Communication Papers of the of the 17th Conference on Computer Science and Intelligence Systems pp. 85–92
DOI: 10.15439/2022F19
ISSN 2300-5963 ACSIS, Vol. 32

Surrogate Estimators for Complex Bi-Level Energy Management

Fatiha Bendali, Jean Mailfert
LIMOS Lab. CNRS/UCA
Clermont-Ferrand, France

Eloise Mole Kamga
Labex IMOBS3, LIMOS Lab
CNRS/UCA, Clermont-Ferrand, France

Alejandro Olivas Gonzalez, Alain Quilliot,
Labex IMOBS3,
LIMOS Lab, UCA/CNRS
Clermont-Ferrand, France
Email: {alain.quilliot@uca.fr}

Hélène Toussaint
LIMOS UCA/CNRS
Clermont-Ferrand, France

***Abstract*—We deal here with the routing of vehicles in charge of performing internal logistics tasks inside some protected area. Those vehicles are provided in energy by a local solar hydrogen production facility, with limited storage and time-dependent production capacities. One wants to synchronize energy production and consumption in order to minimize both production and routing costs. Because of the complexity of resulting bi-level model, we deal with it by short-cutting the production scheduling level with the help of surrogate estimators, whose values are estimated through fast dynamic programming algorithms and through machine learning.**

I. Introduction

THE notion of multi-level decisional [1, 5, 6] model arises when decision is shared between several players. Solving such a model aims at providing a scenario which would be the best (or almost the best) in case all the players accept to submit themselves to a common authority (centralized paradigm), or, if it is not the case (collaborative paradigm), at helping them into the search for a compromise solution [14].

Solving a multi-level model is a difficult task and standard approaches involve decomposition schemes, which may be hierarchical (Benders decomposition of ILP models, …) or transversal (Lagrangean relaxation,…). Still, in both case, a major difficult remains, related to the sensitivity issue, that means about the way one may retrieve information from the handling of the different levels in order to make them interact. So a trend, boosted by the rise of machine learning technology [12, 15], is to bypass some levels of the global model through surrogate constraints or functions, likely to behave as an approximation of the values induced inside the model by optimal decisions taken at those levels.

It is this point of view which we adopt here while dealing with a problem related to the synchronization between local solar energy (hydrogen, PV, [9]) production and its consumption by a fleet of autonomous vehicles. This problem arose in the context of the activities of IMOBS3 (*Innovative Mobility*) Labex in Clermont-Ferrand of the national PGMO program promoted by power company EDF. On one side, the production manager schedules the activity of the micro-plant, which includes not only energy production, but also energy purchase and sale on the market, taking into account that both production costs and production capacities are time-dependent. On the other side, the fleet manager schedules and routes the vehicles in such a way they efficiently achieve a set internal logistic tasks. Both meet in order to perform *refueling transactions*, when the vehicle moves toward the micro-plant in order to refuel. Limited storage capacities impose to synchronize time dependent energy production and consumption. Though many searchers have recently showed interests into the emerging decisional models which are related to the management of renewable energy, they most often focused either on production control and scheduling [1, 10] or one the issues related to consumption [2, 3, 7, 11, 13], without dealing with the synchronized interaction between both levels. We do it here by shortcutting the part of the process related to production scheduling and using a surrogate formulation of the cost related to production. We try two kind of such surrogate formulations: the first one is based upon the fact that, a routing decision Γ being given, an optimal refueling decision may be obtained in polynomial time by solving a simple parametric auxiliary *Refuel*(Γ, β) problem, where β is a cost parameter; the second formulation relies on an approximation by a neural network of the optimal value of the production sub-problem which results from fixing both route decision Γ and related refueling strategy. Though in practice part of the difficulty here is due to uncertainty [8], we suppose, for the sake of simplicity, that all our system behaves in a deterministic way.

So the paper is organized as follows. We first (Section II) describe the problem and its ILP formulation. While doing it, we introduce *strong no_sub-tour* constraints, which may be separated in polynomial time in order to make possible the implementation of a *Branch and Cut* algorithm. Next (Section III) describes an auxiliary parametric problem *Refuel*(Γ, β), which may be solved in polynomial time through dynamic programming. We use it as a surrogate quality estimator, which computes *ad hoc* vehicle routes while bypassing the production part of the problem. In Section IV, we apply machine learning in order to get a fast approximation of the optimal value for the production sub-problem. Section V is devoted to numerical experiments.

II. THE SVREP: SYNCHRONIZED VEHICLE ROUTING/ENERGY PRODUCTION PROBLEM

We consider a fleet of autonomous hydrogen powered vehicles, initially located at a depot 0, and which are required to perform a VRP: *Vehicle Routing Problem* tour, that means to visit a set of stations $\{0,\ldots, M\}$ within a time horizon $[0, TMax]$. Moving from station j to station k requires $\Delta_{j,k}$ time units and an amount $E_{j,k}$ of energy. Those vehicles are constrained by a same fuel storage capacity C^{Veh}, they all start their journey with a same fuel load E_0, and are required to end it loaded with at least E_0 energy units. For the sake of simplicity, we restrict here ourselves to the case when only one vehicle is involved.

Because of this hypothesis about the fuel load of the vehicle at the beginning and at the end of its tour, this vehicle must refuel at least once. It does it while moving to a micro-plant – 1, provided with a storage facility (a tank) with capacity C^{Prod}, filled with hydrogen which may be either produced *in situ*, or bought, or, in case of excess, sold. We restrict ourselves to the case when hydrogen may neither be bought from outside nor sold. Hydrogen is produced from period to period through a combination of photolysis and electrolysis, which makes both its production rate (the amount of energy produced in one period if the micro-plant is activated) and its production cost (the cost of the electricity for the electrolysis) be time dependent. Periods are labeled from 1 to N, and every period has a same duration p in such a way that $TMax = p.N$. The production rate at period i is denoted by R_i and related production cost is denoted by $Cost_i$. Activating the micro-plant requires some human intervention, whose cost is fixed and denoted by C_Act. Refueling the vehicle takes a full period, and safety forbids the micro-plant to be at the same time producing and refueling. A consequence is that the vehicle may not be served as soon as it arrives to the micro-plant, but must wait until the micro-plant achieves its production target. At time 0, the micro-plant is loaded with H_0 energy units, and it should be loaded with at least H_0 at the end of the process. Though in true life uncertainty is part of the problem, we suppose here that production rates are deterministic. An example of cost and production rates is provided by figure 1.

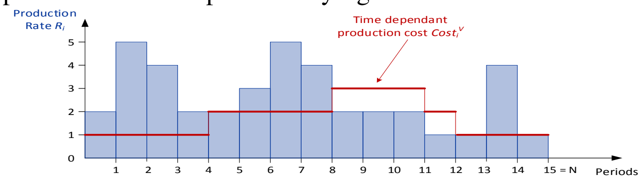

Figure 1. *Time dependent production rates costs for the micro-plant*

Then, resulting **SVREP**: *Synchronized Vehicle Routing and Energy Production* problem is about the computation of the route Γ, of the production schedule and of the *refueling transactions* (j, k, L, T^*) which will tell us the stations j such that the vehicle deviates its route while moving from j to its successor k in Γ in order to go to the micro-plant, related load L and related time T^*. The goal is to minimize a mixed cost $G\_Cost + \alpha.T$, where G_Cost is the global economic cost induced by the production process, T is the time when the vehicle achieves its journey, and α is a time versus money conversion coefficient. Figure 2 below shows an example of synchronization between a route 0, 1, …, M, 0 followed by the vehicle and the production of hydrogen by the micro-plant during periods 1,…, N, in case $p = 2$, $E_0 = 8$, $H_0 = 4$, $TMax = 30$, $C\_Act = 7$, $C^{Prod} = 15$, $C^{Veh} = 15$, $\alpha = 1$. In such a case, the vehicle refuels twice: the first transaction, performed at period 5 between stations 1 and 2, involves 13 fuel units, and the second one, performed at period 13 between stations 3 and 4, involves 12 fuel units. Resulting tour ends at time 30 and production cost is 3.7 + 2 + 6 + 1 = 30. It comes that resulting global cost is 30 + 30 = 60.

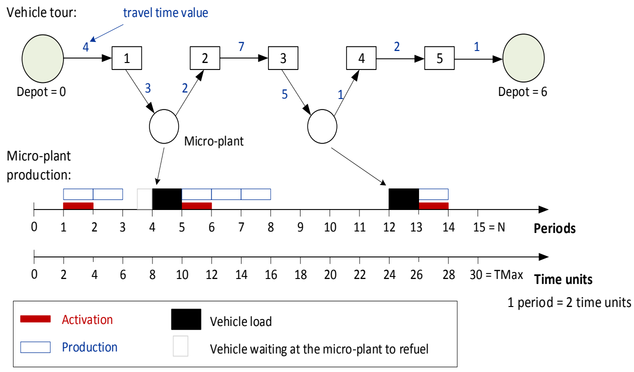

Figure 2. *A* ***SVREP*** *feasible solution.*

A. An ILP Formulation ILP_SVREP of SVREP

SVREP is does not fit the ILP machinery. Still, it is interesting to set such a model **ILP-SVREP**, since it will help us in benchmarking and also since part of this model will play a role inside our surrogate quality estimator oriented approach. We introduce fictitious period 0, a fictitious station M+1 identified with $Depot = 0$, and the following variables:

- **Production variables**:
  - $z = (z_i, i = 0,\ldots, N)$, with $\{0, 1\}$ values: $z_i = 1$ ~ the micro-plant is active during period i.
  - $y = (y_i, i = 1,\ldots, N)$, with $\{0, 1\}$ values: $y_i = 1$ ~ the micro-plant is activated at time $(p\text{-}1).i$.
  - $V^{Prod} = (V^{Prod}{}_i, i = 0,\ldots, N) \geq 0$: $V^{Prod}{}_i$ is the load of the *micro-plant* tank at the end of period i.
  - $\delta = (\delta_i, i = 1,\ldots, N)$, with $\{0, 1\}$ values: $\delta_i = 1$ ~ a refueling transaction takes place during period i.
  - $L^* = (L^*_i, i = 1,\ldots, N) \geq 0$: f $\delta_i = 1$, then L^*_i is the hydrogen transferred to the vehicle during period i.
- **Vehicle variables**:
  - $Z = (Z_{j,k}\ k \neq j$ in -1, 0,…, M+1, with $\{0, 1\}$ values: $Z_{j,k} = 1$ iff the vehicle moves from j to k.
  - $X = (X_{j,k}, k \neq j$ in 0,…, M+1), with $\{0, 1\}$ values: X means the restriction to stations $\{0,\ldots, M\}$ of the vehicle route.
  - $L = (L_j, j = 0,\ldots, M) \geq 0$: if $Z_{j,-1} = 1$ then L_j = quantity of hydrogen loaded by the vehicle at the micro-plant.

- $T = (T_j, j = 0,\dots, M + 1) \geq 0$: T_j = time when the vehicle arrives at j.
- $T^* = (T^*_j, j = 0,\dots, M + 1) \geq 0$: if $Z_{j,-1} = 1$, then T^*_j = time when the vehicle starts refueling.
- $V^{Veh} = (V^{Veh}_j, j = 0,\dots, M + 1) \geq 0$: V^{Veh}_j = hydrogen load of the vehicle when it arrives in j.

- **Synchronization variables**: $U = (U_{i,j}, i = 1,\dots, N, j = 0,\dots, M)$ with {0, 1} values: $U_{i,j} = 1$ ~ the vehicle refuels during period i after traveling from j to the micro-plant.

Then, α being a time vs money coefficient, the *cost* function comes as follows: $\textbf{Cost} = \Sigma_{i=1,\dots,N} (C\_Act.y_i + Cost_i.z_i) + \alpha.T_{M+1}$.

We must now explicit the constraints, while distinguishing those related to production, to routing and to synchronization.

- **Production constraints**:
  - For any $i = 1,\dots, N$: $y_i \geq z_i - z_{i-1}$. (E1)
  - For any $i = 1,\dots, N$: $z_i + \delta_i \leq 1$. (E2)
  - $z_0 = 0$; $V^{Prod}_0 = H_0$; $V^{Prod}_N \geq H_0$. (E3)
  - For any $i = 1,\dots, N$: $V^{Prod}_i = V^{Prod}_{i-1} + z_i.R_i - L^*_i$. (E4)
  - For any $i = 1,\dots, N$: $L^*_i \leq V^{Prod}_{i-1} \leq C^{Prod}$ and $L^*_i \leq \delta_i.C^{Veh}$. (E5)

  Explanation: (E1) expresses the fact that activating the micro-plant at period i means that it was idle at period i-1 and that it becomes active at period i. (E2) means that simultaneously producing and refueling is forbidden. (E3) reflects the initial and final requirements. (E4, E5) express the way the load micro-plant's tank evolves throughout the periods and set bounds on the amount of hydrogen transferred by the micro-plant to the vehicle.

- **Vehicle constraints**:
  - $Z_{M+1,0} = 1$; For any j, $Z_{j,j} = 0$; For any $j \neq M+1$, $Z_{j,0} = 0$ and $Z_{M+1,j} = 0$. (F1)
  - For any $j = 0,\dots, M+1$, $\Sigma_k Z_{j,k} = 1 = \Sigma_k Z_{k,j}$. (F2)
  - $\Sigma_j Z_{-1,j} = \Sigma_j Z_{j,-1} \geq 1$. (F2Bis)
  - For any k, j, $X_{k,j} \geq Z_{k,j}$. (F3)
  - For any $j = 0,\dots, M$, $\Sigma_k X_{j,k} = 1 = \Sigma_k X_{k,j}$. (F3Bis)
  - $V^{Veh}_0 = E_0$; $V^{Veh}_{M+1} \geq E_0$. (F4)
  - For any $j = 0,\dots, M + 1$: $E_{j,-1} \leq V^{Veh}_j \leq C^{Veh}$. (F4Bis)
  - For any $j, k = 0,\dots, M+1$: $Z_{j,k} + (V^{Veh}_k - V^{Veh}_j + E_{j,k})/C^{Veh} \leq 1$. (F5)
  - For any $j,k = 0,\dots, M$: $(X_{j,k} - Z_{j,k}) + (V^{Veh}_k - V^{Veh}_j + E_{j,-1} + E_{-1,k} - L_j)/C^{Veh} \leq 1$. (F6)
  - For any $j = 0,\dots, M$: $L_j \leq C^{Veh} + E_{j,-1} - V^{Veh}_j$. (F7)
  - For any $j = 0,\dots, M$: $L_j \leq (\Sigma_k (X_{j,k} - Z_{j,k})).C^{Veh}$. (F7Bis)
  - $T_0 = 0$; $T_{M+1} \leq TMax$. (F8)
  - For any $j, k = 0, \dots, M$: $Z_{j,k} + (T_j + \Delta_{j,k} - T_k)/TMax \leq 1$. (F9)
  - For any $j, k = 0, \dots, M$: $(X_{j,k} - Z_{j,k}) + (T^*_j + p + \Delta_{-1,k} - T_k)/TMax \leq 1$. (F10)
  - For any $j = 0,\dots, M$: $(\Sigma_k (X_{j,k} - Z_{j,k})) + (T_j + \Delta_{j,-1} - T^*_j)/TMax \leq 1$ (F10Bis)

  Explanation: (F1, F2, F2Bis) mean that the vehicle follows a circular route which visits exactly once every station j and refuels at least once. (F3, F3Bis) mean that vector X describes the route followed by the vehicle when bypassing the micro-plant. Notice that if $X_{j,k} = 1$ and $Z_{j,k} = 0$ then (F3, F3Bis) also implies that $Z_{j,-1} = Z_{-1,k} = 1$. (F4, F4Bis) express the initial and final constraints on the vehicle's tank, as well as the fact that the vehicle must be able to return to the micro-plant as soon as it needs. (F5) expresses the way the load of the vehicle evolves when the vehicle moves from station j to station k without refueling and (F6) expresses what happens when the vehicle refuels between j and k. (F7, F7Bis) set bounds on the fuel amount that the vehicle may receive while moving from station j to its successor k according to X. (F8) sets the initial and final conditions related to the time values. (F9) expresses the evolution of time value T when the vehicle moves from station j to station k without refueling. (F10, F10Bis) express this evolution when the vehicle refuels while moving from j to k, taking into account refueling and waiting times.

- **Synchronization constraints**:
  - For any $j = 0,\dots, M$: $\Sigma_{i=1,\dots,N} U_{i,j} = Z_{j,-1}$. (G1)
  - For any $i = 1,\dots, N$, $\delta_i = \Sigma_{j=0,\dots,M} U_{i,j}$ (G2)
  - For any $j = 0,\dots, M$, $T^*_j = \Sigma_{i=1,\dots,N} p.(i-1).U_{i,j}$. (G3)
  - For any $i = 1,\dots, N$, $j = 0, \dots, M$: $U_{i,j} + (L_j - L^*_i)/C^{Veh} \leq 1$. (G4)

  Explanation: (G1, G2) mean that refueling transactions induce a matching between the refueling periods and the stations j such that the vehicle moves from j to *micro-plant* = - 1 during its trip. (G3) fixes the time when such a refueling transaction involving station j starts. (G4) means that, when a refueling transaction takes place which involves both station j and period i, then received load L_j is equal to (does not exceed) transferred load L^*_i.

B. Enhancing SVREP: Strong No_Sub_Tour Constraints

Above ILP formulation may be enhanced by additional constraints (*Cuts*). The first one tells us that the capacity of the micro-plant imposes at least $(\Sigma_{i=1,\dots,N} R_i.z_i)/C^{Prod} - 1$ refueling transactions and so at least $(\Sigma_{i=1,\dots,N} R_i.z_i)/C^{Prod}$ activations of the production. It may be formulated as follows: $\Sigma_{1 \leq i \leq N} y_i \geq (\Sigma_{i=1,\dots,N} R_i.z_i)/C^{Prod}$. (E6)

Also, we get a lower bound for time value T_{M+1} by setting:

- $T_0 = 0$; $T_{M+1} \geq \Sigma_{j,k=-1,\dots,M+1} \Delta_{j,k}. Z_{j,k}$. (F11)

Finally, we notice that, if the vehicle spends an amount W of energy while moving inside or at the boarder of some station subset A, which does contain the micro-plant – 1 but may possibly contain 0, M+1 or both, then it must move at least

$\lceil W/C^{Veh} \rceil$ times toward the micro-plant in order to refuel. In order to formalize resulting constraint, we denote by J the set {-1, 0, …, M+1} and set, for any such a subset A of J:

- $Cl(A)$ = {arcs $e = (j, k)$ s.t at least j or k is in A};
- $\delta(A)$ = {arcs $e = (j, k)$, s.t $j \notin A$ and $k \in A$};

Also, for any arc $e = (j, k)$, we set:

- $\Pi_{j,k} = E_0$ if $(j, k) = (M+1, 0)$ and $\Pi_{j,k} = C^{Veh}$ else.
- $\Pi^*_{j,k} = C^{Veh} - E_0$ if $(j, k) = (M+1, 0)$ and $\Pi_{j,k} = C^{Veh}$ else.

This leads us to the following *Strong No_Subour* constraint:

- For any subset A of {0, …, M+1},
  $\Sigma_{(j,k) \in \delta(A)} \Pi_{j,k} .Z_{j,k} \geq \Sigma_{(j,k) \in Cl(A)} E_{j,k}.Z_{j,k}$ (F12)
- For any subset A of {0, …, M+1},
- $\Sigma_{(j,k) \in \delta(J-A)} \Pi^*_{j,k} .Z_{j,k} \geq \Sigma_{(j,k) \in Cl(A)} E_{j,k}.Z_{j,k}$ (F12Bis)

Theorem 1: *If {0, 1} vector Z satisfies (F1, F2, 2Bis, F12, F12Bis) if and only if arcs (j, k) such that $Z_{j,k} = 1$ define a collection γ of sub-tours $\gamma_0, \gamma_1, …, \gamma_s, .., \gamma_S$, such that:*

- *γ_0 starts from Depot = 0, ends into micro-plant = - 1, and spend less than E_0 energy;* (I1)
- *γ_1 starts from micro-plant = - 1, ends into Depot = 0, and spend less than C^{Veh} - E_0 energy;* (I2)
- *Routes $\gamma_2, …, \gamma_S$ start from – , end into -1and do not require more than C^{Veh} energy units.* (I3)

Sketch of the Proof: (F1, F2, F2Bis) implies that F gives rise to a collection γ of sub-tours $\gamma_0, …, \gamma_S$, which satisfy (I1, I2, I3) above but for the energy consumption conditions. Then one checks that if some tour γ_s spends more energy than related upper bound, then a subset A of {0, …, M+1} exists which makes Z violate either (F12) or (F12Bis). □

Notice that (F12) contains the standard *no_subtour* constraint for VRP as soon as $E_{j,k} = 0$ implies $j = k$.

C. Separating the Strong No_Sub_Tour Constraints.

Given a rational vector Z solution of the linear program obtained by relaxing the integrality constraints from **ILP-SVREP**, separating (F12) means checking (*separation* process), whether there exists $A \subseteq$ {0, 1, …, M+1} which violates this constraint. Once provided with such an efficient separation procedure, we may perform a *Branch and Cut* process, while using the *callback* mechanisms as implemented in libraries, like CPLEX, GUROBI, … It happens to be the case here:

Theorem 2: *(F12, F12Bis) may be separated in polynomial time through application of a max flow (min cut) procedure on some auxiliary network.*

Sketch of the proof. Let us first deal with (F12). Some vector Z, being given, separating (F12) means searching for $A \subseteq$ {0, 1, …, M+1} which violates (F12), and so for B = {-1, 0, 1, …, M+1} – A, such that $\Sigma_{j,k \in B} Z_{j,k}.E_{j,k} + \Sigma_{(j,k) \in \delta(B)} \Pi_{j,k} .Z_{j,k} < \Lambda = \Sigma_{j,k} Z_{j,k}.E_{j,k}$. (*)

In order to do it, we construct a network G^{Aux}, whose node set is{- 1, 0, 1, …, M+1, M+2) and whose arc set U^{Aux} may be written $U^{Aux} = U \cup Copy(U)$, with:

- U = {all pairs (j, k), j, k = -1, 0, …, M+1} such that $Z_{j,k} \neq 0$: such an arc $u = (j, k)$ is provided with a capacity $w_u = Z_{j,k}.(\Pi_{j,k} - E_{j,k})$;
- With any arc $e = (j, k)$ in U, we associate an arc $u = Copy(e) = (j, M+2)$: such an arc $u = Copy(e)$ $(j, M+2)$ is provided with a capacity $w_u = Z_{j,k}.E_{j,k}$. Then arc set $Copy(U)$ is the set of all arcs $Copy(e)$, $e \in U$.

In order to conclude, we only need to check that: Searching for B such that (*) holds is equivalent to searching for $B \subseteq$ {-1, 0, 1, …, M+1}, such that: $\Sigma_{u \in UAux, \text{ s.t } (origin(u) \in B) \wedge (destination(u) \notin B)} w_u < \Lambda$. It is known that, in case B exists, it may be retrieved through application of the standard *Min Cut* algorithm. We proceed the same way with (F12Bis). □

III. A First Surrogate Estimator approach for SVREP

Since **SVREP** is a bi-level problem the idea here is to focus on the routing level: If vehicle route Γ is fixed, which may be represented by a vector X as in **SVERP** ILP formulation, then the quality $Q\_SVREP(\Gamma)$ of Γ derives from the resolution of **SVREP** with X fixed. Though such a resolution can be performed in pseudo-polynomial time (see [4]), it remains too time costly to be achieved all along an iterative process. So we replace $Q\_SVREP(\Gamma)$ by a surrogate estimator $Q\_Surr(\Gamma)$, easy to compute and likely to provide us with sensitivity information. Let us discuss this estimator.

A. A Parametric P_Refuel Auxiliary Problem

Intuitively, a good route Γ requires little time and few energy. So we define the following auxiliary parametric problem **P_Refuel**(Γ, β), whose optimal value *Refuel*(Γ, β) is going to become our estimator $Q\_Surr(\Gamma)$:

P_Refuel(Γ, β): {β being a production price parameter, compute the refueling stations j in Γ, together with load values L_j in such a way that:

- Resulting values V_j meet constraints (F4,…, F7) of the **SVREP ILP** model;
- α.*TIME* + β.F*UEL* is the smallest possible, where *TIME* (*FUEL*) is the time (energy) required by Γ augmented with the refueling transactions}.

Explanation: Parameter β means a price for energy, Because of this interpretation of β as a price of energy, we consider the mean unit price $\beta_0 = (\Sigma_i Cost_i./R_i)/N$ as reference value.

B. Solving P_Refuel(Γ, β).

Given Γ and β. Let us define the following *refuel* acyclic graph *G_Refuel* = (N, A): (see Fig. 3)

- The node set N of *G_Refuel* is the set of all pairs (j, V),

$j = 0, 1, \ldots, M, 0 \leq V \leq C^{Veh}$.

- A *standard* arc from node (j, V) to node (j', V'), $j < j'$, means that the vehicle moves along Γ from j to j', while refueling immediately before arriving to j'. Then V' is equal to $C^{Veh} - E_{-1,j'}$. If we denote by $j°$ the predecessor of j' in Γ, then the cost of such an arc is equal to $\alpha.TIME(j, j') + \beta.FUEL(j, j')$, where:
  - $TIME(j, j')$ is the time required by moving from j to j' along Γ, augmented with the detour $\Delta_{-1,j'} + \Delta_{j°,-1} + p - \Delta_{j,j°}$.
  - $FUEL(j, j')$, is the energy required by moving from j to j' along Γ, augmented with the detour $E_{-1,j'} + E_{j°,-1} - E_{j,j°}$.

  An *end* arc from node (j, V) to node (M+1, $V' \geq E_0$), means that the vehicle moves from j to M+1 without performing any refueling transaction. We get the cost of such an arc as well as the relationship between V and V' in a straightforward way.

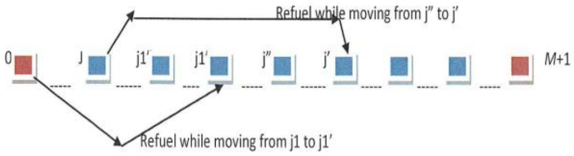

Figure 3: *The G_Refuel Acyclic Graph*.

Theorem 3: *For any β, solving **P_Refuel**(Γ, β) means computing a shortest path from (0, E_0) until (M+1, $V \geq E_0$) and can be solved in polynomial time. Besides, parametric resolution of **P_Refuel**(Γ, β) may be achieved in polynomial time, providing value Refuel(Γ, β) and related tour Γ for all values β.*

Sketch of the proof: The first part of above statement is pure routine. The second part is obtained by observing that the number of values V involved in the path search process does not exceed $M.(M+1)/2$. The last part is obtained by noticing that, for any node in the graph H, Bellman induction process makes appear a polynomial bounded number of frontier values β and related cost values. We get values L_j are obtained by making the vehicle arrive to the micro-plant with the smallest possible load. □

It comes that all values *Refuel*(Γ, β) may be obtained through a unique dynamic programming ***DP_Refuel***(β), together with related refueling stations.

C. Solving SVREP through the use of SURROGATE estimator Refuel(Γ, β)}.

Above sub-sections lead us to introduce the following parametric problem: **SVREP-SURROGATE**(β):{Compute Γ which minimizes *Refuel*(Γ, β)}.

This formulation suggests us to handle any **SVREP** instance according to the following parametric process:

First *Surrogate Estimator SVREP* Resolution Scheme:

1) Initialize (greedy VRP heuristic);
2) Perform a parametric resolution of ***P_Refuel***(Γ, β) and extract related set Ω of critical β values;
3) For every value β in Ω do

   Solve **SVREP_SURROGATE**(β) and get related tour Γ(β), together with vectors Z(β), X(β) and L(β);

   Solve **SVREP** while setting $Z = Z(\beta)$, $X = X(\beta)$ and $L = L(\beta)$ and using the exact algorithm of [4];
4) Keep the best **SVREP** solution obtained this way.

D. Two ways for solving SVREP_SURROGATE

Ler us first introduce the following ILP model:

ILP_SVREP_SURROGATE: {Compute {0, 1} valued vector $Z = (Z_{j,k}\ k \neq j$ in -1, 0,…, M+1: $Z_{j,k} = 1$ iff the vehicle moves from j to k) in such a way that:

- Constraints (F1, F2, F2Bis, F12, F12Bis) are satisfied;
- $\Sigma_{j,k} Z_{j,k}.(\alpha.T_{j,k} + \beta.E_{j,k})$ be the smallest possible}.

Theorem 4: *For any β, solving SVREP_SURROGATE(β) is equivalent to solving ILP_SVREP_SURROGATE.*

Sketch of the proof: F12 and F12Bis ensures us that Z computed this way defines a collection γ of sub-tours $\gamma_0, \gamma_1, \ldots, \gamma_s, .., \gamma_S$, such that (I1), (I2), (I3) of Theorem 1 are satisfied. This collection may be turned into a route Γ, optimal solution of **SVREP_SURROGATE**(β). □

So we may solve **SVREP_SURROGATE**(β) is to apply the branch and cut algorithm which derives from Theorem 1

Another approach (heuristic) consists in implementing a LNS (*Large Neighborhood Search*) algorithm to the route Γ:

***LNS_SVREP_SURROGATE(β)* Algorithmic Scheme**:

- Current Γ being given, apply ***DP_Refuel***(β) and decompose Γ into a collection γ of sub-tours $\gamma_0, \gamma_1, \ldots, \gamma_s, .., \gamma_S$, which meets (I1, I2, I3);
- Apply an *Insertion/Removal* operator, (see Fig. 4):
  - *Removal* step: It removes a set R of q stations (between 5% to 30% of stations in 1,…, M) from Γ. Several strategies are tried:*Random*; *Poor*: Focus on poorly inserted station; *Shaw*: Focus on stations close to each others.
  - *Insertion* step: Reinserts stations of R while making in such a way that (I1,…, I3) remain satisfied. Best insertion strategy is applied.
  - *Permutation* step: It randomly reorders sub-tours $\gamma_2, \ldots, \gamma_S$ in the decomposition γ.

- Retrieve resulting route Γ* from resulting collection γ*, applies ***DP_Refuel***(β)and get the value *Refuel*(Γ*, β).
- If *Refuel*(Γ*, β) is smaller than current value *Refuel*(Γ, β) then replace Γ by Γ*and keep on else try (no more that *Trial* times) again above *Insertion/Removal* operator or decide to stop,

Taken as a whole, the ***LNS_SVREP-Surrogate*** algorithm works as a GRASP process: It randomly builds (through some greedy VRP algorithm) initial routes $\Gamma_1,\ldots,\Gamma_{Init}$, and applies above descent process to every Γ_k, .

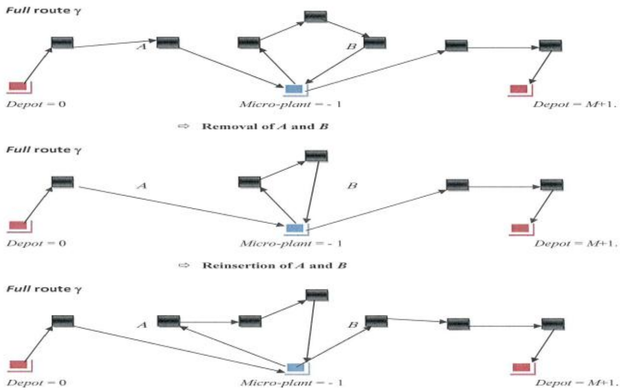

Figure 4: *Insertion/Removal Operator*.

IV. A MACHINE LEARNING ORIENTED APPROACH

We choose here to learn the optimal value *Fuel_Cost*(γ) of the problem **Fuel_Cost**(γ) induced by imposing *Z* and *X* and *L* in the **SVREP** model. In order to do it, we design a neural network *N_Fuel_Cost* which we implement with the *TensorFlow* software and which we train with a large number (6000) of **Fuel_Cost**(γ). Our surrogate problem becomes: **SVREP-ML_SURROGATE**{Compute γ which minimizes *N_Fuel_Cost*(γ)}. We derive:

ML_Surrogate_ *SVREP* Resolution Scheme:

1) Initialize *Init* collections γ;
2) For every γ obtained this way do
   a) Apply the previous LNS descent scheme in order to minimize *N_Fuel_Cost*(γ);
   b) Solve **Fuel_Cost**(γ) while using the dynamic programming algorithm described in [4];
3) Keep the best **SVREP** solution obtained this way.

Description of the *N_Fuel_Cost* neural network.

- The input layer of *N_Fuel_Cost* is 2-sided:
  - On one side, we consider the *production* data, which means production rates R_i and production costs $Cost_i$, $i = 1,\ldots, N$, together with capacity C^{Prod} , fixed activation cost *C_Act* and initial load H_0. In order to control the size of the network, we compress those data as soon as $Q.20 \leq N < (Q+1),20$, by merging consecutive periods $i = Q.q,\ldots, (Q+1),i - 1$ into one macro-period with *ad hoc* production cost and rate.
  - On another side, we consider the *vehicle* data related to γ, which are the number *S* of refueling transaction, the time (periods) windows for those transactions, and related loads $\mu_1,\ldots\mu_S$. So we fix an upper bound equal to 10, and implement a fusion mechanism for the case $S \geq 10$ in order to fix size of those data.
- The 2 first hidden layers involve synaptic arcs which connects periods and transactions to a small number of neighbor periods. Those arcs transport impulsions which take rational values in [0, 1] and so tend to emulate Boolean calculus. They are filtered by standard sigmoid functions and they aim at making appear, as outputs of the third layers:
  - the probability that production takes place at period (macro-period) *i* in case of the *left* side (see Fig. 5) of the network;
  - the probability that the last refueling transaction takes place at period *i* in case of the *right* (according to Fig. 5) side of the network.
  - They involve synaptic arcs which connects periods and transactions to a small number of neighbor periods.
- The last hidden layer derives (*left* side) the production cost from the above mentioned probability vector and (*right* side) the expected period for the latest refueling transaction, and involve RELU activation functions (smooth functions, null when the input is negative, and linear when the inputs are positive).
- Finally the output layer retrieves the final cost $\Sigma_{\,i\,=\,1,\ldots,N}\ (C\_Act.y_i + Cost_i.z_i) + \alpha.T_{M\,+\,1}$ from both quantities computed by the last hidden layer.

The following figure 5 illustrates the construction of *N_Fuel_Cost*, which contains 986 synaptic arcs.

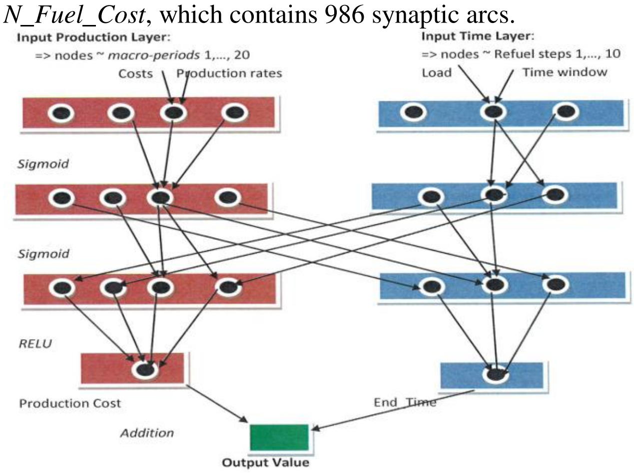

Figure 5: *The Neural Network N_Fuel_Cost*.

V. NUMERICAL EXPERIMENTS

Goal: Evaluate the behavior of surrogate estimators into the resolution of the **SVREP** problem.

Technical Context: We use a processor IntelCore i56700@3.20 GHz, with 16 Go RAM, together with a C++ compiler and libraries CPLEX12 (for ILP models) and TensorFlow/Keras (for machine lerning).

Instances: As for the vehicle part, we generate *M* (between 6 and 50) stations as points with integral coordinates in a 2-dimensional square, and derive Δ, *E* values according to a rounding of the Euclidean and Manhattan distances. We introduce a parameter $Q = V\_TSP/C^{Veh}$ between 3 and 10, where *V_TSP* is an estimation of the energy required by the production, we split the period set into *K* = 2, …, 5 intervals, corresponding to different qualities of the weather and randomly generate rates R_i for every such interval.

Activation cost *C_Act* is generated in such a way that expected activation cost be equal to *R* time the expected production cost, with *R* between 0.2 and 2. The capacity of the micro-plant is generated as equal to $H.C^{Veh}$, with *H* between 1 and 3. Finally, mean production rates are updated in such a way that the maximal production capacity of the micro-plant does not exceed *S.V_TSP*, where *S* is a control parameter between 2 and 4. We use here 10 instances:

TABLE I. CHARACTERISTICS OF THE INSTANCES

Instance	*N*	*M*	*Q*	*K*	*R*	*S*	*H*	*α*
1	45	6	3	3	1	3	2	8
2	45	6	3	3	0.3	2	1	8.5
3	30	8	3	5	0.5	2	3	11
4	40	8	3	2	0.3	2	3	5
5	140	10	3	4	0.25	4	1	6.5
6	140	10	5	4	0.25	4	1	6.5
7	120	15	6	3	2	2.5	2	8
8	120	15	6	5	2	2.5	2	15.5
9	200	20	9	5	0.7	3.5	1	14
10	200	20	9	5	0.25	2	2	18

Outputs: For every instance:

- We apply (Table II) the general **SVREP** ILP model, and get (in less than 1 CPU h), a lower bound *LB_G*, an upper bound *UB_G*, CPU time *T_G*, the value *Relax* of the rational relaxation at the root, and the number *C_G* of *strong no_sub_tour* cuts which were generated.
- We solve (Table 3) **ILP_SVREP_SURROGATE**(β) with β = (Σ $_i$ $Cost_i./R_i$)/*N*, while using the branch and cut algorithm of III.*D*. We keep memory of resulting lower and upper bounds *LB_S1* and *UB_S1*, as well as CPU time *T_S1* and the number *C_S1* of generated cuts (F12, F12Bis). We also try to **ILP_SVREP_SURROGATE** without using *strong no_sub-tour* constraints and get lower and upper bounds *LB_W* and *UB_W*.
- We apply (Table 4) ***LNS_SVREP_SURROGATE***($β_0$) with *Trial* = 2.*M*; *Init* = 10; τ = 15% to **SVREP_SURROGATE,** and keep resulting value *V_Heur*, CPU time *T_Heur*, as well as the number *Try* of trials per iterations the mean number *Iter* of iterations of performed during the descent process. We do the same thing (Table 5) with the procedure **SVREP_ML_SURROGATE**, which involves machine learning. We keep resulting value *V_ML*, CPU time *T_ML*, as well as values *Try* and *Iter*.
- We apply (Table 6) the global First SVREP resolution scheme of Section III.*C* with β = $β_0$ and denote by *W_ILP* the value of **ILP_SVREP_SURROGATE** and by *W_Heur* the value obtained with ***LNS_SVREP_SURROGATE***. We do the same with the resolution scheme involving machine learning and denote by *W_ML* resulting value, and with 5 values β ranging from $β_0$/4 until 4. $β_0$. *W_ILP5* and *W_Heur5* denote the values obtained by applying the global scheme of Section III.*C*, with respectively ***ILP_VREP_SURROGATE*** and ***LNS_SVREP_SURROGATE***.

Those results may be summarized into the following tables:

TABLE 2: BEHAVIOR OF THE GLOBAL ILP MODEL

Inst.	*LB_G*	*UB_G*	*T_G*	*Relax*	*C_G*
1	735.3	735.3	628.4	344.5	6
2	951.3	951.3	265.3	312.0	25
3	709.2	709.2	159.7	413.5	72
4	322.1	508.2	3600	208.7	146
5	969.0	969.0	1819.3	611.7	188
6	1153.4	1486.3	3600	764.5	286
7	1972.7	3065.5	3600	1652.3	330
8	3386.9	4211.8	3600	3008.2	518
9	4923.0	9594.3	3600	4021.2	646
10	5956.6	8560.3	3600	5365.4	601

Comments: As expected, the global ILP model of Section II is in trouble, even on small instances. Still Table 6 will make appear the upper bound *UB_G* is close to optimality.

TABLE 3: BEHAVIOR OF ILP_ SVREP_SURROGATE

Inst.	*LB_S1*	*UB_S1*	*T_S1*	*C_S1*	*LB_W*	*UB_W*
1	684.4	684.4	0.02	2	684.4	684.4
2	835.3	835.3	0.04	6	835.3	835.3
3	705.9	705.9	0.07	7	705.9	705.9
4	474.6	474.6	0.85	50	474.6	474.6
5	916.2	916.2	4.7	46	762.1	916.2
6	1386.3	1386.3	6.97	143	1038.0	1387.0
7	2891.7	2891.7	243.7	296	2220.6	2902.8
8	4020.3	4020.3	522.05	412	2808.2	4022.6
9	9023.6	9077.2	3600	468	3702.5	9077.2
10	7773.0	7773.9	1602.5	286	2891.9	7812.3

Comments: *Strong no_sub_tour* constraints significantly increase our ability to manage **ILP_SVREP_SURROGATE**. Also, we notice that the optimal value of **ILP_SVREP_SURROGATE** provides us, when β = $β_0$ = mean unitary production cost, with a kind of approximation (10% in average) of the optimal **SVREP** value.

TABLE 4:
BEHAVIOR OF THE HEURISTICFOR SVREP_SURROGATE

Inst.	*V_Heur*	*T-Heur*	*Iter*	*Try*
1	684.4	0.01	4.4	8.1
2	835.3	0.01	2.9	6.3
3	705.9	0.01	5.3	4.9
4	474.6	0.02	7.5	6.6
5	916.2	0.02	4.4	10.3
6	1386.3	0.04	5.2	10.6
7	2891.7	0.05	10.3	6.5
8	4047.2	0.07	6.6	9.8
9	9077.2	0.17	11.3	14.5
10	7802.0	0.15	16.5	22.4

Comments: Our Removal/Insertion heuristic provides very good approximations under low computational costs.

TABLE 5.
BEHAVIOR OF THE HEURISTIC FOR SVREP_ML__SURROGATE

Inst.	*V_ML*	*T-ML*	*Iter*	*Try*
1	786.2	0.01	7.2	4.1
2	994.4	0.01	3.7	7.4
3	700.5	0.01	7.1	8.0
4	543.6	0.01	6.4	5.4
5	936.3	0.02	10.6	7.5
6	1627.8	0.02	10.8	9.2
7	3280.9	0.02	8.1	13.1
8	4458.3	0.02	12.7	10.8
9	1046.0	0.03	18.5	11.9
10	9322.7	0.03	10.0	13.5

Comments: The gap between the optimal SVREP value and its approximation through the neural network *N_Fuel_Cost* is rather important, in average around 7%, with a peak at 10%.

TABLE 6: SVREP VALUE OBTAINED WITH $\beta = \beta_0$, AND 5 VALUES β.

Inst.	*W_ILP*	*W_Heur*	*W_ML*	*W_ILP5*	*W_Heur5*
1	735.3	735.3	735.3	735.3	735.3
2	956.3	956.3	951.3	956.3	956.3
3	709.2	709.2	709.2	709.2	709.2
4	535.2	535.2	568.2	508.2	508.2
5	969.2	969.2	990.0	969.0	969.0
6	1493.4	1492.3	1566.3	1486.3	1486.3
7	3003.5	3025.7	3065.5	3003.5	3025.7
8	4274.4	4211.8	4388.0	4211.8	4211.8
9	9342.5	9397.5	9794.7	9248.4	9365.0
10	8473.1	8465.1	8560.3	8420.8	8465.1

Comments: Computing tour Γ while estimating its quality as the optimal value of **SVREP_SURROGATE**(β) is efficient, with a low sensitivity to parameter β. The second approach involving machine learning, though more generic and better fitted to the management of uncertainty, unfortunately happens here to be less efficient.

VI. CONCLUSION

We dealt here with a complex energy management problem, while shortcutting the production sub-problem. We tried 2 approaches: the first consisted in estimating the quality of a routing strategy as the parametric increase of energy and time consumption by the vehicle; the second one consisted in learning, through a neural network, the value of the production sub-problem. Numerical experiments made appear, at least in our case, a better efficiency of the first approach. Still, because the machine learning oriented approach is far more generic, it would be worthwhile to go deeper with it, and make the value learned by a neural network more sensitive to the application of local search operators. It could also help us in managing this uncertainty through a joint estimation of both the expected value of a production cost and of the risk of failing in meeting production requirements according to this cost.

ACKNOWLEDGMENT

Present work was funded by French ANR Labex IMOBS3, and by PGMO Program..

REFERENCES

[1] Y. Adulyasak, J.F. Cordeau, R.Jans :The production routing problem: A review. *Computers & Operations Research*, 55, p 141-152, (2015). doi:10.1016/J.COR.2014.01.011
[2] A.Albrecht, P. Pudney: Pickup and delivery with a solar-recharged vehicle. *Ph.D. thesis Australian Society for O.R* (2013).
[3] C.Artigues, E.Hébrard, A.Quilliot, H.Toussaint: "Models and algorithms for natural disaster evacuation problems". *Proceedings of the 2019 FEDCSIS WCO Conference*, p 143-146, (2019). DOI: http://dx.doi.org/10.15439/978-83-952357-8-8
[4] F.Bendali, J.Mailfert, E.Mole-Kamga, A.Quilliot, H.Toussaint : Pipe-lining dynamic programming procsses in order to synchronize energy production and consumption, *2020 FEDCSIS WCO Conf.*, p 303-306, (2020). DOI: http://dx.doi.org/10.15439/978-83-955416-7-4.
[5] A.Caprara, M.Carvalho, A.Lodi, G.J.Woeinger: A study on the computational complexity of the bilevel knapsack problem; *SIAM Journal on Optimization* 24 (2), p 823-838, (2014).
[6] L.Chen, G.Zhang: Approximation algorithms for a bi-level Knapsack problem; *Theoretical Computer Sciences* 497, p 1-12, (2013).
[7] T. Erdelic, T. Caric: (2019). A survey on the electric vehicle routing problem: Variants and solution approaches. *Journal of Advanced Transportation*, (2019). doi:10.1155/2019/5075671.
[8] S.Fidanova, O.Roeva, M.Ganzha: " Ant colony optimization algorithm for fuzzy transport modelling ". *Proceedings of the 2020 FEDCSIS WCO Conference*, p 237-240, (2020). DOI: http://dx.doi.org/10.15439/978-83-955416-7-4
[9] C.Grimes, O.Varghese, S.Ranjan. Light, water, hydrogen:Solar generation by photoelectrolysis. *Springer-Verlag US*, (2008).
[10] S.Irani, K.Pruhs: Algorithmic problems in power management. *SIGACT News*, 36, 2, p 63-76, (2003).
[11] C.Koc, O.Jabali, J.Mendoza, G.Laporte: The electric vehicle routing problem with shared charging stations. *ITOR*, 26 , p 1211-1243, (2019). doi:https://doi.org/10.1111/itor.12620.
[12] M.Krzyszton: "Adapative supervison: method of reinforcement learning fault elimination by application of supervised learning". *Proceedings of the 2018 FEDCSIS AI Conference*, p 139-149, (2018). DOI: http://dx.doi.org/10.15439/978-83-949419-5-6
[13] G. Macrina, L.D. Pugliese, F. Guerriero: (2020). The green-vehicle routing problem: A survey. In *Modeling and Optimization in Green Logistics, Cham: Springer International Publishing*. p. 1-26, (2020). doi:10.1007/978-3-030-45308-4_1
[14] K.Stoilova, T.Stoilov: "Bi-level optimization application for urban traffic management". *2020 FEDCSIS WCO Conference*, p 327-336, (2020). DOI: http://dx.doi.org/10.15439/978-83-949419-5-6
[15] J. Wojtuziak, T. Warden, O. Herzog: "Machine learning in agent based stochastic simulation: Evaluation in transportation logistics"; *Computer and Mathematics with Applications 64, p 3658-3665, (2012).* https://doi.org/10.1016/j.camwa.2012.01.079.