Daniel Pál, Jakub Urbanský, Ľubomír Beňa, Michal Ivančák, Maksym Oliinyk

Porovnanie optimalizačných metód z hľadiska zadefinovania ich parametrov

Abstrakt: Tento článok je zameraný na opis vybraných optimalizačných metód. Optimalizačná metóda sa veľmi podobá na prirodzenú evolúciu a preto je ju možné nazvať aj ako evolučný algoritmus. Pomocou optimalizačných metód je možné nájsť optimálne riešenie matematickej alebo technickej úlohy. Článok ponúka prehľad potrebných vstupných parametrov, ktoré je potrebné zadefinovať pri používaní rôznych optimalizačných metód, ako sú napríklad Samo Organizujúci sa Migračný Algoritmus (SOMA), Diferenciálna evolúcia (DE), genetický algoritmus alebo Rojenie častíc (PSO).

Kľúčové slová: Optimalizačné metódy, SOMA, PSO, genetické algoritmy.

Abstract: This paper is focused on the description of selected optimization methods. The optimization method is very similar to natural evolution and therefore, it can be called an evolutionary algorithm. By the methods, it is possible to find an optimal solution to a mathematical or technical problem. The article gives an overview of what parameters are necessary to be defined when using various optimization methods, such as the Self Organizing Migration Algorithm (SOMA), Differential Evolution (DE), Genetic Algorithm or Particle Swarming (PSO).
(Comparison of the parameters required for setting different optimization methods)

Keywords: Optimization methods; SOMA; PSO; genetic algorithms.

I. ÚVOD

V posledných rokoch s nástupom rozvoja výpočtovej techniky postupne narastá aj význam používania tzv. netradičných metód oproti klasickým metódam pri výpočte určitých typov úloh. Pomocou klasických metód často nie je možné vypočítať zložité matematické alebo technické úlohy s potrebnou presnosťou. Netradičné metódy však tieto nedostatky odstraňujú.

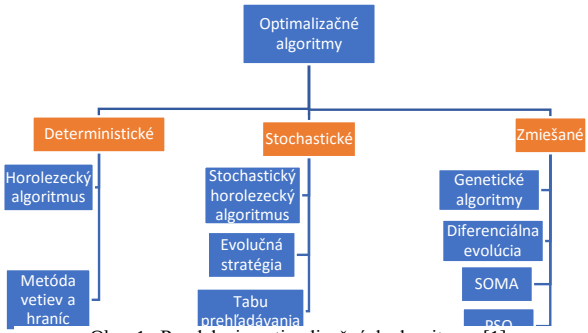
Optimalizačné netradičné metódy sa veľmi podobajú na prirodzenú evolúciu a preto sa často nazývajú aj ako evolučné algoritmy. Tie sú vo všeobecnosti používané na vyriešenie takých úloh, v prípade ktorých je potrebné nájsť najlepšiu variantu z určitej populácie, ktorá sa veľmi podobá na globálne optimum alebo treba nájsť v rámci populácie najmenšiu, respektíve najväčšiu variantu.

V oblasti elektroenergetiky sa vyskytujú rôzne úlohy, ktoré vyžadujú úplne iný postup ako pri klasických metódach riešenia.

Cieľom tohto článku je opísať optimalizačné metódy z hľadiska zadefinovania potrebných vstupných údajov na začatie algoritmov a tiež opisuje aj rozdelenie optimalizačných metód. Popisuje, aké sú výhody, resp. nevýhody jednotlivých skupín týchto algoritmov.

II. OPTIMALIZAČNÉ ALGORITMY

Úlohou optimalizačných metód je zistenie minima (maxima) v určitej populácii, kde pomocou kombinácie všetkých variantov dokáže nájsť najlepšiu z nich. Zvyčajne sú používané tam, kde klasické metódy nedosiahnu požadovanú presnosť alebo ich použitie pre daný typ úlohy nie je možné. Pomocou optimalizačných metód je možné riešiť veľmi široké spektrum úloh. V dôsledku toho je možné optimalizačné metódy rozdeliť na 3 podskupiny. Každá z nich slúži na riešenie iných úloh. Spomínané rozdelenie je možné vidieť na Obr. 1.

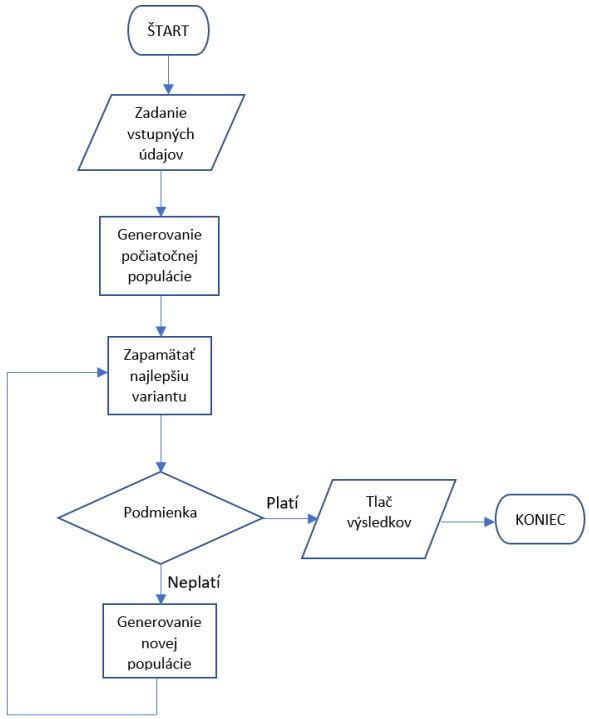


Obr. 1. Rozdelenie optimalizačných algoritmov [1]

- **Deterministické** – algoritmus je založený na presných matematických metódach. Dôležitou požiadavkou je, aby na správnu funkciu boli zadefinované potrebné vstupné predpoklady. Tieto predpoklady sú napríklad:
  - vyriešený problém je lineárny alebo konvexný,
  - vyšetrovaný priestor (populácia) je malý,
  - pri riešení úloh je iba 1 extrém (minimum resp. maximum) [1].
- **Stochastické** – algoritmus vo svojom procese využíva náhodu. Pri procese sa náhodne hľadajú hodnoty argumentov funkcie, ktorých výsledkom je vždy najlepšie riešenie. Nevýhodou týchto algoritmov je to, že pomocou nich nie je možné zistiť, kedy bude úloha vyriešená, keďže používajú iba náhodné hľadanie. Ďalšou nevýhodou je to, že využívanie týchto metód je možné len na vyhľadávanie malého priestoru [1].
- **Zmiešané** – tento algoritmus je kombináciou predošlých dvoch algoritmov, teda stochastického a deterministického algoritmu. Používa výhody obidvoch metód. Hlavnými výhodami používania zmiešaného algoritmu sú:

- rýchlosť;
- žiadne, resp. minimálne požiadavky na vstupné informácie;
- schopnosť vyriešiť viac úloh behom jedného výpočtu [1].

Všeobecný vývojový diagram pre optimalizačné metódy sa nachádza na Obr. 2. Diagramy algoritmov sa od seba navzájom mierne odlišujú. Rozdiely sú napríklad v rozsahu parametrov, vstupných údajov, ktoré vstupujú do výpočtu. Obr. 2 vo všeobecnosti dokáže presne opísať fungovanie algoritmov. Prvou úlohou je zadefinovanie vstupných parametrov, z ktorých program vygeneruje počiatočnú populáciu. Nasleduje podmienka, v prípade ktorej algoritmus vyhodnocuje, či je už dosiahnutá požadovaná hodnota výsledku, napríklad z hľadiska presnosti. Ak požadovaná hodnota výsledku bola dosiahnutá, program prezentuje výsledky. Ak daná hodnota ešte nebola dosiahnutá, algoritmus vygeneruje ďalšiu populáciu. Tento úkon sa opakuje až dovtedy, kým sa nedosiahne požadovaná presnosť.



Obr. 2. Všeobecný vývojový diagram pre optimalizačné metódy [2]

III. SAMO ORGANIZUJÚCI SA MIGRAČNÝ ALGORITMUS

Samo Organizujúci sa Migračný Algoritmus (SOMA – Self-Organizing Migrating Algorithm) je algoritmus, ktorý bol prvýkrát prezentovaný už v roku 1999 [1]. Patrí do 3. skupiny spomínaných metód, a kombinuje výhody stochastických aj deterministických metód.

Parametre SOMA:

Pri definovaní SOMA algoritmu je možné parametre, ktoré majú vplyv na funkciu algoritmu rozdeliť do dvoch skupín:

1. *riadiace parametre* – majú vplyv na kvalitu algoritmu.
2. *ukončovacie parametre* – slúžia na ukončovanie algoritmu na základe vopred zadanej podmienky [1].

Riadiace a ukončovacie parametre SOMA je možné vidieť v Tabuľke I.

TABUĽKA I
Parametre SOMA [1], [3]

Parameter	Typ
Dĺžka cesty	Riadiaci
Krok	Riadiaci
Perturbácia	Riadiaci
Veľkosť populácie	Riadiaci
Dimenzia	Počet argumentov
Migrácia	Ukončovací
Chyba	Ukončovací

Jednotlivé parametre SOMA je možné charakterizovať nasledovne:

Dĺžka cesty – je dĺžka, ktorú aktívny prvok prejde dovtedy, kým sa nezastaví v istej vzdialenosti od vedúceho prvku. Zvyčajný rozsah na nastavenie sa pohybuje medzi hodnotami 1 až 3. Najčastejšie používaná hodnota je 3, čo znamená, že aktívny prvok sa v tomto prípade môže vzdialiť dovtedy, kým cesta nebude trojnásobná. Pri nastavení hodnoty 1 sa aktívny prvok dostane do tej istej pozície, kde je vedúci prvok [1], preto minimálna hodnota má byť väčšia ako 1. Je to potrebné kvôli tomu, aby sa prvky vždy vzdialili od vedúceho prvku a tým zabezpečovali, aby algoritmus po určitom čase našiel riešenie.

Krok – tento parameter slúži na vzorkovanie cesty aktívneho prvku. Pri riešení úloh môžu nastať 2 stavy, a to keď účelová funkcia je známa alebo nie. V prípade poznania účelovej funkcie je možný daný parameter nastaviť na vysokú hodnotu a tým pádom urýchliť výpočet. Pri nepoznaní účelovej funkcie je potrebný daný parameter nastaviť na nízku hodnotu, aby bola mapovaná cesta aktívneho prvku [1].

Perturbácia – je jedna z najdôležitejších riadiacich parametrov, pomocou ktorej je možné ovplyvniť, či sa má aktívny prvok k vedúcemu prvku pohybovať priamo alebo nepriamo. Zvyčajný rozsah môže nadobudnúť hodnotu od 0 do 1 [3].

Veľkosť populácie – pomocou tohto parametra je možné nastaviť veľkosť populácie, t. j. koľko prvkov bude obsahovať populácia. Minimálny počet v rámci populácie sa nastavuje na hodnotu 10, maximálnu hodnotu definuje riešiteľ úloh [1].

Dimenzia – vďaka tomuto parametru je možné nastaviť počet argumentov účelovej funkcie [1].

Migrácia – nastavuje koľkokrát je možné zmeniť miesto prvkov v populácii. Jej minimálna hodnota sa nastavuje na 10, maximálnu hodnotu nastaví užívateľ [3].

Chyba – je ukončovacím parametrom. Pomocou nej sa nastavuje, kedy má algoritmus ukončiť svoju úlohu. Algoritmus zisťuje rozdiel medzi vedúcim prvkom a najhorším prvkom [3]. Ak je hodnota rozdielu menšia ako nastavená hodnota „chyby“, algoritmus ukončí svoju prácu. V opačnom prípade algoritmus bude prebiehať dovtedy, kým spomínaný rozdiel nebude menší.

Algoritmus SOMA funguje na základe vyššie spomenutých parametrov. Pomocou riadiacich parametrov sa nastavuje kvalita algoritmov (ako rýchlo, ako presne bude daný algoritmus fungovať), kým ukončovacie parametre slúžia na ukončovanie programu. Algoritmus sa ukončí, ak je nastavená chyba menšia ako je rozdiel medzi najhorším a najlepším prvkom, prípadne sa algoritmus môže skončiť aj vtedy, ak sa populácia viackrát preorganizuje.

IV. ROJENIE ČASTÍC

Rojenie častíc (PSO - Particle Swarm Optimization). V roku 1995 ho vyvinuli Dr. R. C. Eberhart a Dr. James Kennedy. Pri vytvorení tohto algoritmu sa inšpirovali správaním vtákov. Príspevok o PSO prezentovali na kongrese, ktorý sa zaoberal evolučnými výpočtami [2][4][5].

Parametre PSO je možné vidieť v Tabuľke II.

TABUĽKA II
Parametre PSO [4]

Parameter	Typ
Dimenzia	Počet argumentov
Počet prvkov	Riadiaci
vMax	Riadiaci
C1, C2	Riadiaci
w	Riadiaci
MaxIterácia	Ukončovací

Parametre PSO:

Dimenzia – pomocou tohto parametra je možné nastaviť počet argumentov účelovej funkcie [4].

Počet prvkov – pomocou spomínaného parametra sa nastavuje, z koľkých prvkov sa bude skladať populácia. V prípade algoritmu SOMA je to parameter veľkosť populácie [4].

vMax – je to parameter, ktorý udáva maximálnu rýchlosť prvku. Rýchlosť charakterizuje, ako dôkladne sa vyšetruje priestor v ktorom sa nachádza častica. Ak je rýchlosť malá, vtedy je priestor veľmi dôkladne prešetrovaný a naopak, pri vysokej rýchlosti sa vyšetruje len hrubo [4] - približne ako parameter „krok“ v prípade SOMA algoritmu.

C1, C2 – tieto parametre sú tzv. učiace faktory, ktoré určujú smer cesty jednotlivých častíc. V prípade C2 všetky sa častice posúvajú k najlepšiemu globálnemu riešeniu. V prípade C1 si častica vždy pamätá na svoju najlepšiu hodnotu v populácii a posúva sa k najlepšiemu riešeniu [4].

w – Zotrvačnosť. Udávajú sa 1 až 2 parametre. V prípade jednej hodnoty sa nastavuje jej hodnota, pričom počas výpočtu sa táto hodnota zmenšuje a vždy sa bude kontrolovať menší priestor [4]. V prípade, keď sú zadefinované 2 hodnoty sa definuje jej počiatočná a konečná veľkosť.

V prípade zadefinovania dvoch hodnôt sa w vypočíta podľa vzťahu [6][7]:

$$w = w_{zač.} - \frac{(w_{zač.} - w_{kon.}) * iterácia}{MaxIterácia} \quad (1)$$

Kde: *MaxIterácia* – maximálny počet iterácií

Iterácia – aktuálne poradie iterácie

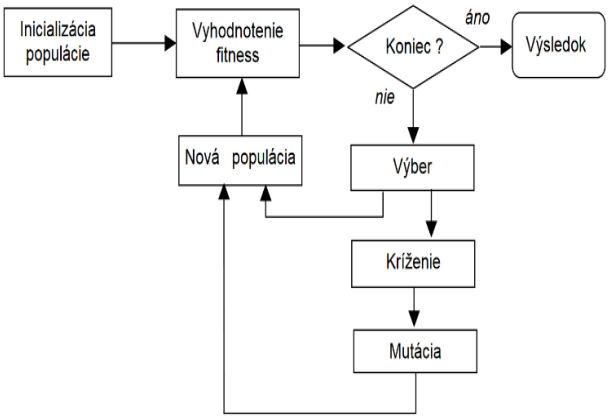
MaxIterácia – je ukončovacím parametrom. Pomocou neho sa nastavuje, koľko iterácií bude urobených dovtedy, kým algoritmus nenájde riešenie [6].

Tieto parametre sú najdôležitejšie pri definovaní algoritmu PSO. Potrebné parametre sa veľmi podobajú na tie parametre, ktoré sú potrebné pri definovaní SOMA algoritmu. V prípade týchto algoritmov je vždy potrebné zadefinovať priestor, kde bude uskutočnený výpočet. Taktiež je potrebné zadefinovať počet prvkov a ukončovaciu podmienku, kedy má algoritmus zastaviť svoju činnosť.

V. GENETICKÉ ALGORITMY

Genetické algoritmy sú jedným z najdôležitejších evolučných algoritmov, boli navrhnuté riešiteľským kolektívom profesora Johna Hollanda z univerzite v Michigane. Svoje výsledky publikovali v knihe „Adaptation in Natural and Artificial Systems“ v roku 1975 [8], [9], [10].

Genetické algoritmy na rozdiel od iných optimalizačných metód nepracujú priamo s parametrami, ktoré pri definovaní úloh treba optimalizovať, ale pracujú s ich reprezentáciou, ako je napr. kríženie, mutácie, atď.



Obr. 3. Schéma genetického algoritmu [11]

Blokovú schému genetického algoritmu znázorňuje Obr. 3. Ako vo všetkých optimalizačných metódach, prvou úlohou je zadefinovanie počiatočnej populácie. Optimalizačné parametre nie sú zadefinované priamo, ale sú definované v génoch a z týchto génoch vzniká reťazec – tzv. chromozóm. Ku každému prvku v populácii je vopred zadefinovaný parameter, nasleduje ich hodnotenie pomocou fitness funkcie, čiže funkcie vhodnosti. Nasleduje podmienka ktorá vyhodnotí, či sú už požadované optimalizované riešenia dosiahnuté (nájdenie minima, maxima, atď.) alebo ešte nie. V prípade dosiahnutia, sa algoritmus ukončí a vypíše výsledky. V prípade nedosiahnutia, algoritmus pokračuje v riešení úlohy. Prvky v populácii sú potom rozdelené do dvoch skupín. Prvá skupina prvkov sa skladá z „najlepších“, ktoré sa už nebudú meniť v populácii. Druhá skupina pomocou kríženia a mutácie bude meniť hodnoty, ktoré sú zadefinované v reťazci. Po vytvorení obidvoch skupín prvkov vzniká nová populácia a algoritmus pokračuje svoju úlohu dovtedy, kým nebudú dosiahnuté požadované podmienky.

VI. DIFERENCIÁLNA EVOLÚCIA

Diferenciálna evolúcia je typ evolučného algoritmu. Patrí do zmiešanej skupiny, kombinuje deterministické a stochastické metódy. Algoritmus vyvinuli K. Price a R. Storn a prvýkrát bol použitý v roku 1995 [1],[12].

Parametre DE je možné vidieť v Tabuľke III.

TABUĽKA III
Parametre diferenciálnej evolúcie [1]

Parameter	Typ
F	Riadiaci
CR	Riadiaci
D	Počet argumentov
Veľkosť populácie	Riadiaci
Generácia	Ukončovací

Parametre diferenciálnej evolúcie:

F – parameter je tzv. mutačná konštanta, ktorá môže nadobúdať hodnotu od 0 do 2 [1].

CR – parameter je možné nazvať ako prah kríženia, ktorý môže nadobúdať hodnotu od 0 do 1. V prípade nastavenia hodnoty 0 sa

populácia zastaví, lebo na tom istom mieste bude vystupovať ako aktuálny prvok [1]. Práve preto je potrebné zadať väčšiu hodnotu ako 0.

Dimenzia (D) – vyjadruje počet argumentov účelovej funkcie.

Veľkosť populácie (NP) – pomocou tohto parametra je možné nastaviť veľkosť, teda počet prvkov populácie. Na fungovanie algoritmu je potrebné, aby počet prvkov v populácii bol aspoň 4, keďže je to minimálna hodnota potrebná na fungovanie algoritmu [1].

Generácia – udáva počet evolučných cyklov, behom ktorých sa celá populácia vyvíja [1]. V prípade porovnania SOMA a diferenciálnej evolúcie, tento parameter je totožný s migráciou.

VII. ZÁVER

Správne nastavenie optimalizačných algoritmov je extrémne sofistikované úloha z dôvodu potreby definovania viacerých parametrov. Treba poznamenať, že aj malá anomália pri zadávaní a špecifikovaní parametrov, môže do značnej miery ovplyvniť výpočty, ktoré sa potom stávajú extrémne nepresné. Preto pri definovaní parametrov je potrebné vždy správne nastaviť jednotku a rozsah. Pri optimalizačných metódach najdôležitejším parametrom je definovanie populácie, algoritmus musí vedieť, v akom priestore musí hľadať riešenie. Druhým najdôležitejším je ukončovací parameter, na základe ktorého algoritmus ukončí svoju úlohu. Tu môže byť zadefinovaná chyba, alebo počet vykonaných krokov.

Výber vhodných optimalizačných algoritmov na konkrétnu úlohu je značne zložitý. Neexistuje univerzálna optimalizačná metóda na riešenie každého typu problému. Pre daný typ úlohy je možné nájsť dostatočne presnú metódu, avšak v prípade aplikácie danej metódy pri inej úlohe môžu vznikať nepresné respektíve nepoužiteľné výsledky. Napríklad pri vyšetrovaní distribučných sústav z hľadiska strát sa neodporúča výber stochastickej metódy, nakoľko metódu je možné používať len na vyhľadávanie malého priestoru. Ďalším závažným dôvodom nevhodnosti tejto metódy v spomínanom príklade je nemožnosť predvídať, kedy daný algoritmus skončí, nakoľko používa len náhodne vyhľadávanie a tým pádom nie je možné okamžite získať potrebné výsledky. Pri riešení týchto úloh je podľa nás najlepšou možnosťou používanie zmiešanej metódy, pretože kombinuje výhody stochastických a deterministických metód. Pomocou nej je možné vypočítať naraz viac parametrov, napríklad pri riešení distribučných sústav je možné vypočítať behom jedného výpočtu okrem celkových strát aj straty na vedení, napätie v uzloch alebo aj prúdy medzi uzlami.

POĎAKOVANIE

Túto prácu podporila Vedecká grantová agentúra Ministerstva školstva, vedy, výskumu a športu Slovenskej republiky a Slovenskej akadémie vied grantom VEGA č. 1/0372/18.

LITERATÚRA

[1] I. Zelinka, „Umělá inteligence v problémoch globální oprimalizace“. BEN – technická literatura, Praha 2002. ISBN 80–7300–069–5.

[2] K. Voratas, „Comparison of three evolutionary algorithms: GA, PSO, and DE“ in Industrial Engineering and Management Systems 2012. s. 215-223. Dostupné na internete: https://www.researchgate.net/publication/260393066_Comparison_of_three_evolutionary_algorithms_GA_PSO_and_DE

[3] D. Davendra, I. Zelinka, „Self-Organizing Migrating Algorithm, Methodology and Implementation“, Springer International Publishing 2016. ISBN 978-3-319-28161-2 – SOMA parameter

[4] PSO Tutorial [online]. [cit. 2019.07.06]. Dostupné na internete: http://www.swarmintelligence.org/tutorials.php

[5] Chapter 4 – Particle Swarm Optimization. [online]. [cit. 2019.07.20]. Dostupné na internete: https://shodhganga.inflibnet.ac.in/bitstream/10603/28526/9/09_chapter4.pdf

[6] L. Rutkowski, M. Korytkowski, R. Scherer, R. Tadeusiewicz, L. A. Zadeh, J. M. Zurada, „Artificial Intelligence and Soft Computing“ in 13th International Conference, ICAISC 2014 Poland. ISBN 978-3-319-07172-5.

[7] A.Sanayei, I. Zelinka, O. E. Rössler, „ISCS 2013: Interdisciplinary Symposium on Complex Systems“ Springer-Verlag Berlin Heidelberg 2014. ISBN 978-3-642-45438-7

[8] M. Juneja and S. K. Nagar, "Particle swarm optimization algorithm and its parameters: A review," *2016 International Conference on Control, Computing, Communication and Materials (ICCCCM)*, Allahbad, 2016, pp. 1-5.

[9] D. Whitley, “A Genetic Algorithm Tutorial”. [online]. [cit. 2019.07.20]. Dostupné na internete: https://www.researchgate.net/profile/Darrell_Whitley2/publication/2425017_A_Genetic_Algorithm_Tutorial/links/563214a108ae506cea68fd96/A-Genetic-Algorithm-Tutorial.pdf

[10] M. Melanie, “An Introduction to Genetic Algorithms” in MIT Press, 1998. ISBN 0-262-13316-4.

[11] I. Sekaj, M. Foltin, „Matlab Toolbox – Genetické Algoritmy“. [online]. [cit. 2019.07.20]. Dostupné na internete: http://dsp.vscht.cz/konference_matlab/matlab03/sekaj.pdf

[12] D. Tayal, C. Gupta, „A New Scale Factor for Differential Evolution Optimization“ in National Conference on Communication Technologies & its impact on Next Generation Computing CTNGC, 2012. Dostupné na internete: https://www.researchgate.net/publication/312173515_A_New_Scale_Factor_for_Differential_Evolution_Optimization

ADRESY AUTOROV

Ing. Daniel Pál, Technická Univerzita Košice, Katedra elektroenergetiky, Mäsiarska 74, Košice, SK 042 10, Slovenská Republika, daniel.pal@tuke.sk

Ing. Jakub Urbanský, Technická Univerzita Košice, Katedra elektroenergetiky, Mäsiarska 74, Košice, SK 042 10, Slovenská Republika, jakub.urbansky@tuke.sk

doc. Ing. Ľubomír Beňa, PhD., Technická Univerzita Košice, Katedra elektroenergetiky, Mäsiarska 74, Košice, SK 042 10, Slovenská Republika, lubomir.bena@tuke.sk

Ing. Michal Ivančák, Technická Univerzita Košice, Katedra elektroenergetiky, Mäsiarska 74, Košice, SK 042 10, Slovenská Republika, michal.ivancak@tuke.sk

Ing. Maksym Oliinyk, Technická Univerzita Košice, Katedra elektroenergetiky, Mäsiarska 74, Košice, SK 042 10, Slovenská Republika, maksym.oliinyk@tuke.sk