Optimal Routing of Electric Vehicles in Networks with Charging Nodes: A Dynamic Programming Approach

Sepideh Pourazarm*, Christos G. Cassandras*, Andreas Malikopoulos†
*Division of Systems Engineering, Boston University, Boston, MA, 02215
{sepid, cgc}@bu.edu
†Urban Dynamics Institute, Oak Ridge National Laboratory, Oak Ridge, TN 37831
andreas@ornl.gov

***Abstract*—Motivated by the significant role of recharging in battery-powered vehicles, we study the routing problem for vehicles with limited energy through a network of charging nodes. We seek to minimize the total elapsed time for vehicles to reach their destinations considering both traveling and recharging times at nodes when the vehicles do not have adequate energy for the entire journey. We have studied the case of homogeneous charging nodes in [1] and generalized it to inhomogeneous charging nodes in [2] by formulating and solving a Mixed Integer Non-Linear Programming problem (MINLP) for a single-vehicle. In this paper, we solve the same problem using Dynamic Programming (DP), resulting in optimal solutions with lower computational complexity compared to [2]. For a multi-vehicle problem, where traffic congestion effects are included, we use a similar approach by grouping vehicles into "subflows" and propose a DP formulation. Our numerical results show that DP becomes prohibitively slow as the number of subflows increases. As in [1] and [2] we resort to an alternative flow optimization formulation leading to a computationally simpler problem solution with minimal loss of accuracy.**

Keywords - Electric Vehicles, Routing, Optimal Recharging Policy, Optimal Control, Dynamic Programming

I. Introduction

The increasing presence of Battery-Powered Vehicles (BPVs), such as Electric Vehicles (EVs) or mobile robots and sensors, has given rise to novel issues in classical network routing problems [3]. There are four BPV characteristics which are crucial in routing problems: limited cruising range, long charge times, sparse coverage of charging stations, and the BPV energy recuperation ability [4] which can be exploited. In recent years, the vehicle routing literature has been enriched by work aiming to accommodate these BPV characteristics. For example, by incorporating the recuperation ability of EVs, extensions to general shortest-path algorithms are proposed in [4] that address the energy-optimal routing problem, with further extensions in [5]. Charging times are incorporated into a multi-constrained optimal path planning problem in [6], which aims to minimize the length of an EV's route and meet constraints on total traveling time, total time delay due to signals, total recharging time and total recharging cost. In [7], algorithms for several routing problems are proposed, including a single-vehicle routing problem with inhomogeneously priced refueling stations for which a dynamic programming based algorithm is proposed to find a least cost path from source to destination. More recently, an EV Routing Problem with Time Windows and recharging stations (E-VRPTW) was proposed in [8], where controlling recharging times is circumvented by simply forcing vehicles to be always fully recharged.

In [2], we studied the vehicle total traveling time minimization problem in a network containing *inhomogeneous* charging nodes, i.e., charging rates at different nodes are not identical. In fact, depending on an outlet's voltage and current, charging an EV battery could take anywhere from minutes to hours. For the single EV routing problem, formulated as a MINLP, we proved certain optimality properties allowing us to reduce the dimensionality of the original problem. Further, by adopting a locally optimal charging policy, we derived a Linear Programming (LP) formulation through which near-optimal solutions are obtained. For a multi-vehicle problem, where traffic congestion effects are included and a system-wide objective is considered, a similar approach was used by grouping vehicles into "subflows". Moreover, we provided an alternative flow-based formulation which reduces the computational complexity of the original MINLP problem by orders of magnitude with numerical results showing little loss in optimality. Despite the properties of the problem that we have exploited, its solution remains computationally demanding for real-time applications. This motivates the study of alternative solution techniques.

Thus, in this paper we formulate the single EV routing problem as a Dynamic Programming (DP) problem by discretizing vehicle residual energy at each node. This model is identical for both homogeneous and inhomogeneous charging nodes and allows us to find an optimal routing and charging policy for both cases in CPU time which is about two orders of magnitude lower compared to [2]. We then study the much more challenging multi-EV routing problem, where a traffic flow model is used to incorporate congestion effects. Similar to [1] and [2], by grouping vehicles into "subflows" we are able to reduce the complexity of the original problem and provide a DP-based algorithm to determine optimal routing and charging policies at the EV subflow level. In this case, the problem size significantly increases with the number of subflows and the DP

The authors' work is supported in part by NSF under Grant CNS-1139021, by AFOSR under grant FA9550-12-1-0113, by ONR under grant N00014-09-1-1051, and by ARO under Grant W911NF-11-1-0227.

algorithm is eventually outperformed by our earlier MINLP approach as the number of subflows increases.

The structure of the paper is as follows. In Section II, we address the single-EV routing problem in a network with inhomogeneous charging nodes and formulate it as a DP problem. We then derive an iterative algorithm to solve it recursively. In Section III, the multi-EV routing problem is also formulated as a DP. Simulation examples are included illustrating our approach and providing insights on the relationship between recharging speed and optimal routes. Conclusions and further research directions are outlined in Section IV.

II. Single Vehicle Routing

We assume, as in [1] and [2], that a network is defined as a directed graph $G = (\mathcal{N}, \mathcal{A})$ with $\mathcal{N} = \{1, \dots, n\}$ and $|\mathcal{A}| = m$. Node $i \in \mathcal{N}/\{n\}$ represents a charging station and $(i, j) \in \mathcal{A}$ is an arc connecting node i to j (we assume for simplicity that all nodes have a charging capability, although this is not necessary). We also define $I(i)$ and $O(i)$ to be the set of start nodes (respectively, end nodes) of arcs that are incoming to (respectively, outgoing from) node i, that is, $I(i) = \{j \in \mathcal{N} | (j, i) \in \mathcal{A}\}$ and $O(i) = \{j \in \mathcal{N} | (i, j) \in \mathcal{A}\}$.

First, we deal with a single-origin-single-destination vehicle routing problem in a network of inhomogeneous charging stations. Nodes 1 and n respectively are defined to be the origin and destination. For each arc $(i, j) \in \mathcal{A}$, there are two cost parameters: the required traveling time τ_{ij} and the energy consumption e_{ij}. Note that $\tau_{ij} > 0$ (if nodes i and j are not connected, then $\tau_{ij} = \infty$), whereas e_{ij} is allowed to be negative due to an EV's potential energy recuperation effect [4]. Letting the vehicle's charge capacity be B, we assume that $e_{ij} < B$ for all $(i, j) \in \mathcal{A}$. Since we are considering a single vehicle's behavior, we assume that it will not affect the overall network's traffic state, therefore, τ_{ij} and e_{ij} are fixed depending on given traffic conditions at the time the single-vehicle routing problem is solved. Clearly, this cannot apply to the multi-vehicle case in the next section, where the decisions of multiple vehicle routes affect traffic conditions, thus influencing traveling times and energy consumption. Since the EV has limited battery energy, it may not be able to reach the destination without recharging. Thus, recharging amounts at charging nodes $i \in \mathcal{N}$ are also decision variables.

First, we briefly review the formulation in [2]. The single vehicle's objective is to determine a path from 1 to n, as well as recharging amounts, so as to minimize the total elapsed time to reach the destination. We formulate this as a Mixed Integer Nonlinear Programming (MINLP) problem:

$$\min_{x_{ij}, r_i,\ i,j \in \mathcal{N}} \quad \sum_{i=1}^{n} \sum_{j=1}^{n} \tau_{ij} x_{ij} + \sum_{i=1}^{n} \sum_{j=1}^{n} r_i g_i x_{ij} \tag{1}$$

$$s.t. \quad \sum_{j \in O(i)} x_{ij} - \sum_{j \in I(i)} x_{ji} = b_i, \quad \text{for each } i \in \mathcal{N} \tag{2}$$

$$b_1 = 1,\ b_n = -1,\ b_i = 0, \ \text{for } i \neq 1, n \tag{3}$$

$$E_j = \sum_{i \in I(j)} (E_i + r_i - e_{ij}) x_{ij}, \ \text{for } j = 2, \dots, n \tag{4}$$

$$0 \leq E_i \leq B, \quad E_1 \text{ given}, \ \text{for each } i \in \mathcal{N} \tag{5}$$

$$x_{ij} \in \{0, 1\}, \quad r_i \geq 0 \tag{6}$$

where $x_{ij} \in \{0, 1\}$, $i, j \in \mathcal{N}$ denotes the selection of arc (i, j), $r_i \geq 0$, $i \in \mathcal{N}/\{n\}$ is for energy recharging amount at node i, E_i represents the vehicle's residual battery energy at node i, and g_i is the charging time per unit of energy for charging node i.

The constraints (2)-(3) stand for flow conservation, which implies that only one path starting from node i can be selected. Constraint (4) represents the EV's energy dynamics. Finally, (5) indicates that the vehicle cannot run out of energy before reaching a node or exceed a given capacity B.

A. Dynamic Programming Formulation

Solving problem (1) - (6) is computationally expensive. In [2] we proceeded by decomposing it into two linear programming (LP) problems finding a near-optimal solution (for networks with inhomogeneous charging nodes). Here, we formulate the same problem in a DP setting and obtain optimal (not just near-optimal) solutions. The algorithm is based on the following formulation.

We define $Q(i, E_i)$ to be the minimum elapsed time, including traveling and recharging times, to the destination node when starting at node i with E_i units of energy. Our goal, therefore, is to determine $Q(1, E_1)$ where E_1 is given. Assuming the EV maximum charging capacity is B, we have to consider all possible values of $E_i \in [0, B]$. To do so, we discretize the range $[0, B]$ and form a set of all possible values for E_i. Our algorithm is centered on the standard principle of optimality [9] based on which, $Q(i, E_i)$ is obtained using the following iterative equation:

$$Q(i, E_i) = \min_{\substack{j \in O(i),\ 0 \leq E_j \leq B \\ \text{s.t } 0 \leq E_j - E_i + e_{ij} \leq B}} \Big[\overbrace{Q(j, E_j)}^{\text{Cost to go}} + \overbrace{\tau_{ij} + (E_j - E_i + e_{ij}) g_i}^{\text{One step cost}} \Big] \tag{7}$$

where the state is $[i, E_i]$ and there are two control variables: the amount to charge at each state, r_i, and the next node to route the EV to, $j \in O(i)$, dictated by the graph topology. The charge amount r_i is constrained by the energy dynamics, $E_j = E_i + r_i - e_{ij}$ and by $0 \leq r_i \leq B$. This iterative process leads to the optimal solution because when an optimal policy is found from state $[j, E_j]$ to the destination for all feasible values of j and E_j, then the route from node i to the destination node via node j will also be optimal. Under proper technical conditions, the iterative process generated through (7) converges to the optimal value of Q. The detailed steps of the DP algorithm for this problem are given next.

Initialization: Based on Lemma 2 in [2], if an EV receives any positive charge in the optimal path, i.e. $\sum_i r_i^* > 0$, the EV residual energy at the destination is zero, i.e. $E_n^* = 0$. Therefore, the cost value at the destination node is $Q^{(0)}(n, 0) = 0$. Motivated by Dijkstra's algorithm for the shortest path problem, we set the initial elapsed time for all other states to infinity, i.e.,

$$Q^{(0)}(n, E_n) = \begin{cases} 0 & \text{if } E_n = 0, \\ \infty & \text{if } E_n > 0. \end{cases} \tag{8}$$

$$Q^{(0)}(i, E_i) = \infty \quad \forall i \in \mathcal{N} \setminus n, \quad 0 \le E_i \le B \tag{9}$$

Iteration steps: The update of Q values can be carried out starting from any state. For convenience, we start at the source node, i.e. $[1, E_1]$. At the kth iteration, the Q values are updated as follows: $Q^{(k)}(n, E_n) = Q^{(0)}(n, E_n)$ and

$$Q^{(k)}(i, E_i) = \min_{\substack{j \in O\{i\},\ 0 \le E_j \le B \\ \text{s.t } 0 \le E_j - E_i + e_{ij} \le B}} [Q^{(k-1)}(j, E_j) + \tau_{ij} + (E_j - E_i + e_{ij}) g_i] \quad \forall i \in \mathcal{N} \setminus n, \quad 0 \le E_i \le B \tag{10}$$

We seek $\lim_{k \to \infty} Q^{(k)}(i, E_i) = Q^*$, therefore, the algorithm stops when $Q^{(k)}(i, E_i) = Q^{(k-1)}(i, E_i)$ for all $i \in \mathcal{N}$, $0 \le E_i \le B$. The optimal route can then be determined by choosing the next state, minimizing $Q(i, E_i)$. Without loss of generality, we re-index nodes so that we may write the optimal path as $P = \{1, ..., m\}$. Then, the optimal charging amount at each node on the optimal path is calculated through $r_{i-1} = E_i - E_{i-1} + e_{i-1,i}$ with $i = 2, ..., m$, and E_1 given.

B. Numerical Example

To investigate the effectiveness of the DP algorithm, we consider a grid graph with 49 nodes and 84 edges as shown in Fig. 1, where the traveling time, τ_{ij}, and energy consumption, e_{ij} on each edge are shown in red and blue numbers respectively. Fig. 1 shows the optimal path for the network with

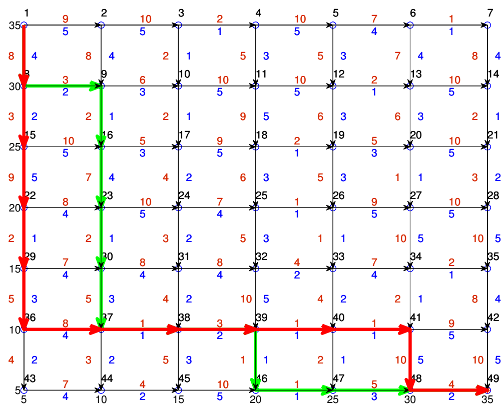

Fig. 1: A 49-node grid network with inhomogeneous charging nodes.

homogeneous charging stations ($G = [g_1, ..., g_{n-1}]$, $g_i = 1 \ \ \forall\ i$) and inhomogeneous charging stations G =[1 1 1 1 1 1 1 5 5 1 1 1 1 1 1 5 1 1 1 1 1 1 5 1 1 1 1 1 1 5 1 1 1 1 1 1 5 5 5 1 1 1 1 1 1 5 5 5] as the green and red routes respectively. For the network with homogeneous stations, the optimal charging policy suggests that the EV requires just enough charge at each node to reach the next node on the optimal path, e.g. if $E_1 = 0$ then $r_i^* = e_{i,i+1}$, $i \in P$. In contrast, for the network with inhomogeneous charging nodes with a G vector as above, the optimal charging amount at each node on the optimal path is as follows:

$$r_1 = 5 \quad r_2 = 5 \quad r_3 = 1 \quad r_{10} = 1 \quad r_{17} = 2 \quad r_{24} = 2$$
$$r_{31} = 5 \quad r_{38} = 0 \quad r_{39} = 0 \quad r_{40} = 6 \quad r_{47} = 0 \quad r_{48} = 0$$

The algorithm execution is very fast for this graph and converges to the optimal solution in 13 iterations in less than 10 sec for both homogeneous and inhomogeneous charging nodes. In contrast, a MINLP solver in [1] and [2] requires more than 1000 sec to find the optimal solution for the same graph with homogeneous charging nodes.

III. Multiple Vehicle Routing

The results obtained for the single vehicle routing problem pave the way for the investigation of multi-vehicle routing, where we seek to optimize a *system-wide* objective by routing and charging vehicles through some network topology. This is a much more challenging problem, the main technical difficulty being the need to consider the influence of traffic congestion on both traveling time and energy consumption.

As in [1], we proceed by grouping subsets of vehicles into N "subflows" where N may be selected to render the problem manageable. Let all vehicles enter the network at node 1 and let R denote the rate of vehicles arriving at this node. Viewing vehicles as defining a *flow*, we divide them into N *subflows* each of which may be selected so as to include the same type of vehicles (e.g., large vehicles vs smaller ones or vehicles with the same initial energy). Thus, all vehicles in the same subflow follow the same routing and recharging decisions so that we only consider control at the subflow level rather than individual vehicles. Clearly, not all vehicles in our system are EVs, therefore, not all of them are part of our optimization process. These can be treated as uncontrollable interfering traffic for our purposes and can be readily accommodated in our analysis, as long as their flow rates are known. However, for simplicity, we will assume here that every arriving vehicle is an EV and joins a subflow. Our objective is to determine optimal routes and energy recharging amounts for each vehicle subflow so as to minimize the total elapsed time of these flows from origin to destination.

A. Mixed Integer Non-Linear Programming Formulation

In [2], we formulated this problem as a MINLP as follows:

$$\min_{\mathbf{x_{ij}}, \mathbf{r_i},\ i,j \in \mathcal{N}} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{N} \left(\tau_{ij}^k(\mathbf{x_{ij}}) + r_i^k g_i x_{ij}^k \right) \tag{11}$$

$$\text{s.t.} \quad \sum_{j \in O(i)} x_{ij}^k - \sum_{j \in I(i)} x_{ji}^k = b_i, \quad \text{for each } i \in \mathcal{N} \tag{12}$$

$$b_1 = 1,\ b_n = -1,\ b_i = 0, \ \text{for } i \ne 1, n \tag{13}$$

$$E_j^k = \sum_{i \in I(j)} (E_i^k + r_i^k - e_{ij}^k(\mathbf{x_{ij}})) x_{ij}^k, \quad j = 2, \dots, n \tag{14}$$

$$E_1^k \text{ is given}, \quad 0 \le E_i^k \le B^k, \quad \text{for each } i \in \mathcal{N} \tag{15}$$

$$x_{ij}^k \in \{0, 1\}, \quad 0 \le r_i^k \le B^k \tag{16}$$

where the decision variables are $x_{ij}^k \in \{0, 1\}$ and r_i^k denoting the arc selection and charging amount at node i respectively, for all arcs (i, j) and subflows $k = 1, \dots, N$. Given traffic congestion effects, the time and energy consumption on each arc depends on the values of x_{ij}^k and the fraction of the total flow rate R associated with each subflow k. Let $\mathbf{x_{ij}} = (x_{ij}^1, \cdots, x_{ij}^N)^T$ and $\mathbf{r_i} = (r_i^1, \cdots, r_i^N)^T$. Then, the traveling time and corresponding energy consumption of the kth vehicle subflow on arc (i,j) are denoted by $\tau_{ij}^k(\mathbf{x_{ij}})$ and $e_{ij}^k(\mathbf{x_{ij}})$ respectively. Similar to the single vehicle case, E_i^k represents the residual energy of subflow k at node i, given by the aggregated residual energy of all vehicles in the subflow. If the subflow does not go through node i, then $E_i^k = 0$.

B. Flow control formulation

Although the MINLP formulation above finds an optimal routing and charging amount for each subflow, its solution is computationally expensive to obtain and the number of decision variables (hence, the solution search space) rapidly increases with the number of subflows, N. In [2], we proposed a non-linear programming (NLP) problem which determines near-optimal routes for all subflows as follows:

$$\min_{\substack{\mathbf{x_{ij}} \\ i,j\in\mathcal{N}}} \quad \sum_{k=1}^{N}\sum_{i=1}^{n}\sum_{j=1}^{n} \left[\tau_{ij}^k(\mathbf{x_{ij}}) + e_{ij}^k(\mathbf{x_{ij}})g_i + K(g_i - g_j)\right] \tag{17}$$

$$K = \begin{cases} (B^k - e_{ij}^k(\mathbf{x_{ij}}))x_{ij}^k & \text{if } g_i < g_j, \\ 0 & \text{otherwise} \end{cases}$$

$$\text{s.t.} \quad \sum_{j\in O(i)} x_{ij}^k - \sum_{j\in I(i)} x_{ji}^k = b_i, \quad \text{for each } i \in \mathcal{N}$$

$$b_1 = 1,\ b_n = -1,\ b_i = 0,\ \text{for } i \neq 1, n$$

$$0 \leq x_{ij}^k \leq 1$$

Note that in the above formulation, we relax the binary variables in (16) by letting $0 \leq x_{ij}^k \leq 1$. Thus, we switch our attention from determining a single path for any subflow k to several possible paths by treating x_{ij}^k as the normalized vehicle flow on arc (i,j) for the kth subflow. Once we find the near-optimal routes, the values of r_i^k, $i = 1, \ldots, n$, $k = 1, \ldots, N$, can then easily be determined by solving an LP problem (for more details refer to [2]). However, there is no guarantee of global optimality. This formulation reduces the computational complexity of the MINLP problem by orders of magnitude with numerical results showing little loss in optimality.

C. Dynamic Programming Formulation

Our goal here is to develop a DP algorithm to solve the same problem and compare its computational cost to the solution methods in [2]. Note that the problem size dramatically increases with the number of subflows, N. Our first step is to construct a new graph at the subflow level, $G_{sf} = (\mathcal{N}_{sf}, \mathcal{A}_{sf})$, given a road network $G = (\mathcal{N}, \mathcal{A})$ and the number of subflows, N. In this graph, each node in $\mathcal{N}_{sf}$ represents a feasible combination of nodes in G among which all subflows may be distributed. To make this clear, consider the road network shown in Fig. 2. In order to map the original graph G into the subflow-level graph G_{sf}, we define each of its nodes as $\mathbf{Y}^i = (y_1^i \ldots, y_N^i)$ where $i = 1, 2, \ldots$ indexes these nodes and y_k^i is the location of the kth subflow in G. Fig. 3 is the subflow-level graph G_{sf} constructed from Fig. 2 when the total inflow, R, is divided into 2 subflows ($N = 2$). In this case, G_{sf} consists of 25 nodes. As an example, in Fig. 3 node 3(2 4) represents a node with index $i = 3$ mapping the first and

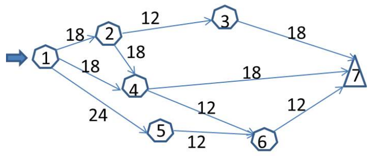

Fig. 2: A 7-node road network with inhomogeneous charging nodes.

second subflows to nodes 2 and 4 in G respectively, i.e., it represents a routing decision at node 1 in G for sublow 1 to travel from 1 to 2 and for sublow 2 to travel from 1 to 4, noting that $O(1) = \{2, 4, 5\}$. Clearly, G_{sf} is much larger than the original road network G, even for $N = 2$. Table I shows the number of nodes and edges in the subflow-level graph for different values of N for the road network in Fig. 2. As

TABLE I: Subflow-level graph size for different number of subflows for road network shown in Fig. 1

Number of subflows (N)	Number of nodes	Number of edges
2	25	54
3	91	268
6	4825	31914

we did for the single EV, we need to consider all possible combinations for the residual energies at each node in the subflow-level graph. Thus, we define $\mathbf{E}^i = [E_1^i, \ldots, E_N^i]$. When a decision is made at a node in G_{sf}, we need to calculate its effect on the travel time and energy consumption over each edge $(i,j) \in \mathcal{A}$ resulting from the traffic added to this edge. This requires information on the number of subflows routed through (i,j). Recalling that $|\mathcal{A}| = m$, let us index all edges $(i,j) \in \mathcal{A}$ as $\{1, \ldots, m\}$. Next, we define an auxiliary vector for each pair $(\mathbf{Y}^i, \mathbf{E}^i)$ in G_{sf} denoted by $\mathbf{S}^i = [s_1^i, \ldots, s_m^i]$ where s_l^i is the number of subflows through the lth edge in $G = (\mathcal{N}, \mathcal{A})$ starting from node $\mathbf{Y}^i \in \mathcal{N}_{sf}$ with residual energies $\mathbf{E}^i$, i.e. $s_l^i \in \{0, 1, \ldots, N\}$ and $l = 1, \ldots, m$. In other words, $\mathbf{S}^i$ is a function of the state variables $(\mathbf{Y}^i, \mathbf{E}^i)$ and includes the data required to calculate traveling time and energy consumption amounts on each edge. Specifically, traveling from node $\mathbf{Y}^i$ to node $\mathbf{Y}^j$ in the G_{sf}, τ_{y_i,y_j}^k and e_{y_i,y_j}^k represent the traveling time and energy consumption on the edge (y_k^i, y_k^j) in the original graph for the kth subflow respectively and their values depend on the traffic congestion on the edge which is a function of $s_l^i \in \mathbf{S}^i$. More precisely, we define the "edge indexing operation", $\delta(y_k^i, y_k^j)$, assigning a single edge index l to a pair of node indices (y_k^i, y_k^j), i.e., $\delta(y_k^i, y_k^j) = l$. Note that s_l^i is updated based on the decision made at node $\mathbf{Y}^i$ which determines the next node, $\mathbf{Y}^j$, and residual energy $\mathbf{E}^j$. Clearly,

$$s_l^i = s_l^j + \sum_{k=1}^{N} \mathbf{1}[\delta(y_k^i, y_k^j) = l] \tag{18}$$

where $\mathbf{1}[.]$ is the indicator function. Thus, the term $\sum_{k=1}^{N} \mathbf{1}[\delta(y_k^i, y_k^j) = l]$ captures the added congestion imposed by edge $(\mathbf{Y}^i, \mathbf{Y}^j) \in \mathcal{A}_{sf}$ on the lth edge in $\mathcal{A}$. Let $\mathbf{S}(i,j)$ be defined as the m-dimensional vector with the qth element

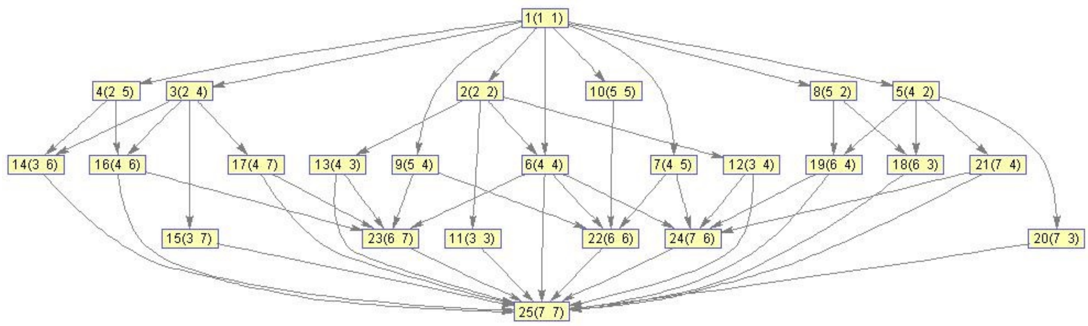

Fig. 3: Subflow-level graph showing all feasible combination of nodes via which subflows may travel

$s_q(i,j) = \sum_{k=1}^{N} \mathbf{1}[\delta(y_k^i, y_k^j) = q]$ and $q = 1, ..., m$. Therefore, (18) can be written in vector form as: $\mathbf{S}^i = \mathbf{S}^j + \mathbf{S}(i,j)$. It's worth to mention that, in contrast with the single-EV problem where we assume fixed parameters for the traveling time and energy consumption on each edge, for the multiple-EV problem these parameters are dependent on the traffic congestion (routing decision) which makes the problem much harder.

We define $Q(\mathbf{Y}^i, \mathbf{E^i})$ to be the minimum total elapsed time to the destination node in G_{sf} starting from node $\mathbf{Y}^i = (y_1^i, ..., y_N^i) \in \mathcal{N}_{sf}$ with $\mathbf{E}^i = (E_1^i, ..., E_N^i)$ units of energy. Our goal then is to determine $Q(\mathbf{Y}^1, \mathbf{E}^1)$ where $\mathbf{Y}^1 = (1, ..., 1)$ and $\mathbf{E}^1$ is a given amount of energy for the whole inflow (divided among suubflows) to the network. Let B^k be the maximum charging amount subflow k can receive based on its vehicle type. Then, we need to consider all possible feasible values of $\mathbf{E}^i = (E_1^i, ..., E_N^i)$ such that $E_j^i \in [0, B^j]$, $\forall j = 1, .., N$ and $\forall i$. To do so we need to dicretize this range accordingly.

The algorithm works based on the following DP formulation over the subflow-level graph. Similar to (7), $Q(\mathbf{Y}^i, \mathbf{E}^i)$ is calculated using the iterative equation

$$Q(\mathbf{Y}^i, \mathbf{E}^i) = \min_{\substack{\mathbf{Y}^j \in O\{\mathbf{Y}^i\},\ 0 \le E_k^j \le B^k \\ \text{s.t } 0 \le E_k^j - E_k^i + e_{y_i,y_j}^k(\mathbf{S}^j) \le B^k \\ k=1,...,N}} \Big[Q(\mathbf{Y}^j, \mathbf{E}^j) + \sum_{k=1}^{N} \tau_{y_i,y_j}^k(\mathbf{S}^j) + \sum_{k=1}^{N} (E_k^j - E_k^i + e_{y_i,y_j}^k(\mathbf{S}^j)) g_k^i + C(\mathbf{Y}^i, \mathbf{Y}^j, \mathbf{S}^j)\Big] \quad (19)$$

In (19), $Q(\mathbf{Y}^j, \mathbf{E}^j)$ denotes the minimum cost to go from node $\mathbf{Y}^j = (y_1^j, ..., y_N^j)$ with residual energies $\mathbf{E}^j = (E_1^j, ..., E_N^j)$ to the destination node. The one-step cost consists of three parts. First, $\sum_{k=1}^{N} \tau_{y_i,y_j}^k(\mathbf{S}^j)$ is the total elapsed time to travel from $\mathbf{Y}^i$ to $\mathbf{Y}^j$ in G_{sf}. The second term, $\sum_{k=1}^{N} (E_k^j - E_k^i + e_{y_i,y_j}^k(\mathbf{S}^j)) g_k^i$, shows the total recharging time, and the third term, $C(\mathbf{Y}^i, \mathbf{Y}^j, \mathbf{S}^j)$ is necessary to evaluate the added edge travel times and energy consumption resulting from the specific routing decision. Note that $\tau_{y_i,y_j}^k(\mathbf{S}^j)$ and $e_{y_i,y_j}^k(\mathbf{S}^j)$ are computed based on the corresponding s_l^i (updated based on a decision at node $\mathbf{Y}^i$). Adding the edge $(\mathbf{Y}^i, \mathbf{Y}^j) \in \mathcal{A}_{sf}$, may change the travel times on the arcs previously used in computing $Q(\mathbf{Y}^j, \mathbf{E}^j)$, and it should be modified accordingly. To do so, we add the term $C(\mathbf{Y}^i, \mathbf{Y}^j, \mathbf{S}^j)$:

$$C(\mathbf{Y}^i, \mathbf{Y}^j, \mathbf{S}^j) = \sum_{l \in A_{ij}} (s_l^j)[\tau_l(s_l^i) - \tau_l(s_l^j)] \quad (20)$$

where $A_{ij} = \{l : s_l^j > 0 \text{ and } s_l(i,j) > 0\}$ for $l = 1, ..., m$, is a set containing the intersection between edges in the route from node $\mathbf{Y}^j$ to the destination and edges in $(\mathbf{Y}^i, \mathbf{Y}^j) \in A_{sf}$.

Recall that the energy dynamics on the optimal path for each subflow are $E_k^j = E_k^i + r_i^k - e_{y_i,y_j}^k(\mathbf{S}^j)$, $k = 1, ..., N$, the constraint $0 \le E_k^j - E_k^i + e_{y_i,y_j}^k(\mathbf{S}^j) \le B^k$ is the feasibility constraint for amount of charge subflow k may receive at each node, r_i^k.

We seek $\lim_{k \to \infty} Q^{(k)}(\mathbf{Y}^i, \mathbf{E}^i) = Q^*$. In the sequel, we describe the detailed steps of the DP algorithm.

Initialization: Based on our analysis in [1] and [2], we know that if subflow k gets charge on the optimal path, the optimal residual energy at the destination for that subflow is zero. Therefore, assuming all subflows will get charge in their journey, it is obvious that the only option for the cost value at the destination node is $Q^{(0)}(\mathbf{Y}^D, \mathbf{E}^D = \mathbf{0}) = 0$ where D is the index of the destination node in the subflow-level graph, e.g. node 25 in Fig. 3. For the other nodes, motivated by Dijkstra's algorithm for the shortest path problem, we set the initial traveling time for all other cases to infinity, i.e.,

$$Q^{(0)}(\mathbf{Y}^D, \mathbf{E}^D) = \begin{cases} 0 & \text{if } \mathbf{E}^D = \mathbf{0}, \\ \infty & \text{if } \mathbf{E}^D > \mathbf{0}. \end{cases} \quad (21)$$

$$Q^{(0)}(\mathbf{Y}^i, \mathbf{E}^i) = \infty \quad \forall \mathbf{Y}^i \in \mathcal{N}_{sf} \setminus \mathbf{Y}^D, \quad 0 \le E_k^i \le B^k \quad (22)$$

Iteration Steps: The update of Q values can be carried out starting from any node. However, we start it at source node. The Q values are updated as follows: $Q^{(k)}(\mathbf{Y}^D, \mathbf{E}^D) = Q^{(0)}(\mathbf{Y}^D, \mathbf{E}^D)$ and

$\forall \mathbf{Y}^i \in \mathcal{N}_{sf} \setminus \mathbf{Y}^D, \quad 0 \le E_k^i \le B^k$:

$Q^{(k)}(\mathbf{Y}^i, \mathbf{E}^i) =$

$$\min_{\substack{\mathbf{Y}^j \in O\{\mathbf{Y}^i\},\ 0 \le E_k^j \le B^k \\ \text{s.t } 0 \le E_k^j - E_k^i + e^k_{y_i,y_j}(\mathbf{S}^j) \le B^k \\ k=1,\dots,N}} \Big[Q^{(k-1)}(\mathbf{Y}^j, \mathbf{E}^j) + \sum_{k=1}^{N} \tau^k_{y_i,y_j}(\mathbf{S}^j)$$

$$+ \sum_{k=1}^{N} (E_k^j - E_k^i + e^k_{y_i,y_j}(\mathbf{S}^j)) g_k^i + C(\mathbf{Y}^i, \mathbf{Y}^j, \mathbf{S}^j)\Big] \qquad (23)$$

The algorithm stops as soon as $Q^{(k)}(\mathbf{Y}^i, \mathbf{E}^i) = Q^{(k-1)}(\mathbf{Y}^i, \mathbf{E}^i) \quad \forall \mathbf{Y}^i \in \mathcal{N}_{sf}, \quad 0 \le E_k^i \le B^k \quad k = 1, ..., N.$

D. Numerical Examples

Consider the 7-node road network in Fig. 2 where the distance of each edge is shown. Similar to our previous work in [1] and [2], in order to model traffic congestion, the relationship between the speed and density of a vehicle flow is estimated as

$$v(k(t)) = v_f \left(1 - \left(\frac{k(t)}{k_{jam}}\right)^p\right)^q \qquad (24)$$

where v_f is the reference speed on the road without traffic, $k(t)$ represents the density of vehicles on the road at time t and k_{jam} denotes the saturated density for a traffic jam. The parameters p and q are empirically identified for actual traffic flows. we assume the energy consumption rates of subflows on arc $(i, j) \in \mathcal{A}$ are all identical, proportional to the distance between nodes i and j in the road network, giving $e^k_{y_i,y_j} = e \cdot d_l \cdot \frac{R}{N}$, where (y_k^i, y_k^j) corresponds to the lth edge in $\mathcal{A}$.

For simplicity we divide the total inflow R into N identical subflows, each of which has R/N vehicles per unit of time. Fig. 3 shows the subflow-level graph for this example for $N = 2$. Now in the subflow-level graph, the time subflow k spends on arc (y_k^i, y_k^j) becomes:

$$\tau^k_{y_i,y_j} = \big(d_l \cdot \frac{R}{N}\big)\big(v_f(1 - (\frac{s_l^i}{N})^p)^q\big)^{-1}$$

s_l^i determines the number of subflows (density) through this edge starting from node $\mathbf{Y}^i$ to the destination node $\mathbf{Y}^D$.

In order to examine the efficiency of the DP algorithm, we solve the problem for the network with homogeneous charging nodes with $g_i = 1 \ \ \forall i \in \mathcal{N}$ for different value of N. Table. II compares the solution and CPU times (computational effort) for different values of number of subflows. It is obvious from our results that as number of subflows, N, increases, DP loses its efficiency and will be computationally more expensive than MINLP. On the other hand, our analysis and numerical examples in [1] and [2] show that our proposed *flow control formulation* for the same problem results in a reduction of about 4 orders of magnitude in CPU time with approximately the same objective function value.

TABLE II: Numerical results for sample problem

	MINLP	**DP**
N	2	2
obj	116.67	116.67
routes	$1 \to 2 \to 3 \to 7$ $1 \to 4 \to 7$	$1 \to 2 \to 3 \to 7$ $1 \to 4 \to 7$
CPU time (sec)	1674.2	79.17
N	3	3
obj	99.68	99.68
routes	$1 \to 2 \to 3 \to 7$ $1 \to 4 \to 7$ $1 \to 5 \to 6 \to 7$	$1 \to 2 \to 3 \to 7$ $1 \to 4 \to 7$ $1 \to 5 \to 6 \to 7$
CPU time (sec)	1752.5	5534.6
N	6	6
obj	99.68	NA
routes	$1 \to 2 \to 3 \to 7(\times 2)$ $1 \to 4 \to 7(\times 2)$ $1 \to 5 \to 6 \to 7(\times 2)$	NA
CPU time (sec)	2579	NA

IV. Conclusions and Future Work

We have studied the problem of minimizing the total elapsed time for energy-constrained vehicles to reach their destinations, including recharging when there is no adequate energy for the entire journey. In [1] and [2], we studied the same problem for networks with homogeneous and inhomogeneous charging nodes respectively. Starting with a MINLP formulation, we derived computationally simpler formulations to solve different versions of the problem. Here, we have formulated it as a DP problem. For a single vehicle, this approach is very efficient and determines an optimal solution in seconds. For a multi-vehicle problem, where traffic congestion effects are considered, we used a similar approach by aggregating vehicles into subflows and seeking optimal routing decisions for each such subflow. In this case, our DP algorithm works well for a small number of subflows but as the number of subflows increases, it loses its efficiency. In [1] and [2] we developed an alternative flow-based formulation which yields approximate solutions with a computational cost reduction of several orders of magnitude, so this can be used in problems of large dimensionality. Numerical examples show these solutions to be near-optimal. For future research, it might be interesting to investigate the potential of using reinforcement learning algorithms that would aim the vehicles to learn online how to minimize the total elapsed time to reach their destinations. In this context, each vehicle through its daily interaction with other vehicles and exploration of different feasible routes could eventually learn the optimal one for a given commute.

References

[1] T. Wang, C. Cassandras, and S. Pourazarm, "Energy-aware vehicle routing in networks with charging stations," in *To appear in Proc. of 2014 IFAC World Congress-arXiv:1401.6478*.

[2] S. Pourazarm and C. Cassandras, "Optimal routing of energy-aware vehicle in networks with inhomogeneous charging nodes," in *Proc. of 22nd IEEE Mediterranean Conference on Control and Automation*, June 2014, pp. 674–679.

[3] G. Laporte, "The vehicle routing problem: An overview of exact and approximate algorithms," *European Journal of Operational Research*, vol. 59, 1992.

[4] A. Artmeier, J. Haselmayr, M. Leucker, and M. Sachenbacher, "The optimal routing problem in the context of battery-powered electric vehicles," in *Workshop: CROCS at CPAIOR-10, 2nd International Workshop on Constraint Reasoning and Optimization for Computational Sustainability*, Bologna, Italy, May 2010.

[5] J. Eisner, S. Funke, and S. Storandt, "Optimal route planning for electric vehicles in large networks," in *Proceedings of the 25th AAAI Conference on Artificial Intelligence*, San Francisco, US, Aug. 2011.

[6] U. F. Siddiqi, Y. Shiraishi, and S. M. Sait, "Multi-constrained route optimization for electric vehicles (evs) using particle swarm optimization," in *2011 11th International Conference on Intelligent Systems Design and Application (ISDA)*, Cordoba, Spain, Nov. 2011, pp. 391–396.

[7] S. Khuller, A. Malekian, and J. Mestre, "To fill or not to fill: The gas station problem," *ACM Transactions on Algorithms*, vol. 7, 2011.

[8] M. Schneider, A. Stenger, and D. Goeke, "The electric vehicle routing problem with time windows and recharging stations," Tech Report, Dept. of Business Information Systems and Operations Research, University of Kaiserslautern, 2012.

[9] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, 4th ed. Athena Scientific, 2012, vol. I.