Proceedings of the 20th Conference on Computer Science and Intelligence Systems (FedCSIS) pp. 577–586 DOI: 10.15439/2025F7738 ISSN 2300-5963 ACSIS, Vol. 43

Vehicle Routing under Complex Access-to-Energy Constraints

Alain Quilliot and Hélène Toussaint
LIMOS Lab.
UCA, CNRS and EMSE
Clermont-Ferrand, France
Email: alain.quilliot@uca.fr

***Abstract*—Photovoltaic platforms enable a single agent to simultaneously act as both a producer and a consumer of power, facilitating *self-consumption* strategies. This trend aligns with the goal of reducing CO2 emissions and is poised to significantly transform the structure of energy markets. It also introduces specific challenges—both tactical (e.g., pricing) and operational (e.g., routing, scheduling)—related to synchronizing energy production with consumption. In this work, we address the problem of efficiently routing a fleet of electric autonomous vehicles (EAVs), using energy that is either produced by a photovoltaic platform or purchased from the general power grid. We propose an exact Mixed-Integer Linear Programming (MILP) formulation of the problem, along with a heuristic approach that approximates the power production component of the model using surrogate representations.**

I. Introduction

RENEWABLE energy sources (e.g., photovoltaic, wind, hydrogen) are driving the emergence of local, in situ producers who simultaneously consume energy—such as factories and farms. In this context, the energy production-consumption process becomes partially endogenous, forming a closed-loop system under the *self-consumption* paradigm [15]. This paradigm is expected to have a significant impact on energy economics [11], [14]. It raises various challenges, ranging from operational issues—related to the scheduling and synchronization of production and consumption—to tactical and strategic decisions, such as pricing, storage, and interaction with the central grid. In scenarios where the decentralized producer-consumer entity is a consortium of independent agents, each with its own schedule and shared access to the production platform, fairly distributing costs and benefits raises the *cooperative* issue.

We consider here the problem of routing a fleet of electric autonomous vehicles (EAVs), powered either by photovoltaic-generated electricity—available at time-dependent rates—or by electricity purchased from the general grid at time-dependent prices. This *Vehicle Routing under Energy Production Costs* (**VR_EPC**) problem requires fleet managers to synchronize vehicle activity with energy production and procurement, while taking into account the limited storage capacities of both the platform and the vehicles.

A. State of the Art

Numerous studies have explored the routing of electric vehicles with the aim of minimizing energy costs or addressing environmental concerns, such as in the *Green Vehicle Routing Problem* (Green VRP), the *Pollution Routing Problem*, and the *Hybrid Vehicle Routing Problem*: [5], [16], [21],[19]). They most often proposed models involving refueling transactions subject to time windows or shared access constraints. For example:

- Erdoğan and al. [6] introduced the *Green VRP*, which minimizes both total travel distance and the number of refueling transactions.
- Franceschetti and al. [8] proposed a *Pollution-Routing* model, incorporating time-dependent costs and an objective function accounting for driver wages and fuel costs.
- Koç and al. [12] focused on determining where and how much vehicles should be recharged, considering access-to-energy constraints.
- In [21], authors addressed customer sequencing, while explicitly accounting for the recharge capacity of the charging stations.

Some authors also dealt with CO2 emitting vehicles, with the purpose of controlling related CO2 emission:

- Raylan and al. [17] minimized emissions resulting from both routing and load-related factors.
- Sachenbacher and al. [18] accounted for time-dependent factors and refueling schemes, adapting shortest path algorithms accordingly.
- Kuo [13] explored speed control to balance energy consumption, distance, and time.
- Schneider and al. [19] developed heuristics for a Vehicle Routing problem with time windows and safety-specific refueling constraints.
- Lajunen [14] performed simulation-based comparisons of energy savings across urban shuttle and bus configurations.

Despite these contributions, few studies have jointly managed both energy production and consumption. Addressing this challenge requires integrating heterogeneous routing and scheduling processes while accounting for storage constraints (see [2], [3], [1]). Some formulations borrow from the *Lot*

Sizing framework or its multi-level scheduling variants (see [4], [10], [20]), but these models involve highly heterogeneous variables and constraints, often leading to poor performance of linear relaxations. Moreover, they implicitly rely on the existence of a single decider provided with full control and full information with respect to the system, inducing an assumption which does not fit most real life contexts. Alternatives such as constraint programming and bi-level formulations may help coordinate interactions across decision levels. Nonetheless, whatever be the point of view (centralized or collaborative), designing efficient solution algorithms remains a major challenge. A recent trend—facilitated by machine learning—is to approximate complex sub-models using surrogate constraints or cost functions.

B. Main Achievements

Since our **VR_EPC** problem is likely to involve in practice collaborative features making it difficult to suppose the existence of a global decider, we adopt here the point of view of the EAV. Also, we skip the uncertainty issue, while supposing that the behavior of our system, and more specifically of the photovoltaic platform, is deterministic. Then we propose two solution approaches for the **VR_EPC** problem.

- The first one is an exact MILP formulation that considers the *vehicle* variables as master variables and includes specific *Recharge Decomposition* constraints requiring the application of a specific separation procedure, which works in polynomial time. This MILP model is solved using a branch-and-cut method. However, this approach becomes computationally inefficient for large instances.
- So we also design 2 heuristic algorithms, which makes the EAV decide under a partial knowledge of the behavior of the photovoltaic platform, while using an approximation of the energy production model. The resulting algorithms pave the way for more efficient handling of uncertainty and collaboration challenges, particularly in scenarios where the photovoltaic (PV) platform and the vehicles are run by distinct players, who do not fully share neither the same goals nor the access to information.

The paper is organized as follows. Section II provides a detailed description of the *Vehicle Routing under Energy Production Costs* problem (**VR_EPC**). In Section III, we present the MILP formulation and its resolution using a branch-and-cut approach. Section IV offers a statistical analysis that provides the basis for the development and testing of 2 local search heuristic. We conclude briefly in Section V.

II. The Vehicle Routing under Energy Production Costs Problem

We consider a photovoltaic (PV) micro-plant, referred to as **PVP**, along with a fleet of electric autonomous vehicles (EAV) in charge of visiting a set of stations. These two components interact through recharge transactions when the vehicles return to the micro-plant to recharge (see Fig. 1). For the sake of simplicity, we limit our study to a single vehicle that must complete a *Traveling Salesman*-type tour, and we assume the system operates in a deterministic manner. Thus, the main components of our target system are:

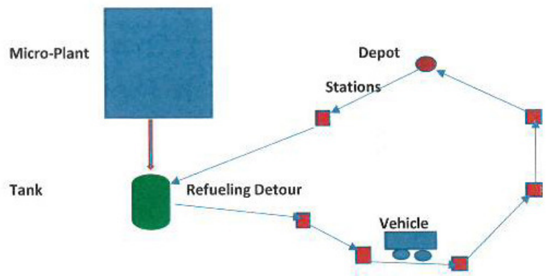

Fig. 1. The PV-Micro-Plant Interacting with a Vehicle Route

- **A Photo-Voltaic Micro-Plant PVP**
The time horizon of **PVP** is divided into N periods $i = 0, \ldots, N-1$, all with a same duration p. Thus the starting timestamp of period i is equal to $p \cdot i$ and its ending timestamp equal to $p \cdot (i+1)$. During a period i, **PVP** is expected to produce R_i energy units. It may also buy an additional amount y_i of power, that cannot exceed a *charge* capacity denoted by C^{Ch}. The cost of buying such an additional energy amount y_i depends on both i and y_i and may be written $\Phi_i(y)$, where Φ_i is a piecewise linear increasing convex function. The convexity of this cost function Φ_i expresses the fact that marginal power purchase prices are usually increasing. **PVP** is provided with a macro-battery, with storage capacity $C^{PVP} \geq C^{Ch}$ and initial load H_0^{PVP}. It must manage its purchase operations in a way which makes it able to meet vehicle's demand without exceeding storage and charge capacities, while ending the process with a load at least equal to H_0^{PVP} and minimizing related purchase costs.
- **An Electric Autonomous Vehicle EVeh and a Set of Stations J**
This vehicle, denoted by **EVeh**, is initially located at a specific station *Depot*. It must visit and service, within the time horizon $[0, N.p]$, a set $\mathbf{J} = \{1, \ldots, M\}$ of stations according to a TSP (*Traveling Salesman*) route Γ, before coming back to *Depot*. Moving from a station j to a station k requires $E_{j,k}$ energy units and $T_{j,k}$ time units, service times being included into the $T_{j,k}$ values. We suppose that vectors T, θ, θ^* $(E, \epsilon, \epsilon^*)$ define a distance on the set defined by $Depot$, **PVP** and **J**. **EVeh** is provided with a battery, with storage capacity C^{Veh} and initial load H_0^{Veh}. It must end with a load at least equal to H_0^{Veh} while minimizing the time when it returns to $Depot$. $Depot$ is considered as a station, identified as $Depot = 0$ at the beginning of the process and $Depot = M+1$ at the end. By the same way, **PVP** is also considered as a station, identified as **PVP** $= -1$

- **Recharge Transactions**
Because of its storage capacity C^{Veh} and the constraint about its load at the end of the process, **EVeh** must periodically move to **PVP** in order to recharge. Such a *recharge transaction* can be achieved in a single period i and takes place between two stations j, k consecutive according to the route Γ: **EVeh** moves from j to **PVP** in order to arrive before time $p \cdot i$, receives some amount $m \leq Inf(C^{Ch}, C^{Veh})$ of energy, and starts again from **PVP** at time $p{\cdot}(i{+}1)$ in order to reach k. Since the vehicle is a kind of robot, this recharge transaction, denoted by $\omega = (i, j, k, m)$, involves some human resource and so it induces a cost Ψ_i. This human resource cost depends on i and is independent on m. For the sake of safety, purchasing power is forbidden during the period i when the recharge transaction ω takes place. Moving from j to **PVP** requires ϵ_j energy units and θ_j additional time, while moving from **PVP** to k requires ϵ^*_k energy units and θ^*_k additional time. A recharge transaction may impose **EVeh** to wait at **PVP** until the beginning of period i in case it arrives before time $p{\cdot}i$. If we denote by τ_j the time when **EVeh** arrives at j, by V_j^{Veh} its energy load and by V_i^{PVP} the energy load of the PV-plant at the beginning of period i, then we must have:
  - $p \cdot i \geq \tau_j + \theta_j$; $\tau_k = p \cdot (i+1) + \theta^*_k$;
  - $V_{i+1}^{MP} = V_i^P - m \geq 0$;
  - The load of **EVeh** at the end of period i must be equal to $V_j^{Veh} - \epsilon_j + m \leq C^{Veh}$;
  - The load of **EVeh** at the beginning of period i must be equal to $V_j^{Veh} - \epsilon_j \geq 0$.

Since the system described this way involves two players, each provided with its own performance criterion, resulting problem might be set according to the bi-objective format. Yet, our focus here is on the algorithms. Therefore, we introduce a *time versus money* coefficient α, and formulate **VR_EPC** as a mono-objective problem. By the same way, since we tend here to adopt the point of view of the vehicle manager **EVeh**, we set related model as a bi-level one.

VR_EPC: *Vehicle Routing with Energy Production Costs* : $\{$Compute the route Γ followed by **EVeh** together with the recharge transactions that link **PVP** and **EVeh** in such a way that:

- All stations are visited once within the time horizon $[0, p \cdot N]$.
- Vehicle storage capacity constraints and energy requirements are satisfied.
- Some extended cost $PCost{+}\alpha.Veh\_Time$ is minimized, where $PCost$ means the optimal cost value induced by the **PVP** sub-problem consisting in deciding the purchase vector $y = (y_i, i = 0, \ldots, N-1)$ in such a way that:
  - It meets **PVP** storage and charge capacities;
  - It meets the needs related to the recharge transactions while allowing **PVP** to end with at least as much energy as when it started;
  - $PCost = \sum_i (\Psi_i \cdot \delta_i + \Phi_i(y_i)).\}$

TABLE I
PRODUCTION RATES, UNIT PURCHASE PRICES AND RECHARGE COSTS

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13
R_i	2	2	0	0	4	4	4	0	0	0	0	2	2	0
P_i	2	3	5	5	1	1	1	5	5	5	5	2	2	5
Ψ_i	1	1	1	1	2	2	1	1	1	1	1	1	2	2

An Example: Let us suppose that:

- M = 5, $N = 14$, $p = 2$; $\alpha = 1$;
- $C^{Veh} = 16, H^{Veh} = 9, C^{PVP} = 20, H^{PVP} = 5, C^{Ch} = 15$;
- Expected productions $R_i, i = 0, \ldots, N-1$ come as in Table I;
- For any period i, function Φ_i comes as: $\Phi_i(y) = P_i \cdot y$, y denoting the additional energy bought at period i and prices P_i come according to table I.

Let us consider a route $\Gamma = \{0, 1, 2, 3, 4, 5, 6\}$ together with time and energy requirements given according to Figure 2:

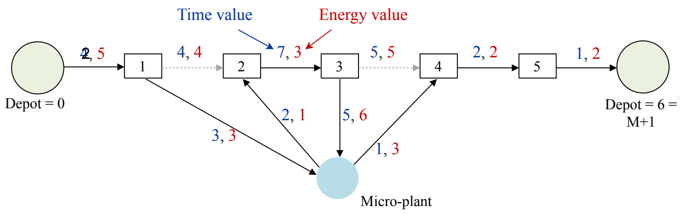

Fig. 2. The Route Γ, together with the time and energy requirements

Then we get a feasible **VR_EPC** solution involving Γ by specifying the recharge transactions as follows:

- A first recharge transaction occurs at period 3, while **EVeh** is moving from station 1 to station 2. Related energy amount m is equal to 10.
- Another recharge transaction occurs at period 11, while **EVeh** is moving from station 3 to station 4. Related energy amount m is equal to 15.
- **PVP** buys 3 energy units at period 0 and 3 energy units at period 1. Those operations induce a purchase cost equal to $3 \cdot 3 + 2 \cdot 3 = 15$.
- The 2 recharge transactions induce a cost equal to 2, which makes the $PCost$ value equal to $15{+}2 = 17$, while the value Veh_Time is equal to $14 \cdot 2 = 28$, inducing a generalized cost value $\alpha \cdot PCost + \beta \cdot Veh\_Time$ equal to $1 \cdot 17 + 1 \cdot 28 = 45$.

III. A MILP MODEL

Though **VR_EPC** is too complex for an exact handling of large size instances, casting it into the MILP format is important, not only because resulting **MILP_VR_EPC** model will provide us with benchmark results in the case of small instances, but also because it will contain a structural analysis of the problem.

A. The Variables of the *MILP_VR_EPC* model

Getting a MILP formulation first requires identifying the variables. Since we adopt here the point of view of the vehicle manager, our master variables are going to be 2 $\{0,1\}$-valued vectors $Z = (Z_{j,k}, k \neq j \in \{-1,0,\ldots,M+1\})$ and $X = (X_{j,k}, k \neq j \in \{0,\ldots,M+1\})$, that respectively describe the routes followed by the vehicle with and without the detours induced by the recharging transactions. Those main *vehicle* variables are going to be completed by auxiliary variables related to the time values when the vehicle visits the stations and to its energy load at this time. We also need *PV-Plant* variables that allow us to describe the evolution along the periods of the power load stored inside the **PVP** macro-battery. Finally, we need *Recharge Transaction* variables, linking the respective trajectories of the vehicle **EVeh** and the PV-Plant **PVP**. This leads us to introduce the following variables:

- **Vehicle** variables:
  - $\{0,1\}$-valued variables $Z_{j,k}, k \neq j \in \{-1,0,\ldots,M+1\}$: $Z_{j,k} = 1$ iff the vehicle moves from j to k: -1 denotes here the micro-plant **PVP**.
  - $\{0,1\}$-valued variables $X_{j,k}, k \neq j \in \{0,\ldots,M+1\}$: $X_{j,k} = 1$ iff the vehicle moves either from j to k or from j to **PVP**, and next from **PVP** to k.
  - Non negative variables $L_j^{Veh}, j = 0,\ldots,M$: L_j^{MP} means the energy transferred to **EVeh** just after j, in case $Z_{j,-1} = 1$;
  - Non negative variables$V_j^{Veh}, j = 0,\ldots,M+1$: V_j^{Veh} means the energy stored by **EVeh** when it arrives at j;
  - Non negative variables $\tau_j, j = 0,\ldots,M+1$: τ_j means the time when **EVeh** arrives at j;
  - Non negative variables $\tau_j^*, j = 0,\ldots,M+1$: If $Z_{j,-1} = 1$ then τ_j^* means the time when **EVeh** starts recharging after station j in case $Z_{j,-1} = 1$.
- **PV-Plant** variables:
  - Non negative variables $y_i, i = 0,\ldots,N-1$: y_i means the energy amount bought by the PV-Plant's during period i;
  - $\{0,1\}$-valued variables $\delta_i, i = 0,\ldots,N-1$: $\delta_i = 1$ means that some recharge transaction takes place at i;
  - Non negative variables $V_i^{PVP}, i = 0,\ldots,N-1,N$: V_i^{PVP} means the energy stored by **PVP** at the beginning of i;
  - Non negative variables $L_i^{PVP}, i = 0,\ldots,N-1$: L_i^{PVP} means the energy transferred from **PVP** period i in case in case $\delta_i = 1$.
- **Recharge Transaction** variables:
  - $\{0,1\}$-valued variables $U_{i,j}, i = 0,\ldots,N-1, j = 0,\ldots,M$: $U_{i,j} = 1$ means that some recharge transactions occurs at period i that involves station j and its sucessor in Γ;
  - Non negative variables $m_{i,j}, i = 0,\ldots,N-1, j = 0,\ldots,M$: $m_{i,j}$ means related amount of energy.

B. Structural Recharge Decomposition Vehicle Constraints

Z and X describe the full route followed by **EVeh**. They must clearly meet the following standard *vehicle routing* constraints:

- $Z_{M+1,0} = 1; \quad \forall j : Z_{j,j} = 0;$ (VR1)
- $\forall j = 0,\ldots,M+1$: $\sum_{k=-1,\ldots,M+1} Z_{j,k} = 1 = \sum_{k=-1,\ldots,M+1} Z_{k,j}$; (VR2)
- $\sum_{j=0,\ldots,M+1} Z_{j,-1} = \sum_{j=0,\ldots,M+1} Z_{-1,j} \geq 1$; (VR3)
- $\forall j,k \in \{0,\ldots,M+1\}$: $X_{j,k} \geq Z_{j,k}$; (VR4)
- $\forall j \in \{0,\ldots,M+1\}$: $\sum_{k=0,1,\ldots,M+1} X_{j,k} = 1 = \sum_{k=0,1,\ldots,M+1} X_{k,j}$; (VR5)

Yet we see that (VR1, ..., VR5) are not enough to ensure that we may interpret Z, X as a route followed by **EVeh**, feasible with respect to the energy requirements. In fact, they are not enough to prevent us from the existence of standard *Traveling Salesman sub-tours*. So we must reinforce them. We do it by noticing that if **EVeh** spends W energy while moving inside or at the boarder of some station subset A which does contain the micro-plant, then it must move at least $\lceil \frac{W}{C^{Veh}} \rceil$ times towards **PVP** in order to refuel. In order to formalize this, we must introduce some additional notations:

- For any such a subset A of $\{-1,0,\ldots,M,M+1\}$:
  - $Cl(A) = \{(j,k)$ s.t at least j or k is in $A\}$;
  - $\delta(A) = \{(j,k)$, s.t $j \notin A$ and $k \in A\}$.
- For any (j,k):
  - $\Pi_{j,k} = E_0$ 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.

Then we become able to derive the following *Recharge Decomposition* constraints that reinforce (VR1, ..., VR5):

- For any $A \subseteq \{0,\ldots,M+1\}$, $\sum_{(j,k)\in\delta(A)} \Pi_{j,k} \cdot Z_{j,k} \geq \sum_{(j,k)\in Cl(A)} E_{j,k} \cdot Z_{j,k}$ (VR6)
- For any $A \subseteq \{0,\ldots,M+1\}$, $\sum_{(j,k)\in\delta(J-A)} \Pi_{j,k}^* \cdot Z_{j,k} \geq \sum_{(j,k)\in Cl(A)} E_{j,k} \cdot Z_{j,k}$ (VR6-Bis)

The following Lemma 1 provides us with the structural meaning of these constraints: It tells us that adding them to (VR1, ..., VR5) ensures us that (Z, X) are going to describe a vehicle route that excludes *sub-tours* and can be decomposed into a sequence of sub-routes ($Depot \rightarrow$ **PVP**), (**PVP** $\rightarrow$ **PVP**) and (**PVP** $\rightarrow Depot$), all feasible with respect to the energy requirements.

Lemma 1: *The VRP constraints (VR1, ..., VR6-Bis) hold if and only if the arcs (j,k) such that $Z_{j,k} = 1$ define a collection γ of sub-tours $\gamma_s, s = 0,\ldots,S$ such that:*

- *γ_0 starts from $Depot = 0$, ends into **PVP** $= -1$, and spends less than H_0^{Veh} energy.* (SUB1)
- *γ_S starts from **PVP** $= -1$, ends into $Depot = 0$, and spends less than $C^{Veh} - H_0^{Veh}$ energy.* (SUB2)
- *For any $s = 1,\ldots,S-1$, γ_s starts from **PVP** $= -1$, ends into -1 and does not require more than C^{Veh} energy.* (SUB3)
- *Every station $j = 1,\ldots,M$ is visited once.* (SUB4)

Proof. We only need to simultaneously follow the TSP route defined by X and its extension defined by Z. **End-Proof**

Remark 1: Constraints (VR6, VR6-Bis) neither tells in which order the sub-routes $\gamma_s, s = 0, \ldots, S$ must be performed, nor it ensures the feasibility of resulting route with respect to the time horizon $p \cdot N$.

Separating the Recharge Decomposition Constraints: As usually when it comes to the management of constraints on the subsets of a given set, we must cope with the separability issue. Given 2 possibly non integral vectors (Z, X), separating the constraints (VR6, VR6-Bis) means checking that all those constraints are satisfied by (Z, X) and, in case they are not, computing a subset $A \subseteq \{0, \ldots, M+1\}$ such that related VR6 or VR6-Bis constraint is violated by (Z, X). Theorem 1 below not only tells us that separating (VR6, VR6-Bis) can be done in polynomial time, but related proof provides us with a polynomial time algorithm that will perform this separation process.

Theorem 1: *The Recharge Decomposition constraints can be separated in polynomial time, by application of a min cost flow algorithm.*

Sketch of the Proof: Let us restrict ourselves to the case of the constraints VR6 (the case of the constraints VR6-Bis is quite similar). Given Z, possibly non integral. Separating VR6 means searching for $A \subseteq \{0, \ldots, M+1\}$, which does not contain -1 and is such that:

$$\sum_{(j,k)\in\delta(A)} \Pi_{j,k} \cdot Z_{j,k} < \sum_{(j,k)\in Cl(A)} E_{j,k} \cdot Z_{j,k},$$

or equivalently for $B = \{-1, 0, \ldots, M+1\} \setminus A$, which contains -1 and is such that: (*)

$$\sum_{(j,k)\ s.t. j\in B, k\notin B} \Pi_{j,k} \cdot Z_{j,k} + \sum_{(j,k)\ s.t. j\in B, k\in B} E_{j,k} \cdot Z_{j,k} < \sum_{(j,k)} E_{j,k} \cdot Z_{j,k}.$$

Let us set $\Delta = \sum_{(j,k)} E_{j,k} \cdot Z_{j,k}$, and let us construct an auxiliary multi-graph $G = (X, A)$ as follows:

- $X = \{Source = -1, 0, \ldots, M+1, M+2 = Sink\}$;
- A is defined as the set of all *simple-arc* $(j,k), j \neq k \in \{-1, 0, \ldots, M+1\}$, augmented, for every j, with *copy-arcs* $(j, M+1)^k, k \neq j \in \{-1, 0, \ldots, M+1\}$, which connect j to $Sink = (M+2)$ and are provided with label k. Every *copy-arc* $a = (j, M+1)^k$ is provided with a weight w_a equal to $E_{j,k} \cdot Z_{j,k}$. Every *simple-arc* $a = (j,k)$ is provided with a weight w_a equal to $\Pi_{j,k} \cdot Z_{j,k} - E_{j,k} \cdot Z_{j,k}$.

Then we easily check that computing B such that (*) holds means computing a cut B' which separates $Source = -1$ from $Sink = M+2$ in G and is such that $\sum_{a\ s.t.\ origin(a)\in B', destination(a)\notin B'} w_a < \Delta$. We know that this can be done in polynomial time through a simple Max-Flow algorithm. **End-Proof**.

C. The *VR_EPC_MILP* MILP formulation

We are now able to extend the structural *vehicle routing* constraints VR1, ..., VR6-Bis into a **VR_EPC_MILP** setting of **VR_EPC**. We do it while distinguishing 3 main groups of constraints:

- The **PV-Plant Constraints**: They involve the variables related to the purchase of power and express the evolution along the periods of the load V_i^{PVP} of the **PVP** battery, the fact that this load cannot exceed the capacity C^{PVP}, and the fact that the final load V_N^{PVP} must be no smaller than the initial load H_0^{PVP}.
- The **Vehicle Routing Constraints**: They contain VR1, ..., VR6-Bis, together with constraints related to the time values when the vehicle arrives to the stations or to **PVP** for the recharging transactions. Those temporal constraints fix the order according to which the sub-routes involved in Lemma 1 are visited. The *Vehicle Routing* constraints also contain constraints which express the evolution along the route defined by (Z, X) of the load V_j^{Veh} of the **EVeh** battery, the fact that this load cannot exceed the capacity C^{Veh}, and the fact that the final load V_{M+1}^{Veh} must be no smaller than the initial load H_0^{Veh}.
- The **Synchronization Constraints**: They link together the **PVP** periods and the time horizon $[0, p \cdot N]$ of the vehicle, and synchronize the energy received by the vehicle and the energy delivered by the PV-Plant during the recharge transactions.

Those constraints may be formalized as follows (for the sake of simplicity, we replace the "Big M" formulations by implications):

VR_EPC Constraints and Objective Function:

- **Objective**: Minimize $\sum_i (\Psi_i.\delta_i + \Phi_i(y_i) + \alpha.\tau_{M+1}$
- **PV-Plant Constraints**
  - $\forall i = 1, \ldots, N-1$: $y_i \leq C^{Ch} \cdot (1 - \delta_i)$; (PC1)
  - $\forall i = 0, \ldots, N$: $V_i^{PVP} \leq C^{PVP}$; (PC2)
  - $V_0^{PVP} = H_0^{PVP}$; $V_N^{PVP} \geq H_0^{PVP}$; (PC3)
  - $\forall i = 1, \ldots, N$: $V_i^{PVP} = V_{i-1}^{PVP} + y_i - L_i^{PVP}$; (PC4)
- **Vehicle Routing Constraints**
  - (VR1, ..., VR6-Bis) involved in Lemma 1;
  - $V_0^{Veh} = H_0^{Veh}$; $V_{M+1}^{Veh} \geq H_0^{Veh}$; (VR7)
  - $\forall j = 0, \ldots, M+1$: $E_{j,-1} \leq V_j^{Veh} \leq C^{Veh}$; (VR8)
  - $\forall j, k = 0, \ldots, M+1$: $X_{j,k} \rightarrow (V_k^{Veh} + E_{j,k} + (Z_{j,k} - 1) \cdot (\epsilon_j + \epsilon_k - E_{j,k})) \leq (V_j^{Veh} + L_j^{Veh})$; (VR9)
  - $\forall j = 0, \ldots, M$: $Z_{j,-1} \rightarrow (V_j^{Veh} \geq \epsilon_j)$; (VR10)
  - $\forall j = 0, \ldots, M$: $Z_{j,-1} \rightarrow (V_j^{Veh} + L_j^{Veh} \leq \epsilon_j + C^{Veh})$; (VR10-Bis)
  - $\tau_0 = 0$; $\tau_{M+1} \leq p \cdot N$; (VR11)
  - $\forall j, k = 0, \ldots, M+1$: $Z_{j,k} \rightarrow (\tau_j + T_{j,k} \leq \tau_k)$; (VR12)
  - $\forall j, k = 0, \ldots, M+1$: $(X_{j,k} - Z_{j,k} = 1) \rightarrow (\tau_j^* + p + T_{-1,k} \leq \tau_k)$; (VR13)
  - $\forall j = 0, \ldots, M+1$: $(Z_{j,-1}) \rightarrow (\tau_j + T_{j,-1} \leq -\tau_j^*)$; (VR13-Bis)
- **Synchronization Constraints**
  - $\forall j = 0, \ldots, M$: $\sum_{i=0,\ldots,N-1} U_{i,j} = Z_{j,-1}$; (SY1)
  - $\forall j = 0, \ldots, M$: $\sum_{i=0,\ldots,N-1} m_{i,j} = L_j^{Veh}$; (SY1-Bis)
  - $\forall i = 0, \ldots, N-1$: $\sum_{j=0,\ldots,M} U_{i,j} = \delta_i$; (SY2)

– $\forall i = 0, \ldots, N-1$:
$\sum_{j=0,\ldots,M} m_{i,j} = L_i^{PVP}$; (SY2-Bis)
– $\forall j = 0, \ldots, M$: $\sum_{i=0,\ldots,N-1} p \cdot i \cdot U_{i,j} = \tau_j^*$; (SY3)
– $\forall i = 0, \ldots, N-1, j = 0, \ldots, M$:
$m_{i,j} \leq Inf(C^{PVP}, C^{Veh}) \cdot U_{i,j}$; (SY4)

Theorem 2: *Above **VR_EPC_MILP** MILP model solves **VR_EPC** in an exact way.*

Proof: Lemma 1 tells us that X, Z define a feasible vehicle route Γ. Constraints (SY1, SY2) link the variables U, m with the route Z and the energy received by the vehicle, with the PV-Plant variables δ and the energy transferred by **PVP**. (SY4) makes those energy amounts be consistent with the storage capacities. (SY3) links the time values of **PVP** and **EVeh**. (PC4) describes the evolution throughout the time of the energy stored by **PVP**. Vehicle constraints (VR7, ..., VR10) describe the evolution throughout the time of the energy stored by **EVeh**. They keep this energy from becoming negative or larger than the capacity C^{Veh}. (VR11, ..., VR13-Bis) describe the evolution of the time while the vehicle follows the route described by X and Z. **End-Proof**

Practical Handling of the VR_EPC_MILP MILP model. We handle **VR_EPC_MILP** through Branch and Cut, while applying the auxiliary graph and the Max-Flow algorithm involved in Theorem 1 in order to separate the *Recharge Decomposition* constraints.

D. Numerical Experiments

Purpose: Evaluating the **VR_EPC_MILP** MILP model.

Technical Context: The experiments are performed on a computer with AMD EPYC 7H12 64-Core processor, and are running under Gnu/linux Ubuntu 20.04.2. The MILP library CPLEX 12.10 is used in a single-thread mode.

1) Instance generation: The main parameters of every instance are the station number M, the period number N, and the length p of the periods. M varies from 5 to 30; N varies from 40 to 320; p takes values in $\{1, 2, 4\}$. We derive from the instances with $p = 1$ the instances with $p = 2$ and $p = 4$ by merging the periods in a natural way, the length $TMax = p \cdot N$ of the time horizon remaining the same.

Vehicle coefficients: The M stations, together with $Depot$ and **HMP** are generated as integral points of a square in the 2D plane. Vectors T, θ, θ^* corresponds to a rounding of the Euclidean distance, and vectors E, ϵ, ϵ^* correspond to the Manhattan distance.

PV-Plant coefficients: We cluster production periods into 5 super-periods of same length, each provided with symbolic mean production and cost values $\overline{R_{cl}}$, $\overline{Cost_{cl}}$ in $\{Low, Medium, High\}$. Then, integral time dependent production coefficients R_i, together with time dependent cost convex piecewise (2 pieces) linear functions $\Phi_i, i = 0, \ldots, N-1$, are generated accordingly in such a way that resulting instance is feasible and that this feasibility requires the purchase of additional power.

Storage capacities: We control the number of recharge transactions by maintaining $\frac{C^{PVP}}{C^{Veh}}$ between 0.5 and 3.

TABLE II
BEHAVIOR OF **VR_EPC_MILP**

Id	(N, M, p)	LBG_LP	UBG_LP
1	(160, 10, 1)	56,7	173
2	(160, 10, 1)	76,6	131
1-2	(160, 10, 2)	58,5	175
2-2	(160, 10, 2)	71,7	135
3	(160, 10, 1)	51,6	171
4	(160, 10, 1)	64,0	125
3-2	(160, 10, 2)	51,7	174
4-2	(160, 10, 2)	63,4	134
5	(240, 20, 1)	43,8	148
6	(240, 20, 1)	59,5	176
7	(240, 20, 1)	83,2	208
5-2	(240, 20, 2)	43,9	148
6-2	(240, 20, 2)	59,1	201
7-2	(240, 20, 2)	82,8	255
8	(320, 30, 1)	58,0	151
9	(320, 30, 1)	51,7	117
10	(320, 30, 1)	63,2	150
8-2	(320, 30, 2)	58,0	153
9-2	(320, 30, 2)	51,2	127
10-2	(320, 30, 2)	63,08	163

Scaling coefficients α: We do in such a way that the weights of respectively $PCost$ and $\alpha.Veh\_Time$ remain integral and comparable.

For every instance Id (Tables II), we provide input values values N, M, p. We tested 242 instances. Yet, we restrict ourselves here to 20 instances, with $p = 1$, referred to as Id, and $p = 2$ referred to as $Id - 2$.

2) Results: For every instance, Table II displays The lower bound LBG_LP and the upper bound UBG_UB computed through Branch and Cut by the CPLEX library in 2 CPU hours (7200 seconds). We boost the computation of upper bounds UBG_UB with the CPLEX parameter $MIP\_emphasis\_switch$.

Comments: Even when allowed to run during 2 hours, the model never reaches optimality, though it often allows the computation of a good feasible solution. That is why we are now going to derive from some statistical analysis some efficient heuristics for our problem.

IV. HEURISTIC HANDLING OF VR_EPC

The bi-level structure of **VR_EPC** and the fact that this bi-level structure will correspond in most cases to some collaborative decision context, suggests to manage **VR_EPC** while acting on the master route Γ through local search operators and devices that anticipate the behavior of resulting sub-problem **VR_EPC**(Γ) through the use of simple devices. There exists many ways to proceed this way. The approach which we are going to propose here relies on a statistical analysis of the correlation which may exist between the optimal value of **VR_EPC** and some key features of the route Γ.

In order to keep on, we need some additional notations:

- We denote by **VR_EPC**(Γ) the **VR_EPC** instance induced by fixing Γ (that means, referring to the **VR_EPC_MILP** MILP model, if we fix X in such a

TABLE III
CORRELATION ANALYSIS

It	1	2	3	4	5	6	7	8	9
$W(\Gamma)$	564	504	500	462	464	435	388	390	378
$L^T(\Gamma)$	197	183	182	165	162	152	135	132	126
NRT	11	10	9	9	9	9	7	7	8

TABLE IV
CORRELATION ANALYSIS

It	10	11	12	13	14	15	16	17	18
$W(\Gamma)$	564	504	500	462	319	315	314	326	324
$L^T(\Gamma)$	197	183	182	165	105	99	101	100	99
NRT	11	10	9	9	6	6	6	7	7

way it meets (VR3, VR4) and does not induce any subtour). We denote by $W(\Gamma)$ its optimal value;
- We denote by $L^T(\Gamma), L^E(\Gamma)$ the lengths of Γ in the sense of respectively vectors T and E.

Let us also recall the way the standard *Traveling Salesman 2_Opt* and *Reloc* operators act on any current route Γ through 2 parameters j_1, j_2 in **J** $+\{Depot\}$:

- *2_Opt*(Γ, j_1, j_2) replaces the moves from j_1 to its successor $\bar{j}_1$ and from j_2 to its successor $\bar{j}_2$ by moves from j_1 to j_2 and from $\bar{j}_1$ to $\bar{j}_2$, and reverses the orientation of Γ between j_2 and $\bar{j}_1$;
- *Reloc*(Γ, j_1, j_2) relocates j_1 between j_2 and $\bar{j}_2$.

A. Experimentally Linking $W(\Gamma)$, $L^T(\Gamma)$ and $L^E(\Gamma)$

Tables III, ..., VI show that $W(\Gamma)$, $L^T(\Gamma)$, $L^E(\Gamma)$ and the number NRT of recharge transactions are strongly correlated along the 36 iterations $It = 1, \ldots, 36$ of a descent loop involving *2_Opt* and *Reloc* and performed on 20 stations. We observe the same kind of correlation when performing the same experimental process on all the instances involved in table II.

B. A Simple PDYN_EPC Algorithm for the Computation of an Upper Bound $W\_Aux(\Gamma)$ of $W(\Gamma)$

Though fixing Γ deeply simplifies the **VR_EPC_MILP** MILP model, it does make it suitable for repeated applications inside a local search process. So we briefly describe here a

TABLE V
CORRELATION ANALYSIS

It	19	20	21	22	23	24	25	26	27
$W(\Gamma)$	205	202	211	200	199	178	178	185	179
$L^T(\Gamma)$	72	70	67	64	64	62	62	61	61
NRT	3	3	4	3	3	4	4	3	3

TABLE VI
CORRELATION ANALYSIS

It	28	29	30	31	32	33	34	35	36
$W(\Gamma)$	173	297	259	267	263	276	275	247	205
$L^T(\Gamma)$	57	97	89	89	87	87	87	76	75
NRT	3	4	3	3	3	4	3	3	4

simple dynamic programming algorithm *PDYN_EPC* which computes an upper bound $W\_Aux(\Gamma)$ of $W(\Gamma)$ under running times that allow its insertion into such a local search process. We design *PDYN_EPC* by noticing that once Γ is fixed, a full solution is determined by the sequence of recharge transactions linking **EVeh** and **PVP** together, augmented with the amounts of power bought during the periods separating 2 consecutive recharge transactions. More precisely, *PDYN_EPC* searches for a path (a sequence of transitions) in a *state* graph whose nodes (the states) and arcs (decisions and transitions) are defined as follows:

- A *state* in the sense of *PDYN_EPC* is a 4-uple $S = (j, i, V_j^{Veh}, V_i^{PVP})$, where j is a station, i a period, V_j^{Veh} is the power load of **EVeh** when it leaves j and V_i^{PVP} the power load of **PVP** at the end of i, with the implicit meaning that a recharge transaction involving i and j has just been performed. Such a state S comes with the smallest cost value $\Pi(S)$ of the cost of a path connecting initial state S_0 to S according to the *PDYN_EPC* process.
- A decision consists, in a natural way, in a 4-uple $(j_1, i_1, m_1, \bar{y})$, where (j_1, i_1, m_1) means the next recharge transaction, and $\bar{y}$ means the power which is going to be bought by **PVP** during the periods $i + 1, \ldots, i_1 - 1$. Resulting state and feasibility constraints derive in a natural way from the **VR_EPC_MILP** setting.
- The cost of related transition is $p \cdot (i_1 - i)$ augmented with the purchase cost of $\bar{y}$. This purchase cost corresponds to the optimal value of some convex optimization program. Since it does not depend on the vehicle inputs, it is computed as part of a pre-process and stored in a table Pr_Cost which, with any pair of periods i, i_1 and any purchase value $\bar{y}$ in a target discrete set, associates the optimal cost of purchasing $\bar{y}$ energy during the periods $i + 1, \ldots, i_1 - 1$.
- Initial state S_0 is a 4-uple $(Depot, -1, H_0^{EVeh}, H_0^{PVP})$. A final state is any 4-uple $(Depot, N - 1, V_j^{Veh} \geq H_0^{EVeh}, V_j^{PVP} \geq H_0^{PVP})$, taking into account that this final state is not related to any recharge transaction and that the cost of related transition must be adapted in order to refer to the time when **EVeh** is back to $Depot$.

We do not detail the description of *PDYN_EPC*, which by many ways works as the standard path search algorithm *A**, and restrict ourselves to say that:

- In order to make this algorithm run fast, we fix an upper bound $NDec$ (in our case, we shall set $NDec = 10$) on the number of feasible decisions that may be tried from a given state S. A consequence is that *PDYN_EPC*$(NDec)$ is only a heuristic algorithm, that provides us with an upper bound for the optimal value $W(\Gamma)$ of **VR_EPC**(Γ).
- We introduce a lower bound computation device, which, to any state $S = (j, i, V_j^{Veh}, V_i^{PVP})$, makes correspond some lower bound $VAL(S)$ of the cost of a path in the state graph that would connect S to some final state. This lower bound device allows us to discard states S which appear to be poorly promising according to related values

$(\Pi(S), VAL(S))$.

C. Two Simple Heuristic Algorithms

As told at the beginning of this section, above experiments suggest handling Γ while partially short-cutting the computation of $W(\Gamma)$ and relying on $L^T(\Gamma)$ and $L^E(\Gamma)$ in order to drive Γ towards good solutions. We do it here in two ways:

- A GRASP Algorithm *GRASP_VR_EPC*(Q), Q being a number of replications, that perform Q descent processes on Q randomly generated linear combinations of $L^T(\Gamma)$ and $L^E(\Gamma)$;
- An Algorithm *Descent_VR_EPC* that pipelines *GRASP_VR_EPC*(Q) and some pseudo-descent process parametrized by 2 flexibility parameters δ^{Time} and δ^{Ener}, that allow modifying the route Γ even when neither δ^{Time} nor δ^{Ener} decreases.

The GRASP Algorithm *GRASP_VR_EPC*.
It considers a replication parameter Q and works as follows:

1) At any replication q, it randomly generates two non negative scaling parameters ω^{Time} and ω^{Ener} together with an initial route $\Gamma(q)$;
2) Then it applies *2_Opt* and *Reloc* until $\Gamma(q)$ becomes a local optimum with respect to the surrogate objective function $\omega^{Time} \cdot L^T(\Gamma)+$ $\omega^{Ener} \cdot L^E(\Gamma)$;
3) Finally it computes $W\_Aux(\Gamma(q))$ while applying the *PDYN_EPC* algorithm and updates the best current route $\Gamma(q_{Best})$. When the whole process is over, it computes the exact value $W(\Gamma(q_{Best}))$ while applying the restriction of **VR_EPC_MILP** to Γ.

This is summarized by Algorithm 1:

Algorithm 1: GRASP_VR_EPC(Replication Parameter Q).

input : The inputs of the **VR_EPC Problem**, that we suppose to be feasible
output: A best route Γ^{Best} and related value W^{Best}

$\Gamma^{Best} \leftarrow Undefined$; $W\_Aux^{Best} \leftarrow +\infty$;
for $q = 0, \ldots, Q$ **do**
  Randomly generate coefficients ω^{Time} and ω^{Ener} ;
  Randomly generate $\Gamma(q)$; `Stop` ← `False` ;
  while not `Stop` **do**
    By applying *2_Opt* or *Reloc*, decrease $\omega^{Time} \cdot L^T(\Gamma(q)) + \omega^{Ener} \cdot L^E(\Gamma(q))$;
    if *Fail(decrease)* **then**
      `Stop` ← `True`
    else
      Modify $\Gamma(q)$ accordingly
    end
  end
  Compute $W\_Aux(\Gamma(q))$ by applying *PDYN_EPC*;
  if $W\_Aux^{Best} > W\_Aux(\Gamma(q))$ **then**
    $\Gamma^{Best} \leftarrow \Gamma(q)$; $W\_Aux^{Best} \leftarrow W\_Aux(\Gamma(q))$;
end
Compute the exact value W^{Best} (by application of the restriction of **VR_EPC_MILP** to Γ^{Best};
return Γ^{Best} and W^{Best}.

The Pseudo-Descent Algorithm *Descent_VR_EPC*: Clearly, it may occur that current route Γ may be improved by operators *2_Opt* and *Reloc* while neither $L^{Time}(\Gamma)$ nor $L^{Ener}(\Gamma)$ may be improved. This suggests allowing the *2_Opt* and *Reloc* operators to slightly deteriorate the values $L^{Time}(\Gamma)$ and $L^{Ener}(\Gamma)$ and deciding about their application after computing $W\_Aux(\Gamma)$ while relying on the algorithm *PDYN_EPC*. More precisely, let us consider 2 parameters $\delta^{Time} > 0, \delta^{Ener} > 0$ and 2 routes Γ_1, Γ_2. We say that Γ_2 deteriorates Γ_1 by no more than $(\delta^{Time}, \delta^{Ener})$ if at least one of the 2 following inequalities holds:

1) $L^{Time}(\Gamma_2) - L^{Time}(\Gamma_1) \leq \delta_1$;
2) $L^{Ener}(\Gamma_2) - L^{Ener}(\Gamma_1) \leq \delta_2$.

Then the parametric *Descent_VR_EPC*($\delta^{Time}, \delta^{Ener}$) algorithm works according to the following descent loop:

1) It initializes Γ as a *good quality* route (for instance by applying *GRASP_SVR_EP*(1) (1 replication));
2) At any iteration of the descent process, it generates all the *2_Opt* and *Reloc* parameters (j_1, j_2) such that resulting route deteriorates Γ by no more than $(\delta^{Time}, \delta^{Ener})$;
3) For any parameter (j_1, j_2) selected this way and any resulting route Γ_1 it tests its impact on the value $W\_Aux(\Gamma_1)$ obtained through application of the *PDYN_EPC* Algorithm;
4) If improving Γ this way is possible, then it does it according to a *Best Descent* strategy else it stops. In case it stops then it compute the exact current value $W(\Gamma)$.

This is summarized by Algorithm 2:

Remark 2: Both *GRASP_VR_EPC* and *Descent_VR_EPC* might be easily extended along more sophisticated metaheuristic scheme, more specifically genetic and ant colony based schemes (see [9] and [7]).

D. Numerical Experiments

Table VII involves the same instances as Table II. It displays:

- The value $GRASP\_50$ obtained by *GRASP_VR_EPC*(50) (50 replications) and related CPU time CPU_Grasp_50.
- The value $GRASP\_150$ obtained by *GRASP_VR_EPC*(150) (150 replications).

Table VIII considers the same instances as Table III and displays The value $Desc\_1 + LS$ obtained by the application of the following pipeline: *GRASP_VR_EPC*(1) (1 replication) $\rightarrow$ *Descent_VR_EPC*(4, 4), together with related CPU time $CPU\_Desc + LS$.

Comments: *GRASP_VR_EPC* with 150 replications most often reaches quasi-optimality. It outperforms the upper bound computed by CPLEX in 4 CPU hours in 8 among the 20 instances and obtains the same value as CPLEX in 6 other instances. This trend is reinforced when $M = 30$.

Algorithm 2: Descent_VR_EPC(Parameters $\delta^{Time} \geq 0, \delta^{Ener} \geq 0$).

input : The inputs of the **VR_EPC Problem**, that we suppose to be feasible
output: A (good quality) route Γ

Initialize Γ and compute an approximation W of $W(\Gamma)$ by applying the *PDYN_EPC* Algorithm (value $W\_Aux(\Gamma)$);
`Stop` $\leftarrow$ `False` ;
while not `Stop` **do**
  Generate all the parameters j_1, j_2 of the *2_Opt* operator such that applying *2_Opt* or *Reloc* to Γ deteriorates Γ by no more than $(\delta^{Time}, \delta^{Ener})$;
  for *Any 3-uple* (Op, j_1, j_2) *(Op meaning either Reloc or 2_Opt) generated this way* **do**
    Apply the *PDYN_EPC* Algorithm to the route resulting from application of (Op, j_1, j_2) to Γ;
  end
  if *A decrease of* $W\_Aux(\Gamma)$ *with respect to W is obtained at least once* **then**
    Replace Γ by $Op(\Gamma, j_1, j_2)$ and W by related W_Aux value (*PDYN_EPC* Algorithm), with (Op, j_1, j_2) related to the best improvement
  else
    `Stop` $\leftarrow$ `False`;
  end
end
Compute the exact value $W(\Gamma)$ (by application of the restriction of **VR_EPC_MILP** to Γ;
return Γ and $W(\Gamma)$.

Applying the pipe-line *GRASP_VR_EPC*(1) (1 replication) $\rightarrow$ *Descent_VR_EPC*(4, 4) improves the value obtained by *GRASP_VR_EP5*(50) in 8 among the 20 instances, and the value obtained by *GRASP_VR_EPC*(150) in 5 among the 20 instances. The running times induced by the calls to *Descent_VR_EPC*(4, 4) are rather high, due to the fact that we did not optimize the transformation of a route Γ into a *PDYN_EPC* input. Also, we may notice that in some cases, *Descent_VR_EPC*(4, 4) yields poor results (if we refer to *GRASP_VR_EP5*(50)) due to a poor initial route computed by a single *GRASP_VR_EPC*(1) run.

V. Conclusion

We dealt here with a complex model which make interact a routing process with an energy production process. W adopted a centralized point of view, which skips uncertainties, and tried both an exact MILP approach and a heuristic approach based upon a statistical correlation analysis. In the future, it will of course be interesting to try other methods. Also we plan dealing with *collaborative* contexts, when several *consumers* interact, each provided with its own agenda and incomplete information, and with the issue related to the uncertainty induced by solar energy production.

TABLE VII
Results for *GRASP_VR_EPC*

Id	$GRASP\_50$	$GRASP\_150$	CPU_Grasp_50
1	171	171	52,9
2	131	131	95,1
1-2	175	175	10,4
2-2	135	135	14,8
3	173	173	58,4
4	129	126	71,4
3-2	176	176	19,8
4-2	138	134	20,8
5	156	156	158,7
6	175	175	333,5
7	214	214	296,2
5-2	158	158	58,5
6-2	181	181	106,3
7-2	226	226	74,6
8	121	121	227,5
9	123	119	225,5
10	159	159	317,6
8-1	135	132	91,7
9-1	128	123	78,7
10-1	163	153	96,7

TABLE VIII
Results for *GRASP_VR_EPC*

Id	$GRASP\_150$	$Desc\_1 + LS$	$CPU\_Desc + LS$
1	171	171	22,7
2	131	134	16,5
1-2	175	175	15,6
2-2	135	140	5,4
3	173	165	197,0
4	126	144	111,0
3-2	176	171	36,7
4-2	134	161	41,1
5	156	168	356,2
6	175	176	565,7
7	214	207	254,4
5-2	158	173	109,8
6-2	181	181	203,6
7-2	226	220	109,0
8	121	121	1998,6
9	119	122	2351,0
10	159	151	876,2
8-1	132	134	773,0
9-1	123	130	629,1
10-1	153	159	633,5

References

[1] C. Artigues, E. Hébrard, A. Quilliot, H. Toussaint, *Models and algorithms for natural disaster evacuation problems*, Proc. 2019 FEDCSIS WCO Conf., p 143-146, 2019. doi.org/10.15439/2019F90

[2] Bendali F., Mole Kamga E., Mailfert J., Quilliot A., Toussaint H.: *Synchronizing Energy Production and Vehicle Routing*. RAIRO-O.R, 55 (4),pp. 2141-2163. (2021). https://doi.org/10.1051/ro/2021093.

[3] F. Bendali, J. Mailfert, E. Mole-Kamga, A. Quilliot, H. Toussaint, *Pipelining dynamic programming process in order to synchronize energy production and consumption*, Proc. 2020 FEDCSIS WCO Conf., p 303-306, 2020. doi.org/10.15439/978-83-955416-7-4.

[4] Drexl M.: Synchronization in vehicle routing–a survey; *Transportation Science* 46, pp. 297-316, (2012). https://dx.doi.org/10.1287/trsc.1110.0400

[5] Erdelic T., Caric T., Lalla-Ruiz E.: A Survey on the Electric Vehicle Routing Problem.*Journal of Advanced Transportation* (2019). https://doi.org/10.1155/2019/5075671.

[6] Erdogan S., Miller-Hooks E.. A green vehicle routing prob-

lem. *Transportation Research Part E* , 109, p 100–114, (2012). https://doi.org/10.1016/j.tre.2011.08.001

[7] S. Fidanova, O. Roeva, M. Ganzha, *Ant colony optimization algorithm for fuzzy transport modelling*, Proc. 2020 FEDCSIS WCO Conference, p 237-240, 2020. doi.org/10.15439/978-83-955416-7-4

[8] Franceschetti A., Demir E., Honhon D., Van Woensel T., Laporte G., and Stobbe M.: A metaheuristic for the time dependent pollution-routing problem. *EJOR*, 259 (3) :972 – 991, (2017). https://doi.org/10.1016/j.ejor.2016.11.026

[9] K. Gmyrek, M. Antkiewicz, P. Myszkowski: Genetic Algorithm for Planning and Scheduling Problem – StarCraft II Build Order case study; *Proceedings of the 18th Conference on Computer Science and Intelligence Systems* ACSIS, Vol. 35, pages 131–140 (2023). DOI: http://dx.doi.org/10.15439/2023F6015.

[10] Hung Y. F., Chien K. L.: A multi-class multi-level capacitated lot sizing model;*Journal of the Operational Research Society*, Palgrave Macmillan;The OR Society, vol. 51(11), pages 1309-1318, (2000). DOI: 10.1057/palgrave.jors.2601026.

[11] Irani S, Pruhs K.: Algorithmic problems in power management. *ACM SIGACT News* 36-2, p 63-76 (2005). DOI:10.1145/1067309.1067324.

[12] Koç C., Jabali O., Mendoza J., Laporte G.. The electric vehicle routing problem with shared charging stations. *ITOR*, 26, (2018). DOI:10.1111/itor.12620

[13] Kuo Y.. Simulated annealing to minimize fuel consumption for the time-dependent VRP. *Comp. Indust. Eng.*, 59 :157–165, (2010). https://doi.org/10.1016/j.cie.2010.03.012

[14] Lajunen A.: Energy consumption and cost analysis of electric city buses. *Transportation Research Part C*, 38 :1–15, (2014). https://doi.org/10.1016/j.trc.2013.10.008

[15] Luthander R., Widen J., Nilsson D., Palm J.: Photovoltaic self-consumption in buildings. *Applied Energy* 142, p 80-94, (2015). https://doi.org/10.1016/j.apenergy.2014.12.028

[16] Macrina G., Di Puglia L., Guerriero F.: The Green-Vehicle Routing Problem: A Survey. *Model. and Optim. in Logistics* p 1-26 (2020). DOI: https://doi.org/10.1007/978-3-030-45308-4_1.

[17] Raylan M., Matos S., Frota Y., and Satoru Ochi L.: Green vehicle routing and scheduling problem with split delivery. *Electronic Notes in Discrete Mathematics*, 69 :13 – 20, 2018. https://doi.org/10.1016/j.endm.2018.07.003

[18] Sachenbacher M., Leucker M., Artmeier A., Haselmayr J.: Efficient energy routing for electric vehicles. Proc *AAAI 2011 Int. Conf*, (2011). https://doi.org/10.1609/aaai.v25i1.7803

[19] Schneider M., Stenger A., Goeke D.: The electric vehicle-routing problem with time windows and recharging stations. *Transportation Science* 48, p 500-520, (2014). DOI:10.1287/trsc.2013.0490

[20] K. Stoilova, T. Stoilov, *Bi-level optimization application for urban traffic management*, Proc. 2020 FEDCSIS WCO Conf., p 327-336, 2020. doi.org/10.15439/978-83-949419-5-6.

[21] Verma A.: Electric vehicle routing problem with time windows, recharging stations and battery swapping stations. *Euro Journal of Transportation Logistics* 7, p 415-451, (2018).