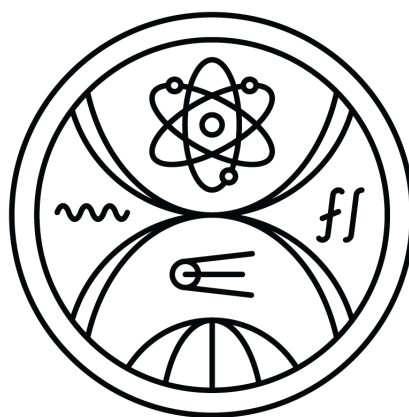


UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

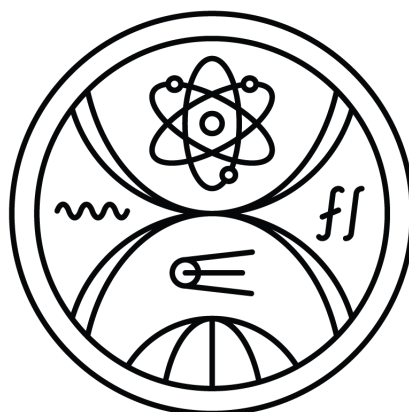


PODPORA INTELIGENTNÉHO RIADENIA
ENERGETICKÝCH SIETÍ
DIPLOMOVÁ PRÁCA

2023

BC. OMAR AL-SHAFE'I

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY



PODPORA INTELIGENTNÉHO RIADENIA
ENERGETICKÝCH SIETÍ
DIPLOMOVÁ PRÁCA

Študijný program: Informatika
Študijný odbor: Aplikovaná informatika
Školiace pracovisko: Katedra aplikovanej informatiky
Školiteľ: prof. RNDr. Mária Lucká, PhD.

Bratislava, 2023

Bc. Omar Al-Shafe'i



Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Omar Al-Shafe'i
Študijný program: aplikovaná informatika (Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor: informatika
Typ záverečnej práce: diplomová
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Podpora inteligentného riadenia energetických sietí
Support for intelligent management of smart energy networks

Anotácia: Stabilita inteligentnej energetickej siete a zabezpečenie dodávok elektrickej energie môže byť potenciálne narušená integráciou obnoviteľných zdrojov energie (napr. fotovoltických panelov), veľkých úložísk energie (batérií), ale aj významným rozvojom elektromobility. Tieto nové prvky sa v súčasnosti v čoraz väčšej miere stávajú súčasťou moderných inteligentných energetických sietí a preto ich integrácia do celej sústavy sa stala mimoriadne dôležitou. Pôvodne jednosmerná sieť, kde sa energia od veľkých výrobcov elektriny dodávala spotrebiteľom, sa postupne mení na obojsmernú sieť, v ktorej sa mnohí odberatelia - vďaka obnoviteľným zdrojom - stávajú súčasne aj drobnými výrobcami – prosumeri. Kvôli garancii stability celej sústavy a zabezpečeniu minimálnej ceny bolo potrebné vytvoriť tzv. agregátorov flexibility, ktorým účastníci siete môžu poskytnúť svoju flexibilitu. Pod flexibilitou pritom rozumieme práva (1) na odber vyrobenej alebo uskladnenej elektrickej energie (napr. z fotovoltiky, batérie alebo elektromobilu) v určitom čase a (2) obmedzenie spotreby (napr. vypnutie kúrenia). Navrhnete a overte model práce agregátora flexibility, ktorý vďaka inteligentným algoritmom a dátovej analýze dokáže optimalizovať tok energie, zabezpečiť stabilné dodávky energie a minimalizovať náklady spotrebiteľov. Svoje riešenie obmedzte pre vybraný typ prosumerov/spotrebiteľov. Použite pritom vhodné optimalizačné metódy, predovšetkým metódy strojového učenia. Svoje riešenie implementujte a porovnajte s existujúcimi riešeniami na dostupných dátach.

Literatúra:

1. Li, T., Sun, B., Chen, Y., Ye, Z., Low, S. H., & Wierman, A. (2020). Real-time Aggregate Flexibility via Reinforcement Learning. 1–17. <http://arxiv.org/abs/2012.11261>
2. Fernando Lezama, Joao Soares, Bruno Canizes, Zita Vale: Flexibility management model of home appliances to support DSO requests in smart grids, Sustainable Cities and Society, Volume 55, 2020, ISSN 2210-6707, <https://doi.org/10.1016/j.scs.2020.102048>
3. Tongxin Li and Bo Sun and Yue Chen and Zixin Ye and Steven H. Low and Adam Wierman: Learning-Based Predictive Control via Real-Time Aggregate Flexibility, IEEE Transactions on Smart Grid, 12 (6), 2021, 97–4913, <https://doi.org/10.1109%2Ftsg.2021.3094719>.



Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

4. Steve Wattam: Artificial intelligence and machine learning approaches to energy demand-side response: A systematic review, Renewable and Sustainable Energy Reviews, Volume 130, 2020, 109899, ISSN 1364-0321, <https://doi.org/10.1016/j.rser.2020.109899>.

Vedúci: prof. RNDr. Mária Lucká, PhD.
Katedra: FMFI.KAI - Katedra aplikovanej informatiky
Vedúci katedry: doc. RNDr. Tatiana Jajcayová, PhD.
Dátum zadania: 13.12.2022

Dátum schválenia: 13.12.2022

prof. RNDr. Roman Ďurikovič, PhD.
garant študijného programu

.....
š student

.....
vedúci práce

Pod'akovanie: Chcel by som pod'akovať mojej školiteľke za cenné rady počas tvorby diplomovej práce.

Abstrakt

Slovenský abstrakt v rozsahu 100-500 slov, jeden odstavec. Abstrakt stručne sumarizuje výsledky práce. Mal by byť pochopiteľný pre bežného informatika. Nemal by teda využívať skratky, termíny alebo označenie zavedené v práci, okrem tých, ktoré sú všeobecne známe.

Kľúčové slová: jedno, druhé, tretie (prípadne štvrté, piate)

Abstract

Abstract in the English language (translation of the abstract in the Slovak language).

Keywords:

Obsah

Úvod	1
1 Východiská	3
1.1 Využité technológie	3
1.1.1 Programovací jazyk Python.	3
1.1.2 Repozitár ACN Portal.	3
1.1.3 Ukážka vstupných dát ACN-Data	5
1.1.4 Architektúra simulátora ACN-Sim	5
1.1.5 Interakcia medzi spotrebitelmi a smart grid	6
1.1.6 Repozitár Adacharge.	6
1.1.7 Repozitár Acnportal-experiments.	6
1.2 Konfigurácia a obmedzenia nabíjacích sietí	6
1.2.1 Typy nabíjania.	6
1.2.2 Rýchlosť nabíjania batérie.	7
1.2.3 Typy nabíjacích staníc.	8
1.2.4 Obmedzenia pri nabíjaní.	8
1.3 Problém nájdenia optimálneho schedulingu.	9
1.4 Uncontrolled charging.	11
1.5 Round Robin.	11
1.6 Triediace scheduling algoritmy	12
1.6.1 Least Laxity First.	13
1.6.2 Smoothed Least Laxity First.	14
1.6.3 Group Least Laxity First.	14
1.6.4 Earliest deadline first.	15
1.6.5 Group Earliest Deadline First.	15
1.6.6 Group priority earliest deadline first (EDF).	15
1.6.7 First-Come First-Served.	15
1.6.8 Last-Come First-Served.	15
1.6.9 Longest Remaining Processing Time.	15
1.6.10 Shortest Job Next.	16

1.7	Kontrolné scheduling algoritmy	16
1.7.1	Typy kontrolných algoritmov	16
1.7.2	Model Predictive Control.	16
1.7.3	Penalised Predictive Control.	17
1.8	Cena nabíjania.	18
2	Návrh riešenia	19
2.1	Least Laxity First	19
3	Návrh softvérového diela	20
4	Výskum	21
5	Výsledky	22
	Záver	23
	Príloha A	26
	Príloha B	27

Zoznam obrázkov

1.1	short	4
1.2	Ukážka vstupných dát ACN-Data	5
1.3	Architektúra simulátora ACN-Sim.	5
1.4	Typy nabíjania elektrických vozidiel	7
1.5	Typy nabíjania elektrických vozidiel	10
1.6	Typy nabíjania elektrických vozidiel	11
1.7	Typy nabíjania elektrických vozidiel	12
1.8	short	18
1.9	short	18

Zoznam tabuliek

Úvod

S neustále pribúdajúcim množstvom elektrických vozidiel na trhu, pribúda aj množstvo elektrickej energie, ktorú treba dodať týmto vozidlám. Kapacity jednotlivých nabíjajúcich staníc nebudú musieť stačiť pre nabitie každého elektrického vozidla požadovaným množstvom energie. Postupne sa tiež odoberatelia elektrickej energie stávajú výrobcami elektrickej energie pomocou obnoviteľných zdrojov. Tým sa jednosmerná sieť, kde veľký výrobca dodáva energiu spotrebiteľom mení na obojsmernú sieť, kde odoberatelia, ktorí sú výrobcami elektrickej energie môžu poskytovať podporné služby.

Riešením týchto problémov sú inteligentné takzvané smart siete. Tieto siete obsahujú agregátor flexibility, ktorému používatelia nabíjacej siete poskytujú flexibilitu. To znamená, že agregátor flexibility má právo riadiť odber elektrickej energie pre elektrické autá a prispôbovať ho podľa potrieb a podľa podmienok siete. Používatelia, ktorí sú tiež výrobcami elektrickej energie z obnoviteľných zdrojov môžu poskytnúť podporné služby prostredníctvom agregátora flexibility (takýchto používateľov nazývame prosumeri), tým že napríklad agregátor flexibility od nich kúpi energiu. Používatelia v takýchto sieťach môžu nastaviť svoje preferencie, ako napríklad, preferovaný čas nabíjania, množstvo požadovanej energie. Agregátor flexibility následne pomocou inteligentných algoritmov vie optimalizovať tok energie. Využíva pritom optimalizačné metódy, alebo metódy strojového učenia (napr. neurónové siete). Agregátor flexibility tiež minimalizuje cenu elektrickej energie pre spotrebiteľov tým, že nakupuje elektrickú energiu keď je najlacnejšia, a predáva elektrickú energiu keď je najdrahšia.

Cieľom tejto práce je overiť model práce agregátora flexibility, ktorý pomocou inteligentných algoritmov a dátovej analýzy vie optimalizovať tok energie, zabezpečiť stabilné dodávky energie a tiež minimalizovať náklady používateľov elektrických vozidiel. Porovnáme naše riešenia tohoto problému s inými, doterajšími riešeniami. [10]

V prvej kapitole rozoberieme východiská pri tvorbe nášho modelu agregátora flexibility, opíšeme existujúce riešenia a aj obmedzenia infraštruktúry pri nabíjaní elektrických vozidiel. V druhej kapitole predstavujeme model práce nášho agregátora flexibility, spomíname inteligentné algoritmy, ktoré sme v našom modeli použili.

V tretej kapitole overujeme model práce agregátora flexibility, kde porovnávame naše riešenie s existujúcimi riešeniami vzhľadom na pomer dodanej energie, množstvo dodanej energie, cenu dodanej energie a počet výmen vozidiel. Vstupné dáta o elektrických autách

získavame z nabíjacích staníc pre elektrické autá: stanica ACN a stanica JPL. Výstup našej implementácie je optimálny scheduling vozidiel v čase. V našej implementácii sa snažíme počet výmen vozidiel minimalizovať, keďže to často vedie k zlému používateľskému zážitku.

1 Východiská

V tejto kapitole popisujeme technologické a teoretické východiská, ktoré využívame v našej implementácii. Technologické východiská tvoria najmä repozitáre, ktoré implementujú existujúce riešenia. Teoretické východiská zostávajú z veľkej časti podobné ako pri existujúcich riešeniach, aby sme vedeli následne efektívne porovnať našu implementáciu s existujúcimi riešeniami.

1.1 Využité technológie

V tejto podkapitole spomíname technológie, na ktorých spočíva naša implementácia. Repozitáre vysvetľujeme hlavne na základe informácií uvedených v ich README.md súboroch a na základe článku [10]. Podrobný návod, ako vytvoriť algoritmy na základe nasledujúcich repozitárov sa nachádza v [10].

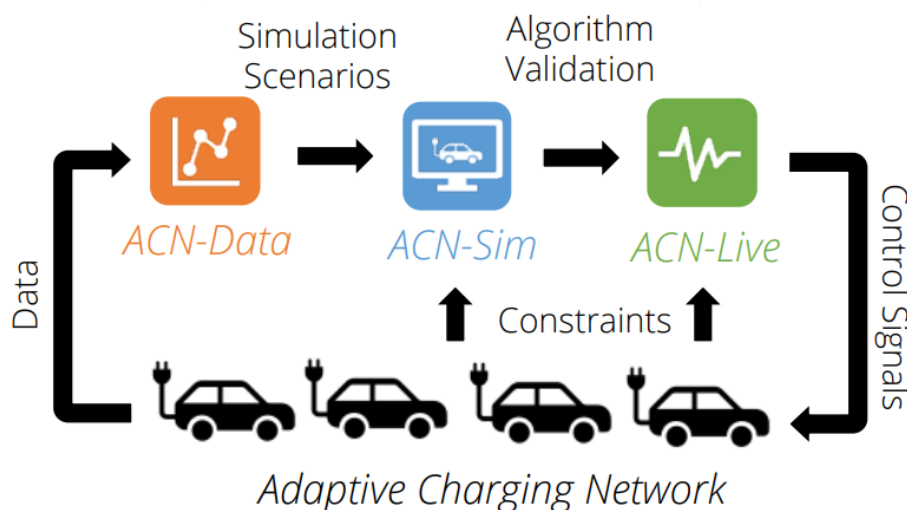
1.1.1 Programovací jazyk Python.

Programovací jazyk Python je veľmi používaným jazykom hlavne na účely dátovej vedy a strojového učenia. Vďaka knižniciam Numpy, Pandas, Matplotlib, Scikit-Learn a iným prostriedkom vieme ľahko ukladať dáta, manipulovať s dátami a aj získať dodatočné informácie o dátach. Napríklad pomocou Scikit-Learn vieme generovať štatistické modely na základe dát získaných z nabíjaciach staníc. Tieto štatistické modely potom môžeme použiť vo forme vstupov do nášho algoritmu. Náš problém je možné riešiť aj pomocou učenia posilňovaním, menovite [12], kde vďaka knižniciam Pytorch a Tensorflow vieme vyriešiť SAC algoritmus. [17, 10]

1.1.2 Repozitár ACN Portal.

Repozitár ACN Portal obsahuje záznamy nabíjania elektrických vozidiel získane z nabíjaciach staníc Caltech, JPL a Office001. Každý záznam nabíjania elektrického vozidla obsahuje jeho príchod, odchod a požadovanú energiu. Repozitár ACN Portal sa nachádza v [5]. Repozitár ACN Portal pozostáva z viacerých komponentov:

1. **ACN-Data:** Dáta získané z nabíjacích staníc pre elektrické vozidlá, konkrétne zo staníc Caltech, JPL a Office001. Vodiči elektrických vozidiel musia cez mobilnú aplikáciu dať oskenovaný QR kód nabíjačky a potom zadať približný čas odchodu a množstvo požadovanej energie. Ak vodič neuvedie informácie cez mobilnú aplikáciu, nabíjačka bude nabíjať len 8 ampérov a ak ani po 15 minútach vodič neuvedie informácie, tak nabíjačka prestane nabíjať. Dáta získané od vodičov elektrických áut, ale aj dáta slúžiace ku konfigurácii siete sú uložené v relačnej databáze. Vďaka tejto úložnej vrstve vieme vytvárať vizualizácie pre vodičov a pre sieťových operátorov. Ide hlavne o vizualizáciu stavu systému pre sieťových operátorov a stav nabíjania elektrických vozidiel pre vodičov.
2. **ACN-Sim:** Simulátor používaný na testovanie a overovanie funkcionality algoritmov, systémov. Simulátor zabezpečuje realistické prostredie na overovanie funkcionality algoritmov, hlavne pre výskumníkov, ktorý nemajú prístup k reálnym nabíjacím systémom pre nabíjanie elektrických vozidiel.
3. **ACN-Live:** Hardvér, ktorý bežia algoritmy nabíjania vozidiel naživo. Keďže má rovnaké rozhranie ako ACN-Sim, tak vieme testovať algoritmy implementované v ACN-Sim, bez žiadnej zmeny kódu. [10, 8]



Obr. 1.1: Architektúra simulátora ACN-Sim.

Obrázok vstupných dát

1.1.3 Ukážka vstupných dát ACN-Data

```

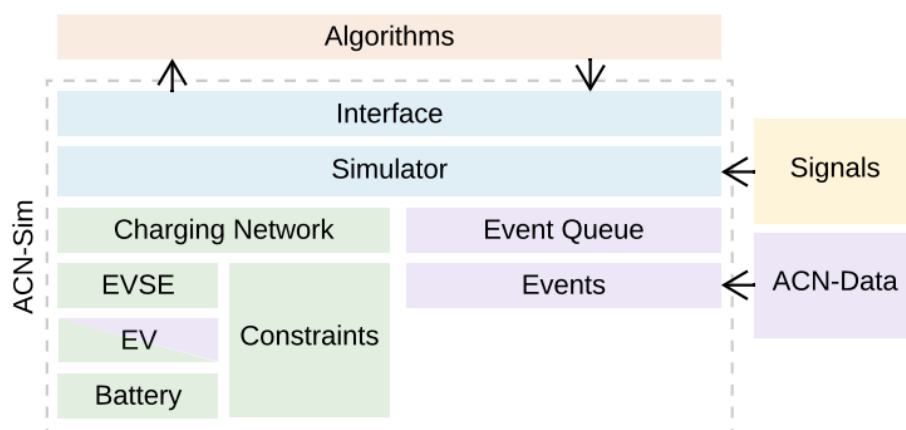
{
  "_id": "5bc90cb9f9af8b0d7fe77cd2",
  "clusterID": "0039",
  "connectionTime": "Wed, 25 Apr 2018 11:08:04 GMT",
  "disconnectTime": "Wed, 25 Apr 2018 13:20:10 GMT",
  "doneChargingTime": "Wed, 25 Apr 2018 13:21:10 GMT",
  "kWhDelivered": 7.932,
  "sessionID": "2_39_78_362_2018-04-25 11:08:04.400812",
  "siteID": "0002",
  "spaceID": "CA-496",
  "stationID": "2-39-78-362",
  "timezone": "America/Los_Angeles",
  "userID": null,
  "userInputs": null
},

```

Obr. 1.2: Vstupné dáta z nabíjania jedného elektrického vozidla 25. Apríla 2018. Zdroj:

Obrázok vstupných dát 1.2 obsahuje viacero parametrov, ktoré používateľ musí zadať, aby mohol nabíjať svoje auto na jednej staníc Caltech, JPL alebo Office001. Iné nabíjacie stanice používajú odlišné vstupné dáta, konkrétny príklad sa nachádza v [11].

1.1.4 Architektúra simulátora ACN-Sim



Obr. 1.3: Architektúra simulátora ACN-Sim.

Obrázok ??vstupných dát

1.1.5 Interakcia medzi spotrebiteľmi a smart grid

1.1.6 Repozitár Adacharge.

Repozitár Adacharge obsahuje optimalizačné algoritmy pre nabíjanie elektrických vozidiel. Tieto optimalizačné algoritmy často využívajú MPC framework a sú schopné nájsť optimálne riešenie pre rôzne cieľové funkcie. Repozitár Adacharge implementuje MPC pomocou konvexnej optimalizácie, preto je implementácia MPC založená na knižnici cvxpy. Podrobnejší popis a zdrojový kód repozitára Adacharge sa nachádza v [4].

1.1.7 Repozitár Acnportal-experiments.

Repozitár Acnportal-experiments uvádza príklady, ako sa repozitár ACN portal dokáže použiť pri jednotlivých experimentoch. Tieto experimenty následne vedia zodpovedať výskumné otázky. Vo vzťahu k našej implementácii sú prioritné experimenty 1.2, 2.1 a 2.2. Krátky popis experimentov:

1. Experiment 1.2: Cieľom experimentu je porovnať rôzne konfigurácie infraštruktúry (level, kapacita transformera a počet nabíjačiek) a aj scheduling algoritmy. Tento experiment ukazuje výhodu smart sietí, lebo smart sieť je schopná minimalizovať kapitálové náklady na kapacitu transformera a aj minimalizovať operačné náklady.
2. Experiment 2.1: Cieľom experimentu je porovnávať výkonnosť algoritmov Round-Robin, First-Come First-Served, Earliest Deadline First, Least Laxity First. Výsledok experimentu má potom byť koľko percent požadovanej elektrickej energie každý algoritmus môže dodať.

Podrobnejší popis experimentov a ich výsledky vieme nájsť v [3].



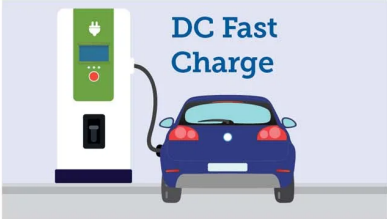
1.2 Konfigurácia a obmedzenia nabíjacích sietí

V tejto podsekcii uvádzame jednotlivé konfigurácie a obmedzenia nabíjacích sietí, na ktorých budeme navrhovať a testovať model agregátora flexibility. Vysvetlíjeme postupne rôzne typy nabíjania, vplyv batérie, obmedzenia sietí atď.

1.2.1 Typy nabíjania.

Pre naše účely sú potrebné 2 typy nabíjania AC level 1 a AC level 2. Nabíjanie AC level 1 je predovšetkým určené pre vlastníkov elektrických vozidiel, ktorý ich chcú nabíjať celý deň (8 až 10 hodín). Rýchlosť nabíjania pri AC level 1 je 1.4 až do 1.9 kW za hodinu. Kapacita nabíjacieho kábla je od 110 až do 120 voltov.

AC level 2 charging je rýchlejší typ nabíjania než AC level 1. Rýchlosť nabíjania pri AC level 2 je od 2.5 do 19.2 Kw. To znamená, že viacero spotrebiteľov sa môže vystriedať pri nabíjaní počas dňa. Kapacita nabíjacieho kábla je dvojnásobne vyššia než pri AC level 1 nabíjaní.

KNOW YOUR EV CHARGING STATIONS		
 <p>AC Level One</p> <p>VOLTAGE 120V 1-Phase AC</p> <p>AMPS 12–16 Amps</p> <p>CHARGING LOAD 1.4–1.9 kW</p> <p>CHARGING TIME 3–5 Miles per Hour</p>	 <p>AC Level Two</p> <p>VOLTAGE 208V or 240V 1-Phase AC</p> <p>AMPS 12–80 Amps (Typ. 32 Amps)</p> <p>CHARGING LOAD 2.5–19.2 kW (Typ. 6.6 kW)</p> <p>CHARGING TIME 12–60 Miles per Hour</p>	 <p>DC Fast Charge</p> <p>VOLTAGE 208V or 480V 3-Phase AC</p> <p>AMPS >100 Amps</p> <p>CHARGING LOAD 50–350 kW</p> <p>CHARGING TIME 60–80 Miles in 20 Minutes</p>

Obr. 1.4: Všetky dnešné typy nabíjania s príslušnými údajmi o rýchlosti nabíjania, kapacite kábla.

Obrázok 1.4 vstupných dát

[18]

UNFINISHED

1.2.2 Rýchlosť nabíjania batérie.

Pomocou triedy Battery, ktorá definuje ideálny model batérie vieme sami definovať vlastné modely batérie. Rýchlosť nabíjania ideálnej batérie je:

$$\hat{r}(t) = \min\{\bar{r}, r(t), \hat{e}(t)\}, \quad (1.1)$$

kde \bar{r} je maximálna rýchlosť nabíjania, $r(t)$ je pilotový signál poslaný do batérie a $\hat{e}(t)$ je nezaplnená energia batérie v čase t v ampéroch.

Existuje aj rozšírenie triedy Battery, ktoré sa nazýva Linear2StageBattery. Linear2StageBattery aproximuje po častiach lineárne nabíjanie elektrických vozidiel s lítiovou batériou. Prvý stav, ktorý nazývame bulk charging trvá zvyčajne od 0 do 70 až 80 percent stavu nabíjania. Rých-

lost' nabíjania batérie Linear2StageBattery je:

$$\hat{r}(t) = \begin{cases} \min\{\bar{r}, r(t), \hat{e}(t)\} & \text{Ak } SoC \leq th \\ \min\{(1 - SoC) \frac{\bar{r}}{1-th}, r(t)\} & \text{inak} \end{cases} \quad (1.2)$$

kde th je hodnota elektrickej energie po tranzícii z bulk stage do absorbtion stage z procesu nabíjania. SoC znamená stav nabíjania batérie.

Tento čiastočný lineárny model je dobrá aproximácia správania batérie počas nabíjania. Žiaľ ani tento model nezachytáva všetky prípady, ktoré môžu správanie batérie zmeniť. [10]

1.2.3 Typy nabíjacích staníc.

Nasledujúce typy nabíjacích staníc pochádzajú z implementácie ACN Portal.

ChargingNetwork

Táto nabíjacia stanica predpokladá, že maximálne jedno elektrické vozidlo môže byť napojené na každú nabíjačku.

StochasticNetwork

Táto nabíjacia stanica pridelí uje nabíjačky vozidlám náhodne. Implementuje čakací front v prípade ak nie sú voľné nabíjačky. Ak nastavíme parameter early departure na pravdivý, tak dovoľíme výmenu vozidla nabíjajúcim na stanici s vozidlom, ktoré sa nachádza v čakacom fronte. Tento typ nabíjacej siete je vhodný pre aplikovanie v reálnom živote, ale aj v pri generovaní udalostí zo štatistických modelov. [10]

1.2.4 Obmedzenia pri nabíjaní.

Nabíjacie systémy často fungujú na princípe radiálnych sietí. Musíme preto obmedziť množstvo voltov prechádzajúcich cez každé úzke miesto siete. Pomocou Kirchhoffových zákonov vieme definovať obmedzenia pri nabíjaní takto:

$$|I_j(t)| = \left| \sum_{i=1}^N A_{ij} r_i(t) e^{j\phi_i} \right| \leq R_j, \quad (1.3)$$

kde R_j je veľkosť prúdu, $I_j(t)$ je prúd prúdiaci cez úzke miesto siete, N je počet nabíjačiek v nabíjacej stanici, $r_i(t)$ je prúd poskytovaný nabíjačkou i v čase t . Dokopy máme T časových krokov. Parametrom ϕ_i vyjadrujeme fázový uhol pre aktuálny fázor, ktorý závisí na tom ako je nabíjačka i zapojená do siete. [10]

1.3 Problém nájdenia optimálneho schedulingu.

Problém nájdenia optimálneho plánu nabíjania (schedulingu) elektrických vozidiel považujeme za optimalizačný problém. Ako vstup všetkých scheduling algoritmov považujeme množinu elektrických vozidiel a parametre na strane agregátora flexibility. Výstupom všetkých scheduling algoritmov je plán nabíjania (scheduling).

Existuje veľa riešení optimalizačného problému nájdenia optimálneho plánu nabíjania vozidiel, lebo môžeme určiť rôzne obmedzenia nabíjacej siete, parametre, a aj cieľové funkcie (Cieľové funkcie určujeme na základe toho, čo chceme v našom pláne nabíjania dosiahnuť). Preto neexistuje jasná matematická formulácia tohoto problému. [14] Nižšie uvádzame scheduling algoritmy, ktoré my využívame pri našej implementácii, alebo ich používame už priamo z existujúcich riešení pri porovnávaní algoritmov.

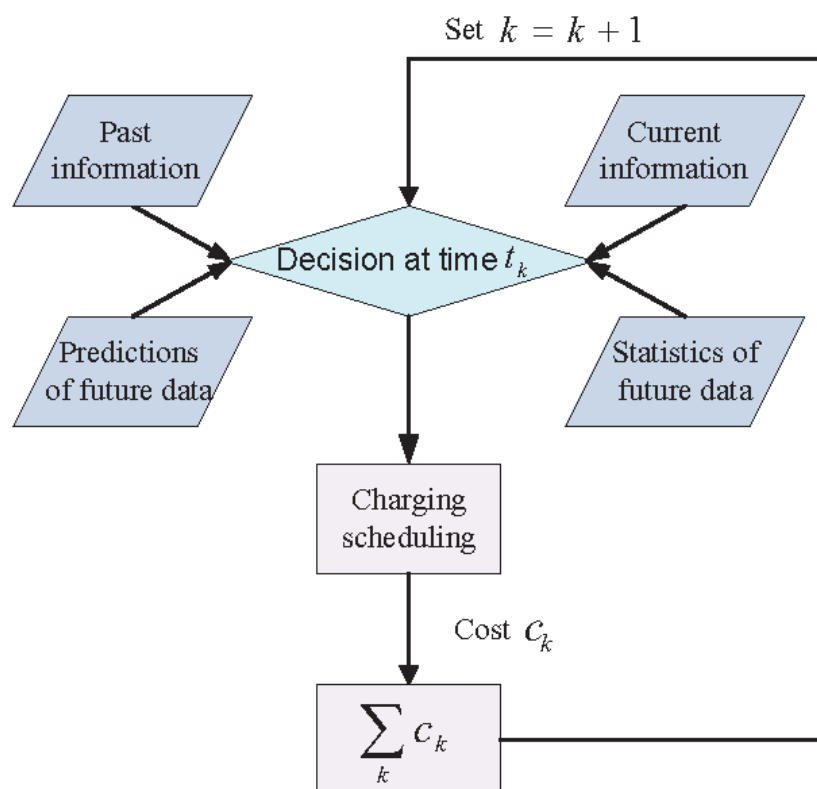
Algoritmy pre nabíjanie elektrických vozidiel spadajú do 2 kategórii. Môžu byť scheduling online algoritmy, alebo offline scheduling algoritmy.

Offline algoritmy

Vyžadujú všetky informácie o všetkých elektrických vozidlách (vrátane parametrov elektrických vozidiel s príchodom v budúcnosti), aby sme sa plnohodnotne vedeli rozhodnúť, aká bude rýchlosť nabíjania.

Online algoritmy

Online algoritmy využívajú len údaje o elektrických vozidlách nachádzajúcich sa na čerpacej stanici. Najväčším problémom, ktorý všetky online algoritmy riešia, aby maximalizovali svoju efektivitu je problém predpovedania príchodov nových elektrických vozidiel do nabíjacej stanice. Optimálnu rýchlosť nabíjania v online algoritmoch môžeme zistiť pomocou riešenia problému konvexnej optimalizácie, alebo problému lineárneho programovania. Pre účely minimalizovania časovej a pamäťovej zložitosti metóda bisekcie a triediace algoritmy sa často využívajú.



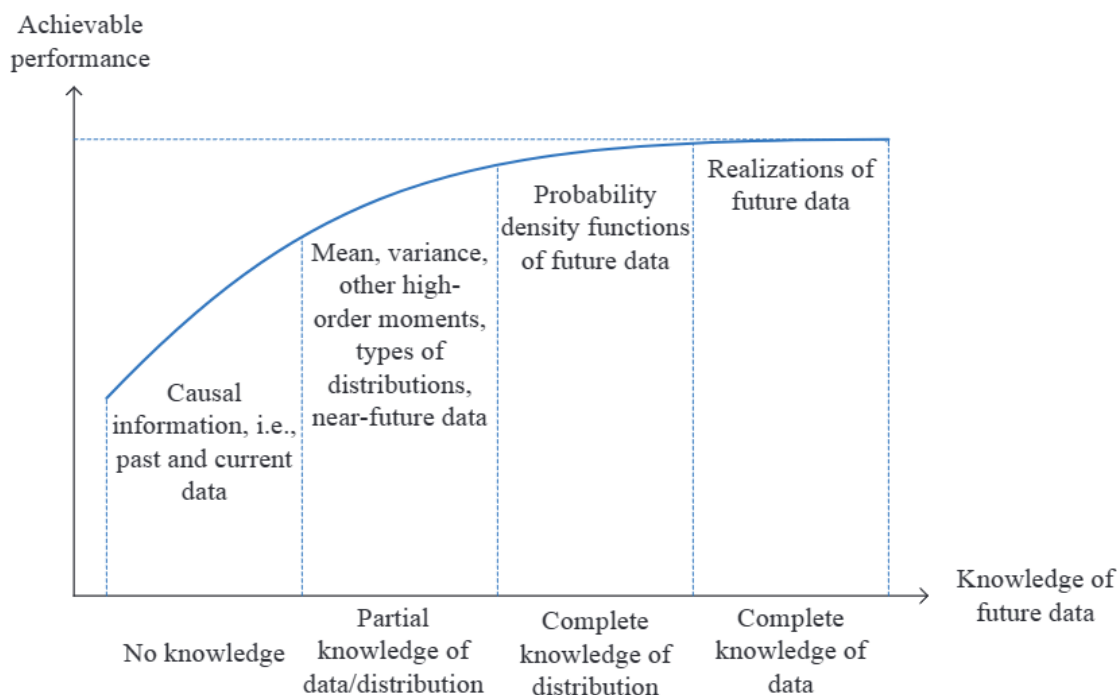
Obr. 1.5: Proces schedulingu pre online scheduling algoritmy pre nabíjanie elektrických vozidiel. Zdroj obrázka je [16]

Obrázok 1.5 ilustruje všeobecne proces schedulingu pre online algoritmy. Rozhodnutie, komu akému elektrickému vozidlu pridelit' koľko energie závisí od viacerých vecí:

1. Informácii v minulosti.
2. Informácii v tomto okamihu.
3. Informácii v budúcnosti.
4. Štatistík o budúcich dáta (ak presné dáta o budúcich príchodoch nemáme).

Porovnanie offline a online algoritmov

Keďže je ťažké získavať informácie o príchode nových elektrických vozidiel (drahé alebo nemožné), tak sa prevažne využívajú v praxi online algoritmy. Tým, že offline algoritmy majú viac informácii o budúcich príchodoch elektrických vozidiel tak konečnom dôsledku generujú lepší scheduling než online algoritmy.



Obr. 1.6: Graf ilustrujúci vplyv určitých vedomostí o budúcich príchodoch elektrických vozidiel na kvalitu algoritmu. Zdroj obrázka je [16]

Náš prístup k riešeniu problému

My riešime online algoritmy, s nižšou komplexitou, kde nepotrebujeme riešiť predikcie o požadovanej energii a iných informáciách o budúcich vozidlách. Namiesto toho, zistíme koľko máme energie dodávať na základe problému prístupnosti. Problém prístupnosti vyriešime tak, že zabezpečíme každému elektrickému vozidlu jeho požadovanú energiu a pritom zachováme obmedzenia našej infraštruktúry nabíjacej siete. [1]

1.4 Uncontrolled charging.

Tento scheduling algoritmus pridelí každému elektrickému vozidlu toľko energie, koľko je možné maximálne prideliť nerátajúc obmedzenia infraštruktúry nabíjacej stanice. Toto je najjednoduchší typ nabíjania, ktorý je stále používaný vo väčšine nabíjacích systémov na svete. [10]

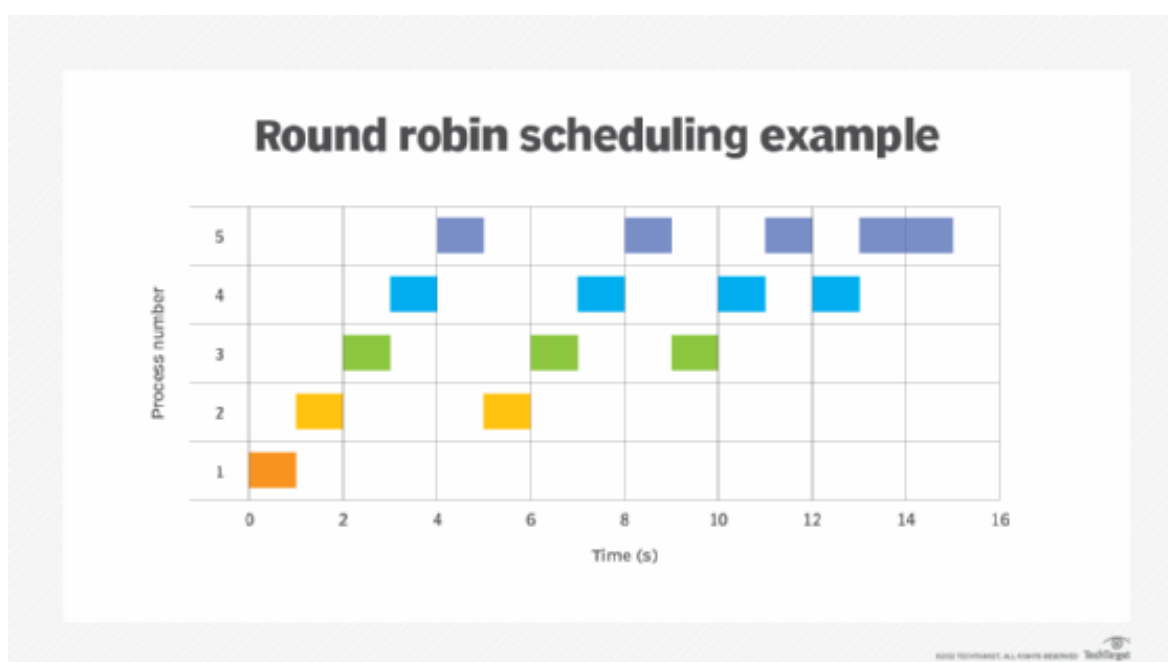
1.5 Round Robin.

Scheduling algoritmus s názvom Round Robin, ktorý nazývame skrátene RR je algoritmus, ktorý je založený na myšlienke zdieľania nabíjacej kapacity medzi všetkými aktívnymi

elektrickými vozidlami. Pre každé elektrické auto kontroluje, či je prístupné zvýšiť nabíjanie o 1Kwh. Potom môžu nastať 2 situácie:

1. Je prípustné zvýšiť nabíjanie o 1Kwh: Nabije auto so zvýšenou rýchlosťou nabíjania a následne elektrické vozidlo dá na koniec frontu.
2. Nie je prípustné zvýšiť nabíjanie o 1Kwh: Nabije vozidlo s fixným nabíjaním, ale nevráti vozidlo do frontu, to znamená, že vozidlo nabíja dovtedy, keď bude možné rýchlosť nabíjania elektrického vozidla zvýšiť.

Vozidlá, ktoré dostali požadovanú elektrickú energiu sú z fronty odstránené. Algoritmus RR skončí vtedy, keď vo fronte nebude žiadne vozidlo. [10]



Obr. 1.7: Pridelovanie času procesom na základe scheduling algoritmu Round Robin. Zdroj obrázka je [16]

Príklad: **UNFINISHED**

1.6 Triediace scheduling algoritmy

Algoritmy, ktoré spomíname v tejto sekcii sú založené na triedení elektrických vozidiel. Rozdiel medzi jednotlivými algoritmi spočíva prevažne v tom, podľa akého parametra alebo podľa akých parametrov triedia. Následne pomocou už spomínanej metódy bisekcie alebo pomocou iných metód (napr. metóda lineárneho vyhľadávania) sa prideluje elektrickým vozidlám energia. Tieto algoritmy môžu byť implementované aj ako online aj ako offline algoritmy. V prípade offline verzie triediacich algoritmov vieme lepšie minimalizovať

cenu, zatiaľ čo v online verzii bez ďalších optimalizácií minimalizovanie nákladov na energiu nevieme veľmi ovplyvniť.

1.6.1 Least Laxity First.

Least Laxity First skrátene nazývame LLF, je scheduling algoritmus, ktorý prideluje vozidlám energiu na základe ich laxity. Na začiatku utriedime vozidlá na základe ich laxít, od najmenšej po najväčšiu a následne im prideliť elektrickú energiu v tom poradí. Nevýhoda scheduling algoritmu LLF je, že spôsobuje časté výkyvy pri dodávaní energie. Dlhodob, taký spôsob nabíjania môže skrátiť životnosť batérii niektorých vozidiel.[1]

Laxitu počítame na základe vzorca:

$$\text{Laxita} = \text{zostávajúci čas do odchodu} - \frac{\text{zostávajúca požadovaná energia}}{\text{rýchlosť nabíjania}}, \quad (1.4)$$

kde predpokladáme, že rýchlosť nabíjania je vždy konštantná.

Príklad

Každé elektrické vozidlo i má definovanú trojicu (a_i, d_i, e_i) , kde a_i je čas pripojenia vozidla na nabíjačku, d_i je čas odpojenia vozidla z nabíjačky a e_i je požadovaná energia. Máme 4 vozidlá ktoré majú nasledujúce parametre:

1. (9, 15, 4).
2. (9, 12, 1).
3. (12, 17, 4).
4. (9, 13, 3).

Zároveň máme 4 nabíjačky v nabíjacej stanici, a preto vieme zabezpečiť nabíjanie vozidiel kedykoľvek, bez použitia prioritnej fronty. Predpokladáme, že cena energie je konštantná. Zvoľme rýchlosť nabíjania každej stanice na 1. Obmedzenia nabíjacej siete sú, že môže každú hodinu dodať len $2Kw$ energie.

Naša úloha je nájsť optimálny scheduling, kde našim cieľom je zabezpečiť požadované množstvo energie pre všetky elektrické vozidlá.

Náš optimálny scheduling problému je:

	t = 9	t = 10	t = 11	t = 12	t = 14	t = 15	t = 16
Vozidlo 1	1	0	1	1	1	-	-
Vozidlo 2	0	1	0	-	-	-	-
Vozidlo 3	-	-	-	1	1	1	1
Vozidlo 4	1	1	1	0	-	-	-

Ak napríklad vymeníme nabíjanie vozidla 2 o 10:00 s nabíjaním vozidla 1, tak nám vzniká iný scheduling, ktorý je tiež optimálny.

Na druhej strane existuje aj neoptimálny scheduling tohoto príkladu, vypojením vozidla 3 o 12:00. Vozidlo 3 v tom prípade by sa vpojilo z nabíjacej siete buď o 9:00, 10:00 alebo 11:00 a napojilo o 12:00.

1.6.2 Smoothed Least Laxity First.

Smoothed Least Laxity First (skrátene sLLF) je scheduling algoritmus založený na algoritme LLF. Tento algoritmus v porovnaní s LLF má vylepšenú úspešnosť v riešení problému prístupnosti. Algoritmus sLLF tiež neobsahuje nepotrebné oscilácie narozdiel od LLF algoritmu.

Laxitu v tomto algoritme budeme počítat' odlišne, než v bežnom LLF, takto:

$$l_i(t) = [d_i - t]^+ - \frac{e_i(t)}{\bar{r}_i}. \quad (1.5)$$

Keďže pri našej implementácii máme k dispozícii len aktívne elektrické vozidlá, tak pre tie vozidlá, ktoré ešte neprišli laxitu nepočítame.

Pre prípad $t < d_i$ počítame $l_i(t + 1)$ takto:

$$\begin{aligned} l_i(t + 1) &= [d_i - (t + 1)]^+ - \frac{e_i(t + 1)}{\bar{r}_i}. \\ l_i(t + 1) &= [d_i - t]^+ - 1 - \frac{e_i(t + 1)}{\bar{r}_i}. \\ l_i(t) + \frac{e_i(t)}{\bar{r}_i} &= [d_i - t]^+. \\ l_i(t + 1) &= l_i(t) + \frac{e_i(t)}{\bar{r}_i} - 1 - \frac{e_i(t + 1)}{\bar{r}_i}. \end{aligned}$$

Kde $e_i(t + 1) - e_i(t) = r_i(t)$ (Množstvo dodanej energie za čas t), tým dostaneme:

$$l_i(t + 1) = l_i(t) - 1 - \frac{r_i(t)}{\bar{r}_i}.$$

Druhý krok algoritmu je maximalizácia minimálnej laxity medzi všetkými elektrickými vozidlami.

UNFINISHED

Výpočtová zložitosť tohoto algoritmu je $O(V + \log(\frac{1}{\delta}))$, kde V je počet aktívnych elektrických vozidiel a δ je level tolerovateľnej chyby. [1]

1.6.3 Group Least Laxity First.

UNFINISHED

1.6.4 Earliest deadline first.

Early deadline first skrátene nazývame EDF, je scheduling algoritmus ktorý prirad'uje elektrickú energiu elektrickým vozidlám na základe času ich odchodu. Prioritu pri nabíjaní sú vozidlá, ktoré majú čas odchodu skorší. [10]

1.6.5 Group Earliest Deadline First.

Group Earliest Deadline First je scheduling algoritmus, ktorého vytvorili nato, aby zlepšil výkonnosť EDF počas preťažená reálnej multimediálnej aplikácie. Tento scheduling algoritmus sa zakladá na myšlienke skupinového schedulingu. To znamená, že elektrické vozidlá, ktoré majú podobné časy príchodu patria do skupiny. Potom sa prirad'uje energia týmto vozidlám na základe scheduling algoritmu Shortest Job First.[15]

1.6.6 Group priority earliest deadline first (EDF).

[15]

1.6.7 First-Come First-Served.

Scheduling algoritmus First-Come First-Served (skrátene FCFS) má veľmi podobnú funkcionalitu ako uncontrolled charging algoritmus. Hlavný rozdiel spočíva v tom, že algoritmus FCFS je časovo oneskorený, lebo obsahuje autá v rade, takže každé elektronické vozidlo si musí počkať pre svoje nabíjanie. [10]

1.6.8 Last-Come First-Served.

Scheduling algoritmus Last-Come First-Served (skrátene LCFS) prirad'uje energiu elektrickým vozidlám, prioritne od vozidla s najneskorším časom pripojenia na nabíjačku po vozidlo s najskorším časom pripojenia na nabíjačku. [10]

1.6.9 Longest Remaining Processing Time.

Scheduling algoritmus Longest remaining processing time je algoritmus, ktorý prideluje prioritu pri nabíjaní elektrických vozidiel takto:

1. Pretriedi vstupné pole aktívnych elektrických vozidiel, v poradí od vozidla s najdlhším zostávajúcim časom nabíjania po vozidlo s najkratším zostávajúcim časom nabíjania.

UNFINISHED

1.6.10 Shortest Job Next.

Algoritmus Shortest Job next, skrátene SJN je scheduling algoritmus, ktorý funguje ako Longest remaining processing time až na jeden rozdiel. Rozdiel je, že pole aktívnych elektrických vozidiel triedi od elektrického vozidla s najkratším zostávajúcim časom nabíjania, po vozidlo s najdlhším zostávajúcim časom nabíjania.

1.7 Kontrolné scheduling algoritmy

V tejto sekcii spomenieme kontrolné algoritmy, s ktorými budeme našu implementáciu porovnávať. Kontrolné algoritmy sú algoritmy, ktoré poskytujú kontrolné akcie, na základe vstupných dát, aby sme zistili nejaký cieľ. Bez kontrolných algoritmov by dnes nemohli fungovať automatické systémy, ktoré často využívame dnes. Tieto algoritmy musia vedieť fungovať v obmedzenom prostredí ako napríklad pri obmedzeniach uvedených v sekcii ??.

[2]

1.7.1 Typy kontrolných algoritmov

open-loop algoritmy

Sú najjednoduchším typom kontrolných algoritmov. Vždy vracajú pre rovnaké vstupné dáta rovnaké výstupné dáta. Používajú sa v systémoch, kde netreba meniť výstupné dáta na základe zmeny vo vstupných dátach.

closed-loop algoritmy

Sú zložitejšie než open-loop algoritmy. Používajú feedback aby zmenili výstupné dáta na základe zmien vo vstupných dátach.

feedback-control algoritmy

Sú najzložitejším typom kontrolných algoritmov. Používajú feedback aby zmenili výstupné dáta na základe zmien vo vstupných dátach.

1.7.2 Model Predictive Control.

Model Predictive Control skrátene MPC je jeden z kontrolných algoritmov. MPC je feedback control algoritmus. MPC využíva prediktívny model, aby zistil kontrolné akcie v budúcnosti, ktoré optimalizujú výkon počas nejakého časového horizontu. Cieľom MPC je zabezpečiť čo najlepší kontrolný výkon, a pritom splňať obmedzenia systému.

Komponenty MPC

MPC sa skladá z 3 komponentov:

1. Matematický model procesu. Tento matematický model zachytáva vzťah medzi vstupom a výstupom a zahŕňa aj dynamiku systému.
2. Optimizátor, ktorý rieši optimalizačný problém ktorý maximalizuje výkonnosť systému a pritom spĺňa obmedzenia systému.
3. Funkcia nákladov, ktorá meria kvalitu riešenia optimizátora.

Výhody MPC

MPC má výhodu pred tradičnými kontrolnými algoritmami, že nemusí mať fixný počet kontrolných parametrov, ktoré sú potom upravované na základe experimentov. MPC využíva dynamický model, aby predikoval správanie v budúcnosti. MPC na rozdiel od tradičných kontrolných algoritmov vie aj riešiť nelineárne kontrolné problémy. Nelineárne kontrolné problémy sú problémy, v ktorých vzťah medzi vstupom a výstupom je nelineárny.

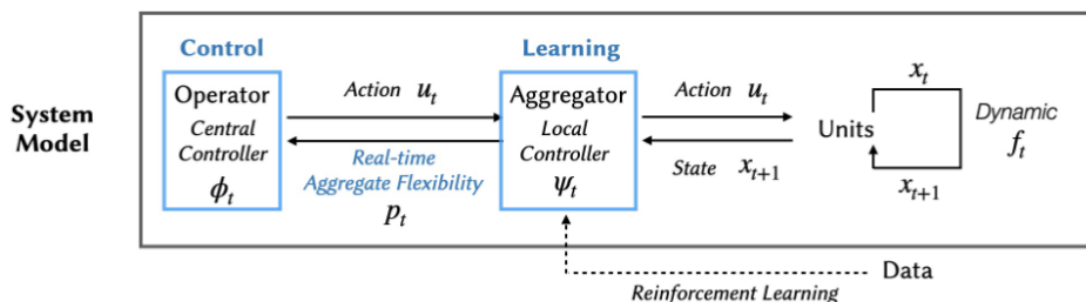
Existujúce riešenia

Veľa existujúcich riešení problému hľadania optimálneho schedulingu využíva Model Predictive Control, skrátene MPC. My sa orientujeme hlavne na algoritmus nazývaný Adaptive Charging Algorithm, ktorý je založený na MPC a rieši pritom problém konvexnej optimalizácie. [10, 11]

1.7.3 Penalised Predictive Control.

Algoritmus Penalised Predictive Control (PPC) je vylepšená verzia algoritmu MPC. Pointa algoritmu spočíva v komunikácii medzi systémovým operátorom a agregátorom. Agregátor poskytuje flexibilitu (naučenú na základe RL algoritmu SAC) systémovému operátorovi, ktorý na základe flexibility vyrieši optimalizačný problém a vráti elektrickú energiu, ktorú potom následne agregátor pomocou vybraného scheduling algoritmu prideli elektrickú energiu od systémového operátora elektrickým vozidlám napojeným na nabíjačky.

Narozdiel od MPC tento algoritmus nevyžaduje veľa premenných, ale na druhej strane obsahuje obsiahly algoritmus SAC na učenie flexibility. Výhodou PPC je tiež, že nemusí vedieť o obmedzeniach siete a stavu, pri riešení optimalizačného problému, lebo to je zahrnuté vo flexibilitate generovanej agregátorom. [13]



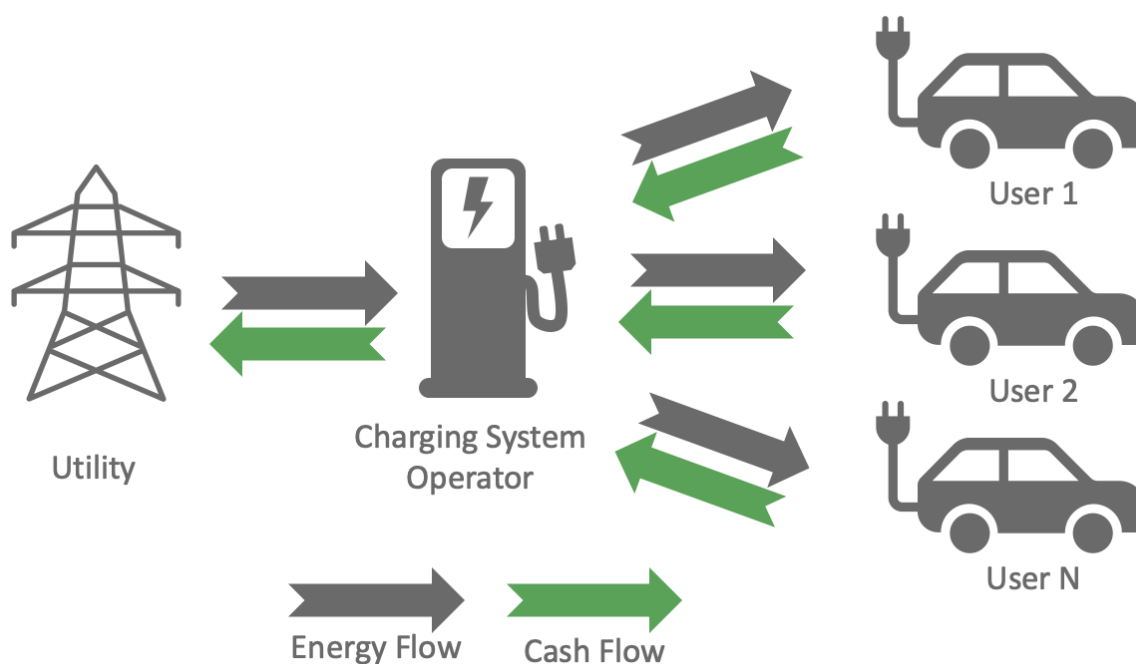
Obr. 1.8: Model systému . Zdroj obrázku je [12]

1.8 Cena nabíjania.

Náš spôsob definovania ceny nabíjania je: Na konci každého mesiaca nastavíme ceny nabíjania ako α_i^t , pre každé nabíjanie i v tom mesiaci. Hovoríme, že e_i je dodaná energia pre spotrebiteľa j počas nabíjania i , kde S_j je množina všetkých nabíjaní používateľa j . Potom náklady za nabíjania spotrebiteľa j sa na konci mesiaca vypočítajú takto:

$$\sum_{i \in S_j} \alpha_i^* e_i. \quad (1.6)$$

Narozdiel od konštantných cien a cien závislých na čase (napríklad v [12]) my používame ceny (tarify), ktoré zachytávajú skutočné ceny energie, preťaženie infraštruktúry a na základe dopytu po elektrickej energii. [6]



Obr. 1.9: Model systému . Zdroj obrázku je [12]

2 Návrh riešenia

2.1 Least Laxity First

3 Návrh softvérového diela

4 Výskum

5 Výsledky

Záver

Literatúra

- [1] Niangjun Chen, Christian Kurniawan, Yorie Nakahira, Lijun Chen, and Steven H. Low. Smoothed least-laxity-first algorithm for ev charging, 2021.
- [2] Collimator. What is a control algorithm? <https://www.collimator.ai/reference-guides/what-is-a-control-algorithm#:~:text=Definition%20of%20a%20control%20algorithm&text=These%20algorithms%20are%20designed%20to,is%20either%20undesirable%20or%20impossible.,> 2023. Accessed: 3rd December, 2023.
- [3] Z. J. Lee and S. Sharma. acnportal-experiments. <https://github.com/caltech-netlab/acnportal-experiments>, Dec. 2023.
- [4] Z. J. Lee and S. Sharma. adacharge. <https://github.com/caltech-netlab/adacharge>, Dec. 2023.
- [5] Z. J. Lee, S. Sharma, and D. Johansson. acnportal. <https://github.com/zach401/acnportal>, Dec. 2023.
- [6] Zachary Lee, John Pang, and Steven Low. Pricing ev charging service with demand charge. *Electric Power Systems Research*, 189:106694, 12 2020.
- [7] Zachary J. Lee, Daniel Johansson, and Steven H. Low. Acn-sim: An open-source simulator for data-driven electric vehicle charging research. In *2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, pages 1–6, 2019.
- [8] Zachary J. Lee, George Lee, Ted Lee, Cheng Jin, Rand Lee, Zhi Low, Daniel Chang, Christine Ortega, and Steven H. Low. Adaptive charging networks: A framework for smart electric vehicle charging, 2020.
- [9] Zachary J. Lee, Tongxin Li, and Steven H. Low. Acn-data: Analysis and applications of an open ev charging dataset. In *Proceedings of the Tenth ACM International Conference on Future Energy Systems, e-Energy '19*, page 139–149, New York, NY, USA, 2019. Association for Computing Machinery.

- [10] Zachary J. Lee, Sunash Sharma, Daniel Johansson, and Steven H. Low. Acn-sim: An open-source simulator for data-driven electric vehicle charging research, 2021.
- [11] Zachary Jordan Lee. *The Adaptive Charging Network Research Portal: Systems, Tools, and Algorithms*. Dissertation, California Institute of Technology, 2021.
- [12] Tongxin Li, Bo Sun, Yue Chen, Zixin Ye, Steven H. Low, and Adam Wierman. Learning-based predictive control via real-time aggregate flexibility. *IEEE Transactions on Smart Grid*, 12(6):4897–4913, nov 2021.
- [13] Tongxin Li, Bo Sun, Yue Chen, Zixin Ye, Steven H. Low, and Adam Wierman. Learning-based predictive control via real-time aggregate flexibility. *IEEE Transactions on Smart Grid*, 12(6):4897–4913, November 2021.
- [14] Joy Chandra Mukherjee and Arobinda Gupta. A review of charge scheduling of electric vehicles in smart grid. *IEEE Systems Journal*, 9(4):1541–1553, 2015.
- [15] Vijayshree Shinde and Seema C. Comparison of real time task scheduling algorithms. *International Journal of Computer Applications*, 158:37–41, 01 2017.
- [16] Wanrong Tang, Suzhi Bi, and Ying Jun Angela Zhang. Online charging scheduling algorithms of electric vehicles in smart grid: An overview. *IEEE Communications Magazine*, 54:76–83, 2016.
- [17] Jake VanderPlas. *Python data science handbook : essential tools for working with data*. O’Reilly Media, Inc, Sebastopol, CA, 2016.
- [18] ZDWL. Ev charger levels. <https://zdw1-tec.com/news/ev-charger-levels/>, 2023. Accessed: 3rd December, 2023.

Príloha A: obsah elektronickej prílohy

Príloha B: Používateľská príručka