A bi-level model for periodical minimizing number of energy-depleting sensors in multi-charging WRSN

Vuong Dinh An

School of Information Communication Technology

Ngày 25 tháng 10 năm 2020

Introduction and some notations

Given a set of sensors $\mathbf{S} = \{S_1, S_2, \ldots S_n\}$ deployed in a Cartesian coordinate $W \times H$. Suppose we've already found the set of charging locations $\mathbf{D} = \{D_1, D_2, \ldots, D_m\}$. Our problem is formulated as follow:

$$F = (\boldsymbol{\pi}, \boldsymbol{\tau})$$

Where, $\boldsymbol{\pi} = \{\pi_1, \pi_2, \ldots, \pi_m\}$ and $\boldsymbol{\tau} = \{\tau_1, \tau_2, \ldots, \tau_m\}$ are charging locations visiting order and charging locations stopping time, respectively. For short, we denote these two variables as charging path and charging time, respectively.

Introduction and some notations

- Let P_m be the set of all permutations of the set $\{1, 2, \ldots, m\}$.
- Let C_m be the set of all Hamilton cycles of MC visiting the set **D**.
- Let p_j be the energy consumption of sensor S_j, $j = 1, 2, \ldots n$.
- Let $P_{u,j}$ be the charging rate from charging location D_u to sensor S_j, $u = \overline{1, m}, j = \overline{1, n}$

Bi-level relation

Since a visiting order $\{\pi_1, \pi_2, \ldots, \pi_m\}$ is a permutation of $\{1, 2, \ldots, m\}$.
Thus, $\boldsymbol{\pi} \in P_m$.
For each $\boldsymbol{\pi}$, let $\mathcal{T}(\boldsymbol{\pi})$ be the set of all feasible charging time of MC at charging locations. Let $\mathcal{T} = \cap_{\pi \in P_m} \mathcal{T}(\boldsymbol{\pi})$
Thus, our problem becomes: Let $F : P_m \times \mathcal{T} \to \mathbb{N}$

$$\min F = (\boldsymbol{\pi}, \boldsymbol{\tau})$$

Such that $\boldsymbol{\tau} \in \mathcal{T}(\boldsymbol{\pi})$
For each $\boldsymbol{\pi}$, we call a pair ($\boldsymbol{\pi}$, $\boldsymbol{\tau}$) a feasible solution.
This relation is bi-level-like, but instead of a lower optimize function, it requires a upper-dependant set. In another word, once can easily follow that each charging time is only feasible with **at least** one charging path.

Bi-level relation

However, we can transform the problem into exactly bi-level model as follow. Let $F : P_m \times \mathcal{T} \to \mathbb{N}$

$$\min F = (\boldsymbol{\pi}, \boldsymbol{\tau})$$

Such that $\boldsymbol{\tau} \in \text{argmin}_{\tau \in \mathcal{T}} F(\boldsymbol{\pi}, \boldsymbol{\tau})$.
Where lower objective function is exact as the upper objective function.
We deal with the feasible set $\mathcal{T}(\boldsymbol{\tau})$ as follow, if $(\boldsymbol{\pi}, \boldsymbol{\tau})$ is not a feasible solution, F will get value $+\infty$ at this point.
Hence, the model becomes exact bi-level model.

Determine total charging time for a period

Since the model is based for optimizing in one period, our proposal algorithm begins with determining the total charging time for a period. From the trivial equation:

$$T = t_{\text{charge}} + t_{\text{move}}$$

And

$$\tau_1 + \tau_2 + \ldots \tau_m = t_{\text{charge}}$$

One can be referred that:

$$\tau_1 + \tau_2 + \ldots \tau_m = T - t_{\text{move}}$$

Consider $\tau_1, \tau_2, \ldots \tau_n$ be symmetric random variables on their domains. Apply expectation formula, it follows:

$$E(\tau_1) + E(\tau_2) + \cdots + E(\tau_m) = T - E(t_{\text{move}})$$

Determine expected value of travelling time

Consider t_{move} be an random variable on its domain and consider c as an discrete random variable on C_m.
Each $c \in C_m$ map into some travelling time t_{move} and since the velocity is considered unchanged, we have a linear relation. By taking expected value of two quantities, one can easily verify that:

$$E(t_{\text{move}}) = \frac{E(c)}{V} \tag{1}$$

In order to calculate $E(t_{\text{move}})$ one can calculate $E(c)$ and follows from the above equation.

Determine expected value of travelling time

Consider the distribution for c is the uniform distribution. Hence,
$E(c) = \frac{\sum C}{|C_m|}$

- To calculate $\sum C$, each $d_{i,j}$ involves into $(m-2)!$ different cycles. Hence:

$$\sum C = (m-2)! \sum_{1 \le i < j \le m} d_{i,j}$$

- $|C_m| = \frac{(m-1)!}{2}$

$$\Rightarrow E(c) = \frac{\sum_{1 \le i < j \le m} d_{i,j}}{\frac{m-1}{2}} = \frac{\sum_{1 \le i < j \le m} d_{i,j}}{\frac{m(m-1)}{2}} \times m = \overline{d} \times m$$

Where $\overline{d}$ is average distance between two sensors in the given Cartesian coordinate. Thus, the expected value of travelling time can be calculated by (1).

Determine expected value of charging time

Recall the equation:

$$E(\tau_1) + E(\tau_2) + \cdots + E(\tau_m) = T - E(t_{\text{move}})$$

Consider $\tau_1, \tau_2, \ldots \tau_m$ be symmetric variables (One can ignore the feasible constraint by taking account into general domain), thus:

$$E(\tau_1) = E(\tau_2) = \cdots = E(\tau_m)$$

Hence:

$$E(\tau_i) = \frac{T - E(t_{\text{move}})}{m} = \frac{T - \dfrac{\overline{d} \times m}{V}}{m}$$

Determine expected value of remaining energy of sensor

The remain energy of sensor S_j after the period with length T is formulated as follow:

$$E_j = E_j^{\text{remain}} - Tp_j + \sum_{u=1}^{m}(P_{u,j} \times \tau_u)$$

Let $P_j = \sum_{u=1}^{m} P_{u,j}$ be the total energy gaining from all charging locations of sensor S_j. By taking expected value both sides:

$$E(E_j) = E_j^{\text{remain}} - Tp_j + \sum_{u=1}^{m}[P_{u,j} \times E(\tau_u)]$$

Thus,

$$E(E_j) = E_j^{\text{remain}} - Tp_j + P_j \times \frac{T - \dfrac{\overline{d} \times m}{V}}{m}$$

Determining period length problem statement

Hence, we simplify the above equation into linear form:

$$E(E_j) = A_j + TB_j$$

$$A_j = E_j^{\text{remain}} - \frac{\overline{d}P_j}{V} \text{ and } B_j = \frac{P_j}{m} - p_j, j = \overline{1, n}. \tag{2}$$

Since the goal is to finding a period length, intuitively, our expected energy of S_j, $j = \overline{1, n}$ should get as much as possible. Thus, we obtain the period length T by optimizing the following problem:

Problem Statement

Determine $T \in \mathbb{R}^+$ such that:

$$\max\{\min_{j=\overline{1;n}} E(E_j)\}$$

Where $E(E_j) = A_j + TB_j, j = 1, 2, \ldots, n$. With $\boldsymbol{A}, \boldsymbol{B}$ define as (2).

We begin the proposal algorithms with a iterative genetic-based algorithm as follow:

- Let the upper level in the mentioned bi-level relation be charging path π . Let the lower level be charging time τ . The domain for evolving two levels called upper society and lower society, respectively.
- Suppose an genetic algorithm named *GAUL* used for evolving upper society and *GALL* used for evolving the lower society.
- The fitness of lower individual is denoted by $F = (\pi_{\text{upper}}, \tau_{\text{lower}})$. The fitness ofupper individual is denoted by $F = (\pi_{\text{upper}}, \tau_{\text{lower}})$, where $\tau_{\text{lower}} \in \text{argmax} F$ where $\pi_{\text{upper}}$is fixed and τ_{lower}varies in its domain set.
- Let $G : P_m \to \mathbb{N}$ be an linear approximate function for upper level. Once can inferred from the fact that our objective function F is take discrete values, hence, its derivative can't be determined. Let ϵ be an threshold for approximating value of G at some points.

Genetic simulation settings

Upper level		
Quantity	Notation	Value
Population	POP_U	15
Number of generation	N_U	15
Crossover rate	C_U	0.7
Mutation rate	M_U	0.1
Lower level		
Quantity	Notation	Value
Population	POP_L	15
Number of generation	N_L	25
Crossover rate	C_L	0.7
Mutation rate	M_L	0.1

Step by step for nested algorithm

- **Step 1:** Initiate upper society with uniformly choose POP_U individuals from set of permutations P_m.
- **Step 2:** Conduct crossover in the upper society using GAUL. The society now may contain more than POP_U individuals.
- **Step 3:** Conduct mutation in the upper society using GAUL.
- **Step 4:** For each upper individual in society, we initiate a lower society with size POP_L. We evolve the lower society by utilizing GALL in traditional genetic algorithm flow. If the lower society is completed evolved, tag the respective upper individual 1, else, tag the upper individual 0. At this step, the upper individual fitness is also calculated.

Step by step for nested algorithm

- **Step 5:** Update the approximate function G with individual tag 1.
- **Step 6:** Select the best individuals in POP_U by GAUL filter.
- **Step 7:** Checking the terminated condition by GAUL criterion. If not meet the termination, the algorithm iterates to step 2.
- **Step 8:** Output the best upper individual with its fitness, charging path and charging time.

The second proposed algorithm is to improve performance of the first one. It also based on bi-level relation and genetic algorithm.

- Remain the bi-level model of charging path and charging time. Remain the GAUL for evolving upper individual.
- The lower level is optimized by a heuristic algorithm, the searching space is $\mathcal{T}(\ \pi_{\text{upper}})$. This is a huge impact, compare to GALL, since the searching space of genetic algorithm is not adequate.
- We simplify the optimization lower level to a linear programming model with some intuitive heuristic constraints.

Heuristic constraints

Since after the T-length charging period, we want as fewest as possible number of depleted sensors. Thus, one of heuristics is to keep the network as much energy as possible. The remain energy of sensor S_j after the period is:

$$E_j^{\text{after}} = E_j^{\text{remain}} - Tp_j + \sum_{u=1}^{m} (P_{u,j} \times \tau_u)$$

Note that, the above relation show the linearity among E_j^{after} and $\boldsymbol{\tau}$. Our goal is to maximize the minimum energy of all sensors after the period. The constraint is sum of $\boldsymbol{\tau}$is up to period length T. Also, the energy transmission from MC is not exceed E_{MC}.

Heuristic model for lower level

The lower level optimization becomes:

Problem Statement

Determine $\boldsymbol{\tau} \in \mathcal{T}$ such that:

$$\max\{\min_{j=1;n} E_j^{remain} - Tp_j + \sum_{u=1}^{m}(P_{u,j} \times \tau_u)\}$$

s.t

$$\sum_{u=1}^{m} \tau_u = T$$

$$\sum_{j=1}^{n}\sum_{u=1}^{m}(P_{u,j} \times \tau_u) \leq E_{MC}$$

Step by step for genetic and heuristic hybrid algorithm

We utilize GAUL implement as in the first algorithm to evolve the upper level society and a heuristic algorithm based on linear programming to find exact the lower level individual satisfy our heuristic, we named the thereafter algorithm $LPLL$. The algorithm is conducted step by step as following:

- **Step 1:** Initiate upper society with uniformly choose POP_U individuals from set of permutations P_m.
- **Step 2:** Conduct crossover in the upper society using GAUL. The society now may contain more than POP_U individuals.
- **Step 3:** Conduct mutation in the upper society using GAUL.

- **Step 4:** For each upper individual in society, utilize the LPLL to find exact lower level individual respect to this upper individual.
- **Step 5:** Select the best individuals in POP_U by GAUL filter.
- **Step 6:** Checking the terminated condition by GAUL criterion. If not meet the termination, the algorithm iterates to step 2.
- **Step 7:** Output the best upper individual with its fitness, charging path and charging time.