60_3_6

"vertically" while in our solution strings are organized "horizontally". In this case the solution provided means connectiong one string around the central obstacle. Our solution reduced both unnecessary cabling and mismatch losses. Practitioners validated our solution.

Figure 6 illustrates the solutions of instance 674_6_12. The instance is medium-sized and is defined for a non-convex surface with one convex obstacle. We observe that modules are organized in groups. In figures 6c and 6d we detail a portion of the solution. We observe that our method finds non-trivial routing solutions that are still practical for installation purposes. On the contrary, the constructive heuristic produces more straight sequences that induce more "jumps" between groups of modules. Our solution reduced both unnecessary cabling and mismatch losses. In this case also, practitioners validated our solution.

7. Conclusions

We have shown an efficient hybridization of machine learning and optimization to tackle a real-world problem. The Photovoltaic Installation Design Problem (PIDP) can be modelled as a location routing problem and solved with the



(a) Provided solution

(b) Our solution

(c) Detail of provided solution

(d) Detail of our solution

Figure 6: Instance 674_6_12

known arsenal of OR methodologies. Results are encouraging: a set of non-dominated solutions is computed in a reasonable amount of time enabling the decision maker to compare different solutions. The approach is shown to be applicable to a production environment. Further work should be concentrated in assessing the quality of the method with the computation of valid lower bounds.

Acknowledgments

This research has been supported by the Swiss Federal Commission for Technology and Innovation (KTI) under the grant Nr.18704.2

[1] Ahn, J., de Weck, O., Geng, Y., Klabjan, D.: Column generation based heuristics for a generalized location routing problem with profits arising in space exploration. European Journal of Operational Research 223(1), 47 – 59 (2012)

[2] Beasley, J.: Route firstcluster second methods for vehicle routing. Omega 11(4), 403 – 408 (1983)

[3] Bishop, J.: Computer simulation of the effects of electrical mismatches in photovoltaic cell interconnection circuits. Solar Cells 25(1), 73–89 (1988)

[4] Carnevale, C., Pisoni, E., Volta, M.: Selecting effective ozone exposure control policies solving a two-objective problem. Ecological Modelling 204(1), 93–103 (2007)

[5] Contardo, C., Cordeau, J.F., Gendron, B.: A GRASP + ILP-based metaheuristic for the capacitated location-routing problem. Journal of Heuristics 20(1), 1–38 (Feb 2014)

[6] Contardo, C., Cordeau, J.F., Gendron, B.: An exact algorithm based on cut-and-column generation for the capacitated location-routing problem. INFORMS Journal on Computing 26(1), 88–102 (2014)

[7] Drexl, M., Schneider, M.: A survey of variants and extensions of the location-routing problem. European Journal of Operational Research 241(2), 283 – 308 (2015)

[8] Farham, M.S., Sral, H., Iyigun, C.: A column generation approach for the location-routing problem with time windows. Computers & Operations Research 90(Supplement C), 249 – 263 (2018)

[9] Fernández-Delgado, M., Cernadas, E., Barro, S., Amorim, D.: Do we need hundreds of classifiers to solve real world classification problems. Journal of Machine Learning Research 15(1), 3133–3181 (2014)

[10] Friedman, J., Hastie, T., Tibshirani, R.: The elements of statistical learning, vol. 1. Springer (2001)

[11] Gray, J.L.: The Physics of the Solar Cell, pp. 61–112. John Wiley & Sons, Ltd (2005)

[12] Green, M.A.: Commercial progress and challenges for photovoltaics. Nature Energy 1, 15015 (2016)

[13] Kang, M.H., Rohatgi, A.: Quantitative analysis of the levelized cost of electricity of commercial scale photovoltaics systems in the us. Solar Energy Materials and Solar Cells 154, 71–77 (2016)

[14] Laporte, G., Nobert, Y., Arpin, D.: An exact algorithm for solving a capacitated location-routing problem. Annals of Operations Research 6(9), 291–310 (Sep 1986)

[15] Nagy, G., Salhi, S.: Location-routing: Issues, models and methods. European Journal of Operational Research 177(2), 649 – 672 (2007)

[16] Peled, A., Appelbaum, J.: Enhancing the power output of pv modules by considering the view factor to sky effect and rearranging the interconnections of solar cells. Progress in Photovoltaics: Research and Applications 25(9), 810–818 (2017)

[17] Peng, Z., Manier, H., Manier, M.A.: Particle swarm optimization for capacitated location-routing problem. IFAC-PapersOnLine 50(1), 14668 – 14673 (2017)

[18] Pisinger, D., Ropke, S.: Large Neighborhood Search, pp. 399–419. Springer US, Boston, MA (2010)

[19] Prins, C., Lacomme, P., Prodhon, C.: Order-first split-second methods for vehicle routing problems: A review. Transportation Research Part C: Emerging Technologies 40(Supplement C), 179 – 200 (2014)

[20] Prodhon, C., Prins, C.: A survey of recent research on location-routing problems. European Journal of Operational Research 238(1), 1 – 17 (2014)

[21] Toro, E.M., Franco, J.F., Echeverri, M.G., Guimares, F.G.: A multi-objective model for the green capacitated location-routing problem considering environmental impact. Computers & Industrial Engineering 110(Supplement C), 114 – 125 (2017)