

A SIMULATED ANNEALING APPROACH TO THE BATTERY ELECTRIC BUS CHARGING PROBLEM

Alexander Brown

College of Engineering
Utah State University

May 5, 2024



College of Engineering
UtahStateUniversity

Outline

Introduction

The Position Allocation Problem Approach With Linear Battery Dynamic

The Simulated Annealing Approach With Linear Battery Dynamics

TODO The Simulated Annealing Approach With Non-Linear Battery Dynamics



Topic

Introduction

The Position Allocation Problem Approach With Linear Battery Dynamic

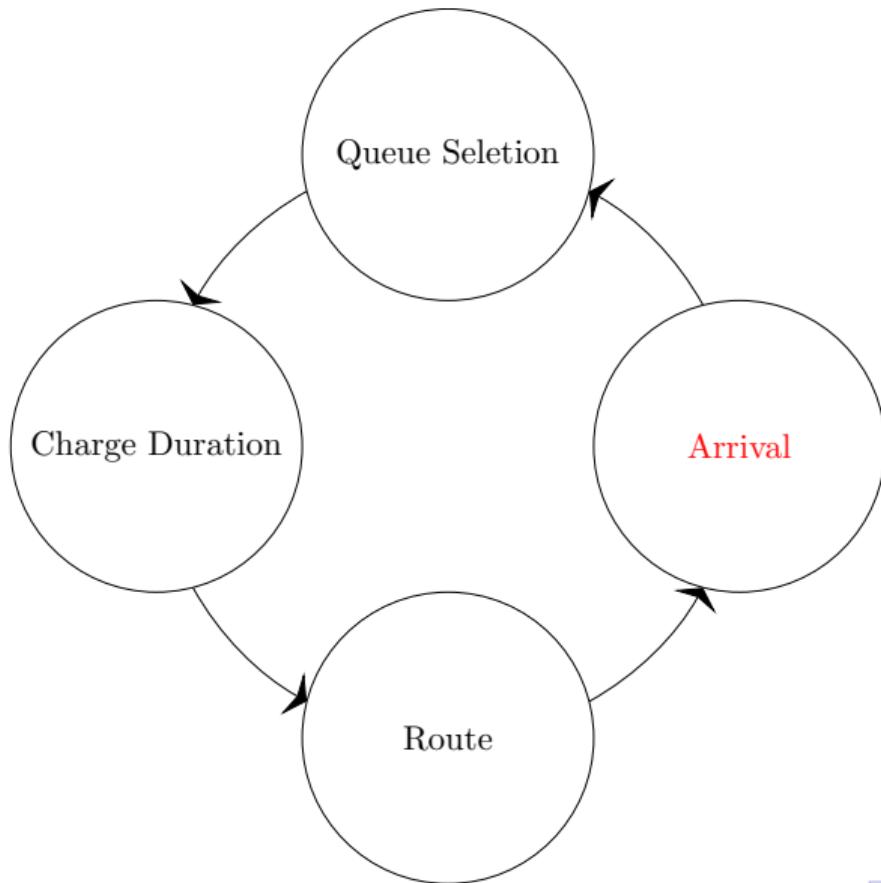
The Simulated Annealing Approach With Linear Battery Dynamics

TODO The Simulated Annealing Approach With Non-Linear Battery Dynamics



Brief State Of The Art

Problem Description



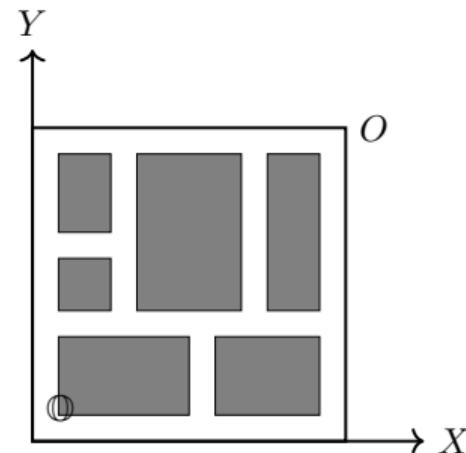
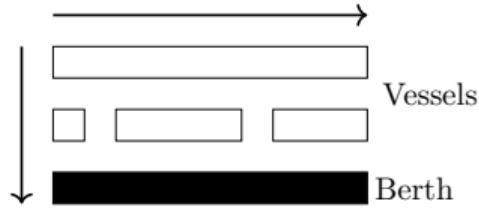
The Berth Allocation Problem¹



¹<https://www.mdpi.com/2077-1312/11/7/1280>

The Berth Allocation Problem

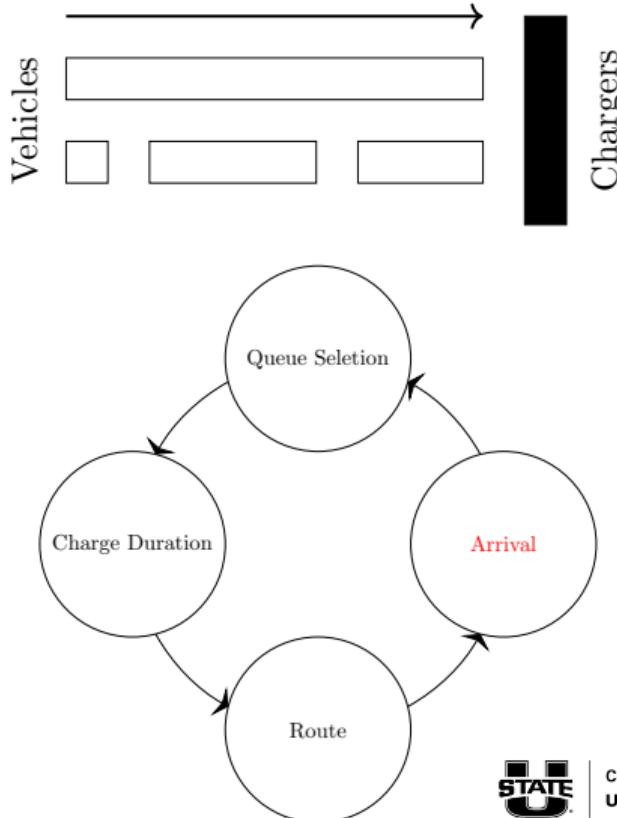
- ▶ Vessels move down toward the quay
 - ▶ Receive service
 - ▶ Exit to the right
-



- ▶ A variant of the rectangle packing problem
- ▶ Solves the problem of optimally assigning incoming vessels to be serviced

The Position Allocation Problem

- ▶ Service flow is left to right
 - ▶ Single charger type
 - ▶ All arrivals are considered unique
 - ▶ Service times are assumed to be known



Topic

Introduction

The Position Allocation Problem Approach With Linear Battery Dynamic

The Simulated Annealing Approach With Linear Battery Dynamics

TODO The Simulated Annealing Approach With Non-Linear Battery Dynamics



Requirements For BEB Implementation

- ▶ Charges must be able to be tracked
- ▶ Service time is unknown
- ▶ Accommodate chargers of different speeds
- ▶ Minimize charger count
- ▶ Minimize consumption cost
- ▶ Encourage slow charger use for battery health



Constraints

$$\min \sum_{i=1}^{n_V} \sum_{q=1}^{n_Q} \left(w_{iq} m_q + g_{iq} \epsilon_q \right)$$

$$u_j - u_i - s_i - (\sigma_{ij} - 1)T \geq 0$$

$$s_i + u_i = d_i$$

$$v_j - v_i - (\psi_{ij} - 1)n_Q \geq 1$$

$$a_i \leq u_i \leq (T - s_i)$$

$$\sigma_{ij} + \sigma_{ji} \leq 1$$

$$d_i \leq \tau_i$$

$$\psi_{ij} + \psi_{ji} \leq 1$$

$$\eta_i + \sum_{q=1}^{n_Q} g_{iq} r_q - \Delta_i = \eta_{\gamma_i}$$

$$\sigma_{ij} + \sigma_{ji} + \psi_{ij} + \psi_{ji} \geq 1$$

$$\eta_i + \sum_{q=1}^{n_Q} g_{iq} r_q - \Delta_i \geq \nu_{\Gamma_i} \kappa_{\Gamma_i}$$

$$\sum_{q=1}^{n_Q} w_{iq} = 1$$

$$\eta_i + \sum_{q=1}^{n_Q} g_{iq} r_q \leq \kappa_{\Gamma_i}$$

$$v_i = \sum_{q=1}^{n_Q} q w_{iq}$$

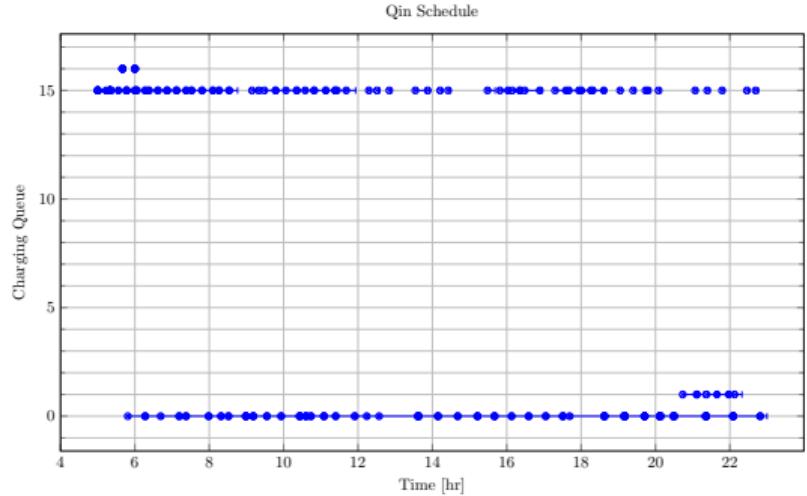
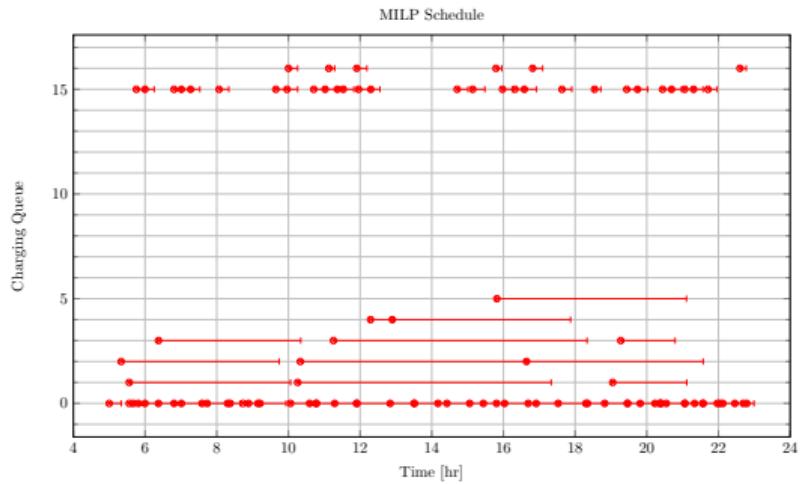


College of Engineering
UtahStateUniversity

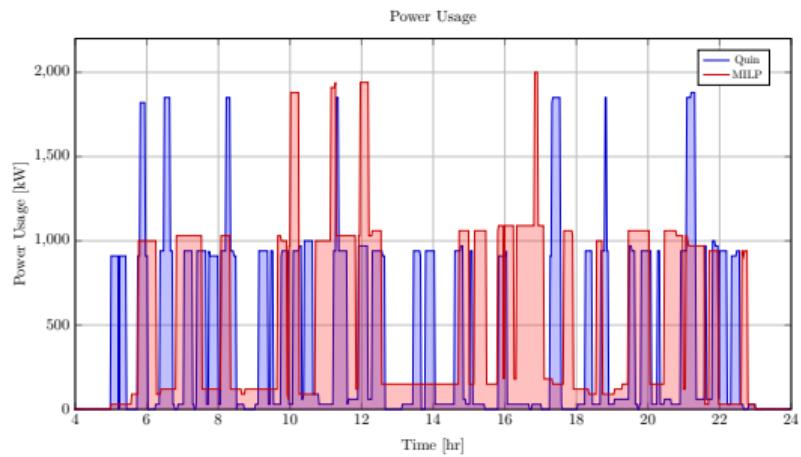
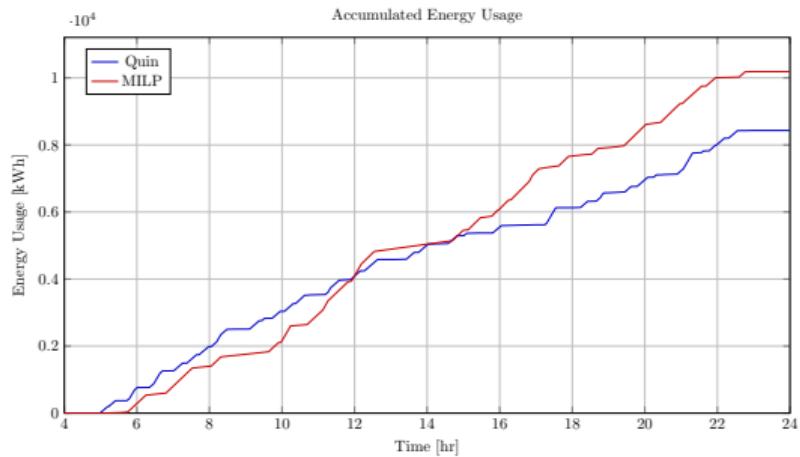
Results

- ▶ Executed for 7200 seconds (2 hours)
- ▶ $T = 24$
- ▶ $n_V = 338$
- ▶ $n_A = 35$
- ▶ $\alpha_i = 90\%$; $\nu_i = 20\%$; $\beta_i = 70\%$
- ▶ $\forall q \in \{n_B + 1, n_B + 2, \dots, n_B + n_C\}; m_q = 1000q$

Schedules



Power And Energy Use



Topic

Introduction

The Position Allocation Problem Approach With Linear Battery Dynamic

The Simulated Annealing Approach With Linear Battery Dynamics

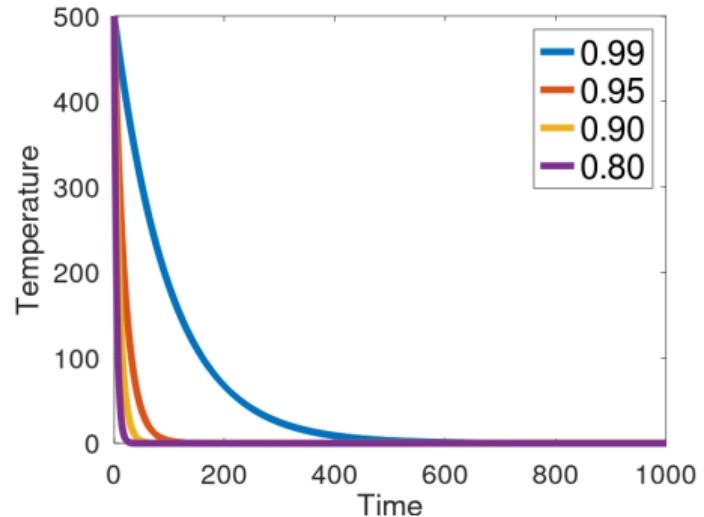
TODO The Simulated Annealing Approach With Non-Linear Battery Dynamics



Simulated Annealing

Cooling Schedule

- ▶ The cooling equation models the rate at which the temperature decreases over time in the SA process.
- ▶ The temperature is high, SA encourages exploration. As the temperature decreases, exploitation is encouraged.



$$t_m = \beta t_{m-1}$$



Acceptance Criteria

$$f(\mathbb{I}, \bar{\mathbb{I}}, t_m) = \begin{cases} 1 & \Delta E > 0 \\ e^{-\frac{\Delta E}{t_m}} & \text{otherwise} \end{cases} \quad (1)$$



Generation Mechanisms - Primitive Functions

- ▶ New Visit: Move a bus from a wait queue to charge queue
- ▶ Slide Visit: Change the charge duration of a visit
- ▶ New Charger: Move a visit to a new charger
- ▶ Wait: Move a visit to its idle queue
- ▶ New Window: Execute Wait then New Visit primitives



Generation Mechanisms - Wrapper Functions

- ▶ Charge Schedule Generation: Iterate through each visit and execute New Visit
- ▶ Perturb Schedule: Randomly execute one of the primitives with a weighted distribution

Objective Function

$$J(\mathbb{I}) = z_d p_d + \sum_{i=1}^{n_V} \left[\epsilon_{q_i} r_{q_i} + z_p \phi_i (\eta_i - \nu_{b_i} \kappa_{b_i}) + z_c r_{q_i} s_i \right]$$

► Demand cost

- $p_{T_p}[h] = \frac{1}{T_p} \sum_{h-\frac{T_p}{dt}+1}^h p_h$
- $p_d = \max(p_{fix}, p_{max})$
- $p_{max} = \max_{h \in H} p_{T_p}[h]$

- $\epsilon_{q_i} r_{q_i}$: Assignment Cost
- $z_p \phi_i (\eta_i - \nu_{b_i} \kappa_{b_i})$: Penalty Function
- $z_c r_{q_i} s_i$: Consumption Cost



Constraints

$$u_j - d_i - (\sigma_{ij} - 1)T \geq 0$$

$$q_j - q_i - 1 - (\psi_{ij} - 1)Q \geq 0$$

$$\sigma_{ij} + \sigma_{ji} \leq 1$$

$$\psi_{ij} + \psi_{ji} \leq 1$$

$$\sigma_{ij} + \sigma_{ji} + \psi_{ij} + \psi_{ji} \geq 1$$

$$s_i = d_i - u_i$$

$$\eta_{\xi_i} = \eta_i + r_{q_i} s_i - \Delta_i$$

$$\kappa_{\Xi_i} \geq \eta_i + r_{q_i} s_i$$

$$a_i \leq u_i \leq d_i \leq e_i \leq T$$



Results

Model	Execution Time [s]	Iteration [s]
MILP	1900	N/A
Quick	1532.8	0.4
Heuristic	1916	0.5

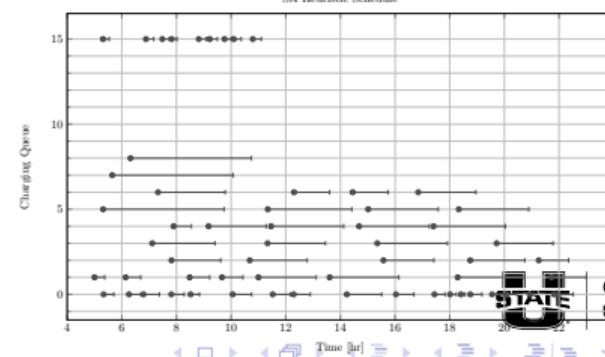
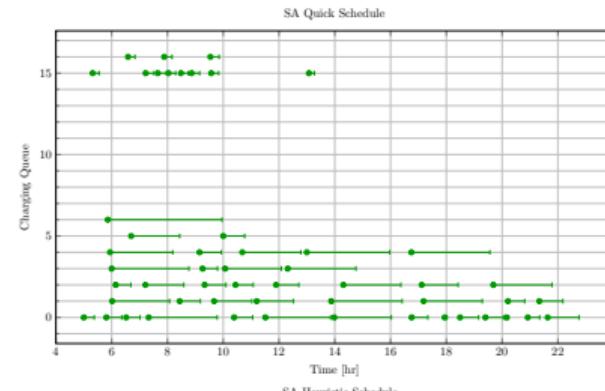
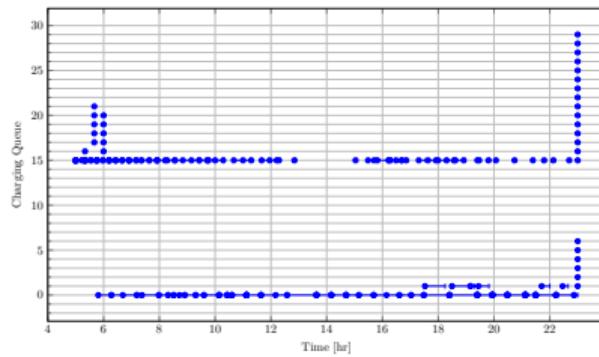
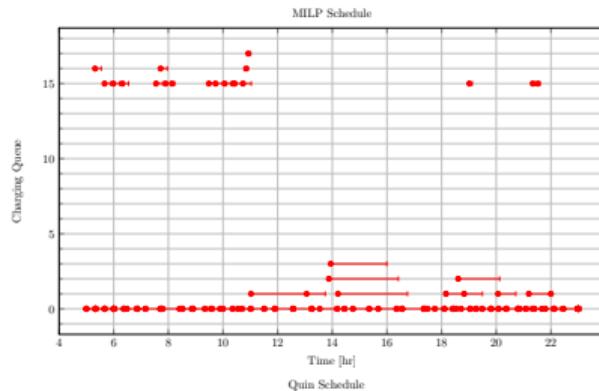
- ▶ $T_0 = 90000$
- ▶ $\beta = 0.997$
- ▶ $|t| = 3797$
- ▶ $n_K = 500$



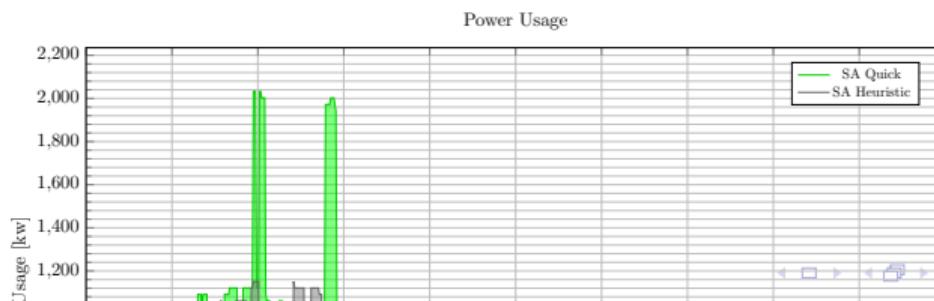
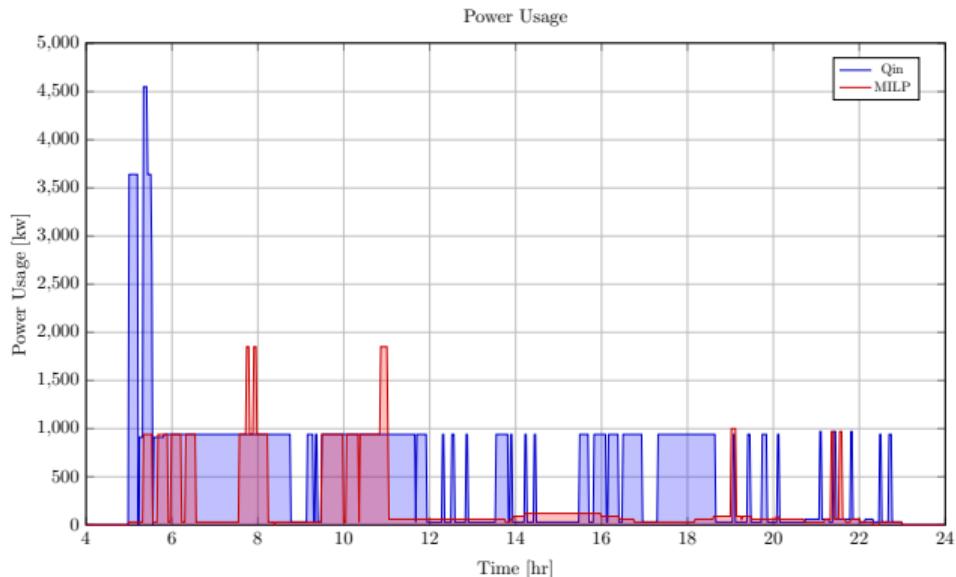
Schedule

- ▶ Point 1
- ▶ Point 2

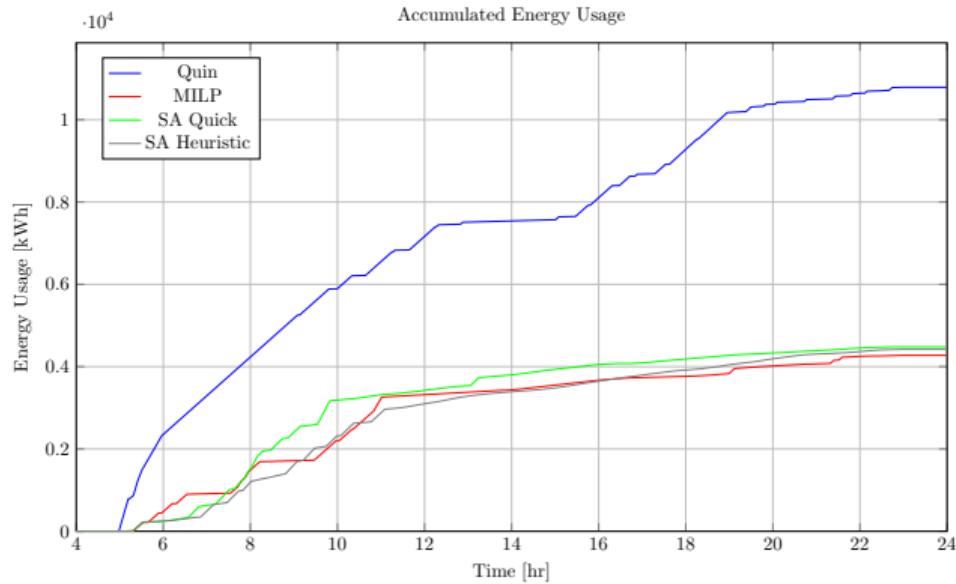
- ▶ Point 3
- ▶ Point 4



Power



Energy



Topic

Introduction

The Position Allocation Problem Approach With Linear Battery Dynamic

The Simulated Annealing Approach With Linear Battery Dynamics

TODO The Simulated Annealing Approach With Non-Linear Battery Dynamics



Introduction

- ▶ Higher fidelity in approximating charge at high SOC
- ▶ Implemented in SA for simplicity



Non-Linear Battery Dynamics Model

- ▶ Show function
- ▶ Show plots



Results

- ## ► Figures!

Appendix



College of Engineering
UtahStateUniversity

Mixed Integer Linear Programming*

$$\max J = \sum_j c_j x_j + \sum_k d_k y_k$$

$$\text{subject to } \sum_j a_{ij} x_j + \sum_k g_{ik} y_k \leq b_i \quad (i = 1, 2, \dots, m)$$

$$x_j \geq 0 \quad (j = 1, 2, \dots, n)$$

$$y_k \in \mathbb{Z}^+ \quad (k = 1, 2, \dots, n)$$

- ▶ J : Objective function
- ▶ $x_j \in \mathbb{R}$ and $y_k \in \mathbb{Z}^+$: Decision Variables
- ▶ $c_j, d_k, a_{ij}, g_{ik}, b_i \in \mathbb{R}$: Parameters



New Visit*

Algorithm 1: New visit algorithm

Algorithm: New Visit

Input: \mathbb{S}

Output: $\bar{\mathbb{S}}$

1 **begin**

```
2    $i \leftarrow \mathbb{S}_i;$                                 /* Extract visit index */
3    $a \leftarrow \mathbb{I}_{i.a};$                             /* Extract the arrival time for visit  $i$  */
4    $e \leftarrow \mathbb{I}_{i.e};$                             /* Extract the departure time for visit  $i$  */
5    $q \leftarrow \mathbb{I}_{i.q};$                             /* Extract the current charge queue for visit  $i$  */
6    $\bar{q} \leftarrow \mathcal{U}_Q;$                          /* Select a random charging queue with a uniform distribution */
7    $C \leftarrow \mathcal{U}_{\mathbb{C}_q};$                       /* Select a random time slice from  $\mathbb{C}_q$  */
8   if  $(\bar{C}, \bar{u}, \bar{d}) \leftarrow \text{findFreeTime}(C, i, q, a, e) \notin \emptyset$  then    /* If there is time available in  $C_q^j$  */
9     | return  $(i, (\bar{q}, \bar{u}, \bar{d}), \bar{C})$                                 /* Return visit */
10  end
11  return  $(\emptyset);$                                 /* Return nothing */
12 end
```



Algorithm 2: Slide Visit Algorithm

Algorithm: Slide Visit

Input: \mathbb{S}

Output: $\bar{\mathbb{S}}$

```
1 begin
2    $(i, \mathbb{I}, \bar{\mathbb{C}}) \leftarrow \text{Purge}(\mathbb{S});$            /* Purge visit  $i$  from charger availability matrix */
3    $C \leftarrow \bar{\mathbb{C}}_{i,q_i};$                          /* Get the time availability of the purged visit */
4   /* If there is time available in  $C$  */
4   if  $(\bar{C}, \bar{u}, \bar{d}) \leftarrow \text{findFreeTime}(C, \mathbb{S}_i, \mathbb{I}_q, \mathbb{I}_{i.a}, \mathbb{I}_{i.e}) \notin \emptyset$  then
5     | return  $(i, \mathbb{I}, (\mathbb{I}_{i.q_i}, \bar{u}, \bar{d}), \bar{\mathbb{C}})$           /* Return updated visit */
6   end
7   return  $(\emptyset);$                                      /* Return nothing */
8 end
```



New Charger

Algorithm 3: New Charger Algorithm

Algorithm: New Charger

Input: \mathbb{S}

Output: $\bar{\mathbb{S}}$

1 **begin**

```
2    $(i, \mathbb{I}, \bar{\mathbb{C}}) \leftarrow \text{Purge}(\mathbb{S});$            /* Purge visit  $i$  from charger availability matrix */
3    $q \leftarrow \mathcal{U}_Q;$            /* Select a random charging queue with a uniform distribution */
4   if  $(\bar{C}, \bar{u}, \bar{d}) \leftarrow \text{findFreeTime}(\bar{\mathbb{C}}_{i,q}, \mathbb{S}_i, \mathbb{I}_q, \mathbb{I}_{i,a}, \mathbb{I}_{i,e}) \notin \emptyset$  then /* If there is time available in
       $C_q$  */
      /* Return visit, note  $u$  and  $d$  are the original initial/final charge times. */
      return  $(i, \mathbb{I}, (q, \mathbb{I}_{i,u}, \mathbb{I}_{i,d}), \bar{\mathbb{C}})$ 
6   end
7   return  $(\emptyset);$            /* Return nothing */
8 end
```



Algorithm 4: Wait algorithm

Algorithm: Wait

Input: \mathbb{S}

Output: $\bar{\mathbb{S}}$

```
1 begin
2    $(i, \mathbb{I}, \bar{\mathbb{C}}) \leftarrow \text{Purge}(\mathbb{S});$  /* Purge visit  $i$  from charger availability matrix */
3    $\bar{\mathbb{C}}'_{\mathbb{I}_{i.r_i}} \leftarrow \mathbb{C}' \cup \{[\mathbb{I}_{i.a}, \mathbb{I}_{i.e}]\};$  /* Update the charger availability matrix for wait queue  $\bar{\mathbb{C}}_{i.q_i}$  */
4   return  $(i, \mathbb{I}, (\mathbb{I}_{i.b}, \mathbb{I}_{i.a}, \mathbb{I}_{i.e}), \bar{\mathbb{C}})$  /* Return visit */
5 end
```



New Window

Algorithm 5: New window algorithm

Algorithm: New Window

Input: \bar{S}

Output: $\bar{\bar{S}}$

```
1 begin
2   |    $\bar{\bar{S}} \leftarrow \text{Wait}(\bar{S});$            /* Assign visit to its respective idle queue */
3   |   if  $\bar{\bar{S}} \leftarrow \text{NewVisit}(\bar{S}) \neq \emptyset$  then
4   |   |   return  $\bar{\bar{S}}$                          /* Add visit  $i$  back in randomly */
5   |   end                                         /* Return visit */
6   |   return ( $\emptyset$ );                          /* Return nothing */
7 end
```



Charge Schedule Generation*

Algorithm 6: Charge schedule generation algorithm

Algorithm: Candidate Solution Generator

Input: \mathbb{S}

Output: $\bar{\mathbb{S}}$

```
1 begin
2   /* Select an unscheduled BEB visit from a randomly indexed set of visits */
3   foreach  $\mathbb{I}_i \in \mathbb{I}$  do
4     |  $(i, \bar{\mathbb{I}}, \bar{\mathbb{C}}) \leftarrow \text{NewVisit}(\mathbb{I}_i, \mathbb{I}, \mathbb{C})$ ;           /* Assign the bus to a charger */
5   end
6   return  $(0, \bar{\mathbb{I}}, \bar{\mathbb{C}})$ 
7 end
```

Algorithm 7: Perturb schedule algorithm

Algorithm: Perturb Schedule

Input: \mathbb{S}

Output: $\bar{\mathbb{S}}$

```
1 begin
2    $p \leftarrow [\text{false}; n_A]$ ;          /* Create vector to track priority routes */
3    $y^i \leftarrow [1.0; n_V]$ ;           /* Create weight vector for index selection */
4   /* Loop through the visits in reverse order
5   foreach  $\mathbb{I}_i \leftarrow \mathbb{I}_{|\mathbb{I}|} \text{ TO } \mathbb{I}_1$  do
6     /* If the current visit is part of a priority route
7     if  $p_{\mathbb{I}_{i,b}} = \text{true}$  then
8       |  $y_{\mathbb{I}_i}^j = y_{\mathbb{I}_{i,\xi}}^j$ ;
9     end
10    /* Else if the current visit's SOC does below the allowed threshold */
```

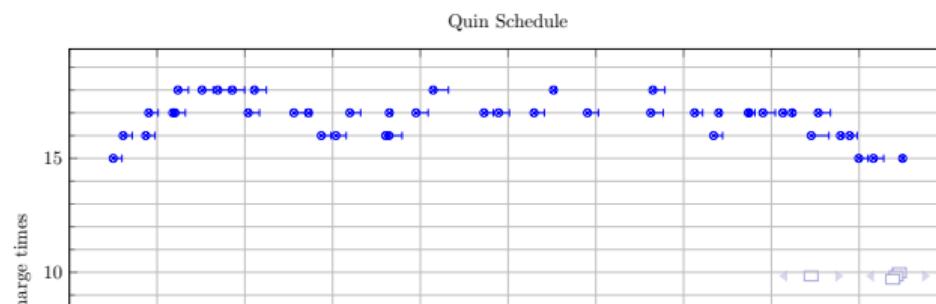
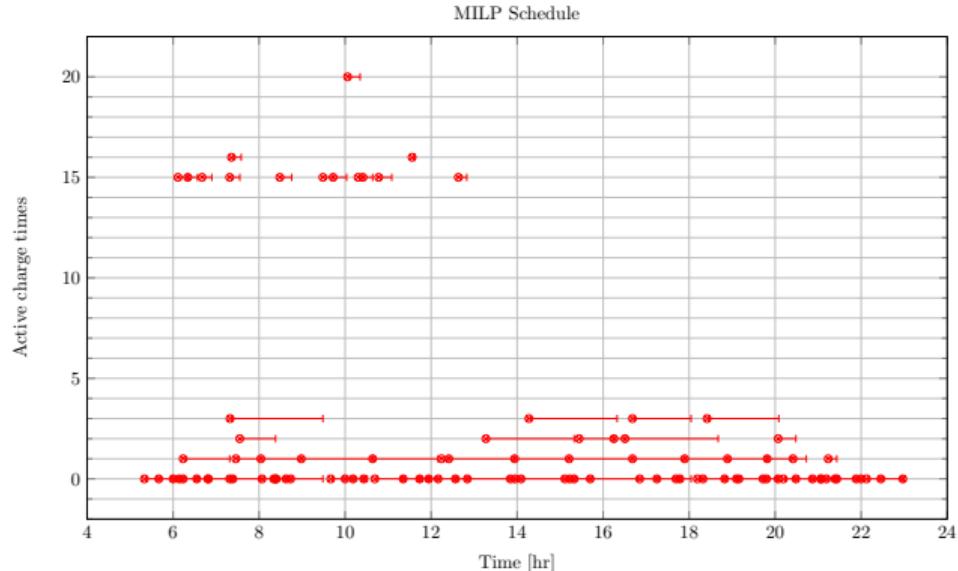
Results - What Is In The Thesis*

Model	Execution Time [s]	Iteration [s]
MILP	3600	N/A
Quick	2275.25	0.25
Heuristic	3640.4	0.4

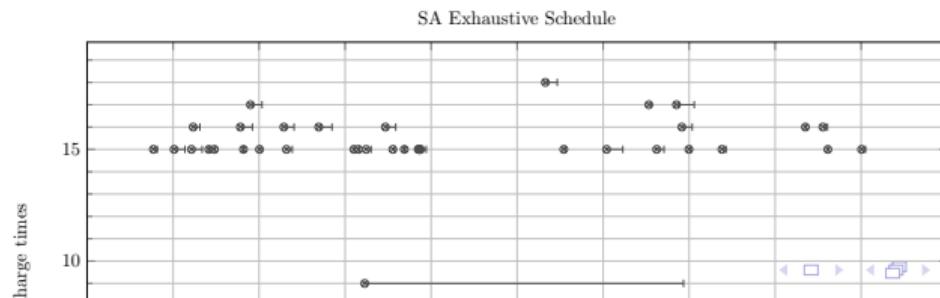
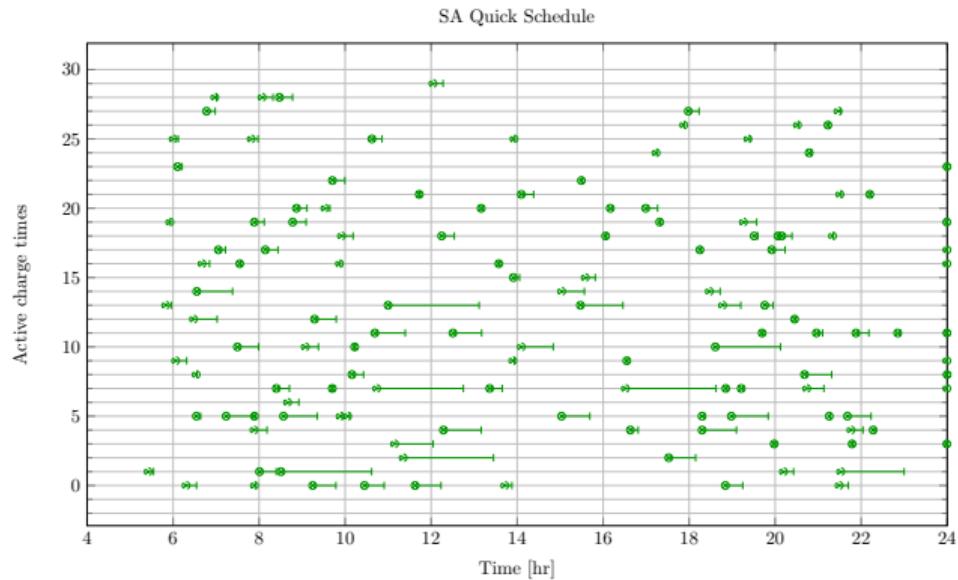
- ▶ $T_0 = 99999$
- ▶ $\beta = 0.999$
- ▶ $|t| = 3797$
- ▶ $n_K = 500$



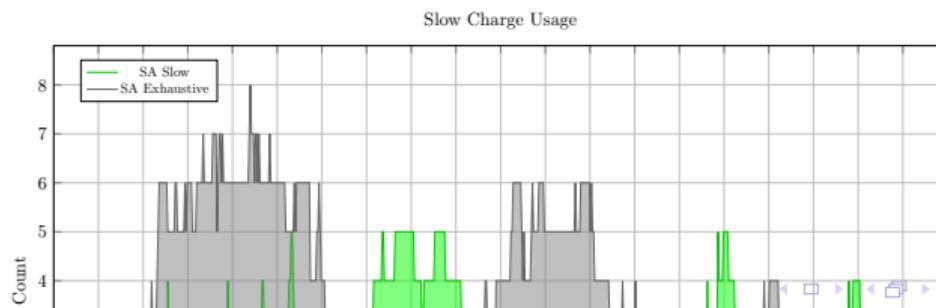
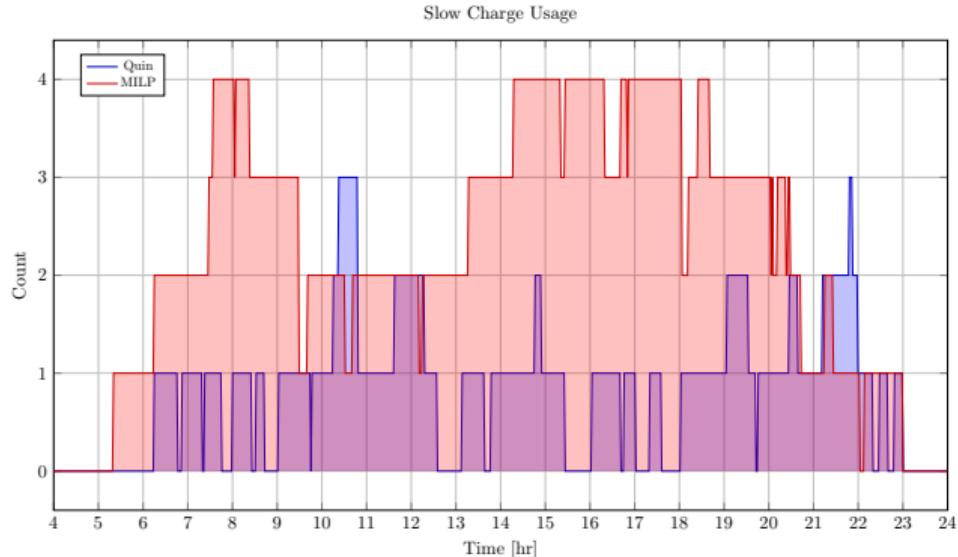
Schedule- What Is In The Thesis



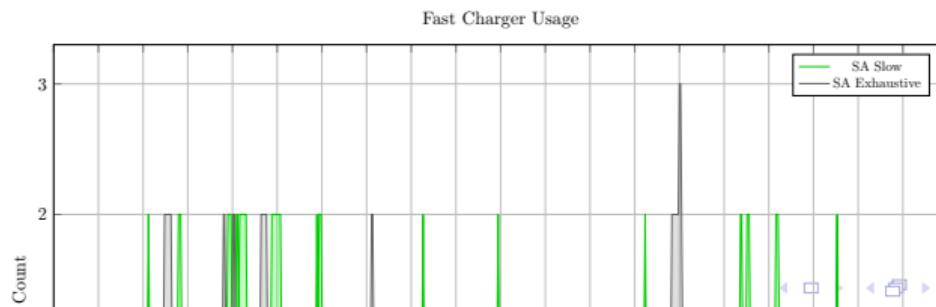
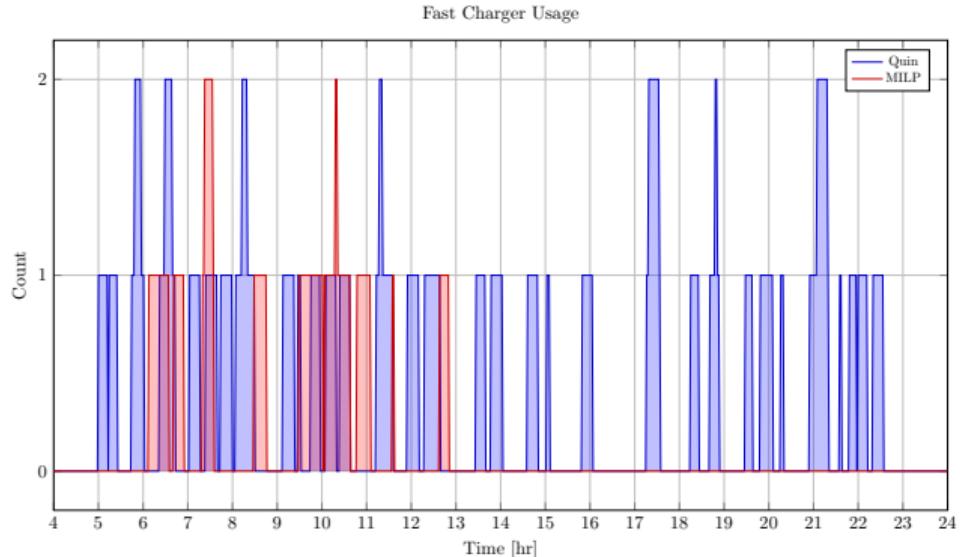
Schedule - What Is In The Thesis



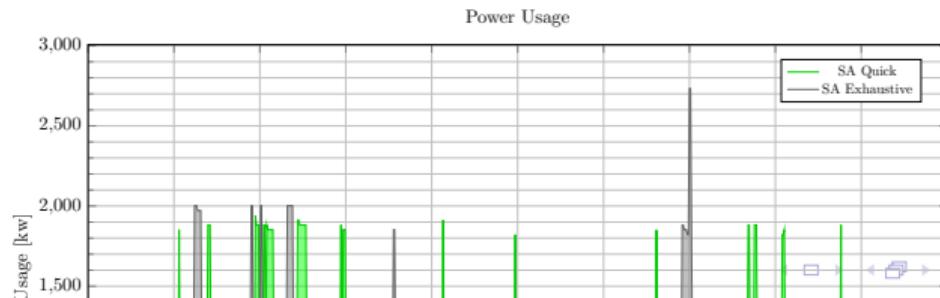
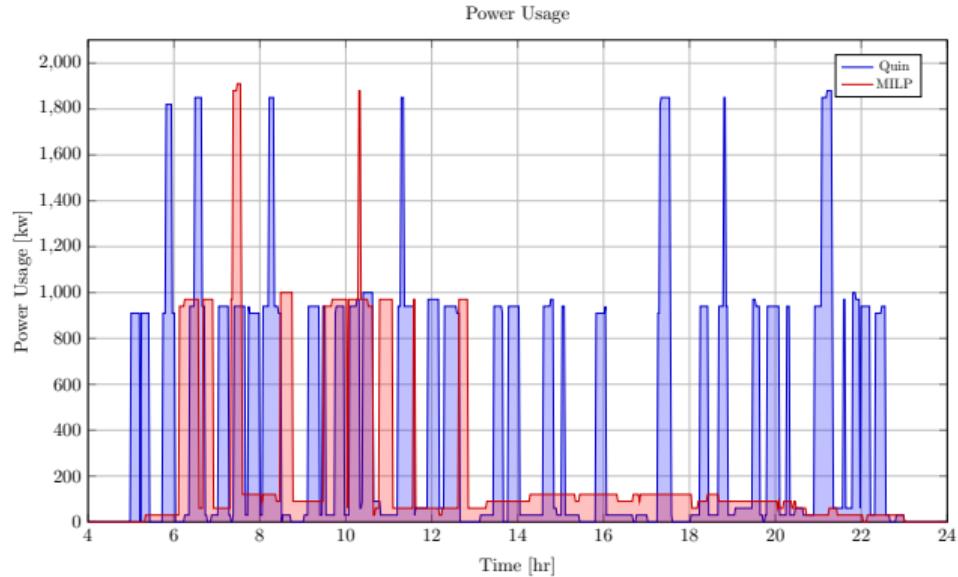
Charger Count - What Is In The Thesis



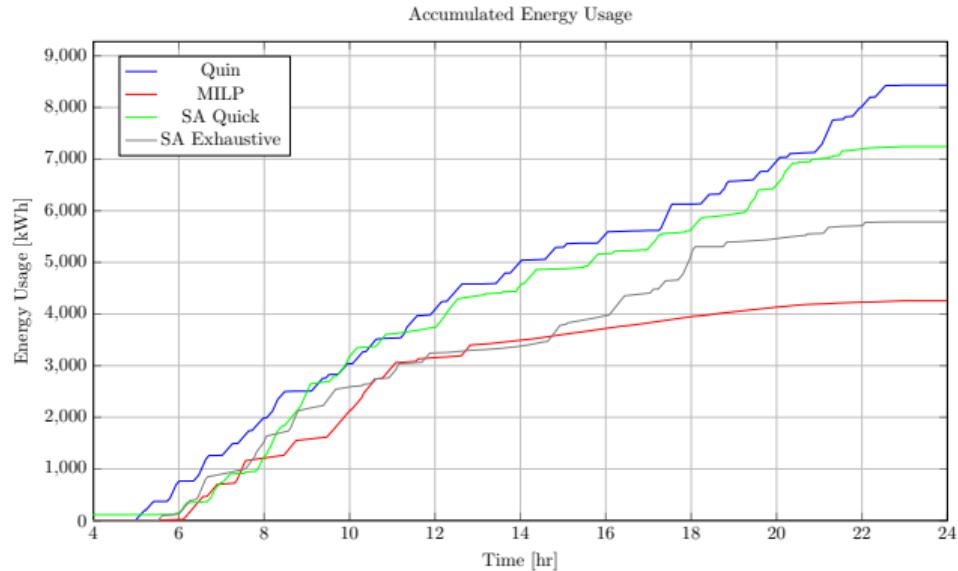
Charger Count - What Is In The Thesis



Power - What Is In The Thesis



Energy - What Is In The Thesis

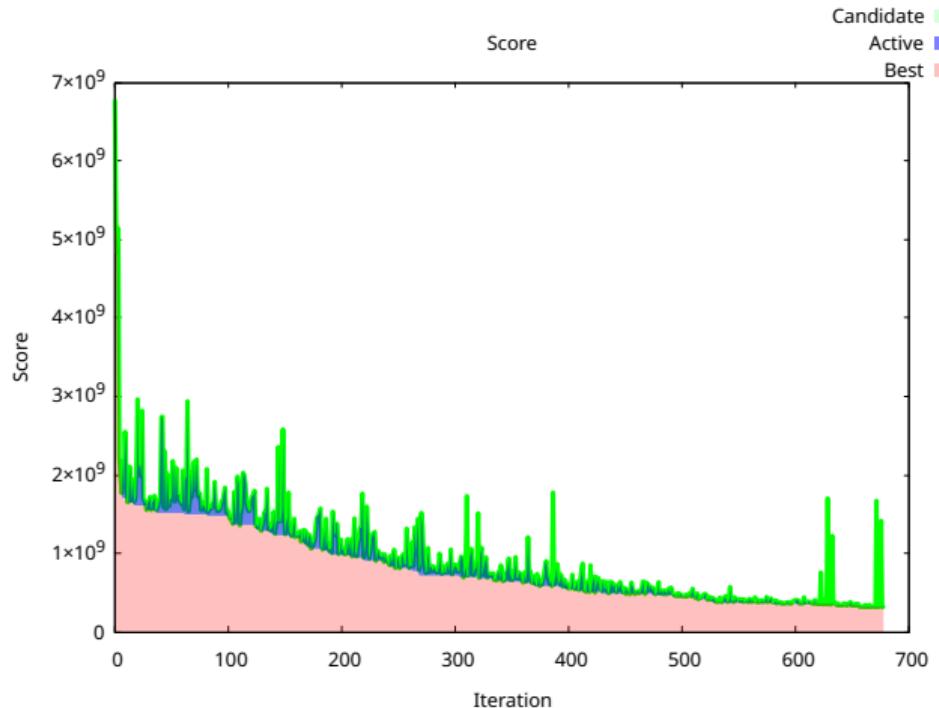


What Happened?



College of Engineering
UtahStateUniversity

How To Resolve This Problem?

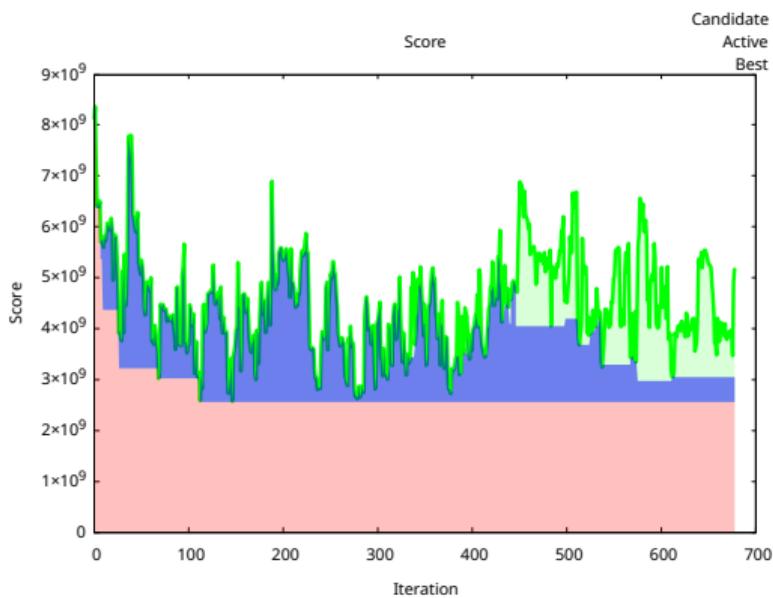


- ▶ Reverse search and weight the visit indices
- ▶ Be more aggressive in exploiting the best solution



Score Convergence Comparison

Before Fix



After Fix

