Annealing algoritmusok alkalmazása egyensúlyi töltetek optimalizációjához

Beliczai Botond Tamás

MVM Paksi Atomerőmű Zrt.
7031 Paks, Pf. 71.

A publikációban egyensúlyi töltetek különböző annealing (lehűtési) algoritmusokkal végzett optimalizációinak eredményeit ismertetem. A számításokhoz – az EGYS reaktorfizikai kód átalakításával – kifejlesztettem a szimulált lehűtés (SA) és a populációs mutációs lehűtés (PMA) algoritmusok visszafűtéssel továbbfejlesztett változatait megvalósító programokat. Az optimalizációk kiinduló tölteteit és kezdeti paramétereit egy általam fejlesztett GUI programmal – ICHTYS – adtam meg. A vizsgálatok során azt tapasztaltam, hogy az SA algoritmus csak speciális körülmények között működött megfelelően. A PMA ezzel szemben a kezdeti paraméterektől függetlenül minden esetben képes volt – egymásnak ellentmondó szempontok alapján is – optimalizálni, amiből következik, hogy ez a visszafűtéses PMA egyensúlyi töltetek optimalizációja esetén egy robusztus optimalizálási algoritmus.

Bevezetés

A Paksi Atomerőmű üzemeltetése szempontjából alapvető fontosságú az egyes reaktorok tölteteinek megfelelő megtervezése (ún. töltettervezés). Ez azt jelenti, hogy az adott reaktor zónájába a friss és a részben kiégett kazettákat olyan módon kell elhelyezni, hogy a blokk a következő kampányban az adott töltettel 100%-os teljesítményen tudjon üzemelni amellett, hogy a nukleáris biztonságot garantáló biztonsági limitek ne sérüljenek.

A töltetek tervezésekor a lehetséges zónaelrendezések csillagászati száma miatt – pl. VVER-440 reaktorok esetén kb.$1{,}3 \cdot 10^{887}$ [1, p. 210] különböző zónaelrendezés valósítható meg – a fenti feltételek figyelembe vétele mellett is lehetőségünk van ún. töltetoptimalizációra. Eszerint a kazettaelrendezések óriási fázisterében egy olyan töltetet keresünk, ami valamilyen szempontból – valamilyen célfüggvény szerint – optimális amellett, hogy bizonyos paraméterekre vonatkozó korlátozásokat (kényszereket) teljesít.

Ugyanakkor tölteteket optimalizálni komoly műszaki kihívást jelent egyrészt a fázistér hatalmas mérete, másrészt pedig a fázistér szerkezete miatt. A célfüggvény értelmezési tartománya és értékkészlete ugyanis diszkrét, emellett a fázistér nagyon sok lokális optimumot tartalmaz [1, p. 211], ami a globális vagy globálishoz közeli optimum megkeresését jelentősen megnehezíti. (Természetesen a fent említett kényszerek optimalizáció során történő figyelembe vétele sem könnyű feladat.)

Jelen cikkben azokat az eredményeket ismertetem, amelyeket VVER-440 reaktorok egyensúlyi tölteteinek annealing algoritmusokkal végzett optimalizációja során kaptam, amely algoritmusok a szakirodalom alapján hatékonyak egy a fentiekben jellemzett fázistér globálishoz közeli optimumának felkutatásában.

Egyedi és egyensúlyi töltetek átrakása

A töltetoptimalizálás során először is azt kell eldöntenünk, hogy egyedi vagy egyensúlyi töltetet szeretnénk optimalizálni. Egyedi töltet optimalizációja esetén ugyanis csak a következő kampányra vonatkozó paraméterek optimuma alapján keressük a megfelelő zónaelrendezést. Egyensúlyi töltet optimalizálásakor ugyanakkor egy több kampányra vonatkozó optimális zónaelrendezést keresünk. (Egyensúlyi töltet alatt egy olyan hipotetikus töltetet értünk, ami úgy jön létre, hogy egymást követő kampányok végén a reaktorfizikai számítások során a zóna elrendezését ugyanolyan módon változtatjuk – tehát a friss és kiégett kazettákat ugyanazokba a pozíciókba helyezzük – mindaddig, amíg az egymást követő kampányok között egy egyensúlyi állapot nem alakul ki.)

Egyensúlyi töltetek esetén az optimalizáció során a vizsgálandó tölteteket az átrakási mátrixuk egyértelműen meghatározza (feltéve, hogy a kazetták berakása kizárólag szimmetrikus kazettamozgatásokkal leírható). Az átrakási mátrix ugyanis azt mutatja meg, hogy egy adott pozícióba helyezett kazetta az előző kampányban a zónában melyik pozícióban volt, illetve friss fűtőelemköteg esetén milyen típusú kazetta került az adott pozícióba. Ennek következtében az átrakási mátrix annyi sorból áll, ahány friss kazettát helyezünk kampányonként a zóna egy szektorába; az oszlopok számát pedig az ún. üzemanyagstratégia határozza meg (hány kampányt tölt el egy kazetta a reaktorban).

Mindebből következik, hogy egyensúlyi töltetek optimalizálásakor egy adott átrakási mátrixból – referenciatöltetből – indulunk ki, és kiszámítjuk ennek a töltetnek a megfelelő paramétereit. Ezt követően kazettacseré(ke)t hajtunk végre, meghatározzuk az új átrakási mátrixot és az új reaktorfizikai paramétereket. Az általunk optimalizálni kívánt célfüggvény (pl. kampányhossz, kirakott kazetták átlagkiégése) értékének

Kontakt: beliczai@npp.hu

Beérkezett: 2022. január 7.
Közlésre elfogadva: 2022. március 21.

változása és egyéb paraméterek – pl. hogy az új töltet biztonsági paraméterei sértik-e a rájuk vonatkozó limiteket – alapján döntünk arról, hogy az új átrakási mátrixot elfogadjuk-e az új referenciatöltetnek. Ha elfogadjuk, akkor az új töltetből kiindulva folytatjuk az optimalizálást, ha pedig nem fogadjuk el, akkor az első töltetet tekintjük továbbra is referenciának.

A fentiekből is látszik, hogy a töltetoptimalizáláshoz kétféle programra és azok egymással való kommunikációjára van szükségünk. Egyrészt valamilyen módszer alapján meg kell határoznunk a kazettacseréket, illetve az átrakási mátrixot. Ezt a programot nevezzük optimalizációs kódnak. Másrészt az átrakási mátrixot át kell adnunk egy reaktorfizikai kódnak, ami az adott töltetet kiértékeli. A számítást követően a célfüggvény és az egyéb paraméterek értékét vissza kell adnunk az első programnak, ami az adatok alapján döntést hoz a következő referenciatöltetről illetve az új kazettacserékről.

A töltetátrakás optimalizálásának módszerei

Az elmúlt évtizedekben nagyon sok különböző optimalizációs eljárást alkalmaztak töltetoptimalizálásra. Ezek az eljárások alapvetően három csoportba oszthatók: a determinisztikus, a sztochasztikus és a mesterséges intelligencián alapuló algoritmusokra ([1, p. 222] alapján). A három különböző csoportba tartozó algoritmusok működéséről az [1] irodalomban olvashatunk részletesen. Jelen fejezetben a cikkben bemutatott számításokhoz használt szimulált lehűtés (SA) és populációs mutációs annealing (PMA) algoritmusokat ismertetem röviden.

A szimulált lehűtés (SA) algoritmus [2] lényege, hogy az optimalizálás során nemcsak akkor fogadunk el egy vizsgált töltetet referenciatöltetnek, ha annak célfüggvény-értéke jobb az aktuális referenciatöltet célfüggvény-értékénél, hanem valamekkora valószínűséggel akkor is, ha rosszabb. Ezt a valószínűséget a következő képlettel határozhatjuk meg:

$$p = \exp\left(-\frac{\delta f}{T}\right) \qquad (1)$$

ahol p a valószínűség, δf a célfüggvény megváltozása, T pedig az aktuális „hőmérséklet". Az algoritmus elején feltételezünk egy kezdeti hőmérsékletet – T_0 általában 1 –, amit minden egyes lépésben csökkentünk ($T_{n+1}=\alpha \cdot T_n$). Az (1) képletből látszik, hogy a kedvezőtlenebb töltet elfogadásának valószínűsége az optimalizálás során folyamatosan csökken, majd az algoritmus végére nullához közelít.

A fenti optimalizációs módszernek az a célja, hogy az optimalizáció elején az algoritmusnak nagyobb szabadságot adjunk, hogy a fázisteret jobban be tudja járni, és ne ragadjon be egy lokális optimumba. (A szimulált lehűtés algoritmusa egy szilárdtestfizikai analógián – a kristályok „lassú" hűtésén – alapul.)

Ugyanakkor töltetoptimalizálás esetén – bármilyen módszert használunk is – fontos figyelembe vennünk bizonyos biztonsági paraméterekre (egyenlőtlenségi tényező, kiégés) vonatkozó limitértékeket. Szimulált lehűtés esetén általában az ún. „bounds cooling" technikát [2] alkalmazzák. Ez azt jelenti, hogy az optimalizáció elején még nem követeljük meg a referenciatöltetektől, hogy a valódi korlátokat – az ún. „hard limiteket" – teljesítsék, hanem ún. „soft limiteket" definiálunk a következő képlettel:

$$p_{soft\ lim} = p_{hardlim} + T_i \cdot (p_{limitmax} - p_{hardlim}) \qquad (2)$$

ahol $p_{softlim}$ az adott lépésben érvényes korlát, $p_{limitmax}$ az algoritmus kezdetén érvényes korlát és $p_{hardlim}$ a valós limit. A képletből jól látszik, hogy az algoritmus végére $p_{softlim} = p_{hardlim}$.

Az elfogadott referenciatöltet paramétereinek az adott lépésben érvényes soft limiteknek kell megfelelnie. Ezeket a soft limiteket – az elfogadási valószínűséghez hasonlóan – folyamatosan hűtjük (ld. (2) képlet), így az algoritmus végén már csak a valós limiteket teljesítő töltetet fogadunk majd el referenciaként.

A PMA algoritmus a szimulált lehűtés továbbfejlesztéseként jött létre [2]. A módszer abban különbözik a szimulált lehűtéstől, hogy megjegyzi az összes korábbi lépésben generált töltetet, és azokból a töltetekből, amelyek az adott lépésben a soft limitet teljesítik, egy populációt hoz létre. A populáció egyedeihez a célfüggvény-értékeik alapján egy valószínűséget rendel (ld. 3. képlet), és a – genetikus algoritmusokban ismert – rulettkerék-módszer [3, p. 48] segítségével határozza meg a következő referenciatöltetet.

$$P_m = \frac{\exp\left\lfloor \frac{z_m - z_{max}}{T_i} \right\rfloor}{\sum_{j=1}^{pop} \exp\left\lfloor \frac{z_j - z_{max}}{T_i} \right\rfloor} \qquad (3)$$

ahol P_m, z_m az m-edik töltet kiválasztásának valószínűsége és célfüggvény-értéke, z_{max} a populációban a legnagyobb célfüggvény-érték, T_i az i. lépés hőmérséklete, pop pedig a populációban levő töltetek száma. (Az egyes töltetek valószínűségeinek meghatározásakor lehetőség van ún. büntetőfüggvények figyelembe vételére is attól függően, hogy az adott töltet a hard limitet teljesíti-e, ld. [2].)

Az EGYS program átalakítása töltetoptimalizációkhoz

Az EGYS programot a Paksi Atomerőműben egyensúlyi töltetek tervezésére fejlesztették ki.

A program az inputban megadott átrakási mátrix alapján – a C-PORCA 2.0 kód segítségével – meghatározza az egyensúlyi töltet egyes reaktorfizikai paramétereit (kampányhossz, kazetták kiégése stb.). A C-PORCA 2.0 program kazettaszintű számításokat végez – amely számításokhoz a HELIOS [4] programmal számított homogenizált hatáskeresztmetszeteket használja –, tehát a kazettán belüli heterogenitásokat nem veszi figyelembe. (A töltettervezés során a részletes számításokhoz manapság használt C-PORCA 7.4 [4] mind a kazettaszintű – a C-PORCA 2.0 kódban alkalmazott algoritmusok segítségével, –, mind a pálcaszintű számításokat elvégzi. A C-PORCA 7.4 verzióval sztochasztikus optimalizálási algoritmusokat a jóval nagyobb gépidőigény miatt csak sokkal körülményesebben lehetne vizsgálni.)

Az EGYS program forráskódját alapul véve valósítottam meg az SA és PMA algoritmusok alapján működő töltetoptimalizációs programokat (sa.exe és pma.exe).

A fejlesztést a Code::Blocks 20.03 fejlesztőkörnyezettel, GNU FORTRAN fordítót használva hajtottam végre.

A programok inputjában a következő opciók adhatók meg: optimalizálandó célfüggvény (kampányhossz, kirakott kazetták átlagkiégése, kq-maximum minimalizálása); maximális lépésszám (az algoritmus alapértelmezés szerint egy adott hőmérséklet elérésekor leáll, a visszafűtések (lásd később) miatt ugyanakkor nem biztos, hogy ezt a hőmérsékletet eléri); α paraméter; a kq-ra és a kiégésre vonatkozó „limitmax" és „hardlim" paraméterek (az optimalizáció során demonstrációs céllal csak ezt a két korlátozó biztonsági paramétert vettem figyelembe); az optimalizáció során nem mozgatható – fixált – kazetták száma és pozíciói.

Az sa.exe és a pma.exe is az első lépésben az EGYS program inputjában megadott átrakási mátrixból indul ki, azt kiértékeli, majd kazettacseréket hajt végre, és így hozza létre a továbbiakban vizsgálandó tölteteket. A kazettacseréket a program a következőképpen valósítja meg.

Minden lépésben az algoritmus meghatározza, hogy két vagy három üzemi kazetta cserélődjön-e a hőmérséklet csökkenésével arányosan változó valószínűségek alapján, majd a cserélhető kazettákból egyenletes eloszlás alapján választja ki a cserélendő fűtőelemkötegeket. Mivel az SZBV-kazetták számossága jóval kisebb, mint az üzemi kazettáké, ezért minden lépésben történő cseréjük szükségtelen. Az optimalizálás elején 20% annak a valószínűsége, hogy SZBV-kazetták is cserélődnek, majd ez a valószínűség az algoritmus végére a hőmérsékletcsökkenés sebességével nullához közelít. A cserélendő SZBV-kazetták is egyenletes eloszlás alapján kerülnek kiválasztásra.

Mindkét programba beépítettem azt is, hogyha az optimalizálás valamilyen oknál fogva megreked, akkor automatikusan fűtsenek vissza. Az sa.exe és a pma.exe esetén ennek az opciónak a megvalósítása bizonyos mértékben különbözik egymástól.

Szimulált lehűtés esetén onnan tudhatjuk, hogy megrekedt az algoritmus, hogy egy adott lépésben a referenciatöltet kq- vagy kiégésértéke jóval nagyobb, mint az aktuális soft limit. A visszafűtést ezért az algoritmus úgy valósítja meg, hogy figyeli a referenciatöltet kiválasztásakor és az adott lépésben érvényben levő soft limitek közötti különbséget, és ha ez egy adott értéknél nagyobb (kq esetén 0,12, kiégés esetén 1,2 MWn/kgU), akkor a hőmérsékletet a referenciatöltet referenciává válása előtti hőmérsékletre emeli vissza.

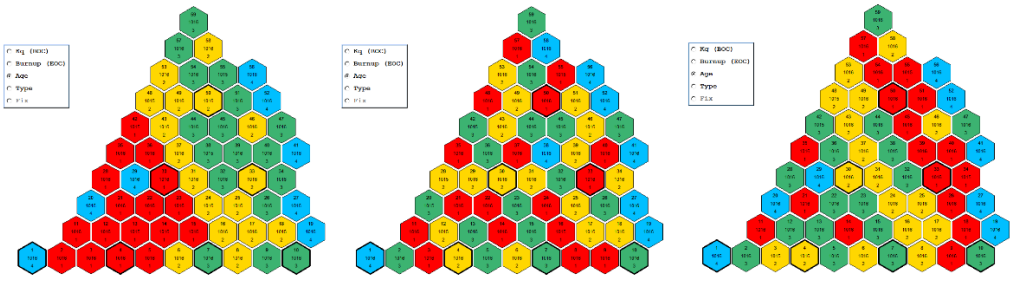
A PMA esetén onnan tudhatjuk, hogy megrekedt az algoritmus, hogy a populációból elfogynak a töltetek. A pma.exe ezért figyeli azt a lépést és hőmérsékletet, amikor a populációban levő zónaelrendezések száma öt alá csökken, és megjegyzi az ezt megelőző lépésben a populációban levő tölteteket (számosságuk minimum öt). Ha ezt követően kiürül a pool, akkor a program egyrészt visszatölti a populációba a megjegyzett zónaelrendezéseket, másrészt pedig a hőmérséklet értékét úgy növeli, hogy az nagyobb legyen a fent megfigyelt hőmérsékletnél.

Optimalizálás előkészítése

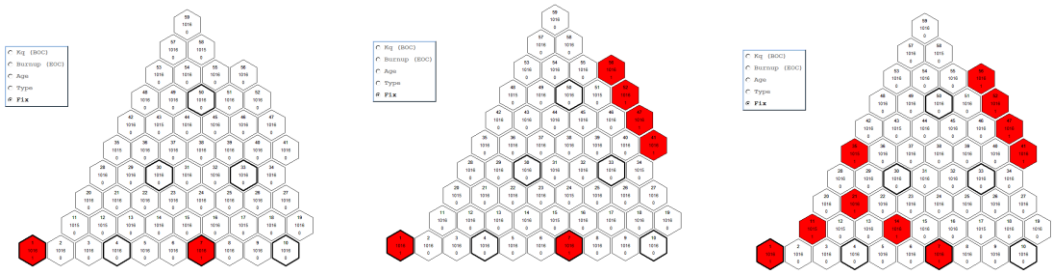
Az optimalizációk során arra voltam kíváncsi, hogy az annealing algoritmusok mennyire hatékonyak optimális töltetek megtalálásában különböző feltételek – kiinduló töltet, fixált kazetták, célfüggvények – mellett. Az SA és a PMA algoritmusokhoz a kiinduló töltetet és az egyéb paramétereket az ICHTYS programmal határoztam meg. (Az ICHTYS egy általam fejlesztett grafikus program, aminek a segítségével egyensúlyi tölteteket lehet vizsgálni, és a töltetoptimalizációk paraméterei könnyen megadhatók.)

A vizsgálatok során egy olyan töltet egyensúlyi kampányait optimalizáltam, amibe kampányonként 102 db friss, Gd-ot tartalmazó kazetta került. A 102-ből 90 db 4,7%-os, míg 12 db 4,2%-os dúsítású kazetta volt.

Rendelkezésemre állt egy kiinduló átrakási mátrix – több évvel ezelőtt végzett számítások alapján –, amit az egyik kiinduló töltetnek választottam. Ezt követően ezt a töltetet mesterségesen elrontottam: létrehoztam egy nagyon előnytelen – hihetetlen nagy kezdeti kqmax-értékű – és egy közepesen jó kiinduló elrendezést (1. ábra).



1. ábra: A „rossz", a „közepes" és a „jó" kiinduló töltet



2. ábra: Kettő, hat és tíz kazetta fixálása

A kazettafixálás hatását is vizsgáltam az optimalizációk hatékonyságával kapcsolatban. Három opciót vettem figyelembe: kevés, közepes mennyiségű és sok kazetta pozíciójának rögzítése az algoritmus kezdetén (2. ábra).

A fentiekben említett célfüggvények mindegyikére végeztem töltetoptimalizációkat (kampányhossz növelése, kqmax csökkentése, kirakott kazetták átlagkiégésének maximalizálása).

A fenti paraméterek többféle variációja alapján végül 18 különböző optimalizációt hajtottam végre ugyanazokat az inputokat figyelembe véve mindkét algoritmussal. Minden számítást az algoritmusok sztochasztikussága miatt háromszor indítottam el, és azokból kiválasztottam a legjobbakat. Ez azt jelenti, hogy összesen 108 futtatást kellett végrehajtanom.

Az inputparamétereket a következőképpen adtam meg.

A kq és kiégés limitmax értékeket a kezdeti töltet alapján határoztam meg – rosszabb zónaelrendezésnél nagyobb kq és kiégés limitmaxra van szükség –, viszont a kq és a kiégés hard limitet 1,25-nek illetve 57 MWn/kgU-nak definiáltam (a valóságban alkalmazott 1,351 és 58,01 MWn/kgU helyett). Ezáltal azt értem el, hogy az algoritmus az 1,351-es és 58,01 MWn/kgU-s limit elérése után is továbbfutott, és próbált optimális tölteteket felkutatni.

Az alfa-paramétert a [3] irodalom alapján 0,995-nek vettem, a maximális lépésszámot pedig 5000-nek. (Ha nem volt visszafűtés, akkor a futtatás kb. 2000 lépésig tartott.)

Eredmények

Az SA optimalizációk eredményei

Az SA-algoritmussal a különböző célfüggvényekre vonatkozó optimalizációk során kapott néhány eredményt az 1. táblázat tartalmazza.

A táblázatban feltüntettem a célfüggvényeket, az optimalizáció kezdetén (kék színnel) és végén (fekete színnel) a referenciatöltet különböző paramétereit (kampányhossz, maximális kq, maximális kiégés) illetve az optimalizáció során a visszafűtések számát különböző kezdeti feltételek mellett (rossz, közepes és jó töltet; sok, „közepes” és kevés fix kazetta).

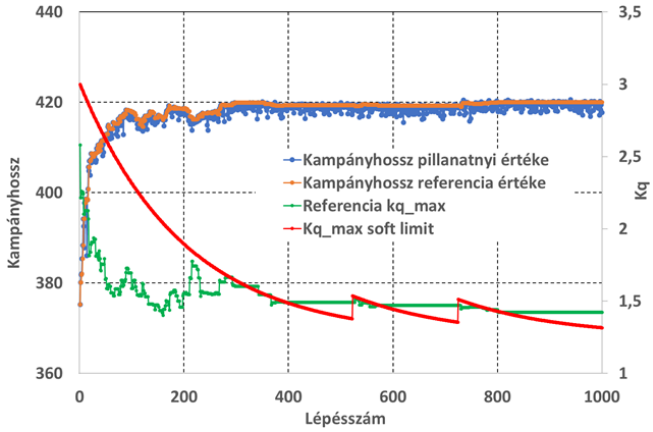
A táblázatból látható, hogy az algoritmus a célfüggvénytől és a fixált kazetták számától függetlenül minden esetben tudta javítani a célfüggvény értékét (sőt, sokszor jelentős mértékben). Az is jól kivehető, hogy az egyes célfüggvények esetén a különböző kiinduló feltételektől függetlenül mindig közel azonos célfüggvény-értékű optimális megoldásokat talált. A fentiek mellett ugyanakkor az is megállapítható, hogy ha a kampányhossz növelésére optimalizáltam, akkor az SA nem tudta a maximális kq-értékét a limit (1,351) alá csökkenteni. (Csak abban az esetben talált a limiteknek megfelelő optimumot, ha kezdetben a jó töltetből indultam ki.)

Az 1. táblázat azt is mutatja, hogy a maximális kq minimalizálásakor a célfüggvény javulásával együtt a referenciatöltet kampányhossza is lényegesen – mintegy 10 nappal – csökkent. A kampányhossznöveléssel kapcsolatos optimalizáció eredményei pedig arra utalnak, hogy minél nagyobb egy töltet kampányhossza, annál nagyobb a maximális kq-értéke a zónában. (Az 1. táblázat bal oldalának utolsó sora ugyanakkor arra utal, hogy ez nem minden esetben van így.) A fentiekből következik, hogy a kqmax csökkentése és a kampányhossz növelése egymásnak ellentmondó szempontok. Ez azért van így, mert az egyenlőtlenség csökkentésekor az algoritmus minél több kis reaktivitású, többéves kazettát szeretne a zóna közepébe helyezni, aminek viszont az a következménye, hogy a beépített reaktivitástartalék – és ezáltal a kampányhossz – csökken. Kampányhossz növelésekor pedig az SA sok friss kazettát igyekszik a reaktor belső pozícióiba tenni, ami viszont a maximális kq növekedésével jár együtt.

Az SA-algoritmus működésének fent említett tulajdonságát támasztja alá az egyik optimalizáció adatai alapján készített 3. ábra. Ennél a számításnál a rossz töltetből kiindulva, sok kazettát fixálva kampányhosszra optimalizáltam (lásd 1. táblázat, bal oldal, 2. sor). A 3. ábrán egyrészt a referencia- és a pillanatnyi célfüggvényértékeket, másrészt a kq soft limitet és a referencia kqmax-ot láthatjuk a lépésszám függvényében. (Csak az első 1000 lépést ábrázoltam. A kiégésre vonatkozó soft limitet az egyes lépésekben a referenciatöltetek teljesítették, ezért csak a kqmax-ra vonatkozó korlátozás jelenik meg a grafikonon.)

1. táblázat: Az SA eredményei a kampányhossz és a kq optimalizálása esetén

	Kampányhossz				Teljesítmény-egyenlőtlenségi tényező (Kq)			
	Ref. Teff	**Ref. Kqm.**	**Ref. Kiegm.**	**Vfut.**	**Ref. Kqm.**	**Ref. Teff**	**Ref. Kiegm.**	**Vfut.**
Kezdet_rosszt	**375,24**	**2,58**	**60,58**		**2,58**	**375,24**	**60,58**	
Rosszt_sokf	420,63	1,45	55,68	20	1,28	408,87	57,97	7
Rosszt_közf	419,86	1,43	54,98	5	1,26	412,42	57,38	1
Rosszt_kevf	421,55	1,47	55,45	18	1,26	406,77	57,01	2
Kezdet_közt	**411,40**	**1,79**	**57,60**		**1,79**	**411,40**	**57,60**	
Közt_közf	417,94	1,39	56,78	32	1,30	407,72	57,57	0
Közt_kevf	419,07	1,38	56,73	33	1,27	408,01	57,68	1
Kezdet_jót	**414,53**	**1,31**	**59,23**		**1,31**	**414,53**	**59,23**	
Jót_közf	418,80	1,35	57,98	14	1,26	410,57	57,54	0



3. ábra: A kampányhossz- és kqmax-értékek a lépésszám függvényében (első 1000 lépés)

A 3. ábrán jól látszik, hogy az algoritmus az egyes lépésekben a legjobb tölteteket próbálja megkeresni, hiszen a célfüggvény pillanatnyi értéke az optimalizáció során csak ritkán nagyobb a referenciaértéknél (kék és narancssárga görbe). A grafikon azt is mutatja, hogy az optimalizáció elején – kb. az első 300 lépésben – az SA az előző referenciánál kisebb értékű referenciákat is elfogad új referenciának, viszont az algoritmus végén a referencia negatív irányba már nem változik. Tulajdonképpen ez az algoritmus lényege.

A 3. ábrán a visszafűtések is jól láthatók: amikor a referencia kqmax és a kq soft limit közötti különbség egy adott értéket meghalad, akkor a rendszer visszafűt, tehát a soft limit értéke egy adott lépésben megnő. (A visszafűtések miatt változik fűrészfogszerűen a piros görbe.) A visszafűtést követően a zöld görbe értékében változásokat láthatunk, ami azt jelenti, hogy változnak a referenciatöltetek. Amikor azonban a soft limit újra a kq referenciaértéke alá csökken, akkor az algoritmus megreked, mert nem talál új referenciát. (A zöld vonal, amikor a piros vonal felett fut, nem változik.) Ennek egyrészt az a magyarázata, hogy az algoritmus kampányhosszra optimalizál, és a nagyobb kqmax-ú tölteteknek általában nagyobb a kampányhossza is (lásd fent). (Ennél a hőmérsékletnél kisebb célfüggvényértékű tölteteket már nagyon kis valószínűséggel fogad el referenciatöltetként.) Másrészt pedig az algoritmus mindig csak a referenciatöltet környezetét vizsgálja – lokálisan vizsgálódik –, hiszen csak egy referenciatöltetet jegyez meg, illetve ezen a hőmérsékleten nagy valószínűség szerint már csak páros cseréket hajt végre.

Ez azt jelenti, hogy az SA a referencia közelében valószínűleg csak olyan, a referenciánál nagyobb kampányhosszal rendelkező tölteteket talál, amelyek kqmax-ja nagyobb a soft limitnél. Tehát hiába van a referencia közelében egy olyan zónaelrendezés, aminek a célfüggvénye jobb a referenciánál, azt az algoritmus nem fogadhatja el a limitsértés miatt, és emiatt megreked. Mindez magyarázatot jelent az SA azon tulajdonságára, hogy egymásnak ellentmondó szempontok esetén csak speciális esetben tud optimalizálni.

A PMA optimalizációk eredményei

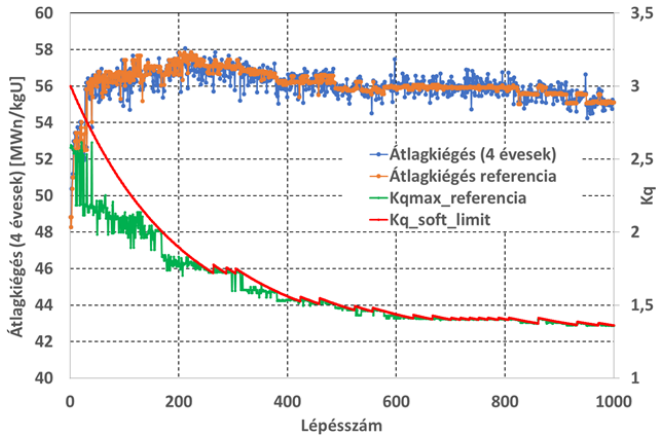
A PMA-algoritmussal a különböző célfüggvényekre vonatkozó optimalizációk során kapott néhány eredményt a 2. táblázat tartalmazza.

A táblázatból jól látható, hogy a PMA-optimalizációk során is minden esetben kedvezően változott a célfüggvény értéke. (Ebben az esetben is jelentős célfüggvény-javulásokat láthatunk a táblázat megfelelő rubrikáiban.) Az SA-val szemben ugyanakkor az algoritmus képes volt olyan optimális zónaelrendezések felkutatására, amelyek teljesítik a limiteket. Az is megállapítható, hogy a PMA ezen képessége nem függött a kezdeti töltet milyenségétől, a fixált kazetták számától vagy a célfüggvénytől, minden vizsgált esetben talált olyan töltetet, ami az előzetes számítások alapján a zónába helyezhető (kqmax<1,351; kiégésmax<58,01 MWn/kgU).

A fenti táblázatból az is jól látható, hogy a PMA-algoritmus esetében is nagyon fontos szerepe van az általam beépített visszafűtésnek. (Mivel ennél az algoritmusnál másképp működik a visszafűtés, mint az SA esetén, ezért a visszafűtések száma közvetlenül nem összehasonlítható.) Sok esetben az algoritmus több tízszer is visszafűt az optimalizáció során, tehát ezen opció nélkül a PMA a pool kiüresedése miatt megállt volna.

2. táblázat: A PMA eredményei a kampányhossz és a kirakott kazetták kiégésének optimalizálásakor

	Kampányhossz				**Kirakott (négyéves) kazetták átlagkiégése**				
	Ref. Teff	**Ref. Kqm.**	**Ref. Kiegm.**	**Vfut.**	**Ref. Kieg4ev**	**Ref. Teff**	**Ref.Kqm**	**Ref. Kiegm.**	**Vfut.**
Kezdet_rosszt	**375,24**	**2,58**	**60,58**		**48,81**	**375,24**	**2,58**	**60,58**	
Rosszt_sokf	420,41	1,35	54,36	0	55,86	410,54	1,35	58,00	99
Rosszt_közf	420,40	1,35	54,58	12	56,33	410,82	1,35	57,91	41
Rosszt_kevf	420,27	1,35	55,47	19	57,26	402,67	1,35	57,99	7
Kezdet_közt	**411,40**	**1,79**	**57,60**		**51,37**	**411,40**	**1,79**	**57,60**	
Közt_közf	419,44	1,35	54,67	10	55,82	409,92	1,31	57,97	73
Közt_kevf	419,25	1,35	54,76	98	57,35	403,66	1,33	58,00	0
Kezdet_jót	**414,53**	**1,31**	**59,23**		**54,13**	**414,53**	**1,31**	**59,23**	
Jót_közf	420,18	1,35	54,00	6	55,65	414,31	1,35	57,91	2



4. ábra: Átlagkiégés-maximalizálás során a paraméterek változása (első 1000 lépés)

A PMA-algoritmusnak az az előnye az SA-val szemben, hogy az összes korábbi töltetet megjegyzi, és a limiteket teljesítő zónaelrendezésekből sorsolja ki a következő kiinduló töltetet. Ez azt jelenti, hogy nemcsak lokálisan – ahogy a szimulált lehűtés –, hanem globálisan is feltérképezi a fázisteret. Ennek következtében az optimalizáció során a folyamatosan változó körülményekhez – kq soft limit, kiégés soft limit – képes alkalmazkodni. Erre mutat példát a 4. ábra, ahol a négyéves kazetták optimalizációjának több paraméterét ábrázoltam a lépésszám függvényében (2. táblázat, jobb oldal, 2. sor).

Az ábra felső részén a referencia- illetve az adott lépésben számított töltet négy éves kazettákra vonatkozó átlagkiégésének értékét láthatjuk. Az adatokból jól kivehető, hogy az algoritmus – hasonlóan az SA-hoz – a legjobb célfüggvényértékkel rendelkező töltetet igyekszik referenciaként kiválasztani. Az is látszik a grafikonon, hogy a PMA első 200 lépése alatt a célfüggvény értéke 48 MWnap/kgU-ról 58 MWnap/kgU-ra nő, viszont az ezt követő 800 lépés alatt ez az érték 55 MWnap/kgU körüli értékre csökken (az optimalizáció végén a referencia-kiégésérték 55,86 MWnap/kgU). Ez a tény is azt mutatja, hogy a PMA egy adaptív algoritmus: hiába a maximalizálás a fő cél, az algoritmus képes figyelembe venni azt, hogy ahhoz, hogy a limiteknek megfelelő töltetet találjon, a referencia célfüggvény-értékét csökkentenie kell (tehát ki tud mozdulni a lokális optimumokból).

Ugyanakkor a 4. ábra alsó része azt is mutatja, hogy ezt a csökkentést a PMA csak olyan mértékben hajtja végre, ami feltétlenül szükséges. A referencia kqmax-értékek ugyanis kb. a 600. lépéstől éppen hogy csak kisebbek a soft limit értékeknél – az ábrán a piros és a zöld vonal nem is nagyon válik el egymástól –, ami azt jelenti, hogy túlságosan alacsony kqmax értékkel rendelkező töltetet – aminek a célfüggvény-értéke túl kicsi lenne – az algoritmus nem választ referenciatöltetnek.

A fentiekből következik, hogy a visszafűtéses PMA egy olyan robusztus algoritmus, ami egymással ellentétes szempontokat is képes figyelembe venni az optimalizálás során, tehát úgy növeli a célfüggvény értékét, hogy közben a biztonsági paraméterekre vonatkozó limiteket is betartja.

Összefoglalás

A cikkben azt vizsgáltam, hogy annealing algoritmusok – SA, PMA – milyen feltételekkel alkalmazhatók egyensúlyi töltetek optimalizációjára.

A szimulált lehűtéssel történő vizsgálatok során azt tapasztaltam, hogy az algoritmus akkor működött jól, ha egy optimálishoz közeli töltetelrendezésből indultam ki vagy pedig olyan szempont szerint optimalizáltam, ami nem mond ellent annak, hogy a korlátozó paramétereket be kell tartani.

A PMA esetén viszont azt tapasztaltam, hogy bármilyen kiinduló állapot és bármilyen célfüggvényre történő optimalizáció esetén talált az algoritmus olyan optimális célfüggvény-értékű tölteteket, amelyek a vonatkozó limitértékeket teljesítik. A PMA működését az általam bevezetett visszafűtés nagyban segítette, hiszen ezen opció nélkül az algoritmus populációja jó néhányszor kiürült volna, és az optimalizáció megállt volna. A fentiek alapján megállapítható, hogy a visszafűtéses PMA a töltetoptimalizáció szempontjából egy robusztus algoritmus. A későbbiekben további kutatás tárgyát képezheti, hogy a PMA algoritmus vajon megőrzi-e ezt a robusztusságát akkor is, ha az optimalizáció során nemcsak kettő, hanem több biztonsági paramétert veszünk figyelembe.

Köszönetnyilvánítás

Jelen cikk az Ipar 4.0 megoldásokat fejlesztő adat- és rendszertudományi szakmérnöki képzés keretében írt szakdolgozat néhány eredményét ismerteti. Szeretném megköszönni prof. dr. Abonyi Jánosnak a fenti szakdolgozat kidolgozásához és megírásához nyújtott sokrétű segítségét.

Irodalomjegyzék

[1] Csom Gyula et al., Atomerőművek üzemtana II/3, Budapest, 2012.

[2] R. v. Geemert, *Reload Pattern Optimization by Application of Heuristic Search and Perturbation Theoretical Methods*, Delft, The Netherlands, 1999.

[3] Várkonyiné Kóczy Annamária et al., Genetikus algoritmusok, Budapest: Typotex Kiadó, 2002.

[4] I. Pos, T. Parko, Z. Kalya, S. P. Szabo, M. Horvath, „C-PORCA 7: a nodal diffusion reactor calculation code to support off-line and on-line core analysis at Paks nuclear power plant," *Kerntechnik 84(4)*, pp. 228-241, 2019.

