A POSITION ALLOCATION PROBLEM APPROACH TO THE BATTERY ELECTRIC BUS CHARGING PROBLEM

Alexander Brown

College of Engineering
Utah State University

May 17, 2024

Outline

College of Engineering
UtahStateUniversity

Topic

College of Engineering
UtahStateUniversity

Introduction

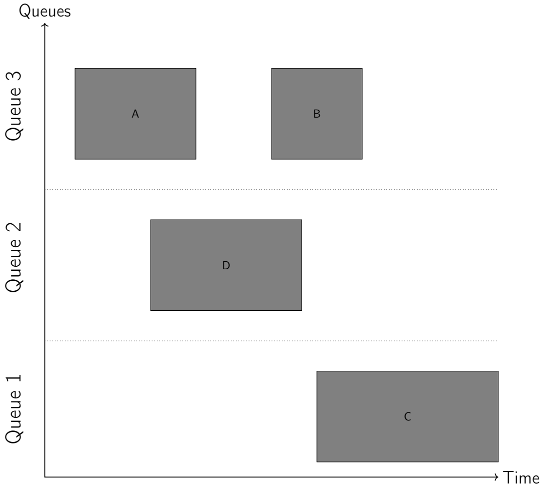
The Position Allocation Problem Approach With Linear Battery Dynamic

The Simulated Annealing Approach With Linear Battery Dynamics

The Simulated Annealing Approach With Non-Linear Battery Dynamics

Problem Description

-	-	6:11 AM	6:16 AM	6:25 AM
6:30 AM	6:34 AM	6:41 AM	6:46 AM	6:55 AM
7:00 AM	7:04 AM	7:11 AM	7:16 AM	7:25 AM
7:30 AM	7:34 AM	7:41 AM	7:46 AM	7:55 AM
8:00 AM	8:04 AM	8:11 AM	8:16 AM	8:25 AM
8:30 AM	8:34 AM	8:41 AM	8:46 AM	8:55 AM
9:00 AM	9:04 AM	9:11 AM	9:16 AM	9:25 AM
9:30 AM	9:34 AM	9:41 AM	9:46 AM	9:55 AM
10:00 AM	10:04 AM	10:11 AM	10:16 AM	10:25 AM
10:30 AM	10:34 AM	10:41 AM	10:46 AM	10:55 AM
11:00 AM	11:04 AM	11:11 AM	11:16 AM	11:25 AM
11:30 AM	11:34 AM	11:41 AM	11:46 AM	11:55 AM
12:00 PM	12:04 PM	12:11 PM	12:16 PM	12:25 PM
12:30 PM	12:34 PM	12:41 PM	12:46 PM	12:55 PM
1:00 PM	1:04 PM	1:11 PM	1:16 PM	1:25 PM
1:30 PM	1:34 PM	1:41 PM	1:46 PM	1:55 PM
2:00 PM	2:04 PM	2:11 PM	2:16 PM	2:25 PM
2:30 PM	2:34 PM	2:41 PM	2:46 PM	2:55 PM
3:00 PM	3:04 PM	3:11 PM	3:16 PM	3:25 PM
3:30 PM	3:34 PM	3:41 PM	3:46 PM	3:55 PM
4:00 PM	4:04 PM	4:11 PM	4:16 PM	4:25 PM
4:30 PM	4:34 PM	4:41 PM	4:46 PM	4:55 PM
5:00 PM	5:04 PM	5:11 PM	5:16 PM	5:25 PM
5:30 PM	5:34 PM	5:41 PM	5:46 PM	5:55 PM
6:00 PM	6:04 PM	6:11 PM	6:16 PM	6:25 PM
6:30 PM	6:34 PM	6:41 PM	6:46 PM	6:55 PM
7:00 PM	7:04 PM	7:11 PM	7:16 PM	7:25 PM
7:30 PM	7:34 PM	7:41 PM	7:46 PM	7:55 PM
8:00 PM	8:04 PM	8:11 PM	8:46 PM	8:25 PM



Brief State Of The Art

Ref	Consumption	Demand	Linear	Non-linear	Charger Min.	Battery Health
[1]	✓		✓			
[2]	✓	✓	✓			
[3]	✓	✓	✓	✓		
[4]	✓	✓	✓			
[5]	✓	✓	✓			
[6]	✓		✓			
[7]	✓			✓	✓	✓
[8]	✓		✓			
[9]	✓		✓		✓	
BPAP	✓		✓		✓	✓
SA BPAP	✓	✓	✓	✓	*	✓

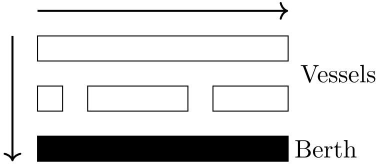
The Berth Allocation Problem[1]

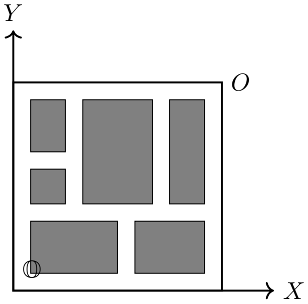
[1] https://www.mdpi.com/2077-1312/11/7/1280

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



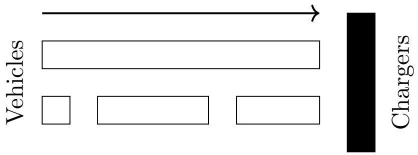


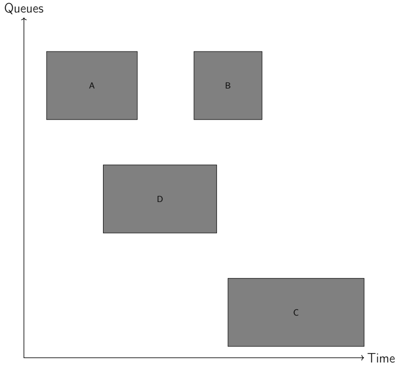
The Position Allocation Problem

College of Engineering
UtahStateUniversity

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

- Discrete charger queues
- Multiple charger types
- Bus can have multiple visits
- Propagate battery SOC across visits
- Unknown charge times

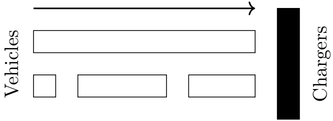


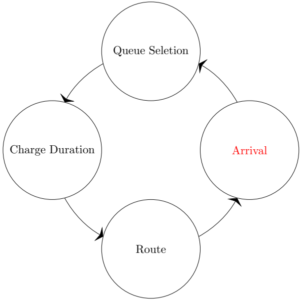


Topic

Requirements For BEB Implementation

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





Mixed Integer Linear Program

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

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

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

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

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

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

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

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

$$s_i + u_i = d_i$$

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

$$d_i \leq \tau_i$$

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

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

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

$$\eta_{\Gamma_b^0} = \alpha_{\Gamma_i} \kappa_{\Gamma_i}$$

$$\eta_{\Gamma_b^f} \geq \beta_{\Gamma_b} \kappa_{\Gamma_b}$$

Constraints

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

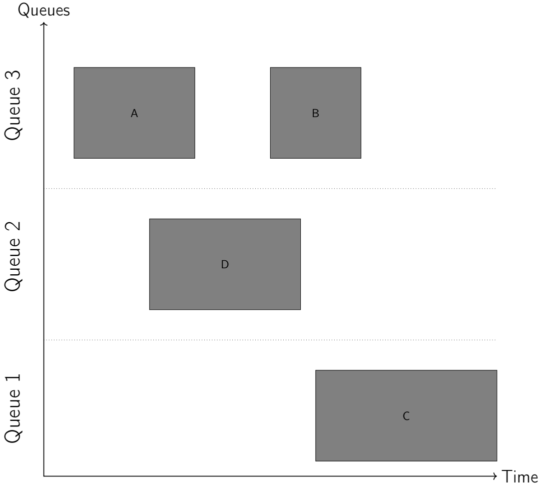
$$v_j - v_i - (\psi_{ij} - 1)n_Q \geq 1$$

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

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

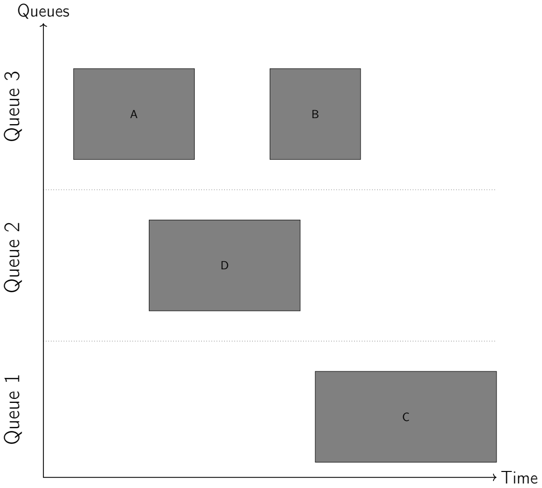
$$\sigma_{ij} + \sigma_{ji} + \psi_{ij} + \psi_{ji} \geq 1$$

- Used to ensure that gray rectangles do not overlap
- σ_{ij} establishes temporal ordering when active
- ψ_{ij} establishes spacial ordering when active

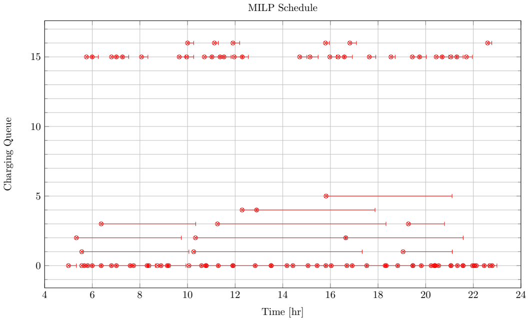


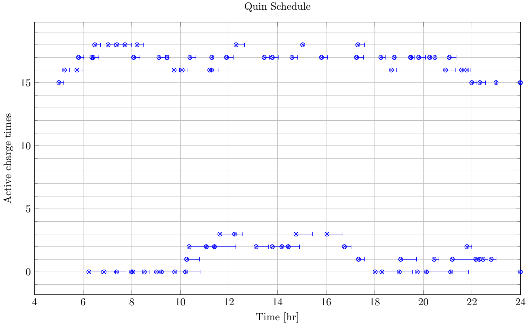
Parameters

Execution	2 hr
T	24 hr
n_V	338
n_A	35
α_i	90%
ν_i	25%
β_i	70%
m_q	1000q



Schedules





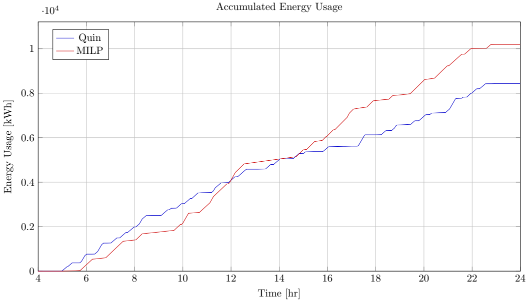
- Slow chargers utilized for frequently
- Fast chargers utilized more sparingly

- Slow chargers used for short durations
- Extensive use of fast chargers

SOC And Energy Use

	MILP	Qin
Mean	265.873	355.93
Min	97.04	0.000
Max	388	368.354

- SOC of Qin allowed to drop to 0
- PAP maintained minimum SOC and final SOC

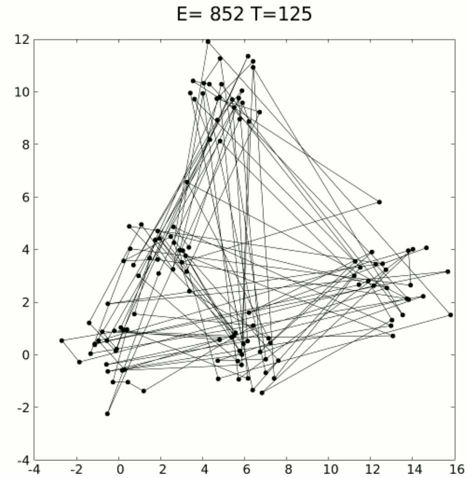


- PAP accumulated energy is larger due to minimum SOC constraints

Topic

Simulated Annealing[2]

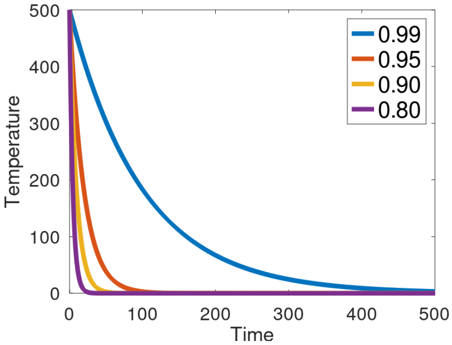
- A probabilistic technique for approximating the global optimum of a given function.
- Often applied to problems that contain many local solutions
- Three key components:
  - Cooling Schedule
  - Acceptance Criteria
  - Generation Mechanisms



[2] https://en.wikipedia.org/wiki/Simulated_annealing

Cooling Schedule

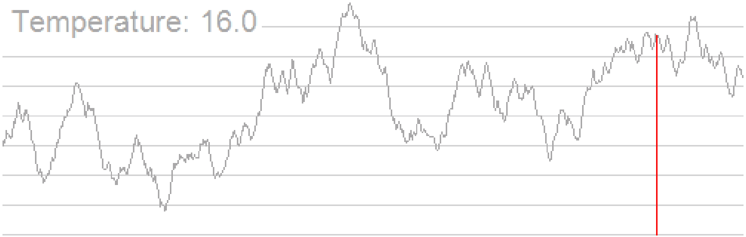
- The cooling equation models the rate at which the temperature decreases over time in the SA process.
- The temperature is high, SA encourages exploration. As the temperature decreases, exploitation is encouraged.

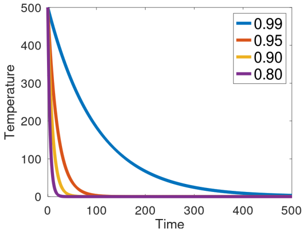


$$t_m = \beta t_{m-1}$$

Acceptance Criteria[3]

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



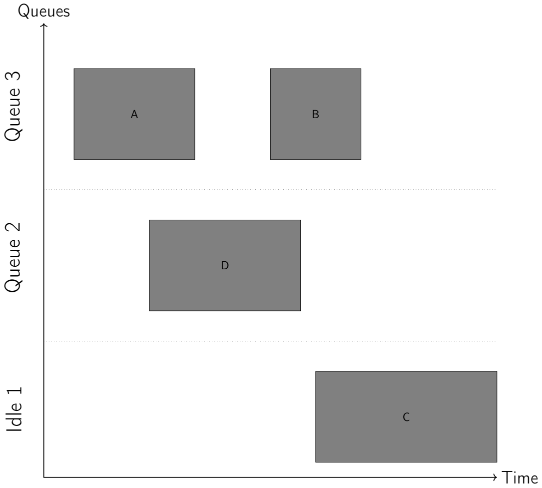


[3] https://en.wikipedia.org/wiki/Simulated_annealing

Generation Mechanisms - Primitive Functions

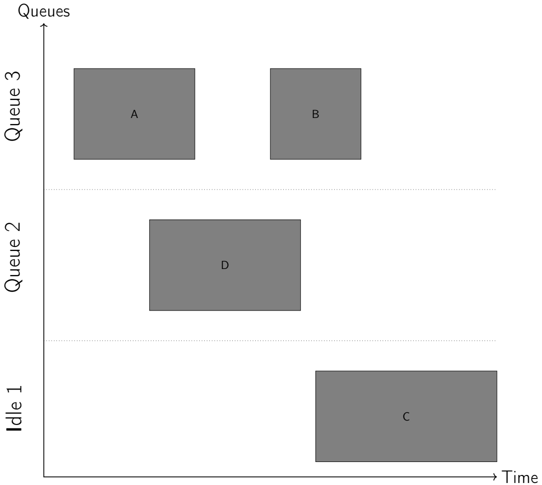
College of Engineering
UtahStateUniversity

- 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
  - $p_{T_p,h} = \frac{1}{T_p} \sum_{k=h-\frac{T_p}{dt}+1}^{h} p_k \ dt$
  - $p_{max} = \max\limits_{k \in [h-\frac{T_p}{dt}+1, h]} p_{T_p,h}$
  - $p_d = \max(p_{fix}, p_{max})$
- $\epsilon_{q_i} r_{q_i}$: 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

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

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

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

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

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

$$s_i = d_i - u_i$$

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

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

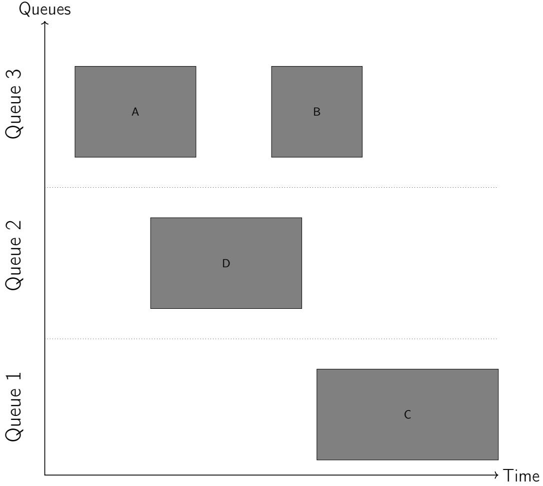
$$a_i \leq u_i \leq d_i \leq e_i \leq \mathcal{T}$$

Parameters

College of Engineering
UtahStateUniversity

Model	Execution Time [s]
MILP	1900
Quick	1532.8
Heuristic	1916

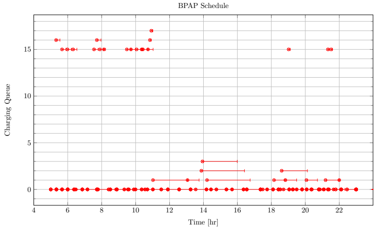
- $T_0 = 90000$
- $\beta = 0.997$
- $|t| = 3797$
- $n_K = 500$

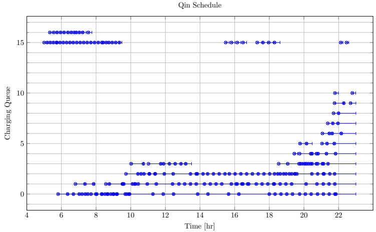


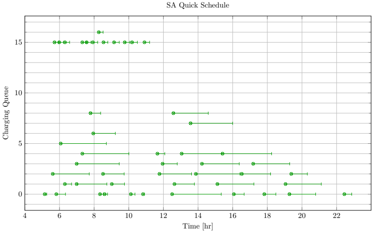
Schedule

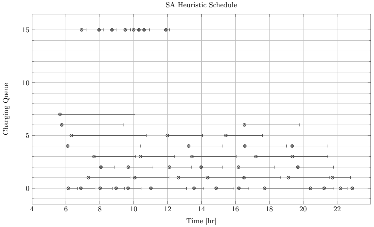
- Qin heavily utilizes fast charger
- BPAP emphasizes slow queues, but utilizes fast chargers readily

- SA techniques heavily utilize slow charging
- SA techniques utilize fast chargers more sparingly

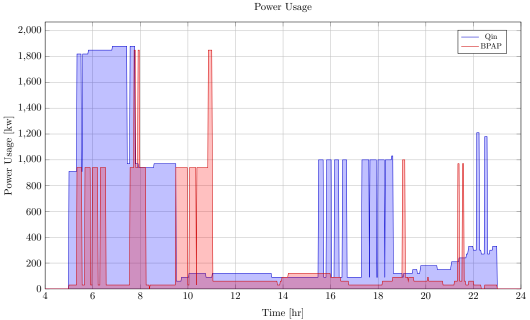


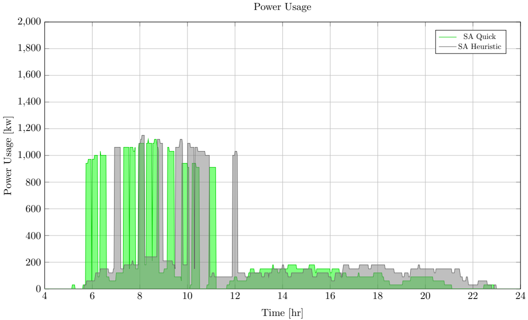






Power

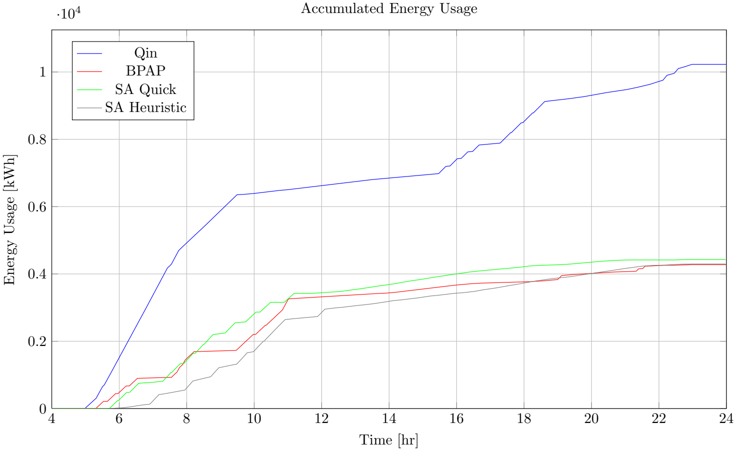




- Heuristic SA maintained the lowest demand peak
- Quick SA peak demand comparable to the PAP peak

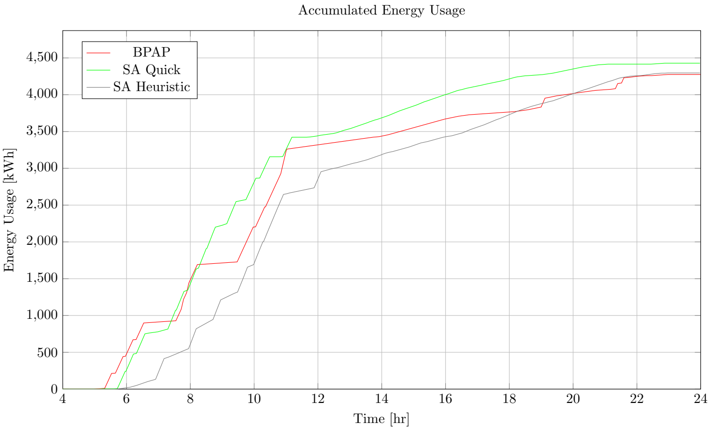
SOC and Energy

- SOC threshold deficits:
  - quick: 2.24 kWh
  - heuristic: 5.74 kWh
- Energy delta:
  - quick: 191.47 kWh
  - heuristic: 58.46 kWh
- Trade off of the lower peak demand



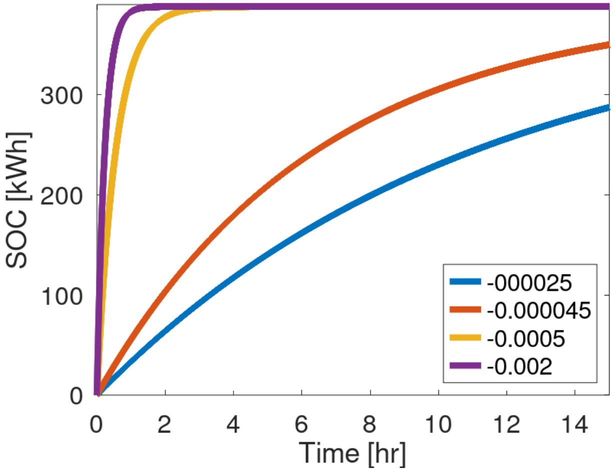
Scores and Energy

Schedule	Score
BPAP	18,500,000
Qin-Modified	34,578,526
Heuristic	11,673,937
Quick	11,234,577



Introduction

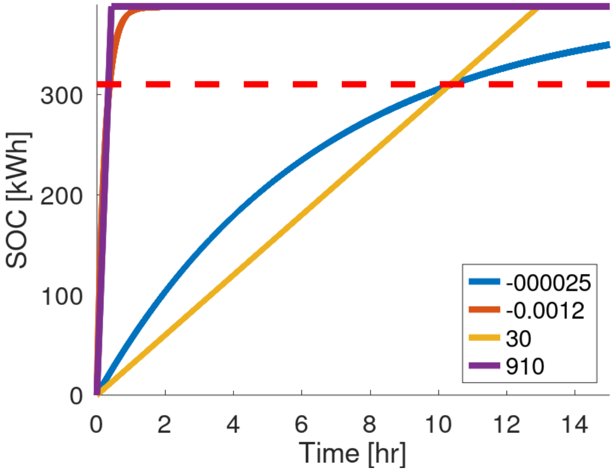
- Higher fidelity in approximating charge at high SOC
- Implemented in SA for simplicity



Non-Linear Battery Dynamics Model

$$\eta_{\xi_i} = \bar{a}_q \eta_i - \bar{b}_q \kappa_{b_i}$$

$$\bar{a}_q = e^{a_q dt} \quad \bar{b}_q = e^{a_q dt} - 1$$

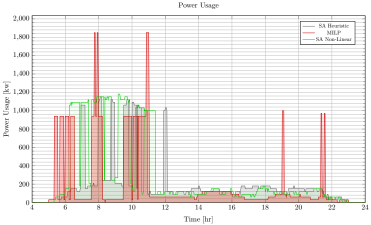


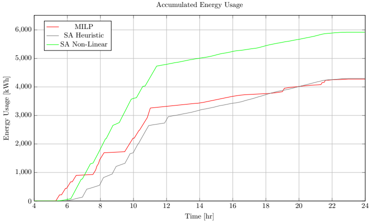
Results

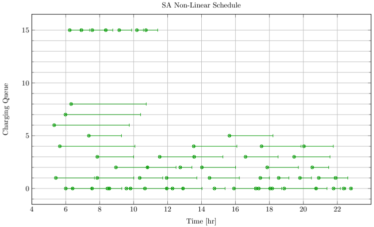
College of Engineering
UtahStateUniversity

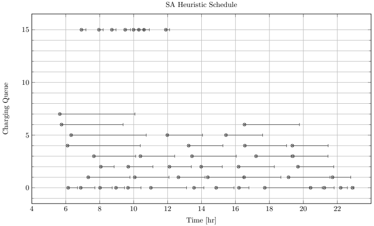
- Minimized demand power
- Larger energy consumption due to fast charger duration

- Similar schedule output, longer fast charger durations due to nonlinear model
- Minimum SOC of 85 kWh









Questions?

Appendix

Mixed Integer Linear Programming

$$
\begin{aligned}
\max\ J &= \sum_j c_j x_j + \sum_k d_k y_k \\
\text{subject to } & \sum_j a_{ij} x_j + \sum_k g_{ik} y_k \le b_i && (i = 1, 2, \ldots, m) \\
& x_j \ge 0 && (j = 1, 2, \ldots, n) \\
& y_k \in \mathbb{Z}^+ && (k = 1, 2, \ldots, n)
\end{aligned}
$$

- 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

SA Primitive Generators

College of Engineering
UtahStateUniversity

New Visit

Algorithm 0: New visit algorithm

Algorithm: New Visit

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

begin
 $i \leftarrow \mathbb{S}_i$; `/* Extract visit index */`
 $a \leftarrow \mathbb{I}_{i.a}$; `/* Extract the arrivial time for visit` i `*/`
 $e \leftarrow \mathbb{I}_{i.e}$; `/* Extract the departure time for visit` i `*/`
 $q \leftarrow \mathbb{I}_{i.q}$; `/* Extract the current charge queue for visit` i `*/`
 $\bar{q} \leftarrow \mathcal{U}_Q$; `/* Select a random charging queue with a uniform distribution */`
 $C \leftarrow \mathcal{U}_{\mathbb{C}_q}$; `/* Select a random time slice from` $\mathbb{C}_q$ `*/`
 if $(\bar{C}, \bar{u}, \bar{d}) \leftarrow$ `findFreeTime(`C, i, q, a, e`)` $\notin \varnothing$ **then** `/* If there is time available in` C_q^j `*/`
  return $(i, (\bar{q}, \bar{u}, \bar{d}), \bar{C})$ `/* Return visit */`
 end
 return $(\varnothing)$; `/* Return nothing */`
end

Slide Visit

Algorithm 1: Slide Visit Algorithm

Algorithm: Slide Visit

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

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

New Charger

Algorithm 2: New Charger Algorithm

Algorithm: New Charger

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

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

Wait

Algorithm 3: Wait algorithm

Algorithm: Wait

Input: $\mathbb{S}$

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

begin

  $(i, \mathbb{I}, \bar{\mathbb{C}}) \leftarrow$`Purge(`$\mathbb{S}$`);` `/* Purge visit` i `from charger availibility matrix */`

  $\bar{\mathbb{C}}'_{\mathbb{I}_{i.\Gamma_i}} \leftarrow \mathbb{C}' \cup \{[\mathbb{I}_{i.a}, \mathbb{I}_{i.e}]\}$; `/* Update the charger availability matrix for wait queue` $\bar{\mathbb{C}}_{i.q_i}$ `*/`

  return $(i, \mathbb{I}, (\mathbb{I}_{i.b}, \mathbb{I}_{i.a}, \mathbb{I}_{i.e}), \bar{\mathbb{C}})$ `/* Return visit */`

end

New Window

College of Engineering
UtahStateUniversity

Algorithm 4: New window algorithm

Algorithm: New Window

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

begin
    $\bar{\mathbb{S}} \leftarrow$ `Wait`$(\mathbb{S})$; `/* Assign visit to its respective idle queue */`
    if $\bar{\bar{\mathbb{S}}} \leftarrow$ `NewVisit`$(\bar{\mathbb{S}}) \notin \varnothing$ **then** `/* Add visit` i `back in randomly */`
        return $\bar{\bar{\mathbb{S}}}$ `/* Return visit */`
    end
    return $(\varnothing)$; `/* Return nothing */`
end

SA Wrapper Functions

Charge Schedule Generation

Algorithm 5: Charge schedule generation algorithm

Algorithm: Candidate Solution Generator

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

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

Charge Schedule Perturbation

College of Engineering
UtahStateUniversity

Algorithm 6: Perturb schedule algorithm

Algorithm: Perturb Schedule

Input: $\mathbb{S}$

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

begin

    $p \leftarrow [false; n_A]$; /* Create vector to track priority routes */

    $y^i \leftarrow [1.0; n_V]$; /* Create weight vector for index selection */

    /* Loop through the visits in reverse order */

    foreach $\mathbb{I}_i \leftarrow \mathbb{I}_{|\mathbb{I}|}$ *TO* $\mathbb{I}_1$ **do**

        /* If the current visit is part of a priority route */

        if $p_{\mathbb{I}_{i.b}} = true$ **then**

            $y^i_{\mathbb{I}_i} = y^i_{\mathbb{I}_{i.\xi}}$;

        end

        /* Else if the current visit's SOC does below the allowed threshold */

        else if $\mathbb{I}_{i.\eta} \leq \nu_{\mathbb{I}_{i.b}} \kappa_{\mathbb{I}_{i.b}}$ **then**

            $p_{\mathbb{I}_{i.b}} = true$; /* Indicate the current BEB's routes are to be prioritized */

            $y^i_{\mathbb{I}_i} = \kappa_{\mathbb{I}_{i.b}}(\nu_{\mathbb{I}_{i.b}} \kappa_{\mathbb{I}_{i.b}} - \mathbb{I}_{i.\eta})$; /* Calculate the weight of the current visit */

        end

    end

    $\mathbb{I}_i \leftarrow \mathcal{W}^{y^i}_{\mathbb{I}}$; /* Select an index with a weighted distribution */

    $i \leftarrow \mathbb{I}_i$; /* Extract visit index */

    $y^p \leftarrow [y^p_1, y^p_2, \ldots]$; /* Define the weight of each primitive generator */

    $PGF \leftarrow \mathcal{W}^{y^p}_{[1,n_G]}$; /* Select a generator function with weighted distribution */

    $\bar{\mathbb{S}} \leftarrow$ PGF($(i, \mathbb{I}, \mathbb{C})$); /* Excecute the generator function */

    return $(0, \bar{\mathbb{I}}, \bar{\mathbb{C}})$

end

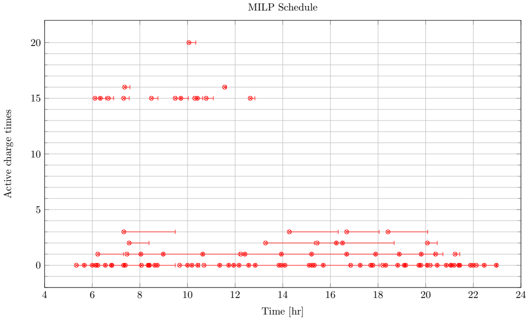
SA Thesis Results

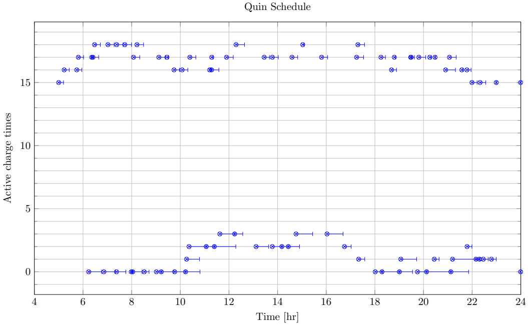
Parameters

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

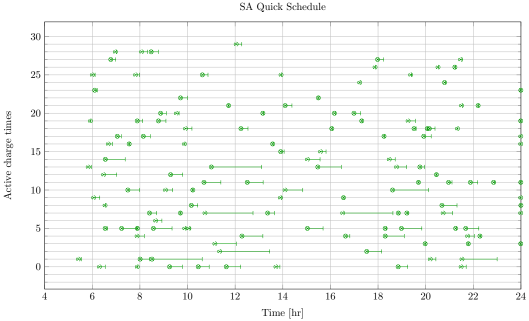
- $T_0 = 99999$
- $\beta = 0.999$
- $|t| = 3797$
- $n_K = 500$

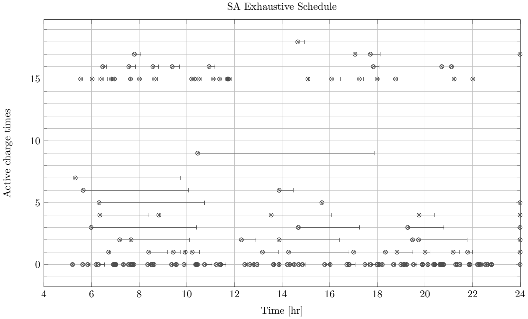
Schedule


MILP Schedule
Active charge times
Time [hr]
0
5
10
15
20
4
6
8
10
12
14
16
18
20
22
24

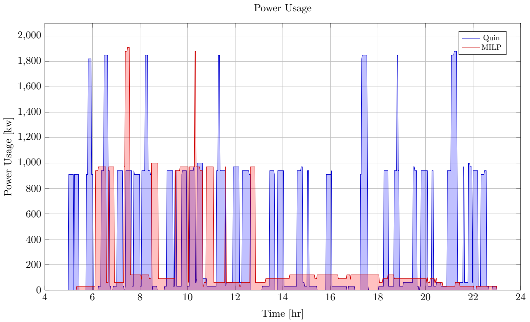

Quin Schedule
Active charge times
Time [hr]
0
5
10
15
4
6
8
10
12
14
16
18
20
22
24

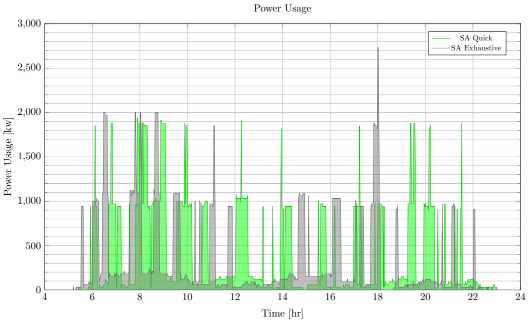
Schedule


SA Quick Schedule
Active charge times
Time [hr]
0
5
10
15
20
25
30
4
6
8
10
12
14
16
18
20
22
24

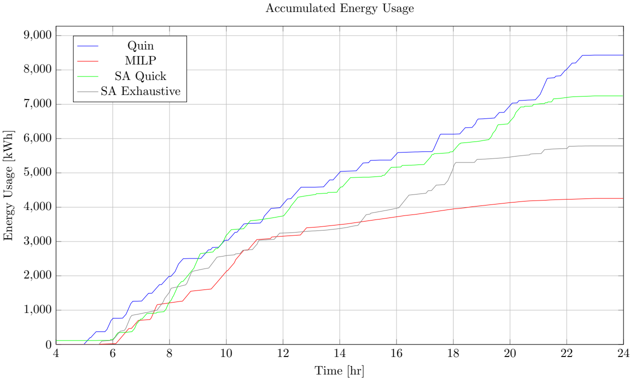

SA Exhaustive Schedule
Active charge times
Time [hr]
0
5
10
15
4
6
8
10
12
14
16
18
20
22
24

Power

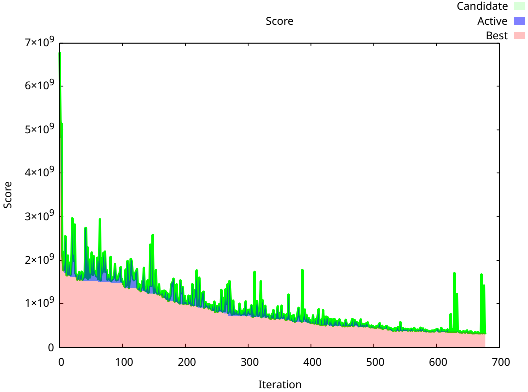

Power Usage
Quin
MILP
Power Usage [kw]
Time [hr]
2,000
1,800
1,600
1,400
1,200
1,000
800
600
400
200
0
4
6
8
10
12
14
16
18
20
22
24


Power Usage
SA Quick
SA Exhaustive
Power Usage [kw]
Time [hr]
3,000
2,500
2,000
1,500
1,000
500
0
4
6
8
10
12
14
16
18
20
22
24

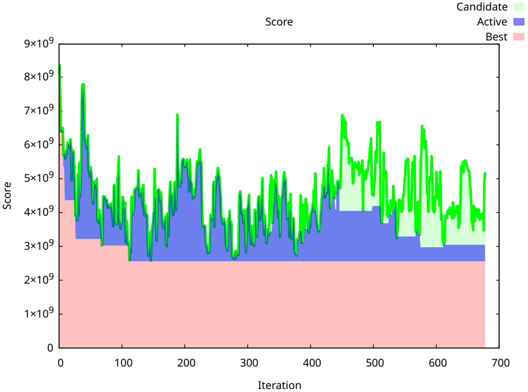
Energy


Accumulated Energy Usage
Quin
MILP
SA Quick
SA Exhaustive
Energy Usage [kWh]
0
1,000
2,000
3,000
4,000
5,000
6,000
7,000
8,000
9,000
Time [hr]
4
6
8
10
12
14
16
18
20
22
24

How To Resolve This Problem?



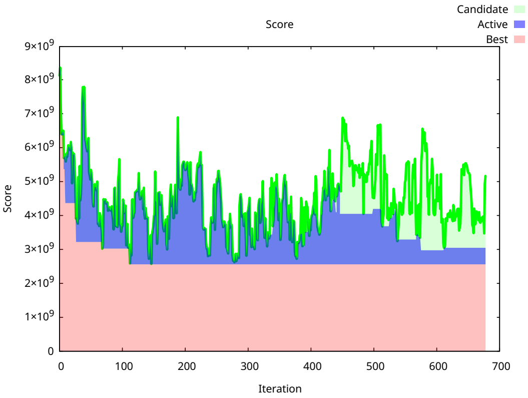
- Reverse search and weight the visit indices
- Be more aggressive in exploiting the best solution



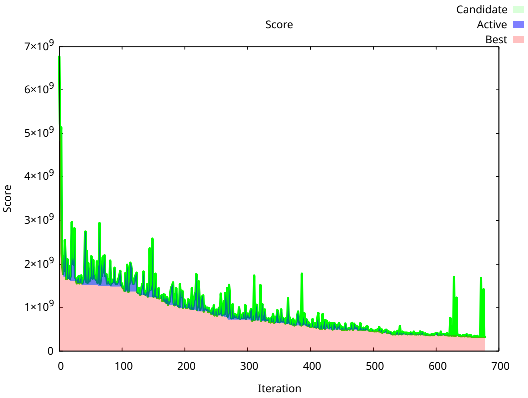
- Candidate solutions diverge
- Hard time handling "difficult" routes

Score Convergence Comparison

Before Fix



After Fix



References

Mengyuan Duan, Geqi Qi, Wei Guan, Chaoru Lu, and Congcong Gong.
Reforming mixed operation schedule for electric buses and traditional fuel buses by an optimal framework.
IET Intelligent Transport Systems, 15(10):1287–1303, Jul 2021.

Oliver Frendo, Nadine Gaertner, and Heiner Stuckenschmidt.
Open source algorithm for smart charging of electric vehicle fleets.
IEEE Transactions on Industrial Informatics, 17(9):6014–6022, September 2021.

Amra Jahic, Mina Eskander, and Detlef Schulz.
Preemptive vs. non-preemptive charging schedule for large-scale electric bus depots.
In *2019 IEEE PES Innovative Smart Grid Technologies Europe (ISGT-Europe)*. IEEE, September 2019.

Daniel Mortensen, Jacob Gunther, Greg Droge, and Justin Whitaker.
Cost minimization for charging electric bus fleets.
World Electric Vehicle Journal, 14(12):351, December 2023.