Laborator 7

Optimizarea multiobiectiv

1. Obiectivele laboratorului

- Recapitularea conceptelor legate de optimizarea multiobiectiv: frontul Pareto
- Însușirea unei metode de optimizare multiobiectiv a unei funcții matematice
- Analiza frontului Pareto și condiția de identificare a frontului
- Însușirea unei metode de stabilire a parametrilor unui sistem de control multiobiectiv, în varianta fuzzy

2. Recapitularea conceptelor legate de optimizarea multiobiectiv

Problemele de optimizare vizează uneori mai multe obiective care sunt în conflict, adică ridicarea performanței aferente unui obiectiv poate duce la scăderea performanței altor obiective (de exemplu: suprareglaj ↔ perioadă de stabilizare). În aceste cazuri se alege o variantă de compromis, iar soluția finală va satisface toate obiectivele parțial.

O soluție posibilă din spațiul de căutare se numește:

- *Pareto-dominată* dacă există o altă soluție posibilă care îmbunătățește unele obiective și nu înrăutățește nici un alt obiectiv.
- *Pareto-optimală* dacă nici un obiectiv nu poate fi îmbunătățit fără a înrăutăți alte obiective.

Frontul Pareto este zona formată din toate soluțiile Pareto-optimale și delimitează spațiul de căutare acolo unde un obiectiv nu poate fi îmbunătățit fără a afecta negativ un alt obiectiv. Frontul Pareto este o linie (dacă există 2 obiective), o suprafață (3 obiective), sau o hiper-suprafață. Toate soluțiile Pareto-dominate se află de aceeași parte a frontului Pareto.

Se consideră că toate soluțiile de pe frontul Pareto sunt în mod egal performante. Alegerea finală a unei soluții implică preferința subiectivă a utilizatorului. De exemplu, se poate opta pentru acea soluție care maximizează (sau minimizează) suma obiectivelor.

La optimizarea multiobiectiv, funcția de performanță este multidimensională cu dimensiunea n: $J : S \rightarrow R^n$, unde $R^n = R \times R \times ... \times R$ este produsul cartezian al mulțimii reale R de n ori. Putem considera că este vorba, practic, de n funcții obiectiv.

Selecția cromozomilor trebuie să păstreze varietatea populației cu privire la îndeplinirea tuturor obiectivelor [Konak et al, 2006], deci este necesară implementarea unui operator de selecție specializat:

- folosirea succesivă a operatorului clasic de selecție, pentru fiecare obiectiv, pentru a genera subpopulații. Generația următoare este compusă din toate aceste subpopulații.
- selecția folosind un indice de performanță care este o sumă ponderată a obiectivelor,

- înlocuirea progresivă a fiecărei soluții Pareto-dominate cu una dintre soluțiile Pareto-optimale care o domină.

În figura de mai jos, există 2 obiective f_1 și f_2 care trebuie minimizate simultan. Pentru fiecare soluție Pareto-dominată există cel puțin o soluție Pareto-optimală care îmbunătățește cel puțin un obiectiv. În practică, se evaluează mai întâi frontul Pareto, iar apoi se alege în mod subiectiv (empiric) una dintre soluțiile respective.

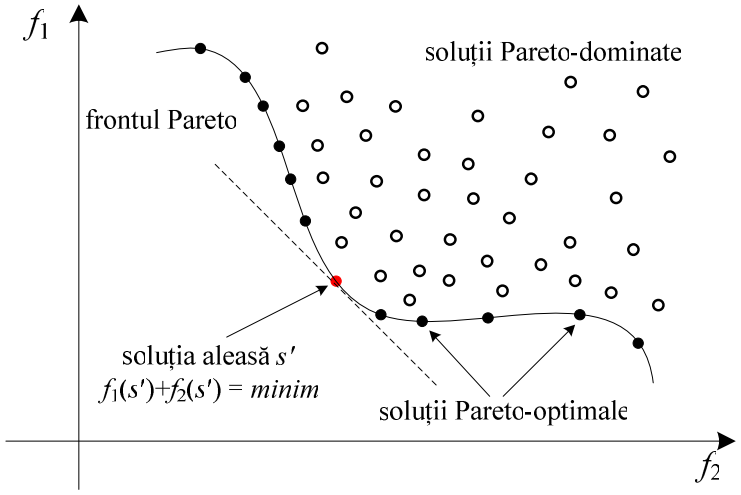

Figura 20. Optimizare multiobiectiv. Frontul Pareto

3. Studiu de caz I: optimizarea multiobiectiv a unei funcții matematice

Problema generală este aceea de a găsi valoarea numerică pentru argumentele x_1, x_2, ..., x_n astfel încât funcțiile obiectiv f_1, f_2, ... f_m: $R^n \rightarrow R$ să fie minimizate (sau maximizate) simultan. Domeniul de definiție al funcțiilor poate fi constrâns, dar este necesar să fie identic pentru toate funcțiile. O notație echivalentă este să considerăm o singură funcție obiectiv $f:R^n \rightarrow R^m$, cu m valori de ieșire. Se cere identificarea frontului Pareto și apoi selectarea unei soluții care satisface o condiție dată.

În exemplul următor, funcțiile obiectiv au o singură variabilă, iar selecția soluției de pe frontul Pareto se va face maximizând o combinație liniară a funcțiilor obiectiv.

3.1. Formularea problemei

Se dă o funcție matematică având o singură variabilă ca intrare notată cu x și două valori de ieșire, notate f_1 si f_2. Funcția f este deci un tuplu de 2 funcții obiectiv:

$f : [-\pi/2, \pi] \rightarrow R \times R$, $f(x) = (f_1 ; f_2) = (\sin(x) ; \cos(x))$

Se cere să se afle valoarea lui x pentru care f_1 și f_2 au valori cât mai mari.

3.2. Rezolvarea problemei

Se vede imediat că maximul pentru funcțiile f_1 și f_2 se atinge pentru $x_1 = 0$, respectiv $x_2 = \pi/2$, prin urmare f_1 și f_2 nu pot fi maximizate simultan cu aceeași valoare a lui x. Spunem deci că cele două obiective sunt în conflict.

Pentru vizualizarea problemei, se dă reprezentarea grafică a celor două funcții mai jos. Se observă că în afara intervalului [0, $\pi/2$] ambele obiective pot fi îmbunătățite

simultan, deci ne așteptăm ca soluția optimă *x* să se găsească în interiorul acestui interval, iar frontul Pareto să fie format din evaluarea lui f_1 și f_2 pentru puncte x, unde $x \in [0, \pi/2]$.

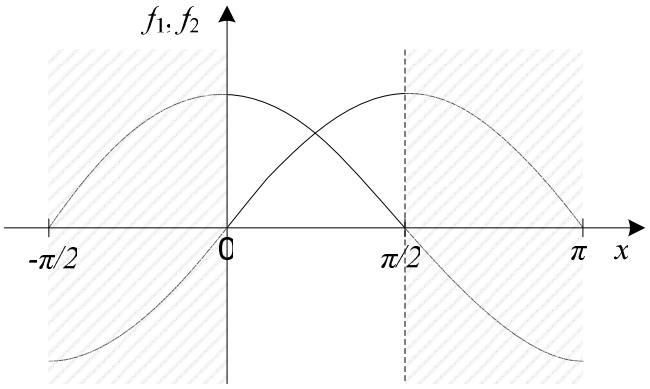

Figura 21. Funcțiile obiectiv pentru studiul de caz I

Pentru a reprezenta grafic frontul Pareto recurgem la expresia lui f_2 în funcție de f_1, și anume: $f_2(f_1)= \cos(\arcsin(f_1))$. Reprezentarea grafică este în figura următoare. În mod normal, la problemele unde rezolvarea nu este directă, frontul Pareto este determinat prin rularea algoritmului genetic până când nu mai apar îmbunătățiri semnificative. Populația din ultima generație conține indivizi care se află, preponderent, pe frontul Pareto.

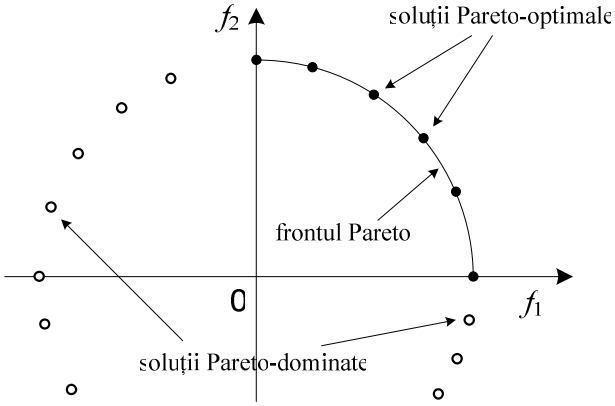

Figura 22. Frontul Pareto pentru studiul de caz I

Pentru alegerea unei soluții dintre cele Pareto-optimale, vom recurge de această dată la suma ponderată a celor două obiective: vom alege *x* pentru care $f = 2 \cdot f_1(x)+3 \cdot f_2(x)$ este maximă. În general, alegerea soluției finale de pe frontul Pareto se poate face fie după reprezentarea lor grafică (alegere subiectivă), fie utilizând un algoritm genetic simplu, pentru care spațiul de căutare va fi compus din soluțiile Pareto-optimale.

3.3. Implementare

Modul de implementare nu diferă substanțial față de algoritmul genetic standard. Printre exemplele descărcare odată cu pachetul *jgap*, în directorul *multiobjective* există un

exemplu *MultiObjectiveExample*, care oferă această implementare și pe care nu îl mai reproducem aici.

În pachetul *jgap* există niște clase cu metode definite pentru rezolvarea problemelor de optimizare multiobiectiv:

- `BulkFitnessFunction`, cu metoda `void evaluate(Population a_subject)`, cu ajutorul căreia se evaluează toate obiectivele pentru întreaga populație. Performanța de îndeplinire a fiecărui obiectiv se setează în obiectul de tip `Chromosome`, cu metoda `setMultiObjectives(List a_values)`. La sfârșitul evoluției programului, populația finală va conține indivizi de pe frontul Pareto.
- Selectorul `BestCromosomesSelector` este cel folosit pentru păstrarea în populație a indivizilor performanți la fiecare dintre obiective.
- `FitnessEvaluator`, cu metoda `isFitter(…)`, folosită pentru alegerea soluției finale de pe frontul Pareto determinat, prin comparația cromozomilor relativ la performanțele de îndeplinire ale fiecărui obiectiv. Semnătura acestei metode este: `boolean isFitter(IChromosome a_chrom1, IChromosome a_chrom2)`

3.4. Testare și concluzii

Frontul Pareto

Dacă se rulează programul pentru 20 de generații, în populația finală se vor regăsi valorile lui x care fac parte din frontul Pareto. Încercarea de a adăuga pe o hartă carteziană puncte cu coordonatele $(f_1(x), f_2(x))$ va rezulta într-un grafic cum este cel din figura 24. Alegerea soluției preferate se poate face acum în mai multe moduri:

- subiectiv, direct de pe grafic.
- *brute force*, calculând valoarea $f = 2 \cdot f_1(x) + 3 \cdot f_2(x)$ pentru fiecare punct de pe frontul Pareto. O alternativă la brute force este căutarea binară, după ordonarea soluțiilor.
- dacă frontul Pareto conține prea multe soluții, se poate folosi un algoritm genetic simplu în loc de abordarea *brute force*, cu funcția de performanță f dată.

Convergența

Dacă lăsăm algoritmul să ruleze mai departe, observăm că, după ce populația s-a stabilizat în jurul frontului Pareto, nu există o convergență standard în sensul în care cea mai bună soluție să nu se mai modifice. Populația conține indivizi de pe frontul Pareto și va baleia în continuare zona respectivă, găsind cu fiecare generație alte soluții diferite de pe frontul Pareto. Nu se poate vorbi despre noțiunea de convergență standard pentru că nu există o soluție finală care să fie cea mai bună.

Identificarea momentului când algoritmul ajunge la frontul Pareto se poate face pe baza definiției Pareto-optimalității: nici o soluție găsită nu mai poate fi îmbunătățită în ceea ce privește *toate* obiectivele. Este necesar deci să verificăm dacă există vreun individ din generația anterioară față de care un individ din generația actuală îmbunătățește toate obiectivele.

4. Studiu de caz II: generarea unui sistem de control multiobiectiv

4.1. Formularea problemei

Se dă un sistem format dintr-un lac de acumulare cu două turbine generatoare de energie electrică, G_1 si G_2, controlate prin debitele s1 și s2. Acestea au expresiile: $s_1=\alpha \cdot c_1$, $s_2=\alpha \cdot c_2$, unde c_1 și c_2 sunt mărimile de control, iar α este constantă. Aportul de apă în lac, *u*, se consideră măsurabil.

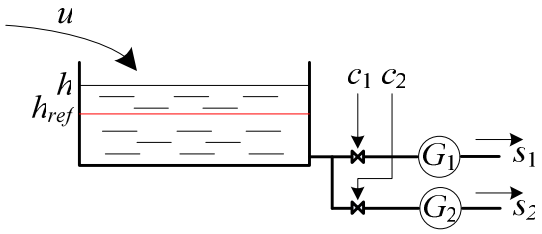

Figura 23. Lac de acumulare cu două generatoare

Sistemul are următorul model discret pentru h – înălțimea apei și P_i – puterea generată de generatorul i:

$$h(k+1) = h(k) + A \cdot \left(u(k) - s_1(k) - s_2(k)\right)$$

$$P_i(k) = K_i \cdot h(k) \cdot s_i(k)$$

, unde A și K_i sunt constante (aria bazinului, respectiv o constantă a generatorului) [Leția et al, 2013]. Se consideră că generatoarele sunt diferite ca și specificații ($K_1 > K_2$).

Se cere să se calculeze parametrii unui sistem de control fuzzy pentru care nivelul apei se menține într-un interval în jurul referinței h_{ref} și energia totală generată este maximă.

4.2. Rezolvarea problemei

Pentru înălțimea coloanei de apă din lac, vom considera nivelele fuzzy descrise în laboratorul 4: *NL*, *NM*, *ZR*, *PM*, *PL*, cu mențiunea că *ZR* corespunde lui h_{ref}. Aceleași valori fuzzy vor fi folosite pentru debitul de intrare u și pentru comenzile c_1 și c_2. Evident, fiecare dintre aceste mărimi poate folosi funcții de apartenență diferite.

Regulile de control fuzzy sunt de forma dată mai jos. Sunt necesare 25 de reguli de control, deci soluția problemei va fi un tabel de dimensiune 5×5 cu perechi de reguli (c_1, c_2).

IF (u IS U AND h IS H) THEN (c_1 IS C_1 AND c_2 IS C_2)

Cromozomul codifică tabelul de reguli fuzzy, deci conține 25 de gene cu valori întregi în intervalul [1, 5], așa cum s-a folosit în laboratorul 4.

Performanța sistemului vizează două obiective care sunt în conflict: minimizarea erorii de nivel $\Delta h = |h_{ref} - h|$ și maximizarea energiei totale generate, ambele pe un orizont de optimizare T. Prin urmare, funcțiile obiectiv au expresiile:

$J_{\Delta h} = \sum_{k=0}^{T} \left| h_{ref}(k) - h(k) \right|$, unde $J_{\Delta h}$ (suma erorilor nivelului) trebuie minimizată

$J_e = E_{tot} = \sum_{k=0}^{T} \left(P_1(k) + P_2(k)\right) \cdot \Delta t_k$, unde J_e (energia totală generată) trebuie maximizată. În acest caz, Δt_k =1 reprezintă perioada de eșantionare.

Pentru a evalua performanța unui cromozom, este necesară simularea sistemului pe întreg orizontul de optimizare și memorarea valorilor nivelului *h* și a puterii generate P_1 si P_2. De menționat că $J_{\Delta h}$ trebuie minimizată, iar J_e trebuie maximizată, deci este nevoie să se inveseze semnul la una dintre ele.

Funcția globală de evaluare a performanței unui sistem de control este: $J = J_e - \mu \cdot J_{\Delta h}$, unde μ este o pondere introdusă cu scopul de a echilibra sensibilitatea lui *J* în raport cu $J_{\Delta h}$. Această funcție va fi folosită în final pentru alegerea unei soluții de pe frontul Pareto.

5. Exerciții

1. Folosind exemplul *MultiObjectiveExample* din pachetul *jgap* ca punct de pornire, construiți programul pentru studiul de caz I. Exemplul din în pachetul *jgap* realizează optimizarea multiobiectiv pentru o funcția de performanță: $f = (f_1, f_2) = (x^2; (x-2)^2)$. Este nevoie deci, să modificați funcția de performanță și funcția care alege soluția finală.
2. După găsirea frontului Pareto, reprezentați grafic soluțiile găsite așa cum este indicat în capitolul 3.4. Rulați algoritmul pentru încă 30 de generații, refăcând de fiecare dată graficul, în locul celui anterior. Ce observați din animația rezultată ?
3. Folosind același exemplu, implementați programul pentru studiul de caz II, în varianta de control fuzzy. Alegeți valori potrivite pentru constantele date și pentru valorile fuzzy astfel încât problema să aibă sens.
4. Pentru studiul de caz II, reprezentați grafic frontul Pareto prin adăugarea de puncte aferente fiecărei soluții pe o reprezentare carteziană ($J_{\Delta h}$, J_e). Analizați forma, capetele și caracteristicile acestuia și înțelegeți legătura dintre parametrii de control fuzzy și frontul Pareto.