UNIUNEA EUROPEANĂ
GUVERNUL ROMÂNIEI MINISTERUL MUNCII, FAMILIEI ŞI PROTECŢIEI SOCIALE AMPOSDRU
Fondul Social European POSDRU 2007-2013
Instrumente Structurale 2007-2013
OIPOSDRU
UNIVERSITATEA TEHNICĂ "GHEORGHE ASACHI" DIN IAŞI

UNIVERSITATEA TEHNICĂ
"GHEORGHE ASACHI" DIN IAŞI

Şcoala Doctorală a Facultății de Inginerie Chimică şi Protecția Mediului

APLICAȚII ALE ALGORITMILOR EVOLUTIVI ÎN OPTIMIZAREA PROCESELOR DIN INGINERIA CHIMICĂ

- REZUMATUL TEZEI DE DOCTORAT -

Conducător de doctorat:
Prof. univ. dr. ing. Silvia Curteanu

Doctorand:
Ing. Renata Furtună (Popa)

IAŞI - 2011

Proiectul „Burse Doctorale - O Investiție în Inteligență (BRAIN)", POSDRU/6/1.5/S/9, ID 6681, este un proiect strategic care are ca obiectiv general „Îmbunătăţirea formării viitorilor cercetători în cadrul ciclului 3 al învăţământului superior - studiile universitare de doctorat - cu impact asupra creşterii atractivităţii şi motivaţiei pentru cariera în cercetare".

Proiect finanțat în perioada 2008 - 2011.

Finanțare proiect: 14.424.856,15 RON

Beneficiar: Universitatea Tehnică "Gheorghe Asachi" din Iaşi

Partener: Universitatea "Vasile Alecsandri" din Bacău

Director proiect: Prof. univ. dr. ing. Carmen TEODOSIU

Responsabil proiect partener: Prof. univ. dr. ing. Gabriel LAZĂR

UNIVERSITATEA TEHNICĂ "GHEORGHE ASACHI" DIN IAŞI

R E C T O R A T U L

Către

_____

_____

Vă facem cunoscut că, în ziua de 28.11.2011, la ora 10:00, în Sala de Consiliu a Facultății de Inginerie Chimică şi Protecția Mediului, va avea loc susținerea publică a tezei de doctorat intitulată:

"APLICAȚII ALE ALGORITMILOR EVOLUTIVI ÎN OPTIMIZAREA PROCESELOR DIN INGINERIA CHIMICĂ"

elaborată de doamna ing. **RENATA FURTUNĂ (POPA)** în vederea conferirii titlului ştiințific de doctor.

Comisia de doctorat este alcătuită din:

1. Prof. dr. ing. Dan Caşcaval, Universitatea Tehnică „Gheorghe Asachi", Iaşi — preşedinte
2. Prof. dr. ing. Silvia Curteanu, Universitatea Tehnică „Gheorghe Asachi", Iaşi — conducător ştiințific
3. Prof. dr. ing. Octavian Păstrăvanu, Universitatea Tehnică „Gheorghe Asachi", Iaşi — membru
4. Conf. dr. Sabin Buraga, Universitatea „Alexandru Ioan Cuza", Iaşi — membru
5. CP I dr. ing. Maria Cazacu, Institutul de Chimie Macromoleculară „Petru Poni", Iaşi — membru

Vă trimitem rezumatul tezei de doctorat cu rugămintea de a ne comunica, în scris, aprecierile dumneavoastră.

Cu această ocazie vă invităm să participați la susținerea publică a tezei de doctorat.

RECTOR,

Prof.univ.dr.ing.**ION GIURMA**

Secretar universitate,

Ing.Cristina Nagîț

Mulțumiri

Doresc să adresez sincere mulțumiri doamnei prof. univ. dr. ing. Silvia Curteanu pentru îndrumarea, sprijinul şi ajutorul necondiționat pe care mi le-a acordat pe întreaga durată a cercetării şi elaborării tezei de doctorat. Realizările mele şi experiența acumulată în domeniul aplicațiilor inteligenței artificiale în modelarea şi optimizarea proceselor din ingineria chimică n-ar fi fost posibile fără îndrumarea domniei sale.

De asemenea, doresc să îmi exprim aprecierea şi deosebita recunoştință față de domnul ş.l. dr. ing. Florin Leon pentru suportul şi indicațiile valoroase oferite în dezvoltarea şi implementarea aplicațiilor software bazate pe instrumente ale inteligenței artificiale.

Mulțumesc tuturor colegilor pentru cadrul profesional creat şi sprijinul acordat de-a lungul anilor de doctorat.

Alese mulțumiri cadrelor didactice de la Facultatea de Automatică şi Calculatoare pentru pregătirea pe care mi-au oferit-o în timpul studiilor universitare.

Aş dori, de asemenea, să mulțumesc domnului prof. univ. dr. Manuel Andrés Rodrigo Rodrigo de la Universitatea Castilla-La Mancha din Spania pentru colaborarea eficientă pe parcursul stagiului de pregătire extern.

Cele mai calde gânduri şi mulțumiri sunt adresate familiei mele pentru înțelegerea, răbdarea şi încurajările acordate în toți aceşti ani.

CUPRINS

1. Introducere

Domeniul ***inteligenței artificiale*** prezintă una din cele mai spectaculoase dezvoltări, datorate în special diversității aplicațiilor în care sunt utilizate tehnicile sale. Instrumentele inteligenței artificiale s-au evidențiat prin eficiența în modelarea şi optimizarea proceselor chimice, cele mai utilizate fiind rețelele neuronale artificiale şi algoritmii evolutivi.

Rețelele neuronale constituie veritabile instrumente de modelare ce posedă abilitatea de a învăța variate tipuri de relații. Fiind capabile de a aproxima orice funcție neliniară continuă, rețelele neuronale oferă o soluție efectivă în modelarea sistemelor neliniare complexe.

Algoritmii evolutivi reprezintă metode eficiente de optimizare globală care folosesc modelul computațional al selecției naturale, bazându-se pe principiile eredității (reproducere, mutație şi recombinare) şi a supraviețuirii celui mai adaptat individ. Soluțiile candidat ale unei probleme de optimizare joacă rolul indivizilor dintr-o populație, iar funcțiile de fitness determină mediul în care "trăiesc" soluțiile.

În ingineria reacțiilor chimice există numeroase procese şi sisteme pentru care nu se cunosc legile fizice şi chimice ce le guvernează sau sunt disponibile cunoştințe limitate, nesigure. În astfel de cazuri, fie nu pot fi dezvoltate modele fenomenologice, fie modelele obținute sunt afectate de erori semnificative. Modelarea empirică, bazată pe date experimentale, devine un instrument util modelării, având şanse reale să furnizeze modele precise. Rețelele neuronale se încadrează în această categorie, funcționând pe principiul „cutiilor negre", respectiv operând cu date intrare – ieşire.

În cazul proceselor chimice complexe, puternic neliniare, pe lângă problema obținerii cu dificultate a unui model mecanicist, trebuie avute în vedere dificultățile legate de rezolvarea modelelor matematice reprezentate, adesea, de sisteme de ecuații diferențiale. Redarea dependenței ieşire – intrare folosind rețele neuronale evită problema rezolvării unor modele matematice.

O altă dificultate majoră în reglarea calității produselor chimice constă în *lipsa măsurătorilor „on-line"* pentru unele variabile şi în *întârzierile substanțiale* ce însoțesc măsurările.

Dificultăți suplimentare în modelarea proceselor pot fi identificate în domeniul complex al proceselor de polimerizare: creşterea semnificativă de viscozitate pe măsură ce reacția avansează, obținerea de molecule cu dimensiuni diferite, mari, mecanisme de reacție variate, complexe, incomplet cunoscute. Din punct de vedere al modelării, schemele cinetice conduc la modele complexe, adesea afectate de erori, dificil de elaborat şi soluționat. În aceste condiții, modelarea neuronală devine o alternativă demnă de luat în considerare, comparativ cu modelarea fenomenologică clasică.

Dificultățile legate de optimizarea unui proces chimic sunt determinate, în special, de natura multi-obiectiv a problemelor, de metoda de soluționare sau de modelul matematic necesar procedurii de optimizare. Optimizarea proceselor chimice este, cel mai adesea, multi-obiectivă, deoarece implică satisfacerea simultană a mai multor obiective, de multe ori contradictorii. Rezolvarea unei astfel de probleme se poate realiza vectorial, obiectivele de realizat fiind componentele unei funcții obiectiv vectoriale, sau scalar, obiectivele fiind combinate ponderat într-o funcție obiectiv scalară.

Din punct de vedere al modalității de soluționare a unei probleme de optimizare, se pot folosi fie metode tradiționale, bazate pe funcții diferențiabile, fie metode bazate pe algoritmi evolutivi, în principal algoritmi genetici. Se pot evidenția câteva dificultăți specifice primei categorii de metode: convergența la soluția optimă depinde de valorile inițiale, există pericolul detectării unor minime locale, algoritmii sunt dependenți de tipul de proces supus optimizării. Algoritmii evolutivi înregistrează, în ultimii ani, progrese rapide datorită faptului că pot fi aplicați cu rezultate bune la procese diverse. Ei nu necesită valori inițiale, conduc, în majoritatea cazurilor, la optimul global, chiar în condițiile existenței optimelor locale, şi nu folosesc derivate ale funcției obiectiv. Tehnicile de optimizare bazate pe algoritmi evolutivi, comparativ cu metodele clasice, sunt recomandate pentru procedee care se desfăşoară „on-line", şi pentru faptul că sunt extrem de robuste. De

asemenea, se pot proiecta diferite variante de algoritmi evolutivi, adaptate procesului supus optimizării, care să beneficieze de performanțe îmbunătățite în ceea ce priveşte convergența spre optimul global, viteza de căutare, reproductibilitatea şi precizia rezultatelor.

1.1 Obiectivele tezei

Teza de doctorat intitulată "***Aplicații ale algoritmilor evolutivi în optimizarea proceselor din ingineria chimică***" îşi propune să abordeze o serie de aspecte legate de aplicarea ***tehnicilor neuro-evolutive*** la modelarea şi optimizarea proceselor din industria chimică. În principal, pornind de la algoritmul NSGA-II, aplicat pe scară largă şi cu rezultate bune proceselor chimice, se urmăreşte elaborarea unor variante îmbunătățite, destinate optimizării modelelor şi proceselor. Modelele incluse în procedurile de optimizare sunt reprezentate de rețele neuronale simple sau rețele neuronale de tip stivă, dezvoltate în variante optime.

Principalele obiective propuse şi rezolvate în această lucrare sunt:

- Elaborarea unor metodologii de modelare bazate pe rețele neuronale care să conducă la determinarea unor modele optime sau aproape optime.
  - metodologie de optimizare a unei rețele neuronale de tip feed-forward, bazată de încercări succesive, constituită algoritmic;
  - metodologie de optimizare a rețelelor neuronale individuale sau agregate în stive folosind un algoritm hibrid care îmbină căutările globală şi locală (NSGA-II-QN).
- Proiectarea unor variante îmbunătățite ale algoritmului NSGA-II şi aplicarea acestora pentru optimizarea modelelor neuronale sau pentru optimizarea multi-obiectiv a unor procese complexe din industria chimică.
  - algoritmul genetic de sortare ne-dominată, NSGA-II;
  - algoritmul genetic de sortare ne-dominată bazat pe codificare cu numere reale şi prevăzut cu operator de transpoziție, NSGA-II-RJG;
  - hiper-euristica evolutivă NSGA-II-QN compusă dintr-un algoritm evolutiv multi-obiectiv (NSGA-II) şi un algoritm de antrenare a unei rețele neuronale (Quasi-Newton).
- Implementarea algoritmilor de modelare şi optimizare în programe proprii, asamblate într-un produs software unitar, prevăzut cu interfață grafică.
- Asocierea strategiilor de modelare şi optimizare cu diferite procese din domeniul ingineriei chimice, în vederea simulării diferitelor caracteristici de proces sau proprietăți ale produselor obținute.
  - sinteza unor polimeri şi copolimeri siloxanici;
  - prepararea unor nanoparticule polimerice cu surfactanți siliconici,
  - polimerizarea radicalică a stirenului;
  - modelarea viscozității unor compuşi pe bază de azometină;
  - sinteza unor hidrogeluri multicomponente pe bază de poliacrilamidă;
  - modelarea procesului de ultrafiltrare pe suport polimeric / electrodepunere pentru recuperarea metalelor grele.

1.2 Structura tezei

Teza este organizată astfel încât fiecare metodologie să fie prezentată distinct, într-un capitol separat, succesiunea lor corespunzând sensului creşterii complexității prin adăugarea diverselor elemente pentru îmbunătățirea performanțelor. Astfel, prima problema abordată este reprezentată de descrierea algoritmului de proiectare a unor rețele neuronale optime deoarece un model neuronal eficient este una din condițiile care determină rezultate satisfăcătoare de optimizare. În continuare sunt prezentați algoritmii evolutivi de tip NSGA-II, în variante îmbunătățite, aplicați pentru optimizarea proceselor sau a rețelelor individuale sau stivă. Fiecare capitol conține descrierea detaliată a câte unui algoritm, urmată de studii de caz care ilustrează şi testează eficiența acestuia. În unele cazuri, acelaşi proces este abordat cu tehnici diferite, pentru compararea rezultatelor în

aceleaşi condiţii. Studiile de caz conţin descrierea bazelor de date experimentale, adaptarea algoritmilor la condiţiile concrete (datele disponibile), prezentarea şi interpretarea rezultatelor.

Din punct de vedere structural, teza este constituită din 8 capitole dintre care capitolul 1 reprezintă introducerea, capitolele 2 şi 3 cuprind aspecte privind fundamentele teoretice şi realizările relevante în domeniu, iar următoarele (capitolele 4-8) contribuţia propriu-zisă a tezei. Fiecare capitol se încheie cu o secţiune de concluzii, iar ultimul capitol cuprinde concluziile generale. Acestea sunt prezentate din două puncte de vedere: pe de o parte contribuţia tezei la rezolvarea obiectivelor IA propuse (optimizare multi-obicetiv cu tehnici neuro-evolutive) şi, pe de altă parte, este considerat fiecare proces abordat, cu procedurile de modelare şi optimizare aplicate.

Lucrarea conţine o serie de scheme ce sintetizează în format grafic subiectele abordate şi principalele rezultate obţinute, astfel încât este uşor de urmărit modul de organizare a problemelor tratate (considerând concluziile parţiale, concluziile finale şi schemele).

Capitolul 1 prezintă o serie de aspecte generale, constituind o introducere în conţinutul tezei. Sunt enumerate obiectivele generale ale tezei, detaliate pe subprobleme, şi este prezentată structura acesteia, incluzând conţinutul fiecărui capitol.

Capitolul 2 prezintă o serie de aspecte teoretice ale reţelelor neuronale, din punctul de vedere al modelelor de optimizat. Astfel, sunt trecute în revistă elementele caracteristice ale reţelelor neuronale artificiale (arhitectură, tipuri de reţele, metode de antrenare), punându-se accent pe metode de determinare a topologiei dat fiind faptul că acesta este unul din obiectivele importante ale tezei.

Capitolul 3, al doilea cu conţinut teoretic, vizează algoritmii evolutivi, ca instrumente de optimizare. Este un capitol cu dimensiuni mai mari, dat fiind faptul că el fixează baza de referinţă a cercetărilor originale. Capitolul este împărţit în trei secţiuni: principii de bază şi evoluţia conceptelor (noţiuni generale, proiectarea unui algoritm evolutiv, variante de algoritmi evolutivi, evaluarea performanţei unui algoritm evolutiv), aplicaţii ale algoritmilor evolutivi (optimizare multi-obiectiv, estimare de parametri, dezvoltarea reţelelor neuronale) şi, în final, algoritmi evolutivi hibrizi, bazaţi pe combinarea căutărilor locală şi globală.

Capitolul 4 începe prezentarea contribuţiei propriu-zise a tezei. Este proiectată o metodologie eficientă, simplu de utilizat, pentru obţinerea de topologii optime pentru reţele neuronale de tip feed-forward. Metodologia este descrisă în detaliu, pornind de la considerentele practice pe care se bazează şi continuând cu algoritmul general proiectat. Cinci studii de caz sunt ataşate metodei, acestea evidenţiind eficienţa şi utilitatea strategiei propuse. Sunt abordate procese complexe de polimerizare (polimerizarea stirenului, sinteza unor polimeri siloxanici, prepararea unor nanoparticule polimerice cu surfactanţi siliconici, obţinerea unor compuşi pe bază de azometină) fiind redate prin simulare caracteristici de reacţie (conversie) sau proprietăţi ale produselor obţinute (masă moleculară, polidispersitate, dimensiunea particulelor, viscozitate) funcţie de condiţii de reacţie (cantităţi de reactanţi şi catalizatori, timp de reacţie, temperatură) sau descriptori structurali (diametrul moleculelor). Metodologia proiectată se bazează pe un algoritm în şase paşi, bine structurat şi robust, destinat determinării reţelelor neuronale optime, cu complexitate structurală minimă şi capacitate de generalizare maximă.

Capitolul 5 are drept conţinut optimizarea multi-obiectiv folosind algoritmul de sortare ne-dominată NSGA-II. Urmând structura generală adoptată, este prezentată mai întâi implementarea algoritmului, detaliind problemele specifice: etapele procedurii de optimizare multi-obiectiv, operatorii genetici şi parametrii de control ai algoritmului, evaluarea performanţelor NSGA-II. Algoritmul este aplicat optimizărilor multi-obiectiv ale proceselor de sinteză a polisiloxanilor şi copolimerilor siloxan-siloxan. Rezultatele obţinute demonstrează eficienţa algoritmului care, în plus, beneficiază de generalitate, putând fi uşor adaptat altor procese, cu funcţii vectoriale de diferite forme şi număr diferit de obiective de realizat.

Capitolul 6 prezintă implementarea şi aplicarea unei variante îmbunătăţite a algoritmului NSGA-II, respectiv algoritmul NSGA-II-RJG, având codificare cu numere reale şi un operator de transpoziţie („jumping genes”). Se foloseşte ca studiu de caz optimizarea procesului de preparare a unor nanoparticule polimerice cu surfactanţi siliconici, respectiv se urmăreşte determinarea

condițiilor optime care conduc la minimizarea diametrului mediu al particulelor şi minimizarea indicelui de polidispersitate. Capitolul cuprinde descrierea algoritmului, evidențiind elementele noi introduse, aplicarea acestuia şi compararea rezultatelor sale cu cele furnizate de NSGA-II.

Capitolul 7 introduce hiper-euristica evolutivă NSGA-II-QN împreună cu două variante ale acesteia particularizate pentru dezvoltarea şi optimizarea unor rețele neuronale simple (NSGA-II-QNN) şi a unor rețele neuronale de tip stivă (NSGA-II-QNSNN). La începutul capitolului sunt tratate aspectele teoretice pornind de la hiper-euristica evolutivă NSGA-II-QN şi particularizând apoi pentru cele două variante ale sale. Ilustrarea modalității prin care se efectuează modelarea neuro-evolutivă pe baza celor două hiper-euristici evolutive se face prin intermediul a două studii de caz. Hiper-euristica NSGA-II-QN este aplicată unui proces de ultrafiltrare pe suport polimeric / electrodepunere pentru recuperarea metalelor grele, iar hiper-euristica NSGA-II-QNSNN este utilizată la modelarea neuro-evolutivă a unui proces de sinteză unor hidrogeluri multicomponente pe bază de poliacrilamidă. Ultima secțiune a acestui capitol prezintă modul de lucru cu interfața grafică utilizator a programului ce implementează algoritmul NSGA-II-QN, în cele două varinate ale sale. Aplicația software include şi posibilitatea folosirii operatorului de transpoziție în cadrul algoritmului de optimizare multi-obiectiv.

Concluziile generale evidențiază în mod sintetic îndeplinirea obiectivelor propuse. Concluziile sunt formulate din două puncte de vedere: pe de o parte sunt descrise tehnicile bazate pe instrumente IA, iar pe de altă parte sunt considerate procese folosite drept studii de caz, menționându-se modelările şi optimizările efectuate, precum şi principalele rezultate obținute. Sunt atașate două secțiuni distincte care sistematizează elementele originale şi direcțiile de continuare a cercetărilor.

La sfârşitul tezei sunt prezentate alfabetic cele 214 referințe, acestea fiind reprezentate de articole consultate şi articole proprii, publicate sau în curs de apariție.

Rezultatele cercetărilor proprii din cadrul tezei de doctorat sunt prezentate în ultima secțiune a tezei, fiind concretizate prin: 5 articole ISI publicate sau acceptate spre publicare, 3 articole publicate sau acceptate în reviste BDI, 1 articol trimis spre publicare într-o revistă ISI, 1 brevet ce conține software original înregistrat la ORDA, precum şi 4 participări la manifestări ştiințifice (2 postere şi 2 comunicări). Se menționează, de asemenea, faptul că o parte din rezultate au fost incluse în 2 granturi de cercetare în calitate de membru.

4. Metodologie de optimizare a parametrilor unei rețele neuronale cu propagare înainte

Pentru ca o rețea neuronală să ofere performanțe maxime în modelarea unui proces chimic, aceasta trebuie să funcționeze la parametri optimi. Prin urmare, este necesară optimizarea atât a elementelor de proiectare ale rețelei, cât şi a parametrilor interni ai acesteia. Metodologia de optimizare a parametrilor unei rețele neuronale cu propagare înainte propusă în acest studiu vine ca o soluție la problema menționată mai sus. Metodologia se bazează pe o serie de considerente practice existente în literatura din domeniu şi construieşte un algoritm în şase paşi pentru optimizarea numărului de straturi ascunse, a numărului de neuroni ascunşi, a funcțiilor de activare, a ratei de învățare şi a termenului moment.

Tipul de rețea neuronală pentru care s-a dezvoltat metodologia de optimizare a parametrilor este perceptronul multistrat (MLP) cu propagare înainte (feed-forward) şi învățare supervizată, folosind algoritmul de învățare cu propagare înapoi a erorii (back-propagation).

4.1 Considerente practice asupra elementelor caracteristice unei rețele neuronale cu propagare înainte

Fiabilitatea şi puterea predictivă a unui model neuronal sunt puternic corelate cu alegerea variabilelor de intrare ale rețelei şi cu disponibilitatea unor date robuste. Datele trebuie să fie

reprezentative, trebuie să acopere domeniul de interes cât mai complet posibil şi să prezinte un nivel satisfăcător de uniformitate.

Este recomandat ca atât datele de intrare, cât şi datele de ieşire folosite la antrenarea reţelei să fie normalizate conform cu funcţiile de activare folosite în primul strat ascuns şi în stratul de ieşire. Înainte de utilizarea reţelei neuronale, ieşirile acesteia trebuie să fie denormalizate folosind operaţia inversă normalizării. Ecuaţiile 4.1 şi 4.2 reprezintă formulele folosite pentru operaţiile de normalizare şi, respectiv, denormalizare.

$$y_{norm} = (y - y_{max}) \frac{up - low}{y_{max} - y_{min}} + up \tag{4.1}$$

$$y = (y_{norm} - up) \frac{y_{max} - y_{min}}{up - low} + y_{max} \tag{4.2}$$

În aceste ecuaţii, y_{norm} este valoarea normalizată a variabilei y, y_{min} şi y_{max} reprezintă valoarea minimă şi, respectiv, valoarea maximă pentru domeniul de definiţie al variabilei y, iar *low* şi *up* sunt limitele inferioară şi superioară ale intervalului de normalizare.

Pentru o mare parte a aplicaţiilor cu reţele neuronale cu propagare înainte, numărul optim de straturi ascunse s-a demonstrat a fi mai mic decât numărul de intrări (Maren et al., 1990). În majoritatea cazurilor, un singur strat ascuns este suficient pentru a calcula zonele de decizie arbitrare pentru ieşiri. Trebuie reţinut faptul că, în situaţia unui set limitat de date, numărul de ponderi sinaptice ale reţelei poate fi redus prin adăugarea unor straturi de neuroni (Fernandes şi Lona, 2005).

Numărul de neuroni din straturile de intrare şi, respectiv, de ieşire este predeterminat de problemă şi, implicit, de datele de antrenare, însă numărul de neuroni din straturile ascunse poate varia.

Numărul de neuroni din straturile ascunse influenţează capacitatea de generalizare a reţelei neuronale cu propagare înainte. Un număr mare de neuroni în straturile ascunse produce o eroare mică la antrenare, dar nu conduce neapărat la o eroare mică pe datele de testare. Adăugarea de unităţi ascunse va avea întotdeauna drept consecinţă o reducere a erorii de antrenare şi va conduce mai întâi la o scădere a erorii obţinute la testare, dar apoi, începând cu un anumit număr de unităţi ascunse, ar putea produce o creştere a erorii la testare, ceea ce înseamnă pierderea capacităţii de generalizare a reţelei neuronale. Aceasta se datorează faptului că reţeaua începe să înveţe exact datele din setul de antrenare în loc să reţină o funcţie generalizată care să poată face calcule pe baza unor date noi. Astfel, apare supra-antrenarea, când reţeaua extrage nu doar informaţiile utile din setul de antrenare, ci şi zgomotul.

Numărul de neuroni din stratul ascuns este dependent de datele de antrenare. O regulă euristică afirmă că numărul de ponderi (numărul de conexiuni dintre neuroni) ar trebui să fie în jurul sau sub o zecime din produsul dintre numărul de vectori de antrenare şi numărul de ieşiri. Cu toate acestea, dacă datele de antrenare sunt relativ puţine, numărul de ponderi poate fi chiar jumătate din produs (Leon, 2006).

Pentru reţele neuronale cu mai mult de o ieşire, Khaw şi colab. (1995) au propus pentru numărul de neuroni din straturile ascunse următorul interval:

Limita inferioară pentru numărul de neuroni din primul strat ascuns: $2Ni + 1$,

Limita superioară pentru numărul de neuroni din primul strat ascuns: $No \cdot (Ni + 1)$,

unde Ni este numărul de intrări în reţeaua neuronală, iar No este numărul de ieşiri.

Pentru mai mult de un singur strat ascuns, numărul de neuroni din primul strat ascuns trebuie să fie de trei ori mai mare decât numărul de neuroni din următorul strat ascuns (relaţia lui Kudrycki).

Tipul funcţiei de activare utilizată are un impact semnificativ asupra vitezei de învăţare, precum şi asupra capacităţii de generalizare a reţelei neuronale. Funcţia de activare este aleasă în funcţie de tipul de date utilizate (binare, bipolare, zecimale etc.) şi tipul de strat (de intrare, ascuns, de ieşire). Stratul de intrare nu are funcţie de activare. Funcţiile de activare ale unităţilor ascunse

trebuie să fie neliniare. În majoritatea aplicațiilor din domeniu este folosită o rețea feed-forward cu un singur strat ascuns de neuroni care au o funcție de activare sigmoidă (sigmoida bipolară este cel mai des utilizată). Pentru neuronii de ieşire se recomandă funcții de activare adaptate distribuției datelor de ieşire.

Selectarea parametrilor pentru algoritmul back-propagation şi setările inițiale ale ponderilor sinaptice influențează atât viteza antrenării, cât şi convergența algoritmului.

Valori prea mari ale ponderilor duc la saturarea derivatelor funcțiilor de activare şi, astfel, modificările ponderilor vor fi aproape de zero. Pentru ponderi inițiale mici, modificările ar fi de asemenea foarte mici, ceea ce ar cauza o încetinire a procesului de învățare şi chiar ar împiedica convergența algoritmului.

Cea mai uşoară metodă pentru inițalizarea ponderilor este alegerea în mod aleatoriu a unor valori dintr-un interval adecvat, precum (-0.1, 0.1) sau (-2, 2). Fyfe (1996) recomandă intervalul $(-\frac{2.4}{F_i}, \frac{2.4}{F_i})$, în care F_i reprezintă numărul de intrări ale neuronului i.

Ca şi în cazul numărului de neuroni, dacă rețeaua va fi lăsată să învețe un număr foarte mare de epoci, eroarea la antrenare va continua să scadă, dar eroarea la testare va începe să crească datorită supra-antrenării şi, prin urmare, capacitatea de generalizare a rețelei va scădea. Totodată, în cazul folosirii unui număr mic de epoci de antrenare va apărea fenomenul de sub-adaptare (engl. „under-fitting") când rețeaua nu poate învăța.

În scopul determinării numărului optim de epoci de antrenare, se poate opri procesul de învățare când eroarea la antrenare scade sub o anumită valoare impusă sau când rețeaua începe supra-antrenarea. Cea de-a doua metodă este cea mai recomandată şi constă în oprirea antrenării când se atinge cea mai bună generalizare (momentul când eroarea pentru setul de validare este minimă).

Rata de învățare determină timpul în care se găseşte minimul în spațiul de căutare al ponderilor, adică viteza de antrenare. Cu cât rata de învățare este mai mică, sunt facuți paşi mai mici în spațiul ponderilor, timpul de învățare este excesiv de mare şi rețeaua este mai puțin capabilă să scape din minimele locale ale suprafeței de eroare. Paşi mai mari în spațiul ponderilor conduc la o antrenare mai rapidă, la abilitatea de a găsi minime locale „mai bune", dar şi la creşterea magnitudinii oscilațiilor pentru eroarea pătratică medie când a fost găsit un minim local pe suprafața de eroare.

O rată de învățare mai mică între stratul ascuns şi cel de ieşire creşte abilitatea rețelei de a distinge diferențe mici între exemple, în timp ce o rată mare între stratul de intrare şi cel ascuns permite extragerea informațiilor mai generale, îmbunătățind acuratețea rețelei neuronale (Fausett, 1994).

Pentru probleme relativ simple, o rată de învățare de 0.7 este acceptabilă, însă în general se recomandă rate de învățare de ordinul 0.05 până la 0.25 pentru a se asigura determinarea soluției de către rețeaua neuronală.

Antrenarea unui perceptron multistrat este deseori destul de lentă, necesitând mii sau zeci de mii de epoci pentru probleme complexe. Una din cele mai cunoscute metode de accelerare a învățării este metoda momentului.

Includerea termenului moment în calculul noilor valori pentru ponderi reprezintă o metoda simplă de creştere a ratei de învățare, evitând totodată instabilitatea. Mărimea ultimei modificări a ponderii este mediată de o constantă moment care poate fi orice număr între 0 si 1. O valoare satisfăcătoare pentru α este 0.9.

Efectele introducerii termenului moment se manifestă prin prevenirea blocării rețelei neuronale în minime locale, găsirea mai rapidă a minimului din spațiul ponderilor şi prin stabilizarea procesului de învățare.

4.2 Algoritm general pentru optimizarea parametrilor unei rețele neuronale cu propagare înainte

În demersul optimizării parametrilor unei rețelei neuronale feed-forward, cercetările noastre au pornit de la o rețea cu arhitectură de tip perceptron multistrat, cu un singur strat ascuns şi parametrii presetați.

Datele folosite la proiectarea rețelei neuronale sunt randomizate şi împărțite în 75% date pentru antrenare şi validare şi 25% date pentru testare. Datele pentru antrenare şi validare sunt împărțite, la rândul lor, în 80% date pentru antrenare şi 20% date pentru validare încrucişată.

Fiecare simulare se face cu validare încrucişată la antrenare, alegându-se astfel numărul optim de epoci de antrenare. S-a observat totuşi că, în unele cazuri, eroarea la validare scade continuu până la epuizarea numărului maxim de epoci. De aceea, trebuie găsit un număr maxim optim de epoci de antrenare. Pentru evitarea minimelor locale, se face antrenarea rețelei de 3 ori, cu ponderi inițializate diferit la fiecare antrenare.

Determinarea valorilor optime pentru numărul de neuroni din stratul ascuns, rata de învățare şi termenul moment se face folosind opțiunea antrenării cu variația unui parametru din NeuroSolutions pentru Excel. Acest proces permite antrenarea unei rețele de mai multe ori, în timp ce se variază unul din parametrii rețelei, ceilalți parametri rămânând constanți.

Determinarea valorilor optime pentru ceilalți parametri – funcția de activare pentru stratul de ieşire, numărul de straturi ascunse – se face prin încercări/experimentări, corelându-se sau comparându-se rezultatele cu parametrii găsiți prin metoda antrenării cu variația unui parametru.

Algoritmul de optimizare a parametrilor rețelei neuronale cu propagare înainte constă în următoarea înşiruire de paşi:

1. *Determinarea numărului optim de neuroni din stratul ascuns al unei rețele neuronale feed-forward cu un singur strat ascuns*

Se aplică şi se corelează regulile euristice enunțate în cadrul secțiunii 4.1.3. Numărul optim de neuroni din stratul ascuns trebuie să satisfacă următoarele condiții:

$$Nw \le 1/10 \cdot Nt \cdot No, \tag{4.4}$$

sau dacă datele de antrenare sunt puține:

$$Nw \le 1/2 \cdot Nt \cdot No. \tag{4.5}$$

Numărul total de ponderi sinaptice ale unei rețele neuronale este calculat după formula:

$$Nw = Ni \cdot Hn + Hn \cdot No. \tag{4.6}$$

Dacă rețeaua neuronală are două sau mai multe ieşiri, se ține cont şi de formulele propuse de Khaw şi colab. (1995):

$$2 \cdot Ni + 1 \le Hn \le No \cdot (Ni + 1), \tag{4.7}$$

unde *Ni* este numărul de intrări ale rețelei neuronale, *Hn* – numărul total de neuroni din stratul ascuns, *Nw* – numărul de ponderi, *Nt* – numărul de vectori de antrenare şi *No* este numărul de ieşiri ale rețelei neuronale.

Se obține astfel un interval în care se variază numărul de neuroni din stratul ascuns pentru a se determina o valoare optimă.

2. *Determinarea valorii optime pentru rata de învățare*

Pornind de la rețeaua optimizată la pasul 1, se variază mai întâi rata de învățare pentru stratul ascuns şi apoi, cu valoarea obținută, se variază rata de învățare pentru stratul de ieşire, ceilalți parametri rămânând cei prestabiliți.

3. *Determinarea valorii optime pentru termenul moment*

Plecând de la ratele de învățare obținute la pasul anterior, se caută o valoare optimă pentru termenul moment, astfel încât să se accelereze învățarea, cu rezultate mai bune la antrenare şi testare. Se variază termenul moment pentru stratul ascuns şi apoi, cu valoarea obținută, se variază

termenul moment pentru stratul de ieşire, ceilalți parametri fiind cei găsiți la paşii anteriori şi cei prestabiliți inițial, pentru care nu s-a aflat încă o valoare optimă.

4. *Determinarea funcției de activare optime pentru stratul de ieşire*

Luând în considerare distribuția datelor de ieşire, se testează o serie de funcții de activare pentru stratul de ieşire al rețelei neuronale optimizată în paşii de mai sus.

5. *Determinarea numărului optim de neuroni din straturile ascunse ale unei rețele neuronale feed-forward cu două straturi ascunse*

Se aplică regulile euristice (4.4), (4.5), (4.7) şi cele enunțate în cadrul secțiunii 4.1.3. pentru o rețea cu două straturi ascunse:

$$Hn1 > 3Hn2, \tag{4.8}$$

$$Nw = Ni \cdot Hn1 + Hn1 \cdot Hn2 + Hn2 \cdot No, \tag{4.9}$$

unde *Hn1* şi *Hn2* reprezintă numărul de neuroni din primul strat ascuns şi, respectiv, din cel de-al doilea strat ascuns. Astfel, se obține un interval de valori pentru neuronii din primul şi cel de-al doilea strat ascuns, iar valorile optime ale acestora pot fi determinate în continuare prin încercare şi eroare.

6. *Optimizarea parametrilor rețelei neuronale cu două straturi ascunse urmărind paşii 2-4*

Se poate optimiza în continuare rețeaua cu două straturi ascunse prin variația valorilor pentru ratele de învățare, termenii moment şi prin modificarea funcției de activare din stratul de ieşire.

În final, se compară rețeaua cu un singur strat ascuns optimizată şi cea cu două straturi ascunse şi se alege cea cu performanțe maxime la antrenare şi testare.

Eroarea pătratică medie minimă obținută la validarea încrucişată (*MinMSECV*), eroarea pătratică medie normalizată obținută la testare (*NMSEtest*) şi coeficientul de corelație liniară la testare (*r*) au fost considerați indici de performanță.

Pentru ca o rețea neuronală cu propagare înainte să aibă o performanță mai bună decât o altă rețea neuronală, indicii de performanță trebuie să îndeplinească următoarele condiții:

$$r1 > r2, \tag{4.13}$$

$$MinMSECV1 < MinMSECV2, \tag{4.14}$$

$$NMSEtest1 < NMSEtest2, \tag{4.15}$$

unde *r1*, *MinMSECV1* şi *NMSEtest1* sunt indicii unei rețele neuronale cu o performanță mai bună decât cea a rețelei neuronale cu indicii *r2*, *MinMSECV2* şi *NMSEtest2*. În scopul a construirii unei formule pe baza celor trei indici de performanță, relațiile de mai sus trebuie compuse:

$$r1 - (MinMSECV1 + NMSEtest1) > r2 - (MinMSECV2 + NMSEtest2) \tag{4.17}$$

Astfel, pentru cuantificarea performanței rețelei neuronale s-a utilizat următoarea formulă:

$$perf\_index = r - (MinMSECV + NMSEtest) \tag{4.18}$$

Cu cât valoarea pentru *perf_index* este mai mare, cu atât este mai ridicată performanța rețelei.

4.3 Studii de caz ce includ aplicarea metodologiei de optimizare a parametrilor unei rețele neuronale cu propagare înainte

4.3.1 Modelarea viscozității reduse a unor compuşi pe bază de poliazometină (studiul de caz 1)

Măsurarea mărimii macromoleculelor este de interes special în industria polimerilor deoarece multe proprietăți fizice şi dinamice sunt legate de mărimea şi forma unei macromolecule. Viscozitatea soluțiilor diluate de polimeri este sensibilă la aceste caracteristici structurale ale macromoleculelor, ceea ce face ca măsurarea viscozității să fie utilă stabilirii de corelații cu proprietățile fizice şi de procesare.

Baza de date experimentale pentru acest studiu de caz a conținut 100 de perechi de date ilustrând valorile viscozității reduse (η_{sp}/c) obținute la diferite concentrații şi temperaturi ale soluției pentru doi copolimeri azometină-siloxan.

Cei doi copolimeri azometină-siloxan, notați SiAz1 şi SiAz2, au fost preparați conform cu procedurile descrise de Vasiliu şi colab. (2005).

Măsurătorile viscozimetrice s-au condus în dimetilformamida (DMF) în domeniul de temperatură 20 – 45°C (±0.01°C). Domeniul de concentrații a fost 5.86 – 10 g dL^{-1} pentru SiAz1 şi 6.3 – 9 g dL^{-1} pentru proba SiAz2.

Neuronii din stratul de intrare al rețelei neuronale feed-forward folosită pentru modelare corespund celor trei variable de intrare: *c* (concentrația soluției), *T* (temperatura) şi *tip* (tipul celor doi copolimeri - SiAz1 şi SiAz2), iar neuronul din stratul de ieşire corespunde variabilei modelate, η_{sp}/c (viscozitatea redusă). Astfel, scopul modelării neuronale este de a evalua dependența viscozității reduse funcție de temperatură şi concentrația soluției, conform cu tipul copolimerului folosit, SiAz1 (tipul 1) sau SiAz2 (tipul 2).

Datele din fişierul de intrare pentru rețeaua neuronală au fost amestecate şi împărțite după cum s-a precizat în secțiunea 4.2. Aşadar, pentru problema studiată, dintr-un set de 100 de exemplare, 60 au fost pentru antrenare, 15 pentru validare încrucişată şi 25 de exemplare au reprezentat setul de date pentru testare.

Paşii metodologiei de optimizare şi rezultatele obținute la fiecare etapă sunt descrise în continuare (Furtună şi colab., 2011a).

1. *Determinarea numărului optim de neuroni din stratul ascuns al unei rețele neuronale cu un singur strat ascuns*

Pentru problema abordată în acest studiu de caz, $Nt = 60$, $Ni = 3$, $No = 1$. Notațiile folosite în această secțiune sunt cele descrise şi utilizate în secțiunea 4.2.

Prin aplicarea regulilor euristice (4.4), (4.5) şi (4.6), au fost obținute următoarele rezultate: $Hn1 \leq 1$ sau dacă setul de date de antrenare e prea mic, $Hn1 \leq 7$. Prin urmare, numărul de neuroni din stratul ascuns a fost variat în intervalul [1, 7], în cadrul procesului de antrenare cu variația unui parametru.

Tabelul 4.2. Rezultatele simulărilor pentru optimizarea numărului de neuroni ascunşi

Nr. simulare	Nr. epocă - CV	Hn	MinMSECV	NMSEtest	r	perf_index
1	1000	7	0.01782	0.1635	0.9186	0.7373
2	1000	6	0.01749	0.1734	0.9150	0.7242
3	1000	5	0.01614	0.1593	0.9208	0.7453
4	1000	4	0.01739	0.1616	0.919	0.7400
5	1000	3	0.01712	0.1654	0.9176	0.7351

Fiecare rețea obținută în urma procesului de antrenare a fost testată pe setul de date de test. În final, rețeaua neuronală cu cele mai bune rezultate la antrenare şi testare a fost aleasă pentru optimizări ulterioare.

Câteva din rezultatele simulărilor sunt prezentate în tabelul 4.2 (Furtună şi colab., 2011a). Indicele de performanță a fost calculat folosind formula (4.18).

Indicii de performanță obținuți pentru diferite sume de neuroni ascunşi şi numărul corespunzător de epoci de antrenare rezultat prin folosirea validării încrucişate, au indicat faptul că o rețea neuronală cu 5 neuroni ascunşi poate asigura cea mai bună performanță. Prin urmare, numărul optim de neuroni ascunşi a fost stabilit la 5.

Rezultatele obținute de către rețeaua neuronală optimizată pentru date care nu au fost folosite la antrenare sau validare încrucişată pot fi observate în figura 4.6.

2. *Determinarea valorii optime pentru rata de învățare*

Ratele de învățare ale rețelei neuronale optimizată la pasul anterior au fost variate în intervalul [0.1, 1]. Metodele de antrenare şi testarea au fost aceleaşi ca şi cele descrise la pasul anterior.

Cel mai bun indice de performanță, 0.7426, a fost obținut pentru o rată de învățare a stratului ascuns de neuroni având valoarea 1 şi o rată de învățare a stratului de ieşire cu valoarea 0.7. Totuşi, performanța rețelei neuronale nu a crescut după aplicarea procesului de antrenare cu variația unui parametru, indicele de performanță obținut fiind mai mic decât cel atins în pasul anterior al algoritmului de optimizare. Prin urmare, valorile parametrilor determinate în pasul precedent au fost menținute pentru următorul pas al metodologiei de optimizare.

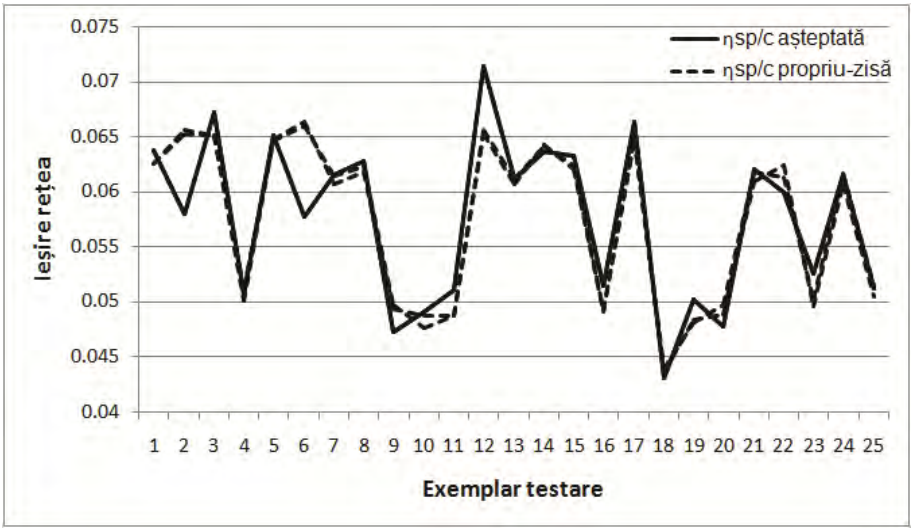


Figura 4.6. Ieşirea aşteptată şi ieşirea actuală a rețelei neuronale optimizată la pasul 1, pentru datele de test

3. *Determinarea valorii optime pentru termenul moment*

Plecând de la rețeaua neuronală optimizată la paşii anteriori şi utilizând acelaşi procedeu de antrenare ca mai sus, termenii moment ai rețelei neuronale au fost variați în intervalele [0.1, 1] şi [0.5, 1].

Pe baza rezultatelor obținute la pasul precedent, a fost optimizat inițial termenul moment pentru stratul ascuns de neuroni şi apoi cel pentru stratul de ieşire.

Cel mai mare indice de performanță, 0.7574, a fost obținut pentru o valoare a termenului moment al stratului ascuns de 0.9 şi o aceeaşi valoare pentru termenul moment al stratului de ieşire.

4. *Determinarea funcției de activare optime pentru stratul de ieşire*

Pentru stratul de ieşire, au fost testate sigmoida unipolară, tangenta hiperbolică liniară şi sigmoida unipolară liniară în scopul optimizării performanței rețelei neuronale.

Cea mai bună performanță a fost obținută pentru funcția de activare sigmoidă, dar indicele de performanță (0.7368) a fost mai mic decât cel obținut pentru tangenta hiperbolică (la pasul anterior, indicele de performanță a fost 0.7574). Astfel, parametrii obținuți în pasul anterior au fost menținuți pentru următorul pas al algoritmului de optimizare.

5. *Determinarea numărului optim de neuroni din straturile ascunse ale unei rețele neuronale cu două straturi ascunse*

Prin aplicarea regulilor euristice (4.4), (4.5), (4.8) şi (4.9), căutarea numărului optim de neuroni din straturile ascunse s-a restrâns la 3 până la 7 neuroni în primul strat ascuns şi un singur neuron în cel de-al doilea strat ascuns.

Plecând de la o rețea neuronală cu parametrii prestabiliți, numărul de neuroni din primul strat ascuns a fost determinat prin metoda încercare şi eroare, folosindu-se procesul de antrenare cu variația unui parametru şi validare încrucişată, pe durata a maxim 1000 de epoci de antrenare. Procedurile de antrenare şi testare au fost aceleaşi ca şi cele utilizate în primul pas al metodologiei de optimizare.

Performanța cea mai ridicată a fost atinsă pentru o rețea neuronală cu topologia 3:6:1:1 (6 neuroni în primul strat ascuns şi 1 neuron în cel de-al doilea strat ascuns).

Rețeaua neuronală cu două straturi ascunse optimizată în această etapă s-a dovedit a avea un indice de performanță (0.7337) mai scăzut decât cel al rețelei neuronale cu un singur strat ascuns (0.7574) optimizată la paşii anteriori.

6. *Optimizarea parametrilor rețelei neuronale cu două straturi ascunse urmărind paşii 2 – 4 de mai sus*

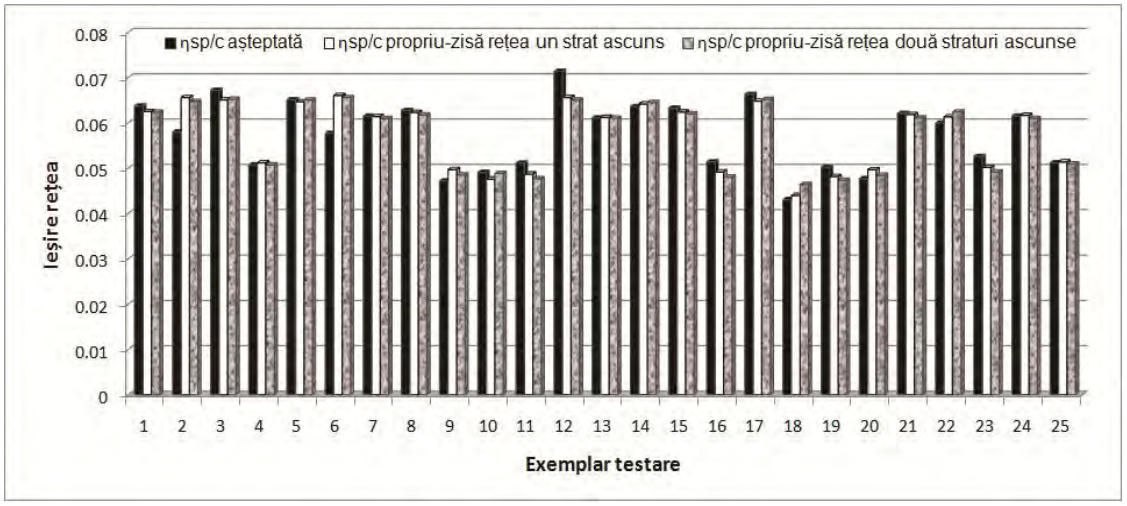


Figura 4.10. Ieşirea aşteptată şi ieşirile propriu-zise obținute cu cele două rețele neuronale, MLP(3:5:1) şi MLP(3:6:1:1) pentru datele de test (pasul 6)

Au fost efectuate o serie de simulări, aplicându-se aceleaşi proceduri ca şi cele utilizate la paşii 2 – 4 ai metodologiei de optimizare. Indicii de performanță obținuți au fost mai mici decât cei calculați pentru rețeaua neuronală optimizată la pasul precedent. Aşadar, valorile parametrilor determinate în etapa anterioară au rămas neschimbate.

Deoarece rețeaua neuronală cu două straturi ascunse (3:6:1:1) a obținut un indice de performanță mai mic decât cel obținut de către rețeaua neuronală cu un singur strat ascuns (3:5:1), acesta din urmă a fost ales pentru efectuarea de predicții pe seturi de date noi. Rezultatele furnizate de către cele două rețele neuronale optimizate la modelarea datelor de test sunt ilustrate în figura 4.10 (Furtună şi colab., 2011a).

4.3.3 Modelarea procesului de sinteză a polisiloxanilor (studiul de caz 3)

4.3.3.2 Baza de date experimentale

Sinteza polidimetilsiloxan-α,ω-diolilor a fost efectuată prin polimerizarea cu deschidere de ciclu a octametilciclotetrasiloxanului, D4, în prezența unui acid solid drept catalizator şi în absența unui solvent (Cazacu şi Marcu, 1995). În urma sintezei, s-a determinat randamentul reacției şi masa moleculară a polimerului obținut.

Baza de date experimentale a constat din 86 de exemple ilustrând valorile conversiei şi masei moleculare obținute cu diferite combinații de valori pentru timpul de reacție, temperatură, cantitatea de catalizator şi cantitatea de apă (Curteanu şi Cazacu, 2008).

4.3.3.3 Modelarea neuronală

Rețeaua neuronală cu propagare înainte utilizată pentru modelarea procesului de sinteză a polisiloxanilor a avut patru intrări, reprezentate de temperatură (T), timp (t), cantitate de catalizator (c_{cat}) şi cantitate de apă (c_{apa}), corespunzătoare celor patru neuroni din stratul de intrare şi două ieşiri, conversia (x) şi masa moleculară medie viscozimetrică (Mv), corespunzând neuronilor din stratul de ieşire.

Datele din fişierul de intrare pentru reţeaua neuronală au fost amestecate şi împărţite astfel încât pentru un set de 86 de exemplare, 52 au fost selectate pentru antrenare, 13 pentru validarea încrucişată şi 21 de exemplare au fost folosite pentru testare.

Datele de intrare au fost scalate in intervalul [-0.9, 0.9] pentru funcţia de activare a stratului ascuns, tangenta hiperbolică, folosind formula (4.1). În cadrul procesului de optimizare a funcţiei de activare a stratului de ieşire, datele experimentale reprezentând ieşirea aşteptată a reţelei neuronale au fost scalate în intervalul [0.05, 0.95] pentru funcţia logistică şi în intervalul [-0.9, 0.9] conform cu domeniul de definiţie al tangentei hiperbolice.

Au fost efectuate o serie de simulări utilizând procesul de antrenare cu variaţia unui parametru, urmând paşii algoritmului de optimizare descris în secţiunea 4.2, care a furnizat cea mai bună reţea neuronală cu un număr optimizat de neuroni în straturile ascunse, rate de învăţare, termeni moment şi funcţii de activare optime (Furtună şi colab., 2011b). Fiecare reţea a fost testată pe baza datelor de test. La final, reţeaua neuronală care a atins cea mai bună performanţă la antrenare şi testare a fost aleasă pentru a fi utilizată în procedeul de optimizare multi-obiectiv a procesului de sinteză a polisiloxanilor.

Reţeaua neuronală cu cea mai ridicată performanţă (1.596 – indice de performanţă total) a avut un singur strat ascuns cu 19 neuroni (4:19:2).

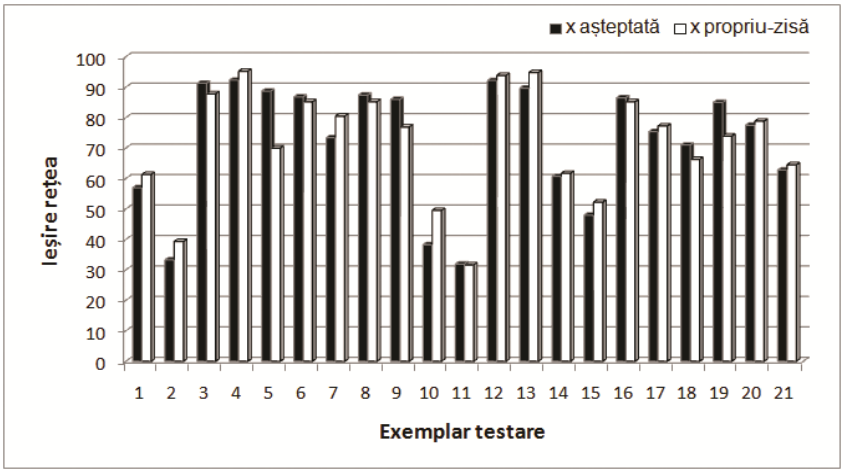


Figura 4.12. Conversia obţinută experimental vs. conversia obţinută cu reţeaua MLP(4:19:2) pentru datele de test

Rezultatele obţinute la testarea reţelei neuronale cu datele de test, care nu au fost folosite la antrenare, sunt reprezentate în figura 4.12, pentru modelarea conversiei şi în figura 4.13 pentru masa moleculară.

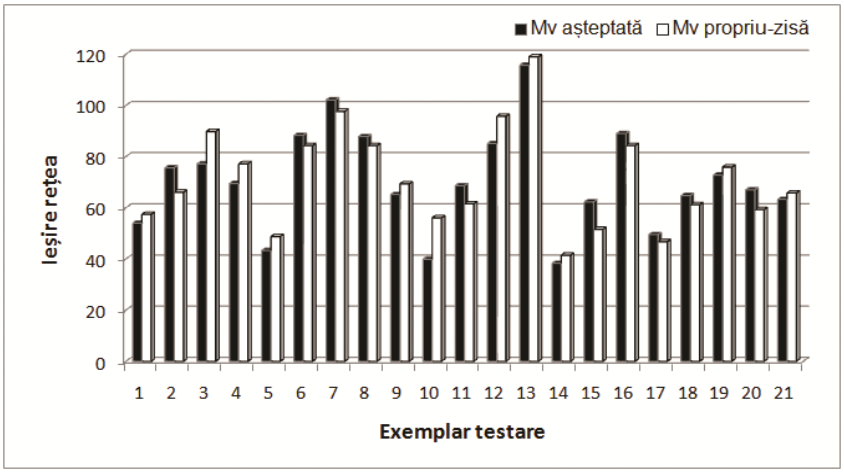


Figura 4.13. Masa moleculară aşteptată (obţinută experimental) vs. masa moleculară obţinută cu reţeaua MLP(4:19:2) pentru datele de test

După obținerea rețelei neuronale, procedura de optimizare a continuat prin includerea modelului neuronal în algoritmul evolutiv multi-obiectiv pentru calcularea funcțiilor de fitness. Rezultatele obținute la optimizarea multi-obiectiv au indicat faptul că este important nu numai să se folosească o rețea neuronală optimă, ci şi una cu performanțe echilibrate pentru cei doi parametri modelați. Astfel, a fost aleasă rețeaua cu o performanță globală apropiată de cea optimă (1.585 – indicele de performanță total), MLP(4:9:2), dar cu performanțe pentru modelarea celor două ieşiri mai bine echilibrate.

4.3.4 Modelarea sintezei nanoparticulelor polimerice cu surfactanți siliconici (studiul de caz 4)

4.3.4.2 Baza de date experimentale

Racleş şi colab. (2006, 2009) au aplicat principiile nanoprecipitării pentru a obține nanoparticule polimerice cu nuclee variate, în prezența surfactanților siliconici originali. Este binecunoscut că, în general, mărimea particulelor şi stabilitatea acestora depinde de mulți factori (Horn şi Rieger, 2001), dintre care concentrația şi raportul solvent/apă sunt foarte importante.

O serie de nanoparticule polimerice au fost obținute, utilizând acelaşi surfactant siloxanic. Dispersiile particulelor au fost analizate prin difuziune dinamică a luminii (DLS) şi informațiile despre diametrul mediu al particulelor şi indicele de polidispersitate rezultate au constituit cele 49 de modele pentru antrenarea şi testarea rețelelor neuronale.

4.3.4.3 Modelarea neuronală

Rețeaua neuronală folosită pentru modelarea dependeței diametrului mediu (*Zave*) față de concentrația de surfactant (*Cs*), concentrația de polimer (*Cp*) şi temperatura de stocare (*T*) a fost un perceptron multistrat (MLP) cu trei intrări şi o ieşire. Similar, pentru modelarea variației indicelui de polidispersitate (*PDI*) în funcție de concentrațiile de surfactant şi de polimer a fost utilizat un perceptron multistrat (MLP) cu două intrări şi o ieşire. În al doilea caz, temperatura de stocare nu a fost folosită ca intrare, deoarece valorile PDI la 4˚C au fost aceleaşi ca şi cele obținute la 40˚C.

Parametrii rețelelor neuronale au fost optimizați folosind algoritmul descris în secțiunea 4.2, cu scopul de a se obține o performanță maximă pentru modelele neuronale (Furtună şi colab., 2011c).

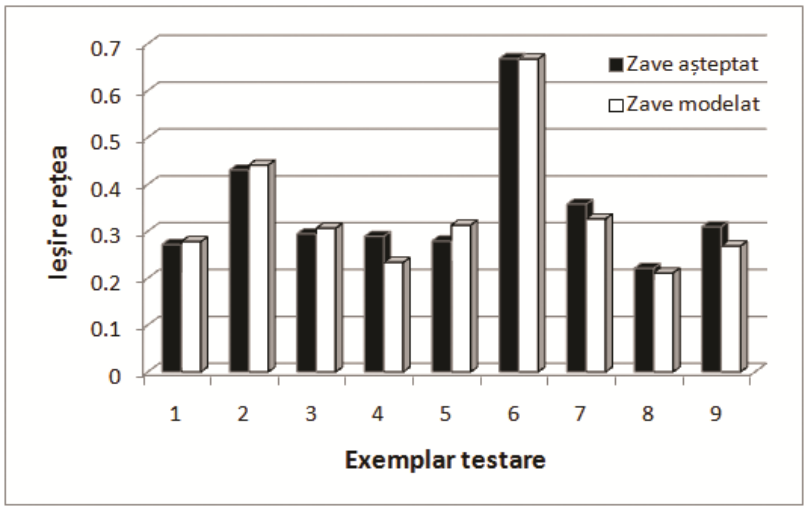


Figura 4.16. Rezultatele obținute la testare de către rețeaua neuronală ce modelează diametrul mediu

În urma testelor făcute asupra celui mai performant model neuronal determinat, s-a observat că folosirea a două rețele neuronale separate, cu câte o singură ieşire fiecare, a condus la rezultate mai bune decât cele obținute cu o rețea neuronală având două ieşiri.

Cele mai bune rețele neuronale obținute au fost MLP(3:9:1) pentru modelarea diametrului mediu şi MLP(2:6:1) pentru modelarea indicelui de polidispersitate. Performanțele obținute la testare de către cele două rețele neuronale optimizate, când li s-au prezentat date care nu au fost folosite la antrenare, sunt ilustrate în figurile 4.16 şi 4.17 pentru modelarea diametrului mediu şi, respectiv, a polidispersității.

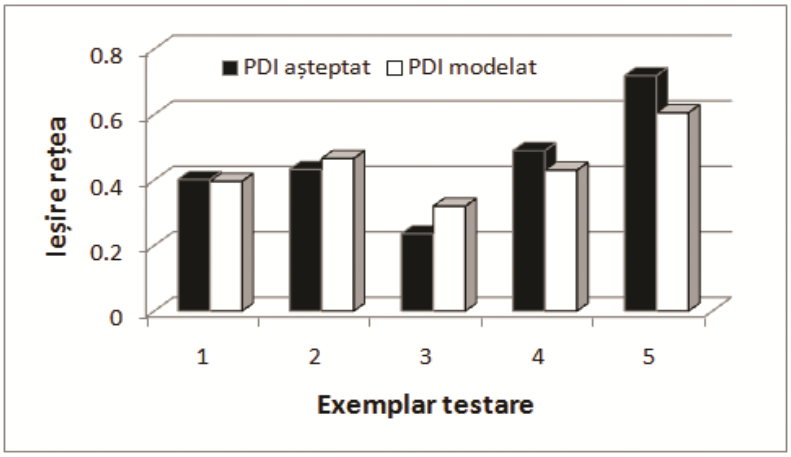


Figura 4.17. Rezultatele obținute la testare de către rețeaua neuronală ce modelează indicele de polidispersitate

4.3.5 Modelarea procesului de polimerizare prin radicali liberi a stirenului (studiul de caz 5)

4.3.5.2 Baza de date experimentale

Un model matematic complet bazat pe ecuații de conservare de masă pentru elementele din amestecul de reacție a fost elaborat şi rezolvat de Curteanu (2003), folosind momentele de distribuție ale concentrațiilor. Acest model a constituit simulatorul pentru producerea bazei de date experimentale folosită la modelarea cu rețele neuronale. Astfel, pentru concentrația inițiatorului şi temperatură s-au ales domenii specifice polimerizării în suspensie a stirenului, respectiv 10 ÷ 55 mol/m^3 (pas de variație 5) şi 60 - 90°C (pas de variație 10). Referitor la timpul de reacție, s-a considerat intervalul 0 – 2000 minute, având în vedere faptul că la concentrații de inițiator şi temperaturi mici, timpul de reacție este mai lung. În plus, pasul de reacție de 10 minute garantează reproducerea corectă a curbelor de variație a conversiei şi maselor moleculare, în condițiile în care, la un moment dat, dependent de condițiile inițiale de lucru, apare o auto-accelerare a reacției, deci pe curbele de conversie şi mase moleculare apar salturi bruşte.

4.3.5.3 Modelarea neuronală

Trei variabile au reprezentat intrările modelelor neuronale: concentrația de inițiator (I_0), temperatura (T) şi timpul de reacție (t), şi alte trei variabile au constituit ieşirile rețelelor: conversia monomerului (x), masa moleculară numerică medie (Mn) şi masa moleculară gravimetrică medie (Mw).

Cele 3494 exemplare din fişierul de la intrarea rețelei neuronale au fost randomizate şi împărțite în 2097 pentru antrenare, 524 pentru validare încrucişată şi 873 pentru testare.

Parametrii rețelei neuronale au fost optimizați prin intermediul procesului de antrenare cu variația unui parametru şi urmărindu-se algoritmul descris în secțiunea 4.2. Fiecare rețea neuronală obținută a fost testată utilizând setul de date de test. În cele din urmă, rețeaua cu cea mai ridicată performanță la antrenare şi la testare a fost aleasă pentru optimizări ulterioare.

Prin aplicarea regulilor euristice (4.4), (4.5) şi (4.6), a rezultat $Hn1 \leq 105$. Astfel, numărul de neuroni din stratul ascuns a fost variat în intervalul [5, 105] cu pasul 5. Cea mai bună performanță a fost atinsă pentru un număr de 45 de neuroni ascunşi.

Ratele de învățare şi termenii moment au fost variați în intervalul [0.1, 1]. Performanța rețelei neuronale nu a crescut în urma optimizării acestor parametri (*perf_index* mediu = 0.8525), prin urmare, ratele de învățare şi termenii moment ai rețelei neuronale au fost păstrați la valorile lor inițiale (Curteanu şi colab., 2010).

Valorile datelor de la intrare au fost scalate pentru funcția de activare a stratului ascuns, tangenta hiperbolică, în intervalul [-0.9, 0.9]. Valorile datelor de la ieşire au fost normalizate conform cu funcția de activare folosită pentru stratul de ieşire. Simulările realizate cu o funcție de activare liniară în stratul de ieşire al rețelei neuronale, nu au oferit o mai bună performanță pentru aceasta. Cea mai mare valoarea obținută pentru *perf_index* mediu a fost 0.8513, în condițiile în care valoarea maximă obținută pentru *perf_index* mediu înaintea optimizării funcției de activare a stratului de ieşire, a fost 0.8534.

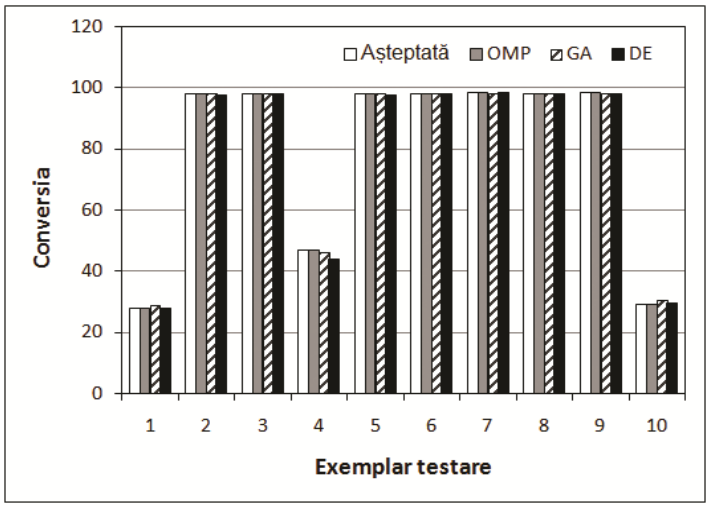


Figura 4.18. Conversia monomerului obținută experimental şi prin predicții realizate de rețele neuronale optimizate prin cele trei metode (ST, GA, DE)

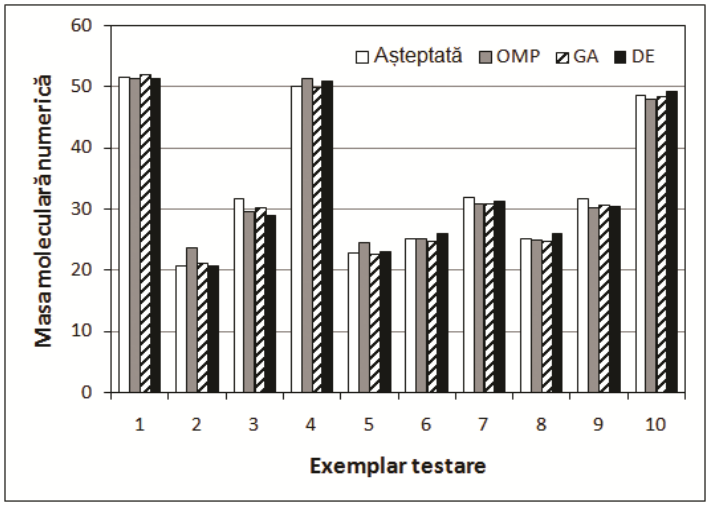


Figura 4.19. Masa moleculară medie numerică obținută experimental şi prin predicții realizate de rețele neuronale optimizate prin cele trei metode (ST, GA, DE)

Algoritmul de optimizare s-a terminat cu dezvoltarea unei rețele neuronale cu două straturi ascunse optimizată şi comparația performanței acesteia cu cea a rețelei neuronale cu un singur strat ascuns, determinată anterior.

Prin compararea indicilor de performanță obținuți de rețeaua cu un singur strat ascuns (*perf_index* mediu = 0.8534) şi de cea cu două straturi ascunse (*perf_index* mediu = 0.9060), s-a constatat că cea de-a doua este mai performantă.

Rezultatele furnizate de metodologia de optimizare a parametrilor unei rețele neuronale cu propagare înainte (ST) au fost comparate cu cele obținute prin alte două metode evolutive: algoritmi genetici (GA) şi evoluție diferențială (DE). Figurile 4.18, 4.19 şi 4.20 prezintă performanțele obținute prin cele trei metode în faza de testare. Se poate observa că cele trei metode dau rezultate comparabile a căror acurateţe poate fi considerată a fi foarte bună.

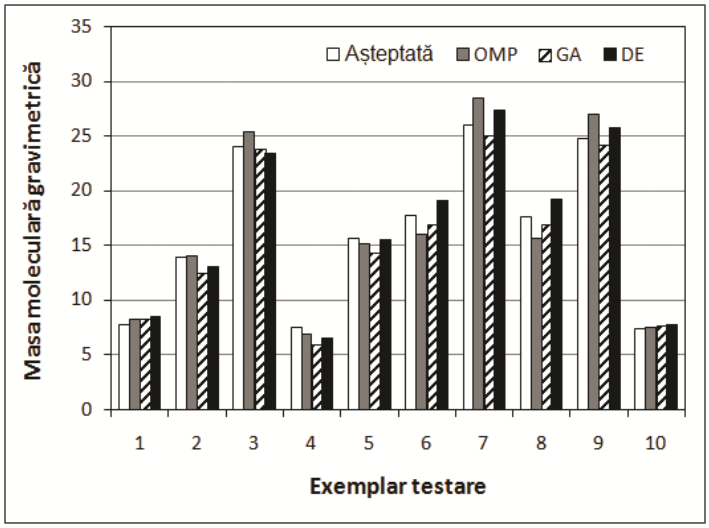


Figura 4.20. Masa moleculară medie gravimetrică obținută experimental şi prin predicții realizate de rețele neuronale optimizate prin cele trei metode (ST, GA, DE)

5. Optimizarea multi-obiectiv folosind algoritmul genetic de sortare ne-dominată NSGA-II

Algoritmul evolutiv multi-obiectiv NSGA-II (Deb şi colab., 2002) a fost adaptat şi aplicat la optimizarea unor procese de polimerizare complexe. Pentru implementarea acestei versiuni a algoritmului NSGA-II a fost creat un software original.

5.1 Etapele procedurii de optimizare multi-obiectiv

Algoritmul NSGA-II este nucleul procedurii de optimizare multi-obiectiv. Acesta este un algoritm iterativ care porneşte de la un număr de soluții (indivizi sau cromozomi) generate aleatoriu într-un anumit interval şi evoluează către soluții mai bune într-un număr prestabilit de iterații (generații de soluții). Soluția oferită de algoritmul NSGA-II este reprezentată de un front optim Pareto.

Implementarea software a procedurii de optimizare a fost realizată folosind limbajul de programare C#, pentru fiecare etapă a algoritmului NSGA-II fiind creată o funcție specifică (Furtună şi colab., 2010). Schema logică a procedurii de optimizare bazată pe algoritmul NSGA-II este prezentată în figura 5.1, unde N_{pop} este mărimea populației, N_{gen} este numărul maxim de generații, N_T – dimensiunea turneului, P_C – probabilitatea încrucişării şi P_M – probabilitatea mutației. Săgeata gri din figură indică faptul că tranziția dintre blocurile conectate prin aceasta are loc numai în prima iterație a algoritmului.

5.2 Operatorii genetici şi parametrii de control ai algoritmului NSGA-II

NSGA-II a fost adaptat optimizării multi-obiectiv a unor procese de polimerizare complexe prin folosirea codificării cu numere reale a soluțiilor şi prin utilizarea unor operatori genetici specifici acestui tip de codificare.

În implementarea software propusă pentru algoritmul NSGA-II, se folosesc cromozomi (soluții) compuşi din gene codificate cu valori reale, egale numeric cu variabilele de decizie ale problemei de optimizat. Constrângerile variabilelor de decizie sunt incluse în codificarea acestora prin limitarea intervalului de valori în care genele sunt inițializate aleatoriu.

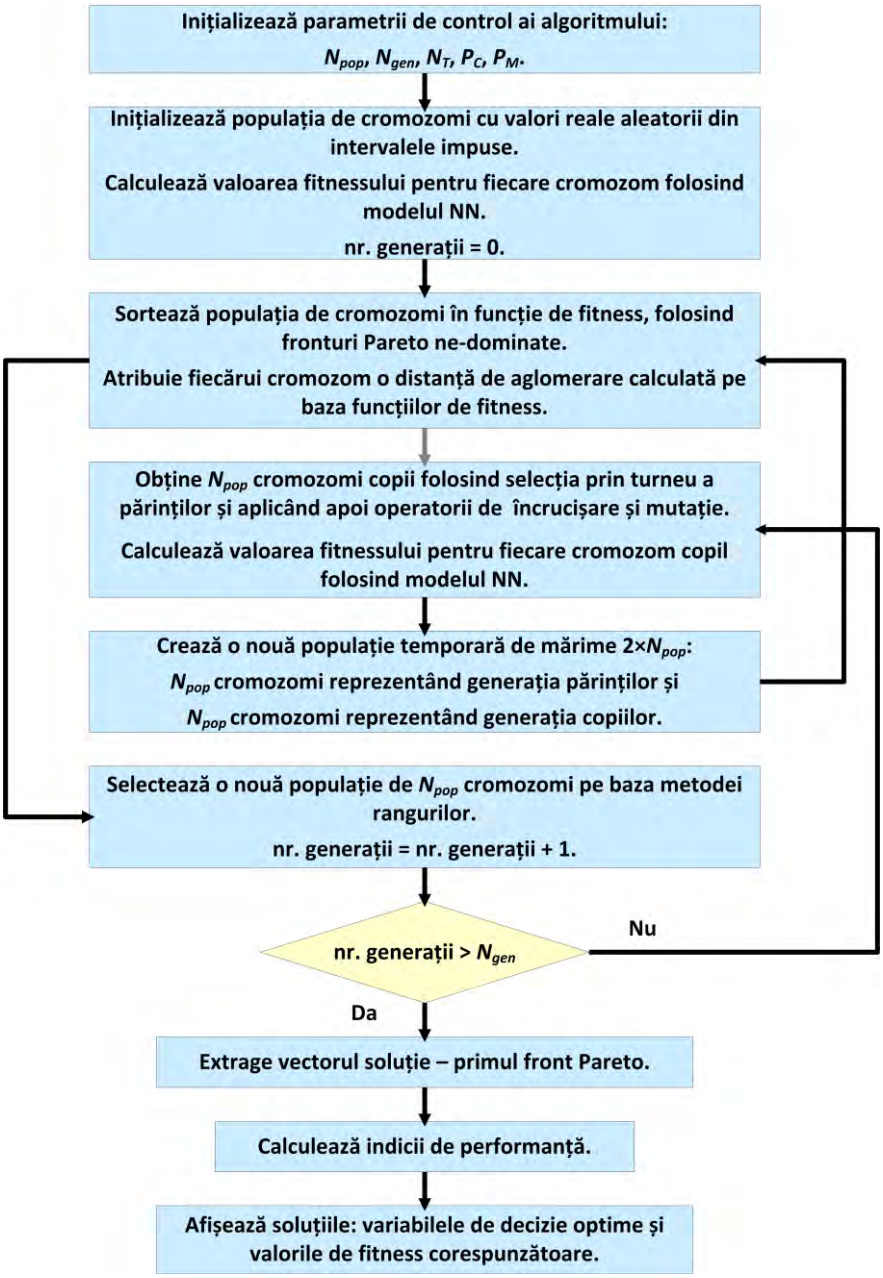


Figura 5.1 Schema logică a procedurii de optimizare multi-obiectiv bazată pe algoritmul NSGA-II

Selecția noii populații se face prin metoda rangurilor, selecția părinților prin turneu, mutația prin varianta resetării, iar recombinarea este aritmetică, cu un singur punct, diferit pentru fiecare genă.

Varianta algoritmului NSGA-II propusă în acest studiu include un nou operator pentru eliminarea soluțiilor duplicat din noua populație. Acest operator este aplicat fiecărui cromozom copil rezultat după încrucişare şi mutație, în cadrul procesului de creare a populației temporare. Cromozomul care urmează să fie inclus în noua populație temporară este comparat cu fiecare dintre cei deja aleşi şi, în cazul în care acesta se dovedeşte a fi unic, este adăugat la noua populație.

5.3 Evaluarea performanțelor algoritmului NSGA-II

Rezultatele algoritmului NSGA-II au fost evaluate prin intermediul timpului de convergență (*Ct*), a metricii de spațiere (*S*) şi a metricii de acoperire a setului (*CS*). Luând în considerare faptul că o soluție mai bună are un timp de convergență mai scurt şi o metrică de spațiere mai mică, a fost creată următoarea formulă pentru evaluarea setului de soluții optime obținut:

$$sol\_perf = 0.001 \cdot Ct + 0.999 \cdot S\,. \tag{5.2}$$

Sol_perf reprezintă o sumă ponderată, în care ponderea pentru timpul de convergență a fost aleasa ca fiind 0.001 pentru a scala valorile acestuia (exprimate în secunde) în acelaşi interval cu valorile obținute pentru metrica de spațiere. În consecință, ponderea pentru metrica de spațiere a fost 0.999, deoarece a fost considerată mai importantă obținerea unei distribuiri echidistante a soluțiilor în setul optim Pareto decât un timp de convergență foarte rapid. Cu cât *sol_perf* este mai aproape de 0, cu atât timpul de convergență şi distribuirea soluțiilor de-a lungul frontului optim Pareto sunt mai eficiente (Furtună şi colab., 2011b).

Corelarea metricii de acoperire a setului cu indicele de performanță *sol_perf* oferă o perspectivă globală asupra performanței algoritmului evolutiv multi-obiectiv NSGA-II.

5.4 Studii de caz asupra optimizării multi-obiectiv a unor procese din ingineria chimică folosind algoritmul NSGA-II

5.4.1 Optimizarea multi-obiectiv a procesului de sinteză a polisiloxanilor (studiul de caz 1)

5.4.1.1 Problema de optimizare

Premisele acestui studiu de caz au fost prezentate în secțiunea 4.3.3. Optimizarea multi-obiectiv a polimerizării cationice eterogene a octametilciclotetrasiloxanului reprezintă continuarea cercetării asupra eficienței instrumentelor inteligenței artificiale în suplinirea experimentelor chimice costisitoare.

Funcția multi-obiectiv vectorială, a cărei maximizare se urmăreşte de-a lungul procesului de optimizare, a fost compusă din două funcții de fitness, corespunzătoare celor două obiective conflictuale (Furtună şi colab., 2011b):

$$F(T,t,c_{cat},c_{apa}) = (x,\ -\left|1-\frac{Mvd}{Mv}\right|)\,, \tag{5.3}$$

unde *T* este temperatura, *t* – timpul de reacție, c_{cat} – cantitate de catalizator, c_{apa} – cantitatea de co-catalizator, *x* – conversia monomerului, *Mv* – masa moleculară viscozimetrică obținută prin procesul de sinteză şi *Mvd* – masa moleculară viscozimetrică impusă.

Intervalele de valori pentru cele patru variabile de decizie şi pentru conversie şi masa moleculară viscozimetrică, derivate din setul de date experimentale (Cazacu şi Marcu, 1995), au fost:

$$\begin{aligned} &T\text{: } 50 - 110^{\circ}\text{C},\ t\text{: } 10 - 180 \text{ min},\\ &c_{cat}\text{: } 0.5 - 3.5\ \%,\ c_{apa}\text{: } 0.1 - 0.8\ \%,\\ &x\text{: } 0 - 100\ \%,\ Mv\text{: } 31900 - 91600. \end{aligned} \tag{5.4}$$

5.4.1.2 Rezultatele optimizării multi-obiectiv

Au fost efectuate o serie de simulări folosind diferite valori pentru parametrii algoritmului NSGA-II, în scopul determinării valorilor pentru care algoritmul oferă cel mai bun randament în optimizarea procesului de polimerizare studiat. Valoarea impusă pentru masa moleculară viscozimetrică a fost 80000.

Folosind o probabilitate de încrucişare (P_C) de 0.9 şi o probabilitate de mutaţie (P_M) de 0.03, numărul de indivizi dintr-o populaţie (N_{pop}) a fost variat între 10 şi 500 şi numărul maxim de generaţii (N_{gen}) a fost variat între 50 şi 1000. Apoi, cu numărul de indivizi dintr-o populaţie şi numărul maxim de generaţii stabilite la valorile optimizate obţinute anterior, probabilitatea de încrucişare a fost variată între 0.1 şi 0.9, iar probabilitatea de mutaţie între 0.01 şi 0.8.

Cum era de aşteptat, timpul de convergenţă a crescut odată cu numărul de generaţii. De asemenea, s-a observat că soluţiile au început să fie distanţate uniform atunci când mărimea populaţiei a crescut, dar numai până la un anumit punct (N_{pop} = 300). Creşterea în continuare a dimensiunii populaţiei nu a dus la o scădere a valorii metricii de spaţiere. Cele mai bune cinci valori obţinute pentru *sol_perf* şi valorile corespunzătoare pentru mărimea populaţiei şi numărul maxim de generaţii folosite de algoritmul evolutiv pot fi observate în tabelul 4.1 (Furtună şi colab., 2011b).

Tabelul 5.1. Timpul de convergenţă şi metrica de spaţiere rezultate pentru diferite combinaţii N_{pop} şi N_{gen}, cu P_C = 0.9 şi P_M = 0.03

Nr. simulare	Parametrii NSGA-II		Ct(s)	S	sol_perf
	N_{pop}	N_{gen}			
11	100	50	1.27	0.0655	0.0667
12	100	100	2.34	0.0563	0.0586
16	300	50	10.88	0.0372	0.0480
17	300	100	24.14	0.0398	0.0639
21	500	50	35.69	0.0307	0.0664

Pentru calcularea metricii de acoperire a setului, a fost elaborată o tehnică originală bazată pe comparaţia dintre vectorii soluţie (fronturile Pareto) obţinuţi de NSGA-II. Comparaţia a fost făcută din punct de vedere al procentului de soluţii dintr-un vector care sunt dominate de soluţii din celălalt vector. Vectorul soluţie care a avut cele mai multe elemente ne-dominate a fost considerat mai bun şi a fost ales pentru compararea cu următorul vector soluţie obţinut de NSGA-II. Tehnica de comparaţie s-a bazat pe proprietatea de tranzitivitate a relaţiei de dominanţă Pareto care este o relaţie de ordine parţială (Coello Coello şi colab., 2007).

Trei dintre fronturile Pareto obţinute în urma căutării combinaţiei optime pentru dimensiunea populaţiei şi numărul maxim de generaţii sunt ilustrate în figura 5.2, unde se poate observa dominanţa dintre vectorii soluţie.

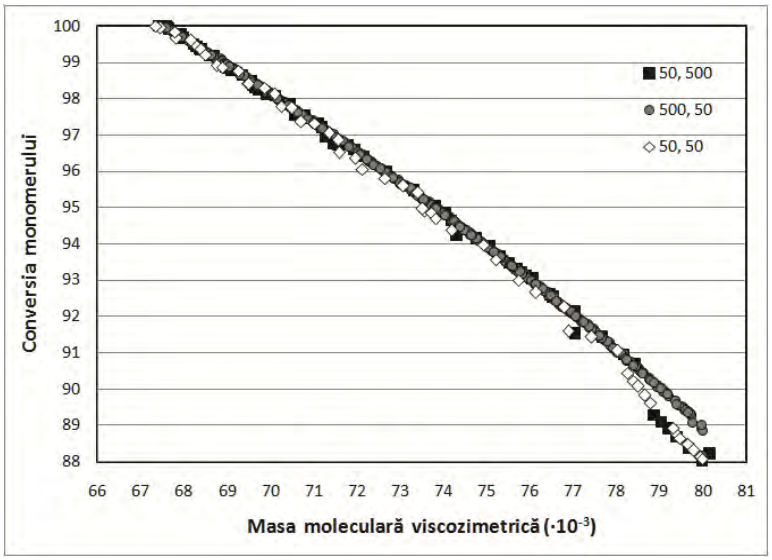


Figura 5.2. Fronturile Pareto obţinute cu diferite combinaţii pentru perechea de parametri (N_{pop}, N_{gen}): (50, 500), (500, 50) şi (50, 50)

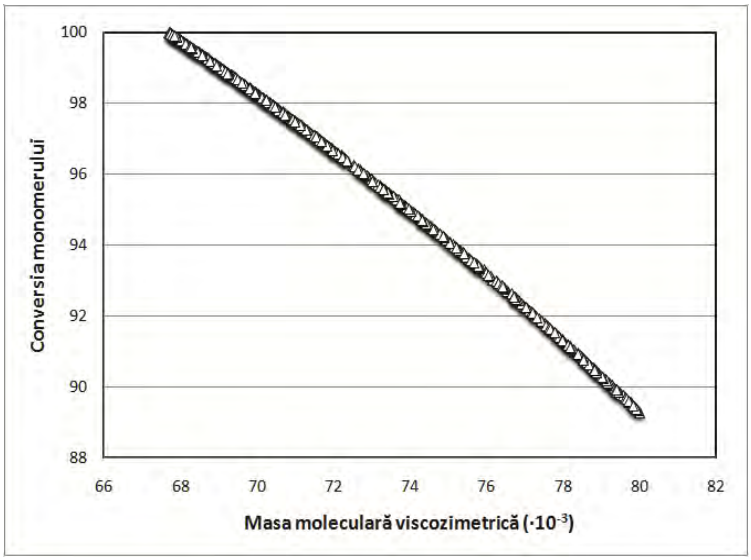
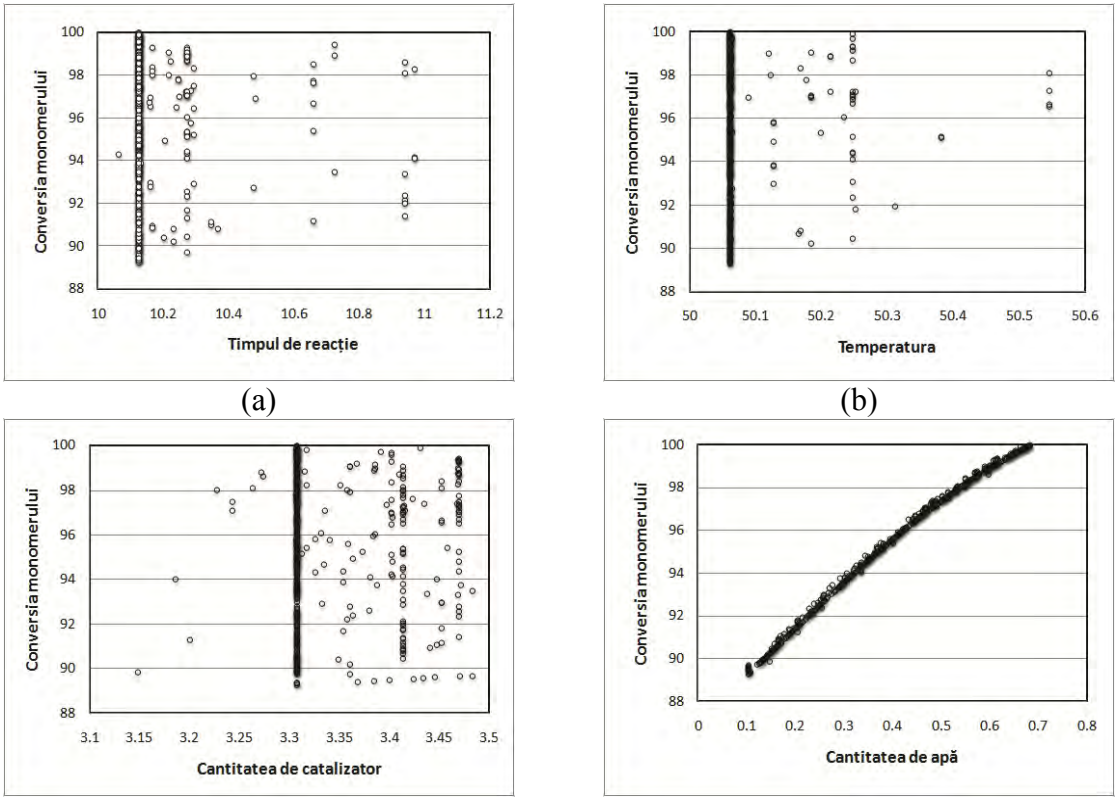


Figura 5.3. Frontul optim Pareto pentru *Mvd* = 80000

Vectorul soluție cel mai dominant s-a dovedit a fi cel obținut cu o populație de 500 de indivizi şi un număr maxim de 500 de generații (Furtună şi colab., 2011b).

Corelând ierarhizarea perechilor de parametri ai algoritmului NSGA-II din punct de vedere al indicelui *sol_perf* cu cea realizată din punct de vedere al metricii de acoperire a setului, combinația N_{pop} = 500 şi N_{gen} = 50 s-a demonstrat a fi cea mai bună (Furtună şi colab., 2011b).

Aceeaşi procedură de comparație ca şi cea folosită pentru determinarea celor mai bune valori pentru N_{pop} şi N_{gen} a fost utilizată în selectarea celor mai potrivite valori pentru probabilitățile de încrucişare şi mutație ale NSGA-II care oferă cel mai dominant front Pareto.



(a) (b)

(c) (d)

Figure 5.4. Condițiile de reacție: (a) timpul de reacție, (b) temperatura, (c) cantitatea de catalizator şi (d) cantitatea de apă care au maximizat conversia monomerului

S-a constatat că valorile cele mai bune pentru probabilitățile de încrucișare și mutație au fost 0.1 și, respectiv, 0.2. Aceste valori au condus, de asemenea, la obținerea celui mai scurt timp de convergență și a celei mai uniforme spațieri a soluțiilor din frontul Pareto.

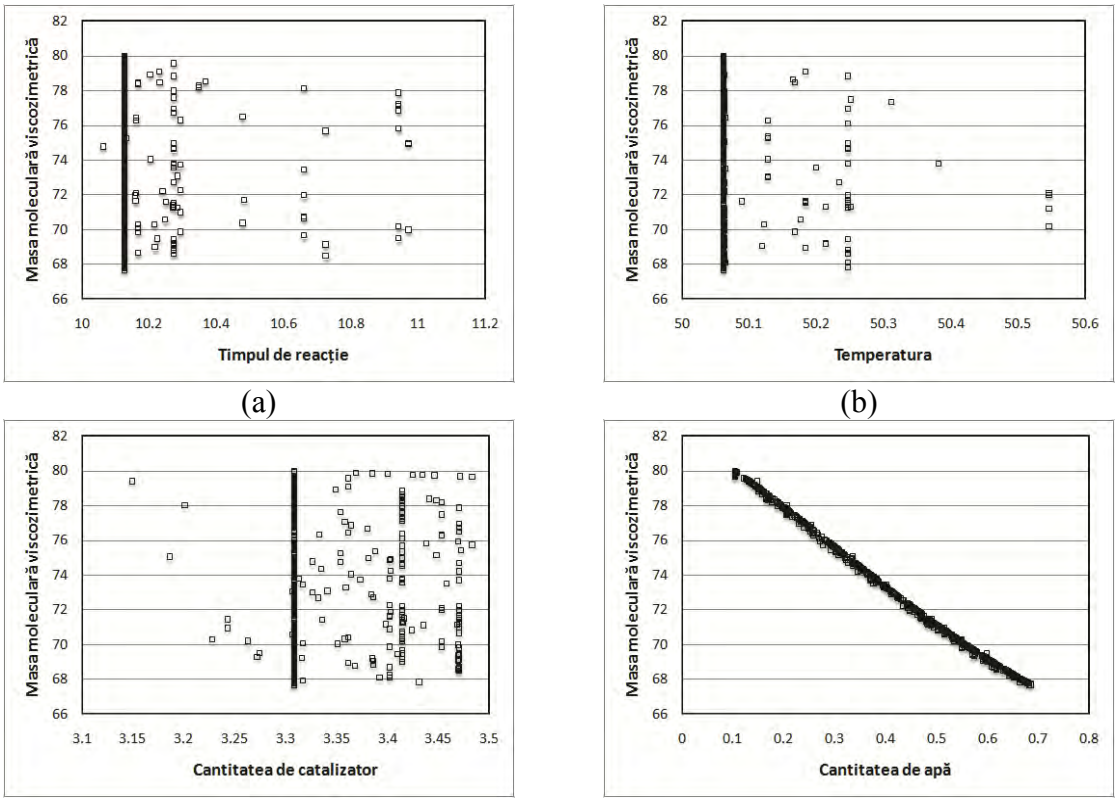


Figura 5.5. Condițiile de reacție: (a) timpul de reacție, (b) temperatura, (c) cantitatea de catalizator și (d) cantitatea de apă pentru obținerea masei moleculare impuse

Frontul optim Pareto obținut prin utilizarea NSGA-II cu valorile optimizate ale parametrilor este prezentat în figura 5.3, unde se poate remarca principala caracteristică a frontului optim Pareto, și anume faptul că un obiectiv este optimizat doar în detrimentul celuilalt.

Figurile 5.4 și 5.5 ilustrează influența condițiilor de reacție asupra conversiei monomerului și, respectiv, asupra masei moleculare viscozimetrice. Aceste variabile de decizie corespund câte unui punct din frontul Pareto prezentat în figura 5.3.

5.4.2 Optimizarea multi-obiectiv a procesului de sinteză a copolimerilor siloxan-siloxan (studiul de caz 2)

5.4.2.2 Problema de optimizare

Optimizarea procesului de sinteză a copolimerilor siloxanici a inclus următoarele elemente:

- Modelul matematic al procesului inclus în procedura de optimizare

Modelul este reprezentat de o rețea neuronală de tip feed-forward cu arhitectura MLP(4:10:2). Cele patru variabile de intrare ale rețelei neuronale au fost temperatura procesului de copolimerizare (T), timpul de reacție (t), cantitate de catalizator (c_{cat}) și compoziția inițială a amestecului de reacție ($f1$). Mărimile de ieșire au fost conversia la sfârșitul reacției (x) și compoziția finală a copolimerului ($F1$).

Modelul neuronal poate fi reprezentat astfel:

$$\text{NN [Intrări: } T, t, c_{cat}, f1\text{; Ieșiri: } x, F1\text{]}. \tag{5.5}$$

- Funcția vectorială multi-obiectiv compusă din două funcții de fitness

$$F(T,t,c_{cat},f1)=(x,(F1-F1d)^2). \tag{5.6}$$

Procedura de optimizare a urmărit maximizarea funcției vectoriale multi-obiectiv prin maximizarea conversiei monomerului (x) şi realizarea unei valori impuse ($F1d$) pentru compoziția copolimerului sintetizat (prin minimizarea diferenței dintre valoarea obținută pentru compoziția copolimerului şi valoarea impusă). Funcția obiectiv a fost formulată vectorial, cele doua obiective fiind tratate simultan în cadrul procedurii de optimizare.

- Variabilele de decizie

Limitările care sunt prezente în reactoarele industriale sunt foarte importante pentru a defini intervalul de variație a parametrilor şi pentru a elimina posibile soluții fără relevanță practică.

Domeniile de variație ale parametrilor (variabilelor de decizie) considerați şi notațiile folosite au fost:

$$\begin{gathered} T\text{: } 20-110^{\circ}\text{C},\ t\text{: } 0.25-2.25 \text{ ore}, \\ c_{cat}\text{: } 0.5-5\ \%, f1\text{: } 0-1, \\ x\text{: } 0-100\ \%\ (90\ \%\ \text{practic}),\ F1\text{: } 0-1. \end{gathered} \tag{5.7}$$

- Metoda folosită pentru rezolvarea problemei de optimizare

Procedura de optimizare a avut ca punct central algoritmul genetic de sortare nedominată NSGA-II (Deb şi colab., 2000) care a fost folosit pentru obținerea frontului optim Pareto şi a variabilelor de decizie corespunzătoare soluțiilor ne-dominate din frontul Pareto (figura 5.7).

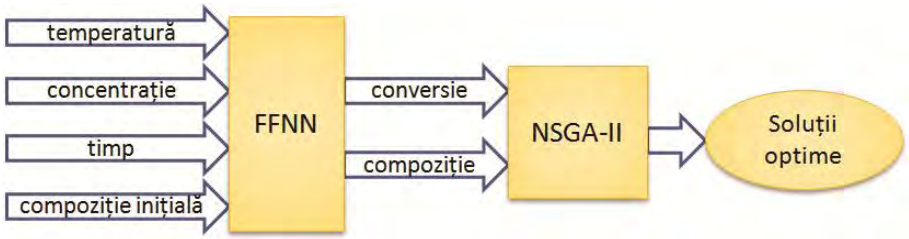


Figura 5.7. Schema procedurii de optimizare aplicată procesului de sinteză a copolimerilor siloxanici

5.4.2.3 Rezultatele optimizării multi-obiectiv

Parametrii algoritmului NSGA-II au fost determinați prin metoda încercărilor succesive şi au fost aleşi pe considerente de echilibru între costul computațional şi performanța algoritmului. Au fost efectuate o serie de simulări folosind diferiți parametri pentru NSGA-II. Dimensiunea turneului a fost setată de la început la 3 cromozomi. Au fost testate diferite valori pentru mărimea populației de cromozomi (50, 100, 200, 500) şi pentru numărul maxim de generații (100, 200, 500, 1000), cu probabilitatea pentru încrucişare setată la valoarea 0.9 şi probabilitatea pentru mutație la valoarea 0.03. O populație de 50 de indivizi şi un număr de 500 de generații s-au dovedit a fi suficiente pentru obținerea unui front optim Pareto cu soluții distribuite aproximativ uniform.

Simulările au continuat folosind dimensiunea populației setată la 50 şi un număr maxim de 500 de generații. Probabilitatea încrucişării a fost variată între 0.1 şi 0.9, iar probabilitatea mutației între 0.01 şi 0.8. S-a observat că se obțin funcții de fitness mai bune pentru o valoare a probabilității încrucişării cuprinsă între 0.7 şi 0.9. Cele mai bune rezultate s-au obținut pentru probabilitatea încrucişării de 0.9 şi probabilitatea mutației de 0.1. Aşadar, valorile optime pentru parametrii algoritmului NSGA-II au fost:

- ✓ mărimea populației (N_{pop}) = 50,
- ✓ numărul maxim de generații (N_{gen}) = 500,
- ✓ probabilitatea de încrucişare (P_C) = 0.9,
- ✓ probabilitatea de mutație (P_M) = 0.1.

Analizând rezultatele obținute în urma simulărilor, se poate constata că pentru *F1* se ating valori apropiate de *F1d*, dar conversia reacției, *x*, este relativ mică. În cazul copolimerilor dimetilmetilvinilsiloxan, valorile relativ scăzute pentru conversia reacției pot fi explicate prin reversibilitatea procesului. Poziția de echilibru depinde de o serie de factori, printre care natura substituentului. Prin urmare, prin creşterea cantității de substituenți, echilibrul se deplasează spre stânga şi, ca rezultat, conversia va fi mai mică.

Parametrii cei mai potriviți s-au dovedit a fi: 50 de indivizi, 500 de generații, rata încrucișării de 0.9 şi rata mutației de 0.1. Frontul Pareto corespunzător acestor valori este ilustrat în figura 5.9.

Examinând frontul Pareto se observă că, trecând de la un punct la altul spre oricare din cele două extremități, o funcție obiectiv se îmbunătățeşte, iar cealaltă se înrăutățeşte; un obiectiv este optimizat în detrimentul celuilalt. Dacă conversia este mai mare atunci compoziția copolimerului se îndepărtează de valoarea dorită (0.7). Pe de altă parte, dacă compoziția copolimerului se apropie de valoarea impusă, conversia reacției începe să scadă.

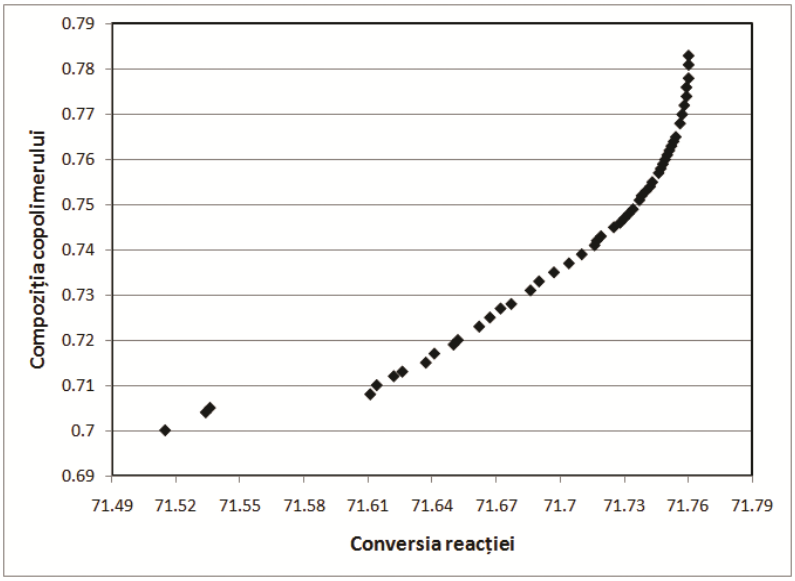


Figura 5.9. Frontul optim Pareto pentru N_{pop}=50, N_{gen}=500, P_C=0.9, P_M=0.1

La sfârşitul procedurii de optimizare, se poate concluziona că o soluție satisfăcătoare este obținerea compoziției impuse de 0.7 pentru copolimer, la o conversie a reacției de 71.52%. Pentru obținerea acestor rezultate, procedura de optimizare a indicat ca fiind condiții optime de reacție:

- timpul de reacție = 1.99 ore,
- temperatura de reacție = 108.64°C,
- concentrația catalizatorului = 4.99%,
- compoziția inițială a amestecului de reacție = 0.82.

6. Optimizarea multi-obiectiv folosind algoritmul evolutiv NSGA-II-RJG

Algoritmul evolutiv multi-obiectiv NSGA-II-RJG a fost creat pornind de la algoritmul NSGA-II, în scopul îmbunătățirii performanțelor acestuia, prin introducerea unui nou operator genetic artificial: operatorul de transpoziție („jumping genes”) adaptat codificării cu numere reale (RJG).

6.1 Operatorul genetic de transpoziție „jumping genes”

Ori de câte ori un singur cromozom este ales pentru o operație de transpoziție, se aplică operatorul de *mutație aleatorie (uniformă)*, pe când în cazul a doi cromozomi diferiți, este exploatat

operatorul de *încrucişare aritmetică* pentru efectuarea operațiilor de transpoziție. Având în vedere că operațiile sunt efectuate cu variabile de decizie reale, nu există nici un motiv de diferențiere între operatorii de transpoziție omologi din codificarea binară (*decupează şi inserează* şi *copiază şi inserează*). Această metodă de aplicare a operatorului „jumping genes" diferă de cel propus de Nawaz Ripon şi colab. (2007) prin faptul că utilizează alte tipuri de mutație şi recombinare, iar cei doi cromozomi care reprezintă operanzii sunt selectați înaintea operațiunii de transpoziție propriu-zise şi nu în cadrul acesteia. Mutația aleatorie folosită ca parte a operatorului de transpoziție este un echivalent al mutației macro-macro utilizată de către Kasat şi Gupta (2003) pentru operatorul „jumping genes" binar.

Figura 6.1 ilustrează implementarea operatorului de transpoziție aplicat unor parametri reali. P_{RJG} reprezintă probabilitatea de efectuare a operației de transpoziție. Aceasta are o valoare cuprinsă între 0 şi 1.

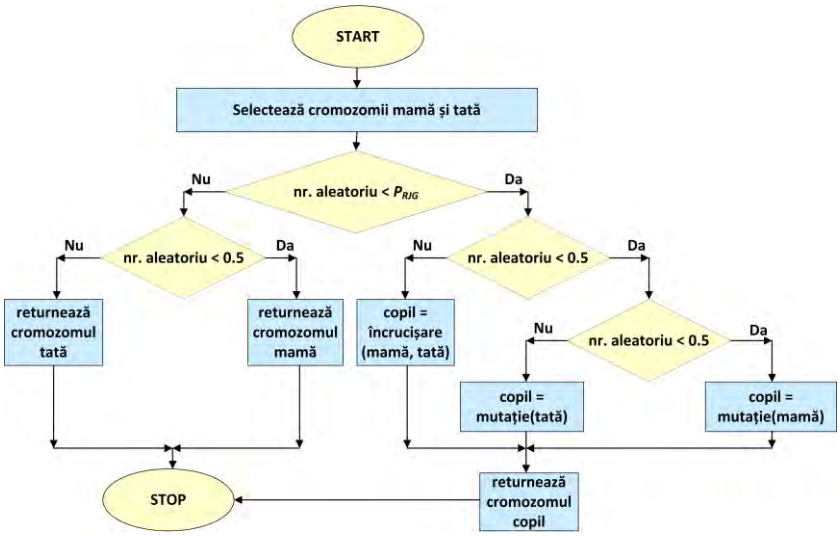


Figura 6.1. Schema logică a operației de transpoziție pentru cromozomi codificați cu numere reale

6.2 Procedura de optimizare multi-obiectiv bazată pe NSGA-II-RJG

Operatorul „jumping genes" a fost inclus în varianta pentru cromozomi codificți cu numere reale a algoritmului NSGA-II, detaliată în capitolul 5, pentru a contracara tendința de scădere a diversității soluțiilor ce apare datorită folosirii elitismului. Operatorul RJG a fost introdus în algoritmul NSGA-II după operația de selecție a părinților pentru crearea unei noi populații şi înaintea operațiilor de încrucişare şi mutație.

NSGA-II-RJG păstrează caracteristicile algoritmului NSGA-II, şi anume codificarea cu numere reale a soluțiilor şi folosirea operatorilor genetici potriviți acestui tip de codificare (încrucişarea aritmetică şi mutația aleatorie). Pentru alegerea unei noi populații de cromozomi se foloseşte selecția bazată pe ranguri de ne-dominanță Pareto, iar desemnarea părinților pentru crearea de noi cromozomi se face aplicând selecția prin turneu. Criteriul de oprire a căutării este acelaşi ca în cazul algoritmului NSGA-II: realizarea unui număr prestabilit de generații de cromozomi (soluții).

6.3 Optimizarea multi-obiectiv a procesului de preparare a nanoparticulelor polimerice cu surfactanți siliconici folosind NSGA-II-RJG (studiu de caz)

6.3.1 Problema de optimizare

Scopul optimizării a fost minimizarea, în acelaşi timp, a diametrului mediu (*Zave*) al particulelor de surfactant siliconic şi a indicelui de polidispersitate (*PDI*), prin determinarea

valorilor optime pentru variabilele de decizie: concentrația de surfactant (*Cs*), concentrația de polimer (*Cp*) şi temperatura de stocare (*T*). Funcția obiectiv vectorială a algoritmului NSGA-II-RJG a fost calculată cu două rețele neuronale optimizate, corespunzătoare celor două funcții de fitness atașate fiecărei soluții considerate de-a lungul procedurii de optimizare multi-obiectiv. Rețelele neuronale au fost determinate folosind metodologia de optimizare a parametrilor unei rețele neuronale cu propagare înainte, descrisă în capitolul 4. Modelele neuronale rezultate au fost MLP(3:9:1) pentru modelarea diametrului mediu funcție de concentrația de surfactant, concentrația de polimer şi temperatura de stocare şi MLP(2:6:1) pentru modelarea indicelui de polidispersitate funcție de concentrațiile de surfactant şi polimer (Furtună şi colab., 2011c).

Funcția multi-obiectiv vectorială maximizată de-a lungul procesului de optimizare a fost construită din două funcții de fitness determinate de cele două obiective divergente:

$$F(Cs, Cp, T) = (-Zave, -PDI) \tag{6.1}$$

Limitele variabilelor de decizie, impuse prin setul de date experimentale folosite la modelarea procesului de sinteză, au fost:

$$Cs\text{: } 0.5 - 2 \text{ gL}^{-1},\ Cp\text{: } 0.5 - 2\%,\ T\text{: } 4^{\circ}\text{C sau } 40^{\circ}\text{C} \tag{6.2}$$

6.3.2 Rezultatele optimizării multi-obiectiv

Procedura de optimizare multi-obiectiv bazată pe NSGA-II-RJG a pornit de la algoritmul NSGA-II, pentru care s-au determinat cele mai potrivite valori ale parametrilor de control. Apoi a fost introdus noul operator de transpoziție, urmărindu-se totodată găsirea celei mai bune probabilități pentru efectuarea acestei operații. Măsurarea timpului de convergență şi calcularea metricilor de spațiere şi de acoperire a setului pentru ambii algoritmi s-a dovedit edificatoare în acest caz.

Criteriul utilizat în selectarea celor mai bune valori pentru parametrii algoritmilor NSGA-II şi NSGA-II-RJG s-a bazat pe cel mai convenabil compromis care ar putea fi făcut între gradul de dominanță al frontului Pareto rezultat (calculat cu metrica de acoperire a setului) şi performanța soluțiilor obținute (cuantificată cu indicele *sol_perf* definit prin ecuația 5.2).

Au fost efectuate o serie de simulări pentru a determina cea mai bună combinație de valori pentru mărimea populației (N_{pop}) şi numărul maxim de generații (N_{gen}). Pentru numărul de indivizi dintr-o populație au fost testate valorile 10, 50, 100, 300 şi 500, iar pentru numărul maxim de generații, valorile 50, 100, 300, 500 şi 1000, probabilitățile de încrucişare şi mutație fiind menținute la 0.9 şi, respectiv, 0.03 pe parcursul simulărilor.

După aplicarea tehnicii de evaluare a performanțelor pe baza metricii de acoperire a setului, cei mai buni parametri s-au dovedit a fi N_{pop} = 500 şi N_{gen} = 1000. Pentru aceste valori însă, metrica *sol_perf* a fost considerabil mai mare în comparație cu valorile sale obținute cu alte combinații între mărimea populației şi numărul maxim de generații. Prin urmare, s-a intersectat clasificarea parametrilor N_{pop} şi N_{gen} din punct de vedere al indicelui *sol_perf* cu ierarhizarea făcută după metrica de acoperire a setului şi a rezultat combinația N_{pop} = 500 şi N_{gen} = 50 (*Ct* = 26.92 s, *S* = 0.0040, *sol_perf* = 0.0067). Având stabilite cele mai potrivite valori pentru mărimea populației şi, respectiv, numărul maxim de generații, simulările au continuat cu testarea diferitelor valori pentru probabilitatea de încrucişare (0.1, 0.3, 0.5, 0.7 şi 0.9) şi probabilitatea de mutație (0.01, 0.05, 0.2, 0.5 şi 0.8).

De-a lungul simulărilor s-a observat că timpul de convergență a scăzut odată cu creşterea probabilității de mutație, ceea ce poate fi explicat prin faptul că operația de mutație previne blocarea procesului de căutare într-un minim local.

Procedura de comparație bazată pe metrica de acoperire a setului a evidențiat faptul că frontul Pareto obținut pentru P_C = 0.1 şi P_M = 0.01 a dominat toate fronturile Pareto rezultate pentru celelalte combinații de valori testate. Cu toate acestea, indicele de performanță a soluțiilor obținute pentru o probabilitate de încrucişare de 0.1 şi o probabilitate de mutație de 0.01 a fost mai mare decât valorile realizate cu o mare parte din celelalte combinații de valori. Aşadar, în acest caz,

alegerea celor mai potrivite probabilități de recombinare şi mutație a ținut cont doar de indicele *sol_perf*. Cea mai mică valoare *sol_perf* (0.0053) a fost obținută atât pentru P_C = 0.3 şi P_M = 0.8 cât şi pentru P_C = 0.9 şi P_M = 0.8. Având în vedere că fronturile Pareto determinate cu cele două perechi de valori au dominat 17.4% şi, respectiv, 24% din soluțiile celui mai dominant front Pareto, probabilitatea de încrucişare de 0.9 şi probabilitatea de mutație de 0.8 au fost considerate cele mai potrivite (*Ct* = 13.20 s, *S* = 0.0040).

Operatorul RJG a fost introdus după determinarea celor mai adecvați parametri pentru algoritmul NSGA-II, iar cea mai indicată probabilitate pentru operația de transpoziție a fost determinată prin aceeaşi metodă folosită în cazul parametrilor algoritmului NSGA-II. Au fost testate valorile 0.1, 0.3, 0.5, 0.7 şi 0.9 pentru P_{RJG} într-o serie de simulări.

Deoarece metrica de acoperire a setului a indicat că probabilitatea cea mai bună pentru operația de transpoziție a fost de 0.1 şi indicele de performanță a soluțiilor corespunzător acestei valori a fost minim (*sol_perf* = 0.0055, *Ct* = 13.81 s, *S* = 0.0041), P_{RJG} utilizat în procedura de optimizare multi-obiectiv cu NSGA-II-RJG a fost 0.1.

După introducerea operatorului de transpoziție adaptat codificării cu numere reale în algoritmul NSGA-II cu parametrii de control îmbunătățiți, cei doi algoritmi (NSGA-II şi NSGA-II-RJG) au fost comparați din punct de vedere al rezultatelor obținute în urma optimizării procesului de sinteză a nanoparticulelor polimerice.

Prin calcularea metricilor de acoperire a setului pentru fronturile Pareto obținute cu cei doi algoritmi, s-a constatat că vectorul soluție oferit de NSGA-II domină vectorul soluție obținut folosind NSGA-II-RJG, deoarece CS (X_NSGA-II, X_NSGA-II-RJG) > CS (X_NSGA-II-RJG, X_NSGA-II). După cum reiese din ecuațiile (6.1) şi (6.2), vectorul soluție obținut cu NSGA-II domină 33.6% dintre soluțiile din frontul Pareto determinat cu NSGA-II-RJG, pe când cel obținut cu NSGA-II-RJG domină doar 12.6% din elementele vectorului soluție realizat cu NSGA-II.

$$CS(X\_NSGA-II-RJG, X\_NSGA-II) = 0.126 \tag{6.1}$$

$$CS(X\_NSGA-II, X\_NSGA-II-RJG) = 0.336 \tag{6.2}$$

Timpul de convergență şi metrica de spațierea realizate cu NSGA-II-RJG (*Ct* = 13.81 s şi *S* = 0.0041) au fost puțin mai mari decât cele obținute cu NSGA-II (*Ct* = 13.20 s, *S* = 0.0040). Prin urmare, se poate afirma că, pentru problema particulară abordată în acest studiu de caz, introducerea operatorului de transpoziție nu conduce la îmbunătățirea diversității soluțiilor sau la scăderea timpului de convergență al algoritmului. Acest rezultat întăreşte concluzia care reiese din studiul comparativ realizat de către Nawaz Ripon şi colab. (2007), şi anume că eficiența operatorului „jumping genes" adaptat codificării cu numere reale depinde de problema la care este aplicat.

Figura 6.3 prezintă o comparație a fronturilor Pareto obținute cu cei doi algoritmi. Discontinuitățile din fronturile Pareto se datorează regiunilor deconectate din spațiul de căutare a soluțiilor.

Deoarece prin simpla introducere a operatorului de transpoziție în cadrul algoritmului NSGA-II cu parametrii de control optimizați şi prin optimizarea probabilității operației de transpoziție nu s-au obținut rezultate mai bune, simulările au continuat cu determinarea celor mai potriviți parametri de control pentru algoritmul NSGA-II-RJG. Astfel, setând probabilitatea de transpoziție la 0.5 (valoarea cea mai utilizată în studiile asupra operatorului „jumping genes"), ceilalți parametri de control au fost testați în acelaşi mod ca şi cei ai algoritmului NSGA-II, iar rezultatele au fost evaluate utilizând aceleaşi metrici de performanță (timpul de convergență, metrica de spațiere şi metrica de acoperire a setului).

Corelând ierarhizarea perechilor de parametri N_{pop} şi N_{gen} ai algoritmului NSGA-II-RJG din punct de vedere al indicelui *sol_perf* cu cea realizată considerând metrica de acoperire a setului, valorile 100 şi, respectiv, 300 au reieşit ca fiind cele mai bune (*Ct* = 5.52 s, *S* = 0.0041, *sol_perf* = 0.0047).

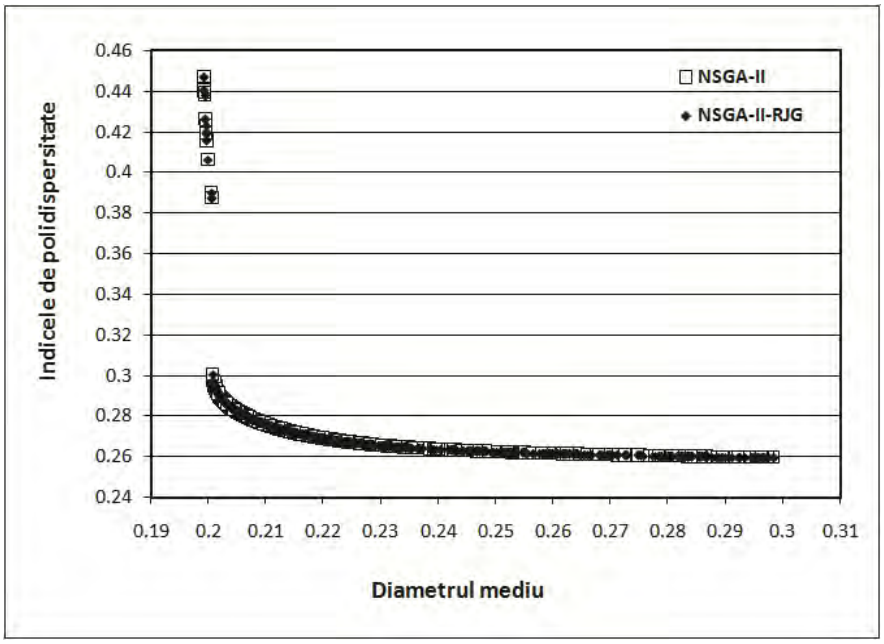


Figura 6.3. Fronturile Pareto obținute cu NSGA-II şi NSGA-II-RJG

Procedura de evaluare bazată pe metrica de acoperire a setului a indicat faptul că o probabilitate de încrucişare de 0.9 şi o probabilitate de mutație de 0.8 au condus la obținerea celui mai dominant front Pareto. Cu toate acestea, valorile 0.9 şi 0.5, care au corespuns următorului vector soluție în ordine descrescătoare a dominanței, au fost alese pentru cei doi parametri de control, deoarece indicele *sol_perf* obținut în acest caz a fost mai mic (*sol_perf* = 0.0044, *Ct* = 2.03 s, *S* = 0.0042).

Procesul de determinarea a celei mai bune probabilități de transpoziție a oferit valori foarte similare pentru metrica de acoperire a setului. Prin urmare, alegerea celei mai potrivite valori pentru P_{RJG} a fost făcută pe baza indicelui de performanță *sol_perf* şi, astfel, a fost aleasă o probabilitate de 0.3 pentru operația de transpoziție (*Ct* = 2.09 s, *S* = 0.0036, *sol_perf* = 0.0038).

Algoritmul NSGA-II-RJG optimizat a avut un timp convergență mai bun (*Ct* = 2.0938) şi o diversitate a soluțiilor mai mare (*S* = 0.0036) decât algoritmul NSGA-II optimizat (*Ct* = 13.2031 s, *S* = 0.0040). Cu toate acestea, frontul Pareto obținut cu NSGA-II l-a dominat pe cel obținut cu NSGA-II-RJG, după cum arată metrica de acoperire a setului (ecuațiile (6.3) şi 6.4)).

$$CS(X\_NSGA-II-RJG, X\_NSGA-II) = 0.042 \quad (6.3)$$

$$CS(X\_NSGA-II, X\_NSGA-II-RJG) = 0.51 \quad (6.4)$$

Concluzionând, se poate afirma că algoritmul NSGA-II-RJG cu N_{pop} = 100, N_{gen} = 300, P_C = 0.9, P_M = 0.5 şi P_{RJG} = 0.3 a avut o performanță mai bună decât algoritmul NSGA-II cu N_{pop} = 500, N_{gen}, = 50, P_C = 0.9, şi P_M = 0.8 atunci când au fost aplicați optimizării multi-obiectiv a procesului de sinteză a nanoparticulelor polimerice cu surfactanți siliconici. O comparație a fronturilor Pareto obținute cu cei doi algoritmi având valori îmbunătățite pentru parametrii de control este dată în figura 6.4.

Temperatura de stocare a influențat doar diametrul mediu, pe când concentrația de surfactant şi concentrația de polimer au avut un impact asupra ambelor obiective conflictuale ale optimizării. De exemplu, o concentrație de surfactant de 2 gL^{-1}, o concentrație de polimer cuprinsă între 0.9% şi 1.02% şi o temperatură de stocare de 4˚C au condus la obținerea unor valori minime pentru diametrul mediu, producând punctele de la extremitatea superioară a frontului Pareto afişat în figura 6.4 (pentru NSGA-II-RJG). Punctele din regiunea de mijloc a aceluiaşi front Pareto au fost realizate folosind o concentrație de surfactant între 0.6 şi 0.8 gL^{-1}, o concentrație de polimer între 0.5% şi 0.63% şi o temperatură de stocare de 40˚C. Minimizarea indicelui de polidispersitate a fost realizată cu o concentrație de surfactant între 0.5 şi 0.6 gL^{-1}, o concentrație de polimer între 0.55% şi 0.67%

şi o temperatură de stocare de 40˚C, creând astfel extremitatea inferioară a frontului Pareto (figura 6.4). Cea mai mică valoare pentru indicele de polidispersitate a fost obținută utilizând o concentrație de surfactant de 0.5 gL^{-1}, o concentrație de polimer de 0.53% şi o temperatură de depozitare de 4˚C.

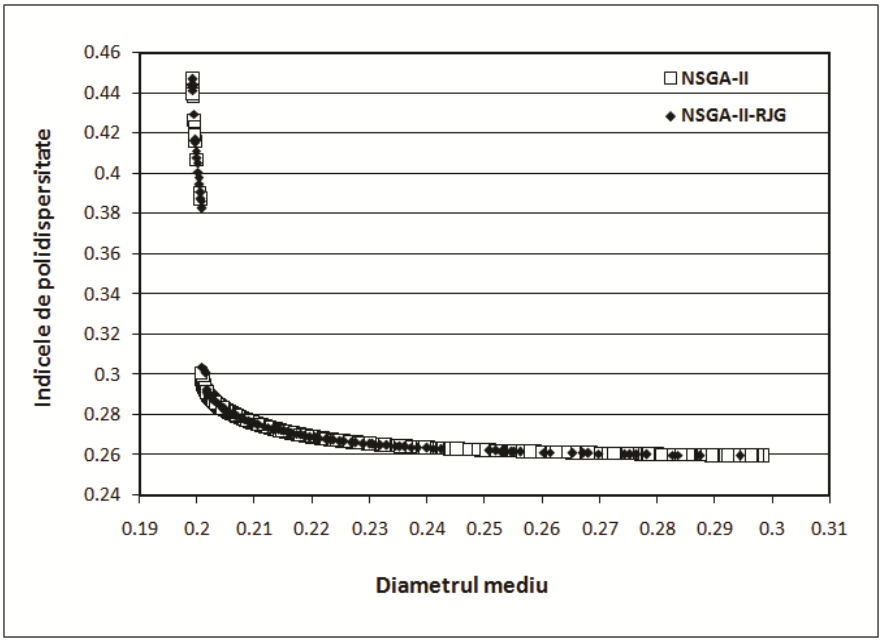


Figura 6.4. Fronturile Pareto obținute cu NSGA-II şi NSGA-II-RJG optimizat

7. Modelarea neuro-evolutivă folosind hiper-euristica NSGA-II-QN

7.1 Hiper-euristica evolutivă NSGA-II-QN

Hiper-euristica evolutivă NSGA-II-QN a fost proiectată în scopul dezvoltării şi optimizării unor modele neuronale care să ofere performanțe maxime la antrenare şi testare şi să aibă, în acelaşi timp, o complexitate structurală cât mai scăzută. Algoritmul a fost implementat folosind limbajul de programare C#, rezultând o aplicație software originală (Popa - Furtună şi Curteanu, 2010).

Hiper-euristica evolutivă NSGA-II-QN, pe baza căreia a fost construită aplicația software, este compusă din algoritmul evolutiv multi-obiectiv NSGA-II şi algoritmul de căutare locală Quasi-Newton (QN). Algoritmul Quasi-Newton a fost integrat în algoritmul NSGA-II şi a fost utilizat pentru antrenarea rețelelor neuronale supuse optimizării, la fiecare iterație a algoritmului evolutiv.

Algoritmul NSGA-II a fost folosit pentru obținerea unui set de compromisuri optime între cele două obiective conflictuale urmărite: erori cât mai mici la antrenare şi testare, pe de o parte, şi un număr cât mai scăzut de ponderi ale rețelei neuronale, pe de altă parte. Setul de soluții ne-dominate (set de soluții optime Pareto) ce corespunde compromisurilor optime găsite de NSGA-II reprezintă soluția problemei de optimizare multi-obiectiv a rețelelor neuronale.

Algoritmul de antrenare Quasi-Newton este un algoritm de căutare locală care determină cel mai bun set de ponderi sinaptice, urmărind minimizarea erorii la antrenare. Acest algoritm a fost preferat față de algoritmul back-propagation, deoarece este mai rapid şi mai eficient în rezolvarea problemelor de aproximare de funcții. Gradienții funcției de eroare au fost calculați prin tehnica de propagare înapoi. Criteriul de oprire a procesului de antrenare a fost scăderea gradientului erorii sub o valoare prestabilită.

Paşii care descriu principiul de funcționare a hiper-euristicii evolutive propuse sunt enumerați în organigrama prezentată în figura 7.1. Săgeata punctată din diagramă a fost folosită pentru a exprima faptul că tranziția descrisă are loc doar o singură dată în algoritm şi, prin urmare, aceasta nu creează o buclă infinită.

În figura 7.1, N_{pop} reprezintă mărimea populației, N_{gen} – numărul maxim de generații, N_T – dimensiunea turneului, P_C – probabilitatea încrucișării și P_M – probabilitatea mutației. *TolG* este toleranța gradientului folosită în algoritmul Quasi-Newton, *maxTrIter* este numărul maxim de iterații pentru algoritmul de antrenare, N_{VD} – numărul de variabile de decizie ale problemei de optimizare, N_{FF} – numărul de funcții de fitness, *Ni* – numărul de intrări ale rețelei neuronale, *No* – numărul de ieșiri ale rețelei neuronale, *Ntr* – numărul de vectori de antrenare și *Nts* este numărul de vectori de test folosiți la dezvoltarea rețelei neuronale. *Hn_total* este numărul total de neuroni ascunși ai rețelei și *perf_index* este indicele de performanță al acesteia, calculat cu formula 4.18.

Hiper-euristica evolutivă NSGA-II-QN a fost adaptată pentru două tipuri diferite de rețele neuronale: rețele neuronale de tip stivă, varianta denumită NSGA-II-QNSNN, și rețele neuronale simple, varianta notată NSGA-II-QNN.

7.2 Modelarea neuro-evolutivă cu rețele neuronale de tip stivă

Ca nucleu al hiper-euristicii evolutive NSGA-II-QNSNN, algoritmul NSGA-II efectuează o căutare globală în spațiul soluțiilor și oferă un întreg front optim Pareto sub forma unei serii de compromisuri între două obiective contradictorii: maximizarea capacității de generalizare a rețelei neuronale de tip stivă și minimizarea complexității sale structurale. Aceste două obiective conflictuale sunt atinse prin determinarea numărului optim de rețele neuronale individuale agregate în stivă, a ponderilor optime asociate fiecărei ieșiri a rețelelor individuale din stivă și a numărului optim de neuroni ascunși din fiecare rețea individuală ce compune stiva. Performanța rețelei neuronale de tip stivă este evaluată la fiecare iterație a hiper-euristicii NSGA-II-QNSNN prin calcularea indicelui *perf_index*. Așadar, funcția multi-obiectiv maximizată în cadrul procedurii de optimizare este compusă din două funcții de fitness și are următoarea expresie:

$$f(NNno, w_{jk}, Hn_k) = (-(1 - perf\_index), -\sum_{k=0}^{NNno} Hn_k), \tag{7.1}$$

în care *NNno* este numărul de rețele neuronale individuale din stivă, w_{jk} este ponderea corespunzătoare ieșirii j a rețelei numărul k din stivă, Hn_k este numărul total de neuroni ai rețelei individuale k.

Ieșirea efectivă a rețelei neuronale stivă este determinată prin utilizarea seturilor de date de antrenare și de test și prin aplicarea mediei ponderate descrisă în ecuația (7.2). Ponderile utilizate în ecuația (7.2) sunt ponderile atașate fiecărei ieșiri a rețelelor individuale din stivă, optimizate prin algoritmul NSGA-II. Media ponderată a fost preferată față de media aritmetică simplă pentru că aceasta favorizează rețelele neuronale individuale cu performanțe mai bune, îmbunătățind, în acest fel, nivelul global de performanță al stivei.

$$y_j = \sum_{k=0}^{NNno} w_{jk} \cdot y_{jk} \tag{7.2}$$

y_j este ieșirea numărul j a rețelei neuronale stivă și y_{jk} este ieșirea j a rețelei neuronale k din stivă. Rețeaua stivă are același număr de intrări și de ieșiri ca și fiecare rețea componentă.

Structura cromozomilor cu care lucrează algoritmul evolutiv multi-obiectiv din cadrul hiper-euristicii NSGA-II-QNSNN este alcătuită din gene codificate cu numere reale, reprezentând variabilele de decizie: numărul de rețele neuronale individuale din stivă (maxim 5), ponderile pentru fiecare ieșire a rețelelor individuale combinate în stivă și numărul de neuroni ascunși al fiecărei rețele neurale. Constrângerile variabilelor de decizie au fost incluse în codificarea lor. Structura cromozomului poate fi observată în tabelul 7.1, unde notațiile folosite sunt aceleași ca în ecuația (7.1).

Pentru a evita efortul de calcul al hiper-euristicii evolutive numărul de rețele neuronale individuale incluse în stivă a fost limitat la 5. Ponderile pentru fiecare ieșire a rețelelor individuale combinate în stivă au avut valori între 0 și 100%.

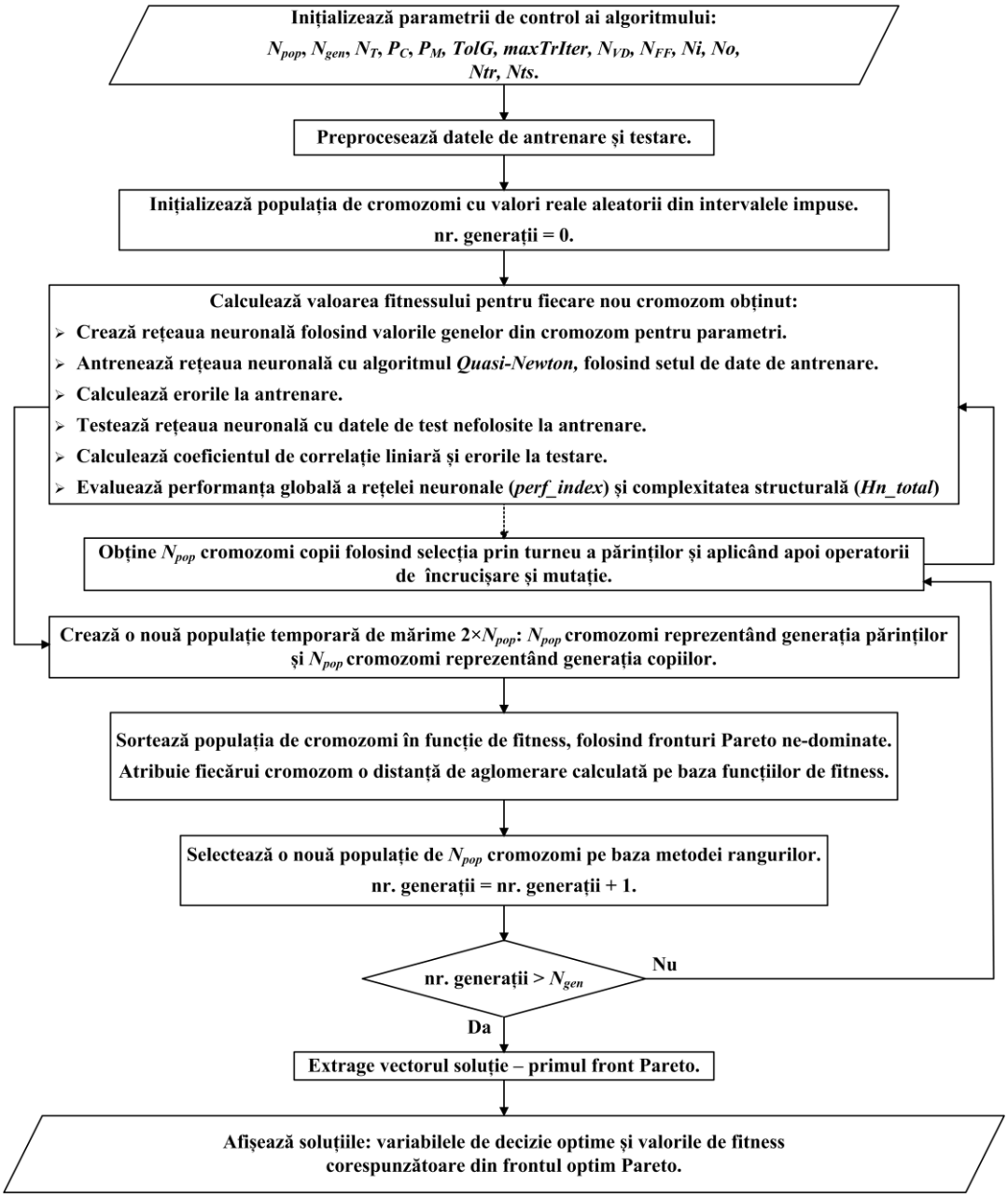


Figura 7.1. Schema logică a hiper-euristicii evolutive NSGA-II-QN

Tabelul 7.1. Structura cromozomului folosit de NSGA-II-QNSNN

$NNno$	w_{j1}	w_{j2}	w_{j3}	w_{j4}	w_{j5}	Hn_1	Hn_2	Hn_3	Hn_4	Hn_5

În stabilirea numărului maxim permis de neuroni ascunşi pentru fiecare rețea neuronală din stivă s-a ținut seama de considerațiile practice enunțate în capitolul 4 (formulele 4.4, 4.5, 4.6, 4.7).

Fiecare rețea neuronală din stivă a avut un singur strat ascuns de neuroni, limitând astfel complexitatea rețelei neuronale stivă şi, implicit, timpul şi memoria necesare hiper-euristicii evolutive pentru a converge la o soluție optimă. Tangenta hiperbolică a fost funcția de activare a stratului ascuns în fiecare rețea neuronală individuală din stivă, fiind preferată datorită neliniarității şi diferențiabilității sale. Datele de intrare din baza de date experimentale au fost scalate în intervalul [-0.9, 0.9] pentru tangenta hiperbolică, folosind formula 4.1.

7.3 Modelarea neuro-evolutivă cu rețele neuronale simple

Varianta hiper-euristicii NSGA-II-QN adaptată pentru dezvoltarea unor rețele neuronale simple (NSGA-II-QNN) diferă de varianta pentru rețele neuronale stivă prin variabilele de decizie care sunt optimizate, obiectivele conflictuale fiind aceleaşi.

NSGA-II-QNN urmăreşte optimizarea următoarelor variabile de decizie: numărul de straturi ascunse, numărul de neuroni din fiecare strat ascuns, funcțiile de activare ale straturilor ascunse şi ale stratului de ieşire şi ponderile sinaptice (incluzând biasurile) ale rețelei neuronale. Astfel, structura unei soluții (cromozom) este cea ilustrată în tabelul 7.2, în care N_H reprezintă numărul de straturi ascunse din rețea, Hn_i – numărul de neuroni din cel de-al i-lea strat ascuns (i = 1 sau 2), F_{Hi} – funcția de activare al celui de-al i-lea strat ascuns şi F_O – funcția de activare a stratului de ieşire.

Tabelul 7.2. Structura cromozomului folosit de NSGA-II-QNN

N_H	Hn_1	Hn_2	F_{H1}	F_{H2}	F_O

Complexitatea structurală a unei rețele este cuantificată prin numărul de ponderi sinaptice ale acesteia. Funcțiile obiectiv vectoriale ale NSGA-II-QNN sunt calculate cu formula (7.3), în care *Nw* este numărul de ponderi.

$$f(N_H, Hn_k, F_{Hi}, F_O) = (-(1 - perf\_index), -Nw), \tag{7.3}$$

În scopul limitării timpului de execuție al hiper-euristicii NSGA-II-QNN şi a memoriei necesare procesului de căutare, numărul de straturi ascunse a fost maxim 2, numărul de neuroni din fiecare strat ascuns a fost cuprins între 1 şi 20, pentru funcțiile de activare ale straturilor ascunse s-au folosit tangenta hiperbolică şi sigmoida unipolară (logistica), iar pentru stratul de ieşire au fost testate funcțiile liniară, tangentă hiperbolică şi logistică.

7.4 Studii de caz pentru aplicarea modelării neuro-evolutive a unor procese din ingineria chimică folosind hiper-euristica NSGA-II-QN

7.4.1 Modelarea neuro-evolutivă a procesului de sinteză a unor hidrogeluri multicomponente pe bază de poliacrilamidă (studiul de caz 1)

7.4.1.1 Premisele cercetării

Acest studiu de caz urmăreşte dezvoltarea şi optimizarea unei rețele neuronale de tip stivă utilizată pentru modelarea procesului de sinteză a unor hidrogeluri multicomponente pe bază de poliacrilamidă. Acesta este un proces chimic foarte complex şi nu se cunoaşte un model fenomenologic care să reproducă legile fizice şi chimice ce-l guvernează, rețelele neuronale fiind recomandate ca alternative pentru modelarea acestuia.

Procedeul de obținere a datelor experimentale folosite în acest studiu a fost descris în două lucrări anterioare (Curteanu şi colab., 2008a; Curteanu şi colab., 2009) care prezintă, de asemenea, rezultatele obținute prin modelarea procesului de polimerizare cu o rețea neuronală simplă. O altă abordare pentru modelarea sintezei hidrogelurilor multicomponente pe bază de poliacrilamidă a fost raportată de către Leon şi colab. (2010). Aşadar, studiul de caz 1 a reprezentat o continuare a cercetărilor precedente, urmărindu-se îmbunătățirea rezultatelor modelării neuronale prin găsirea unei metode eficiente de dezvoltare şi optimizare a rețelelor neuronale de tip stivă.

7.4.1.2 Baza de date experimentale

Baza de date experimentale a fost obținută în urma sintetizării prin reacții simultane de polimerizare/reticulare a unor hidrogeluri mono- şi multicomponente pe bază de poliacrilamidă. Parametrii de interes în acest studiu experimental (Curteanu şi colab., 2009) au fost cantitatea de polimer reticulat obținută, gradul de umflare şi constanta vitezei de umflare.

Baza de date experimentale disponibilă a constat din 178 de date care au fost amestecate şi apoi împărţite în 134 pentru antrenarea reţelei neuronale şi 44 pentru testare. În scopul evitării supra-antrenării, reţelele neuronale individuale din stivă au fost instruite cu seturi de date diferite din punct de vedere a ordinii acestora. Datele utilizate pentru antrenarea şi testarea reţelei neuronale stivă au fost scalate în conformitate cu funcţiile de activare folosite pentru straturile ascunse şi de ieşire ale reţelelor individuale (formula 4.1).

Au fost luate în considerare şapte variabilele de intrare pentru fiecare reţea neuronală individuală: *CM* (concentraţia de monomer), *CI* (concentraţia de iniţiator), *CA* (concentraţia agentului de reticulare), *PI* (cantitatea de polimer de incluziune), *T* (temperatura), *t* (timpul de reacţie) şi tipul de polimer inclus (*Pt*), codificat ca 1 - nici un polimer adăugat, 2 - amidon, 3 - poli (alcool vinilic) (PVA) şi 4 - gelatină. Ieşirile modelelor neuronale şi, implicit, ieşirile reţelei neuronale stivă, au fost η (randamentul în polimer reticulat) şi α (gradul de umflare). Astfel, modelarea cu reţele neuronale a stabilit influenţa condiţiilor de reacţie asupra randamentului reacţiei şi gradului de umflare.

Setul de date experimentale poate fi vizualizat în tabelele 1 - 4 din lucrarea elaborată de Curteanu şi colab. (2009). Intervalele de valori pentru fiecare intrare a reţelei neuronale au fost: [1, 4] pentru tipul de polimer inclus, [0.23, 3.15] mol/l pentru concentraţia de monomer, [0.06, 9.39] mol/l pentru concentraţia de iniţiator, [0.5, 12.33] mol/l - concentraţia agentului de reticulare, [0, 1.5] g - cantitatea de polimer de incluziune, [29, 64]°C - temperatura şi [1, 7] h pentru timpul de reacţie. Domeniile de valori pentru cele două ieşiri ale reţelei neuronale au fost: [0, 98.15]% pentru randamentul în polimer reticulat şi [0, 126] pentru gradul de umflare.

7.4.1.3 Rezultatele modelării neuro-evolutive

Parametrii modelaţi în acest studiu au fost randamentul reacţiei, care caracterizează productivitatea procesului de sinteză a hidrogelurilor multicomponente pe bază de poliacrilamidă, şi gradul de umflare, o proprietate a hidrogelurilor. Predicţia acestor parametri este importantă pentru practica experimentală, fiind utilă pentru planificarea sau chiar substituirea experimentelor consumatoare de timp şi materiale.

Aplicând algoritmul NSGA-II-QNSNN, au fost efectuate o serie de simulări şi teste pentru dezvoltarea şi optimizarea unei reţele neuronale de tip stivă.

Performanţa reţelei neuronale stivă rezultate de-a lungul procedurii de optimizare a fost testată, la fiecare iteraţie a algoritmului evolutiv, pentru modelarea şi predicţia randamentului polimerizării şi a gradului de umflare maxim în corelaţie cu condiţiile de reacţie iniţiale (concentraţia de monomer, concentraţia de iniţiator, concentraţia agentului de reticulare, cantitatea de polimer de incluziune, timpul de reacţie, temperatura şi tipul de polimer inclus).

Figura 7.3 prezintă structura reţelei neuronale de tip stivă utilizată în modelarea neuro-evolutivă (Furtună şi colab., 2011d).

Numărul de reţele neuronale din stivă a fost de maxim 5, iar numărul de neuroni ascunşi din fiecare reţea a fost limitat la intervalul [2, 15] din considerente legate de timp şi memorie. Aceste limitări au fost impuse pe baza unor serii de teste efectuate cu hiper-euristica evolutivă. În urma acestora, s-a observat că performanţa reţelei neuronale stivă s-a îmbunătăţit uşor atunci când au fost utilizaţi mai mult de 15 neuroni în straturile ascunse ale reţelelor individuale, însă timpul necesar pentru ca algoritmul să conveargă a fost considerabil mai mare decât atunci când au fost folosiţi mai puţini neuroni ascunşi. Mai mult, rezultatele optimizării multi-obiectiv au arătat că, de obicei, 2 sau 3 reţele neuronale în stivă au fost suficiente pentru a furniza o precizie satisfăcătoare şi o capacitate de generalizare bună. Acest fapt este susţinut de rezultatele prezentate în tabelul 7.6. Aplicaţia software a fost testată pe un procesor "Intel Core 2 CPU 6600", la 2.40 GHz şi memorie RAM de 2 GB.

Arhitectura de tip perceptron multistrat (MLP) a fost aleasă pentru toate reţelele neuronale din stivă datorită faptului că aceasta s-a dovedit a fi cea mai potrivită pentru modelarea sintezei hidrogelurilor, când a fost comparată cu alte tipuri de arhitecturi (reţele neuronale feed-forward

generalizate, rețele neuronale modulare, rețele neuronale de tip Jordan Elman), într-unul din studiile noastre anterioare (Leon şi colab., 2010).

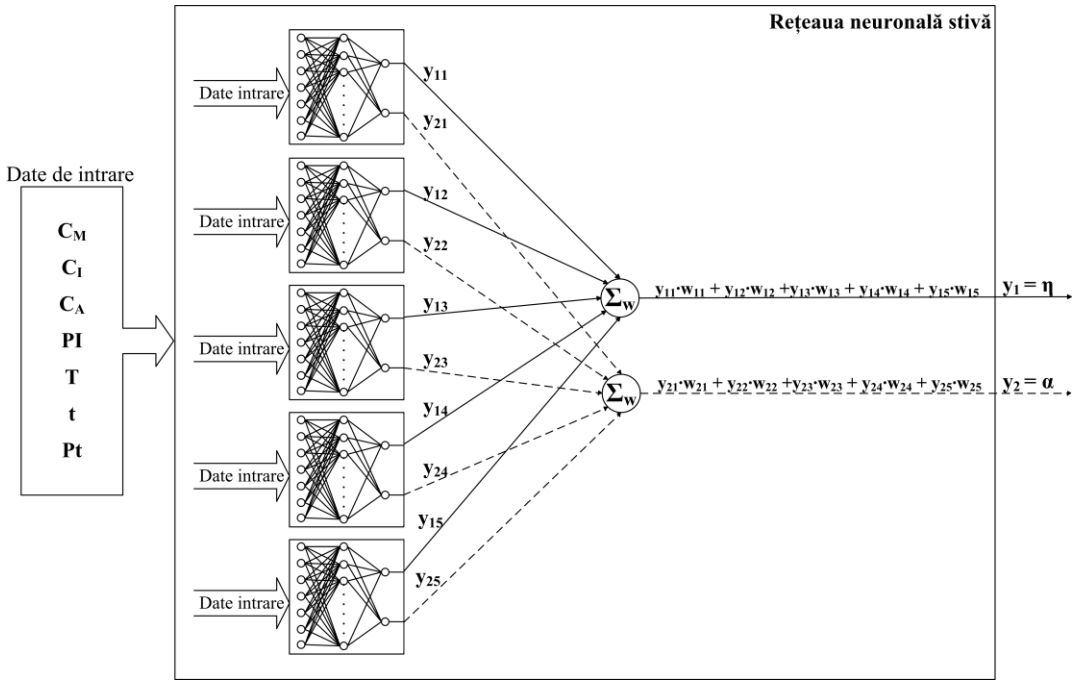


Figura 7.3. Structura rețelei neuronale de tip stivă utilizată la modelarea sintezei unor hidrogeluri multicomponente

Pentru a nu complica structura cromozomului utilizat în NSGA-II, ce ar duce la creşterea timpului de calcul, funcțiile de activare pentru neuronii din straturile de ieşire ale rețelelor neuronale individuale au fost determinate prin încercări succesive. Au fost testate diferite combinații de funcții de activare pentru cei doi neuroni din straturile de ieşire, iar rezultatele obținute au fost comparate în termeni de performanță. Rețelele neuronale individuale comparate au avut aceeaşi structură (15 neuroni ascunşi), iar funcțiile de activare testate pentru neuronii din stratul de ieşire au fost: funcția liniară, funcția logistică şi tangenta hiperbolică (tanh). Folosind formula (4.1), datele experimentale folosite ca exemplu pentru ieşirea rețelei neuronale au fost scalate în intervalul [0.05, 0.95] pentru funcția logistică şi în intervalul [-0.9, 0.9] pentru tangenta hiperbolică.

Condiția de oprire a antrenării rețelelor neuronale a fost reducerea gradientului erorii sub 0.001. Numărul maxim de iterații pentru algoritmul de antrenare Quasi-Newton a fost stabilit la 300.

Cea mai bună performanță la modelarea sintezei hidrogelurilor multicomponente a fost obținută de către rețeaua neuronală cu funcții de activare liniare pentru ambii neuroni din stratul de ieşire (*perf_index* = 0.761). Având în vedere aceste rezultate, rețelele neuronale individuale incluse în stivă au avut funcții de activare liniare în stratul de ieşire.

Rețele neuronale simple din stivă au avut acelaşi număr de straturi ascunse (un strat), aceleaşi funcții de activare pentru straturile ascunse şi, respectiv, de ieşire, şi au fost instruite cu acelaşi algoritm de antrenare (Quasi-Newton). Diferențele au constat în numărul de neuroni ascunşi, ponderile sinaptice şi ponderile asociate ieşirilor rețelelor individuale care alcătuiesc cele două ieşiri ale stivei. Ponderile sinaptice ale rețelelor au fost inițializate cu valori generate uniform aleatoriu în intervalul [-0.5, 0.5].

Atât studiile noastre anterioare (Furtună şi colab., 2011b), cât şi alte studii din domeniu (Aguirre şi Tanaka, 2005; Brizuela şi Aceves, 2003) au evidențiat faptul că valorile parametrilor de control ai algoritmului evolutiv NSGA-II au o influență importantă asupra performanțelor sale. Prin urmare, au fost efectuate o serie de simulări cu valori diferite pentru aceşti parametri. Au fost testate diferite combinații de valori pentru numărul de cromozomi dintr-o populație (N_{pop}) şi pentru

numărul maxim de generații (N_{gen}), în timp ce probabilitățile de încrucişare şi mutație au fost stabilite la 0.9 şi, respectiv, 0.06. Unii cercetători (Nawaz Ripon şi colab., 2007) au sugerat că probabilitatea de mutație trebuie să fie egală cu raportul dintre 1 şi numărul de variabile de decizie (gene din cromozom) folosite de algoritmul evolutiv, iar o valoare egală sau mai mare decât 0.7 pentru probabilitatea de încrucişare duce, de obicei, la obținerea unor valori mai bune pentru funcțiile de fitness.

După determinarea celor mai potrivite valori pentru N_{pop} şi N_{gen}, au fost determinate cele mai bune valori pentru probabilitatea de încrucişare (P_C) şi probabilitatea de mutație (P_M) prin testarea de valori diferite. Cele care conduc la obținerea unui timp de convergență mai bun pentru NSGA-II şi la cel mai convenabil compromis între indicele de performanță şi numărul total de neuroni ascunşi ai rețelei neuronale stivă au fost alese pentru parametrii hiper-euristicii evolutive NSGA-II-QNSNN. S-a constatat că utilizarea unui număr maxim de generații mai mare decât 150 şi creşterea dimensiunii populației peste 50 au condus la un timp de convergență mai mare de 5 ore, cu nicio îmbunătățire semnificativă în capacitatea de generalizare a rețelei neuronale stivă sau la nivelul numărului total de neuroni ascunşi. De asemenea, s-a observat că o probabilitate de mutație mai mică decât 0.06 a condus la o creştere a timpului de convergență.

Câteva dintre rezultatele obținute prin simulări sunt prezentate în tabelul 7.4, în care timpul de convergență (*Ct*) este exprimat în minute. Indicele de performanță (*perf_index*) şi numărul total de neuroni ascunşi (*Hn_total*), prezentați în tabel, au fost selectați dintr-un set de soluții ne-dominate şi corespund rețelei neuronale stivă care a realizat cea mai ridicată performanță.

Tabelul 7.4. Cele mai bune compromisuri între indicele de performanță, numărul total de neuroni ascunşi şi timpul de convergență, obținute cu diferite valori pentru parametrii de control ai algoritmului NSGA-II

N_{pop}	N_{gen}	P_C	P_M	perf_index	Hn_total	Ct (min)
10	50	0.9	0.06	0.9015	26	18
10	50	0.9	0.01	0.9161	43	35
10	50	0.7	0.09	0.9206	39	31
10	150	0.8	0.06	0.9243	49	171
50	50	0.9	0.06	0.9237	51	114
50	50	0.9	0.01	0.9217	44	391
50	150	0.7	0.09	0.9286	63	279
50	250	0.7	0.09	0.9220	48	412
100	50	0.9	0.06	0.9210	39	306
100	150	0.8	0.06	0.9293	65	892

Setul de parametri selectat pentru a fi utilizat în optimizarea multi-obiectiv a rețelei neuronale stivă a fost: $N_{pop} = 50$, $N_{gen} = 150$, $N_T = 3$, $P_C = 0.7$, $P_M = 0.09$.

NSGA-II-QNSNN a furnizat un set de compromisuri la fel de bune între complexitatea rețelei neuronale stivă şi capacitatea sa de generalizare, sub forma un front optim Pareto (figura 7.4). Discontinuitatea care poate fi observată în frontul Pareto din figura 7.4 este cauzată de zonele deconectate din spațiul soluțiilor.

Setul de soluții optime ne-dominate corespunzătoare fiecărui punct din frontul optim Pareto, determinat prin procedura de optimizare multi-obiectiv, este prezentat în tabelul 7.6. Fiecare linie din tabel conține parametrii optimizați ai unei rețele neuronale stivă dintr-un set de rețele neuronale stivă la fel de bune. Liniile din tabelul 7.7 prezintă cele două obiective conflictuale care corespund variabilelor de decizie din rândul cu acelaşi număr din tabelul 7.6 (Furtună şi colab., 2011d).

Cel mai ridicat indice de performanță (0.928) a fost realizat de o rețea neuronală stivă formată din 5 rețele individuale, având un total de 63 de neuroni ascunşi, într-un timp de convergență de 279 de minute. Cea mai scăzută complexitate structurală a fost obținută pentru o rețea neuronală stivă

compusă din 2 rețele individuale şi având un total de 4 neuroni ascunşi, cu un indice de performanță de 0.636, după cum se poate observa în tabelul 7.7. Un compromis echilibrat (de mijloc) între complexitatea structurală a stivă şi indicele de performanță al acesteia este reprezentat de perechea de obiective contradictorii *perf_index* = 0.89 şi *total_Hn* = 16, care a fost obținută cu o rețea neuronală stivă constând din 2 rețele neuronale individuale.

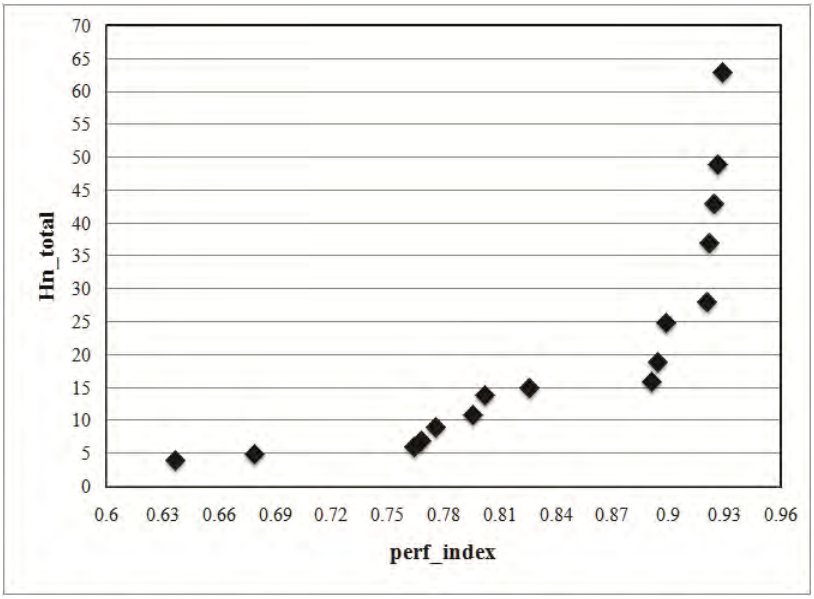


Figura 7.4. Frontul optim Pareto determinat prin aplicarea hiper-euristicii NSGA-II-QNSNN

Tabelul 7.6. Setul de soluții optime ne-dominate prin aplicarea hiper-euristicii NSGA-II-QNSNN

Nr.	NNno	w_{11}	w_{12}	w_{13}	w_{14}	w_{15}	w_{21}	w_{22}	w_{23}	w_{24}	w_{25}	Hn_1	Hn_2	Hn_3	Hn_4	Hn_5
1	2	0.17	0.83	0	0	0	0.47	0.53	0	0	0	2	2	0	0	0
2	2	0	1	0	0	0	0.25	0.75	0	0	0	2	3	0	0	0
3	2	0.08	0.92	0	0	0	0.25	0.75	0	0	0	2	4	0	0	0
4	2	0.22	0.78	0	0	0	0.22	0.78	0	0	0	3	4	0	0	0
5	2	0.31	0.69	0	0	0	0.40	0.60	0	0	0	5	4	0	0	0
6	3	0.02	0.42	0.56	0	0	0.12	0.34	0.54	0	0	2	4	5	0	0
7	2	0.97	0.03	0	0	0	0.40	0.60	0	0	0	11	3	0	0	0
8	2	0.80	0.20	0	0	0	0.30	0.70	0	0	0	11	4	0	0	0
9	2	0.01	0.99	0	0	0	0.05	0.95	0	0	0	2	14	0	0	0
10	2	0	1	0	0	0	0.20	0.80	0	0	0	5	14	0	0	0
11	2	0.03	0.97	0	0	0	0.18	0.82	0	0	0	11	14	0	0	0
12	3	0.02	0.53	0.45	0	0	0.02	0.37	0.61	0	0	2	14	12	0	0
13	3	0.08	0.54	0.38	0	0	0.07	0.44	0.49	0	0	11	14	12	0	0
14	4	0.03	0.38	0.27	0.32	0	0.06	0.43	0.33	0.18	0	5	14	12	12	0
15	4	0.01	0.36	0.28	0.35	0	0.05	0.40	0.39	0.16	0	11	14	12	12	0
16	5	0.02	0.26	0.24	0.26	0.22	0.06	0.42	0.26	0.12	0.14	11	14	12	12	14

Analizând rezultatele afişate în tabelele 7.6 şi 7.7, se poate concluziona că o rețea neuronală stivă cu două rețele componente, una având 14, iar cealaltă 2 sau 5 neuroni ascunşi, poate oferi o performanță foarte bună la modelarea sintezei hidrogelurilor multicomponente pe bază de

poliacrilamidă. Deşi un indice de performanţă mai mare de 0.92 a fost realizat doar cu reţele neuronale stivă compuse din mai mult de două reţele individuale, s-a constatat că una dintre reţelele neuronale din stivă a avut o influenţă foarte mică asupra rezultatelor acesteia. După cum se poate observa în linia 12 din tabelul 7.6, una dintre cele trei reţele neuronale din stivă (cea cu 2 neuroni ascunşi) are o contribuţie de doar 2% la ieşirile stivei.

Tabelul 7.7. Indicii de performanţă şi numărul total de neuroni obţinuţi cu hiper-euristica NSGA-II-QNSNN

Nr.	MSEtrain	MSEtest	r	perf_index	Hn_total
1	0.109	0.044	0.789	0.636	4
2	0.105	0.037	0.821	0.679	5
3	0.084	0.027	0.875	0.764	6
4	0.087	0.027	0.881	0.767	7
5	0.080	0.026	0.882	0.776	9
6	0.072	0.024	0.892	0.796	11
7	0.047	0.027	0.876	0.802	14
8	0.047	0.023	0.895	0.825	15
9	0.024	0.016	0.930	0.89	16
10	0.027	0.015	0.936	0.894	19
11	0.021	0.015	0.934	0.898	25
12	0.019	0.011	0.951	0.921	28
13	0.018	0.011	0.951	0.922	37
14	0.020	0.010	0.955	0.925	43
15	0.017	0.010	0.954	0.927	49
16	0.016	0.012	0.956	0.928	63

7.4.2 Modelarea neuro-evolutivă a unui proces de ultrafiltrare pe suport polimeric / electrodepunere pentru recuperarea metalelor grele (studiul de caz 2)

7.4.2.1 Baza de date experimentale

Scopul principal al celui de-al doilea studiu a fost determinarea unei reţele neuronale optime pentru modelarea unui proces de recuperare a metalelor grele printr-un procedeu în două etape integrate: 1) ultrafiltrare pe suport polimeric; 2) electrodepunere. Datele folosite la modelarea neuro-evolutivă au fost obţinute din experimente efectuate în Departamentul de Inginerie Chimică din cadrul Universităţii Castilla-La Mancha. Aceste date pot fi clasificate în două grupuri:

1) experimente de electrodepunere cu o celulă de electrod rotativ, la scară de laborator, şi cu soluţii monometalice de Cu^{2+} (Llanos şi colab., 2009).

2) experimente de electrodepunere cu o celulă de electrod rotativ, la scară de laborator, cu amestecuri bimetalice de Cu^{2+} şi Zn^{2+}. Aceste rezultate sunt încă nepublicate.

Baza de date experimentale a constat în 190 de date dintre care 152 (80%) au fost utilizate la antrenare şi 38 (20%) la testare.

Ţinând seama de natura tehnicii electrochimice folosite (electrodepunerea), parametrii care afectează cel mai puternic performanţa (viteza şi eficienţa) procesului au fost selectaţi ca variabile de intrare pentru reţeaua neuronală:

- rata de rotaţie a electrodului rotativ, N (min^{-1})
- densitatea de curent, j ($mA \cdot cm^{-3}$)
- pH-ul, pH
- concentraţia iniţială de cupru din reactor, Cu_0 (ppm)
- curentul electric aplicat, Q ($A \cdot h \cdot dm^{-3}$)

- raportul inițial Zn/Cu, *Zn/Cu*.

În ceea ce privește variabilele de ieșire, au fost selectați cinci parametri care ajută la urmărirea evoluției concentrației de metale, vitezei și calității procesului de electrodepunere:

- concentrația de cupru din reactor, *Cu* (ppm)
- concentrația de zinc din reactor, *Zn* (ppm)
- viteza de depunere, r_D (mg·min^{-1})
- randamentul de curent, *E*
- puritatea cuprului, *Cu_pur* (%)

7.4.2.2 Rezultatele modelării neuro-evolutive

Modelarea neuro-evolutivă presupune, în primul rând, determinarea celor mai potrivite valori pentru parametrii de control ai hiper-euristicii NSGA-II-QNN, și anume: mărimea populației (N_{pop}), numărul maxim de generații (N_{gen}), probabilitatea de încrucișare (P_C), probabilitatea de mutație (P_M), numărul de epoci utilizat în timpul antrenării (N_E), numărul maxim de iterații de antrenare (*maxTrIter*), dimensiunea maximă a ratei de învățare a algoritmului Quasi-Newton ($maxN_S$) și toleranța gradientul folosită în algoritmul Quasi-Newton (*TolG*).

Au fost efectuate o serie de simulări și teste, iar câteva rezultate reprezentând soluțiile de la extremitățile fronturilor Pareto obținute atunci când s-au utilizat diferite combinații de valori pentru parametri sunt prezentate în tabelul 7.8. Coloanele „*Cel mai mare perf_index / Nw*" și „*Cel mai mic perf_index / Nw*" conțin soluțiile de la extremitățile frontului Pareto obținut cu valorile parametrilor din coloanele 2 – 9. Coloana „*Ct (min)*" ilustrează timpul de convergență al hiper-euristicii evolutive exprimat în minute.

Tabelul 7.8. Rezultatele simulărilor cu diferite combinații de valori pentru parametrii de control ai hiper-euristicii NSGA-II-QNN

Nr.	N_{pop}	N_{gen}	P_C	P_M	N_E	Max TrIter	Max N_S	TolG	Cel mai mare perf_index / Nw	Cel mai mic perf_index / Nw	Ct (min)
1	5	5	0.9	0.1	10	200	10	1.0e-3	0.95884 / 197	0.01300 / 17	13
2	5	5	0.8	0.17	10	200	20	1.0e-4	0.98690 / 425	0.93746 / 101	40
3	5	5	0.7	0.17	10	200	20	1.0e-4	0.99094 / 563	-0.34636 / 17	58
4	10	10	0.8	0.17	10	200	10	1.0e-3	0.99364 / 469	0.01555 / 17	177
5	10	10	0.7	0.17	10	200	10	1.0e-3	0.98457 / 395	0.01662 / 17	48
6	10	10	0.9	0.17	10	200	10	1.0e-3	0.99382 / 563	-0.34513 / 17	185
7	10	10	0.9	0.17	20	300	10	1.0e-5	0.98549 / 283	0.02884 / 17	212
8	10	10	0.7	0.17	10	300	30	1.0e-4	0.99123 / 443	-0.80425 / 123	246
9	10	15	0.7	0.17	10	200	10	1.0e-3	0.98879 / 446	0.01515 / 17	242
10	15	10	0.9	0.17	10	200	10	1.0e-3	0.98569 / 267	0.76483 / 41	87
11	15	15	0.8	0.17	10	200	10	1.0e-3	0.99132 / 314	0.02902 / 17	182
12	15	20	0.9	0.17	10	200	10	1.0e-3	0.99145 / 509	0.01933 / 17	365
13	**20**	**10**	**0.8**	**0.17**	**10**	**200**	**10**	**1.0e-3**	**0.99226 / 395**	**0.01587 / 17**	**190**
14	20	50	0.7	0.17	10	200	10	1.0e-3	0.99226 / 503	0.01671 / 17	1337

Valorile parametrilor de control au fost selectate pe baza performanței obținute de către modele neuronale rezultate și a timpului necesar algoritmului pentru a converge. Cele mai bune trei soluții determinate au fost comparate în funcție de dominanța Pareto și distribuirea soluțiilor în frontul Pareto. Comparația este ilustrată în figura 7.10.

Luând în considerare faptul că frontul Pareto obținut la simularea numărul 13 (evidențiată cu caractere aldine în tabelul 7.8) a avut cea mai uniformă răspândire a soluțiilor, acesta a fost considerat cel mai bun rezultat.

Frontul optim Pareto obținut cu hiper-euristica evolutivă este ilustrat în figura 7.11.

Frontul Pareto poate fi împărțit în 3 regiuni:

A. regiunea în care soluțiile favorizează performanța la modelare a rețelei neuronale în detrimentul complexității sale structurale

B. regiunea în care cele două obiective divergente sunt echilibrate; nu se pune accent nici pe complexitatea structurală, nici pe performanțele rețelei;

C. regiunea în care este preferată o complexitate structurală mai mică față de o performanță mai mare a rețelei neuronale.

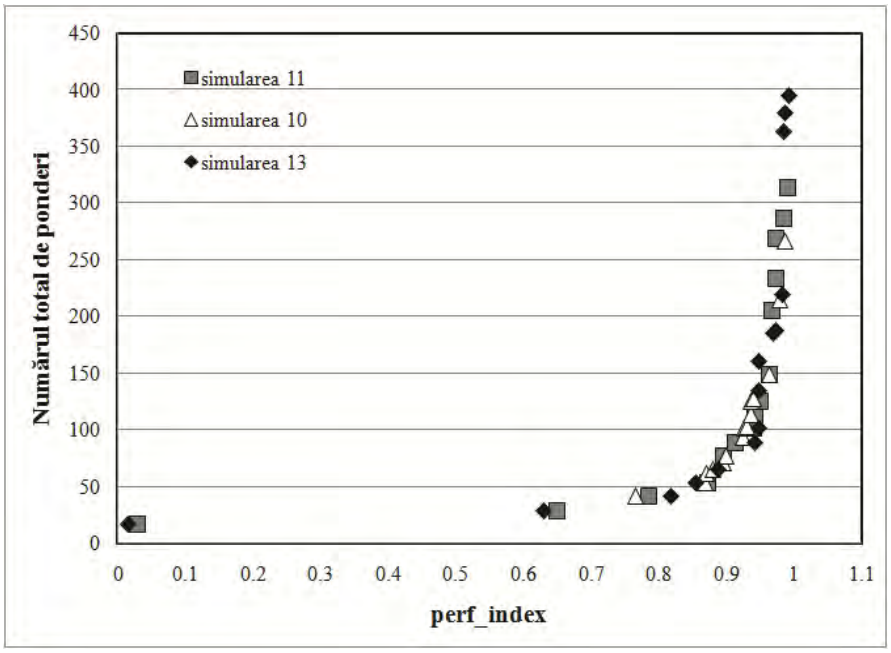


Figura 7.10. Cele mai bune fronturi Pareto obținute cu NSGA-II-QNN

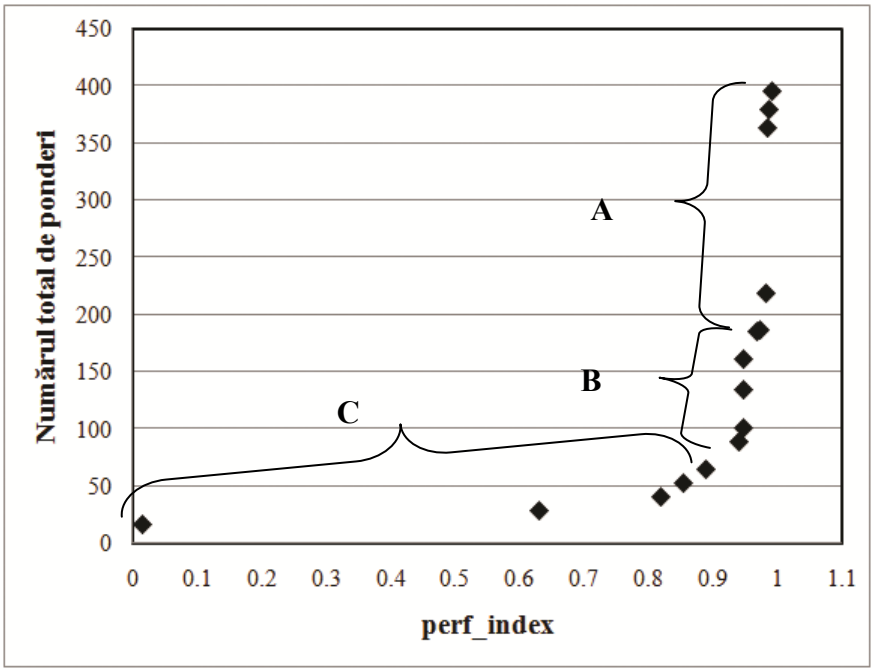


Figura 7.11. Frontul optim Pareto obținut cu NSGA-II-QNN

Parametrii fiecărei rețele neuronale corespunzătoare punctelor din frontul optim Pareto rezultat prin tehnica de optimizare multi-obiectiv sunt prezentați în tabelul 7.10.

Erorile obținute la modelare de către fiecare rețea neurală din setul de soluții optime Pareto sunt prezentate în tabelul 7.11. Formulele utilizate pentru calcularea erorilor sunt (4.10), (4.11) şi (4.12).

Tabelul 7.10. Setul de soluții optime Pareto determinate de NSGA-II-QNN

Nr.	perf_index	Nw	N_H	Hn_1	Hn_2	F_{H1}	F_{H2}	F_O
1	0.01587	17	1	1	0	tanh	-	tanh
2	0.63056	29	1	2	0	tanh	-	tanh
3	0.8187	41	1	3	0	tanh	-	tanh
4	0.85527	53	1	4	0	tanh	-	tanh
5	0.88823	65	1	5	0	tanh	-	tanh
6	0.94117	89	1	7	0	tanh	-	tanh
7	0.94691	101	1	8	0	tanh	-	tanh
8	0.94702	134	2	3	12	logistic	logistic	linear
9	0.94711	161	1	13	0	tanh	-	linear
10	0.96878	185	1	15	0	tanh	-	tanh
11	0.97279	187	2	10	7	tanh	tanh	linear
12	0.98226	219	2	10	9	tanh	logistic	linear
13	0.98444	363	2	19	9	tanh	logistic	linear
14	0.98647	379	2	10	19	tanh	logistic	linear
15	0.99226	395	2	12	17	tanh	logistic	linear

Tabelul 7.11. Erorile obținute la modelare de către fiecare rețea neurală din setul de soluții optime Pareto

Nr.	perf_index	Nw	MSE train	NMSE test	r test
1	0.01587	17	0.32106	0.37992	0.71685
2	0.63056	29	0.19089	0.10961	0.93106
3	0.8187	41	0.09864	0.05422	0.97157
4	0.85527	53	0.06744	0.05632	0.97902
5	0.88823	65	0.05096	0.03900	0.97819
6	0.94117	89	0.03433	0.01627	0.99177
7	0.94691	101	0.02819	0.01716	0.99227
8	0.94702	134	0.03296	0.01264	0.99262
9	0.94711	161	0.02464	0.01760	0.98934
10	0.96878	185	0.01244	0.01194	0.99316
11	0.97279	187	0.00838	0.01196	0.99314
12	0.98226	219	0.00400	0.00899	0.99525
13	0.98444	363	0.00131	0.00966	0.99541
14	0.98647	379	0.00132	0.00778	0.99557
15	0.99226	395	0.00134	0.00397	0.99757

7.5 Interfața grafică utilizator a programului ce implementează hiper-euristica NSGA-II-QN

Interfața grafică utilizator a programului ce implementează hiper-euristica NSGA-II-QN a fost realizată folosind limbajul de programare C#, urmărindu-se facilitarea interacțiunii unui utilizator neexperimentat cu metoda de optimizare multi-obiectiv a unei rețele neuronale. În cadrul acestei aplicații software a fost introdus operatorul de transpoziție, existând astfel posibilitatea alegerii algoritmului NSGA-II-RJG ca metodologie de optimizare multi-obiectiv. În acest caz, hiper-euristica evolutivă este notată NSGA-II-RJG-QN.

7.5.1 Fereastra de start a aplicației software

Fereastra de start a aplicației software prezintă opțiunea alegerii uneia dintre cele două variante de optimizare: a unei rețele neuronale simple sau a unei rețele neuronale de tip stivă. Opțiunea de optimizare a unei rețele neuronale simple este selectată implicit. Utilizatorul poate bifa cealaltă variantă, iar în urma apăsării butonului „Continuă", se va executa fie hiper-euristica NSGA-II-(RJG)-QNN pentru optimizarea unei rețele neuronale simple, fie hiper-euristica NSGA-II-(RJG)-QNSNN pentru optimizarea unei rețele neuronale stivă.

7.5.2 Fereastra principală a aplicației software

Acționarea butonului „Continuă" din fereastra de start duce la deschiderea unei noi ferestre pentru introducerea valorilor parametrilor de control ai hiper-euristicii evolutive şi pentru afişarea rezultatelor optimizării multi-obiectiv. Aceasta este fereastra principală a aplicației.

Odată ce butonul „Continuă" a fost apăsat şi s-a deschis următoarea fereastră, fereastra de start nu mai este accesibilă utilizatorului. Prin urmare, dacă acesta doreşte să îşi schimbe opțiunea pentru tipul de rețea neuronală optimizată, trebuie să închidă fereastra curentă şi să ruleze din nou aplicația. Închiderea ferestrei curente determină oprirea execuției aplicației.

7.5.2.1 Introducerea parametrilor de control ai hiper-euristicii evolutive

Parametrii de control ai hiper-euristicii evolutive constau din parametrii algoritmului evolutiv multi-obiectiv, parametrii algoritmului de antrenare a rețelelor neuronale şi parametrii de control ai rețelei neuronale ce urmează a fi optimizată. Valorile acestor parametri sunt presetate, iar utilizatorul le poate modifica ținând cont de procesul ce urmează a fi modelat cu rețeaua optimizată, de mesajele de eroare la antrenare, de soluția finală obținută şi de timpul de execuție al hiper-euristicii.

Dacă se alege o valoare diferită de 0 pentru probabilitatea de transpoziție, atunci algoritmul NSGA-II-RJG va fi folosit pentru optimizarea multi-obiectiv a rețelei neuronale. În caz contrar, este folosit algoritmul NSGA-II.

Există câteva restricții pentru valorile parametrilor de control ai algoritmului evolutiv care au fost implementate sub forma unor mesaje de eroare ce apar în urma acționării butonului „START".

În ceea ce priveşte algoritmul de antrenare Quasi-Newton, utilizatorul are acces doar la condiția de oprire a antrenării exprimată fie printr-un număr maxim de iterații, fie prin atingerea unei anumite valori prag pentru gradientul erorii. Aceşti doi parametri au o influență importantă asupra ratei de succes în antrenarea rețelelor neuronale.

Mesajele de eroare la antrenare din partea dreaptă sus a ferestrei principale a aplicației software reprezintă indicii asupra corectitudinii valorilor setate pentru numărul maxim de iterații ale algoritmului şi pentru toleranța gradientului erorii. Dacă numărul maxim de iterații este prea mic, atunci antrenarea se va opri înaintea atingerii valorii dorite pentru gradientului erorii, mesajul de eroare afişat în câmpul dedicat mesajelor de eroare la antrenare fiind „*S-a atins numarul maxim de iterații*". Dacă toleranța pentru gradient este prea mică, atunci atingerea numărului maxim de iterații poate totuşi conduce la o antrenare satisfăcătoare a rețelelor neuronale. Utilizatorul va putea evalua capacitatea de generalizare a rețelelor prin intermediul indicelui de performanță la antrenare şi testare, la finalul procedurii de optimizare, când are loc afişarea rezultatelor în tabelul din jumătatea de jos a ferestrei principale. Dacă performanța rețelelor din setul optim Pareto nu este cea aşteptată, atunci cei doi parametri de control ai algoritmului Quasi-Newton vor trebui modificați în următoarea rulare a aplicației software.

Este necesar ca utilizatorul să stabilească un număr maxim de iterații destul de mare, astfel încât antrenarea rețelei sa aibă loc cu succes, dar în acelaşi timp să folosească o toleranță nu prea mică pentru gradientul erorii, astfel încât antrenarea să nu dureze inutil de mult. Valorile prestabilite în interfața grafică pentru aceşti parametri pot fi considerate ca referință (figura 7.23). Există situația când antrenarea unei rețele neuronale durează prea mult şi atunci acest proces este finalizat forțat, afişându-se mesajul de eroare „*Antrenare finalizată forțat*". Rețeaua neuronală pentru care se

întâmplă aceasta este eliminată din populația de cromozomi (setul de soluții potențiale ale problemei).

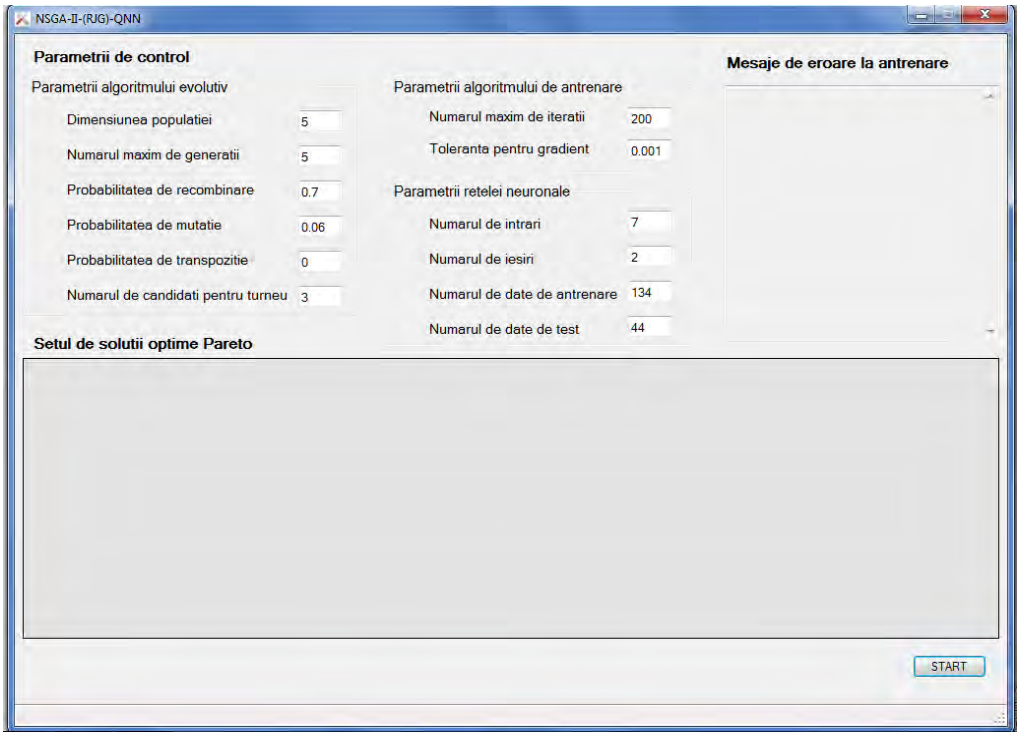


Figura 7.23. Fereastra pentru introducerea parametrilor de control şi afişarea rezultatelor hiper-euristicii NSGA-II-(RJG)-QNN

În câmpul pentru mesajele de eroare la antrenare se afişează timpul scurs în total în secunde, la finalul execuției aplicației. Acest câmp nu poate fi modificat de către utilizator, ci doar citit şi copiat.

Pentru organizarea datelor de antrenare şi testare, utilizatorul trebuie să ia în considerare indicațiile prezentate în capitolul 4, secțiunea 4.1.1. Datele experimentale nu trebuie sa fie normalizate înainte de introducerea în program, deoarece această operație este efectuată automat de către aplicația software. În cazul optimizării unei rețele neuronale simple, utilizatorul trebuie să creeze un fişier text salvat cu denumirea „antren.txt” în acelaşi director în care se află aplicația software. Acest fişier text trebuie să conțină doar date numerice separate prin caracterul „**,**” care să fie aranjate în câte un vector de antrenare/test pe un rând. Datele pentru testare trebuie să succeadă celor de antrenare.

În cazul rețelelor neuronale de tip stivă sunt necesare cinci fişiere text cu denumirea „antren1.txt”, „antren2.txt”, „antren3.txt”, „antren4.txt” şi „antren5.txt”, pregătite în acelaşi mod ca şi fişierul folosit pentru antrenarea şi testarea unei rețele neuronale simple. Cele cinci fişiere pentru antrenarea celor cinci rețele individuale din stivă trebuie să difere prin ordinea vectorilor de antrenare.

După apăsarea butonului „START” are loc execuția propriu-zisă a hiper-euristicii evolutive, fereastra principală prezentându-se sub forma celei ilustrate în figura 7.29.

Execuția hiper-euristicii evolutive începe cu inițializarea populației de cromozomi care este semnalată în bara de stare din josul ferestrei principale prin mesajul „*Se creează populația inițială* ...”. Mesajele de eroare la antrenare sunt înscrise în câmpul dedicat acestora în timp real. Utilizatorul poate opri execuția programului la orice moment de timp prin închiderea ferestrei principale a aplicației software.

Căutarea propriu-zisă a soluției optime începe după terminarea creării populației inițiale. Din acest moment bara de stare va afişa mesajul „*Căutarea soluției optime*” urmat de o bară de progres ce ilustrează grafic numărul de iterații efectuate de către hiper-euristica evolutivă. În continuarea

barei de progres sunt prezentați timpul estimat pentru determinarea soluției optime şi timpul scurs de la începerea căutării în formatul „ore:minute:secunde”, după cum se poate observa în figura 7.30.

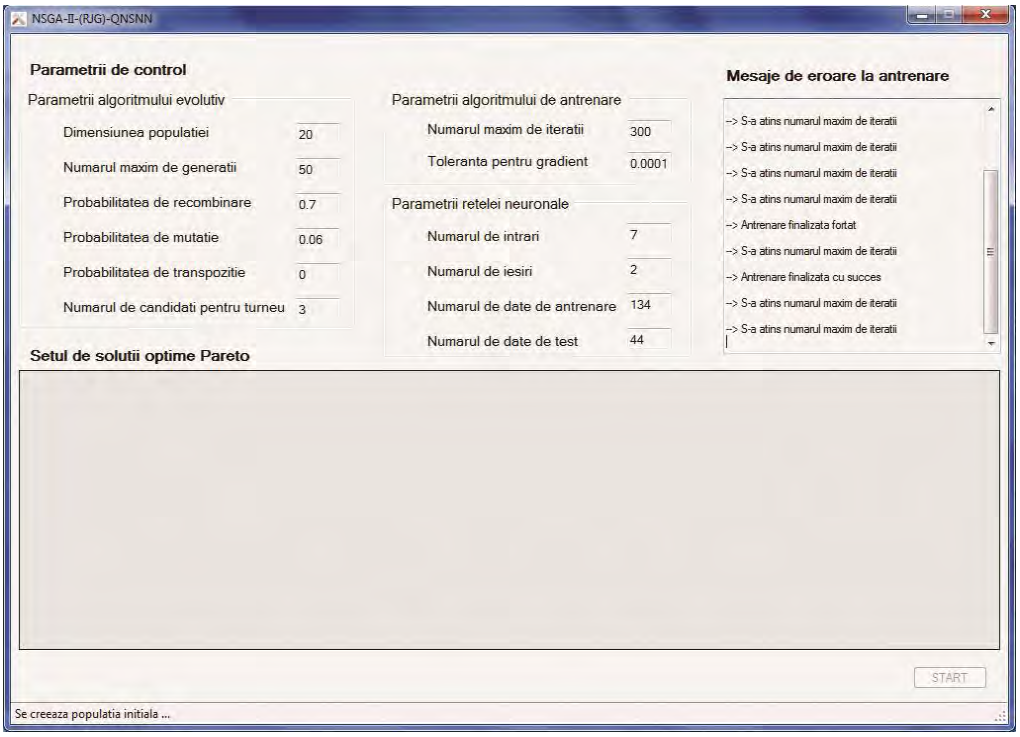


Figura 7.29. Fereastra principală a aplicației software în timpul execuției programului

Figura 7.30. Bara de stare a aplicației software în timpul execuției programului

7.5.2.2 Prezentarea rezultatelor hiper-euristicii evolutive

Odată cu acționarea butonului „START” se crează un fişier text, denumit „log.txt”, care înregistrează valorile stabilite pentru parametrii de control şi rezultatele finale ale căutarii soluției optime globale, precum în exemplele din figurile 7.31 şi 7.33. Aceste informații sunt adăugate în fişier după fiecare execuție a programului, astfel încât „log.txt” va conține parametrii de control şi rezultatele hiper-euristicii evolutive de la prima creare şi până în momentul scrierii curente. Acest fişier este util atunci când se doreşte analiza influenței parametrilor de control asupra rezultatelor căutării.

Dedesubtul fiecărei soluții din setul optim Pareto sunt prezentate eroarea pătratică medie (*MSE*) obținută la antrenare, eroarea pătratică medie normalizată (*NMSE*) obținută la testare şi coeficientul de corelație liniară (*r*) la testare pe baza cărora a fost calculat indicere de performanță (*perf_index*). Valorile obținute de către rețeaua corespunzătoare soluției prezentate la testarea pe un set de date noi, ce nu au fost folosite la antrenare, sunt enumerate sub titlul „*Ieşirile rețelei la testare*” (figurile 7.31 şi 7.33). Fiecare coloana reprezintă valorile obținute la una dintre ieşirile rețelei neuronale pentru fiecare vector de test. Aceste valori pot fi uşor importate într-o aplicație de calcul tabelar (de exemplu Microsoft Excel) pentru realizarea de grafice şi calculul unor erori suplimentare. Tot pentru realizarea de grafice şi calcule suplimentare este creat fişierul text „solutia.txt” la finalul execuției aplicației software. Fiecare rând din acest fişier conține soluțiile din setul optim Pareto şi funcțiile fitness corespunzătoare ordonate în aceeaşi manieră ca şi în fişierul „log.txt”, urmate de eroarea pătratică medie obținută la antrenare, eroarea pătratică medie normalizată obținută la testare şi coeficientul de corelație liniară la testare. Informațiile din fişierul „solutia.txt” sunt separate prin caracterul „,”. Spre deosebire de fişierul „log.txt”, fişierul „solutia.txt” este rescris după fiecare rulare a aplicației software şi, deci, va conține doar rezultatele

ultimei execuții a hiper-euristicii evolutive. Ambele fișiere sunt salvate în directorul în care se află aplicația software.

Pe lângă afișarea în câmpul pentru mesajele de eroare la antrenare, timpul total de execuție al hiper-euristicii evolutive exprimat în secunde este înscris și în fișierul „log.txt”, după ieșirile rețelei neuronale la testare. Această informație este utilă atunci când se dorește compararea diferitor setări pentru parametrii de control ai hiper-euristicii din punct de vedere al timpului de execuție.

Afișarea rezultatelor finale în interfața grafică se face folosind un tabel ce conține pe fiecare linie soluțiile din setul optim Pareto, funcțiile de fitness corespunzătoare din frontul optim Pareto (indicele de performanță și numărul total de neuroni), eroarea pătratică medie obținută la antrenare, eroarea pătratică medie normalizată obținută la testare și coeficientul de corelație liniară la testare. Tabelul rezultatelor pentru NSGA-II-(RJG)-QNN diferă de cel afișat pentru NSGA-II-(RJG)-QNSNN în ceea ce privește coloanele care conțin soluțiile din setul optim Pareto, după cum se poate observa în figurile 7.35 și 7.36.

Datele din tabel nu pot fi modificate de către utilizator, însă pot fi selectate celulă cu celulă și copiate în diferite tipuri de fișiere. Liniile din tabel pot fi sortate crescător sau descrescător, după informațiile dintr-o coloană. Atât liniile, cât și coloanele pot fi redimensionate, iar poziția coloanelor în tabel poate fi schimbată.

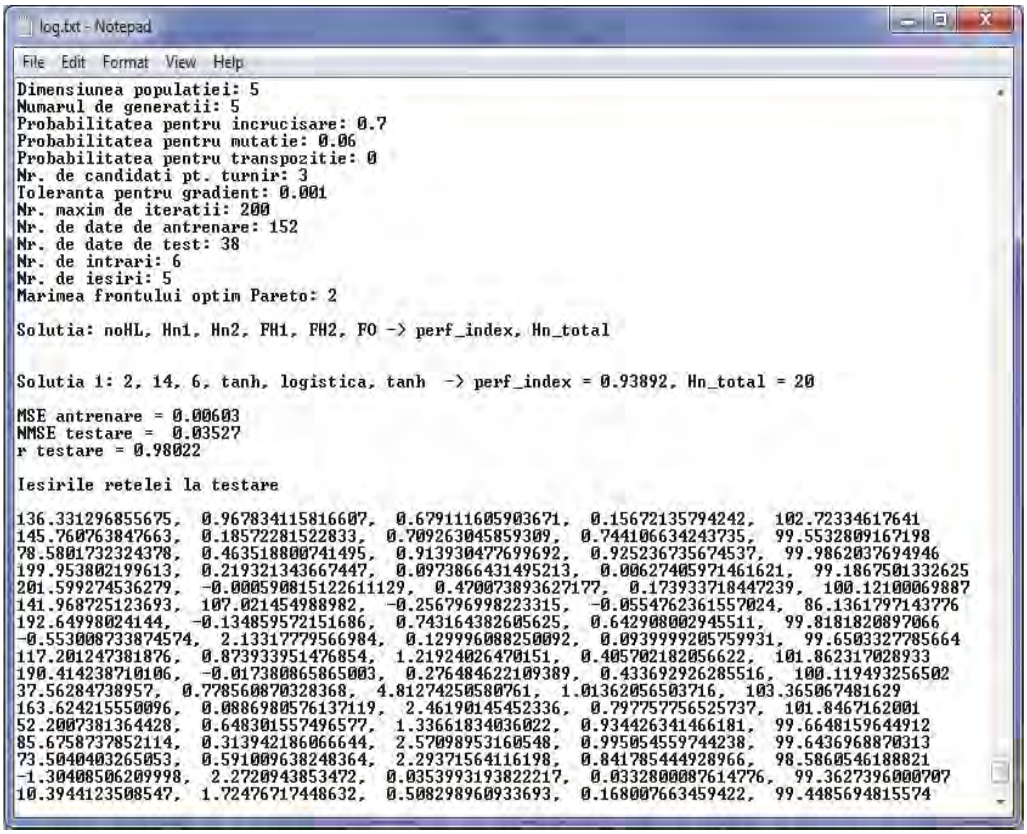


Figura 7.31. Informațiile scrise în fișierul „log.txt” în cazul execuției hiper-euristicii NSGA-II-(RJG)-QNN

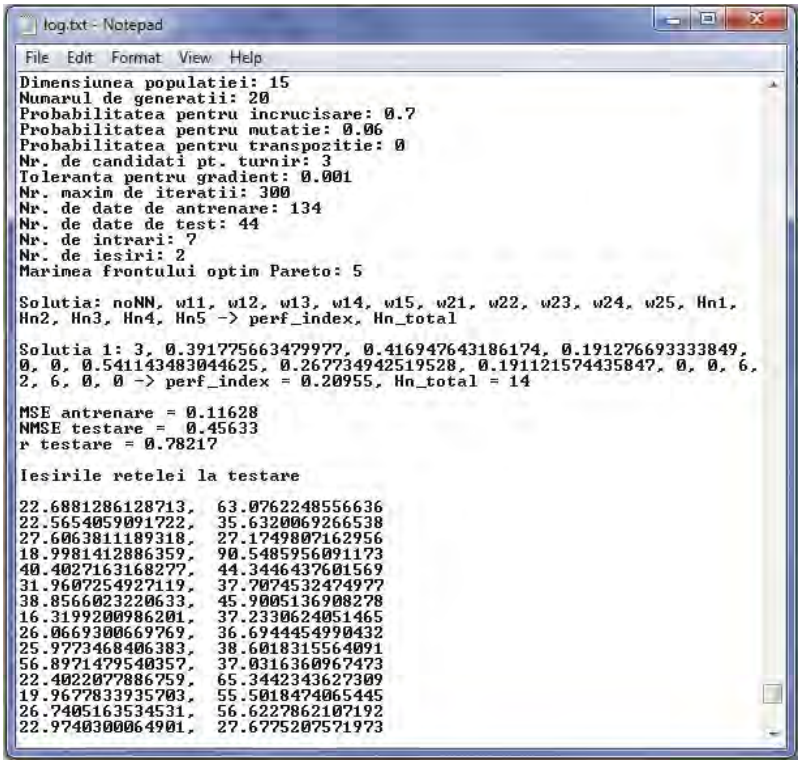


Figura 7.33. Informațiile scrise în fișierul „log.txt” în cazul execuției hiper-euristicii NSGA-II-(RJG)-QNSNN

Setul de solutii optime Pareto

	Numarul de straturi ascunse ale retelei neuronale	Numarul de neuroni din primul strat ascuns	Numarul de neuroni din cel de-al doilea strat ascuns	Functia de activare a neuronilor din primul strat ascuns	Functia de activare a neuronilor din cel de-al doilea strat ascuns	Functia de activare a neuronilor din stratul de iesire	Indicele de performanta	Numarul total de neuroni	Eroarea patratica medie la antrenare	Eroarea patratica medie normalizata la testare	Coeficientul de corelatie liniara la testare
▶	1	5	0	logistica	-	liniara	0.8392	5	0.0911	0.0438	0.9741
	2	5	5	logistica	tanh	tanh	0.8496	10	0.0493	0.0648	0.9637
	2	9	6	tanh	logistica	liniara	0.9283	15	0.0255	0.0294	0.9832

Figura 7.35. Tabelul din interfața grafică cu rezultatele hiper-euristicii NSGA-II-(RJG)-QNN

Setul de solutii optime Pareto

	Numarul de retele neuronale din stiva	Ponderea iesirii 1 a retelei 1	Ponderea iesirii 1 a retelei 2	Ponderea iesirii 1 a retelei 3	Ponderea iesirii 1 a retelei 4	Ponderea iesirii 1 a retelei 5	Ponderea iesirii 2 a retelei 1	Ponderea iesirii 2 a retelei 2	Ponderea iesirii 2 a retelei 3	Ponderea iesirii 2 a retelei 4	Ponderea iesirii 2 a retelei 5	Numarul de neuroni ascunsi ai retelei 1	Numarul de neuroni ascunsi ai retelei 2	Numarul de neuroni ascunsi ai retelei 3
▶	1	1	0	0	0	0	1	0	0	0	0	4	0	0
	2	0.9876	0.0124	0	0	0	0.36	0.64	0	0	0	15	4	0
	2	0.5862	0.4138	0	0	0	0.3829	0.6171	0	0	0	4	12	0
	3	0.44	0.2137	0.3463	0	0	0.0612	0.4102	0.5286	0	0	4	2	6
	3	0.848	0.0821	0.0699	0	0	0.2784	0.1507	0.5709	0	0	4	1	6
	4	0.3552	0.3287	0.3109	0.0052	0	0.2023	0.3833	0.3885	0.0259	0	6	13	12
	5	0.2274	0.1618	0.0382	0.2298	0.3428	0.1869	0.2948	0.2652	0.1734	0.0797	11	4	14

Setul de solutii optime Pareto

Numarul de neuroni ascunsi ai retelei 1	Numarul de neuroni ascunsi ai retelei 2	Numarul de neuroni ascunsi ai retelei 3	Numarul de neuroni ascunsi ai retelei 4	Numarul de neuroni ascunsi ai retelei 5	Indicele de performanta	Numarul total de neuroni	Eroarea patratica medie la antrenare	Eroarea patratica medie normalizata la testare	Coeficientul de corelatie liniara la testare
4	0	0	0	0	0.4998	4	0.0793	0.2862	0.8653
15	4	0	0	0	0.595	19	0.0423	0.2532	0.8905
4	12	0	0	0	0.5639	16	0.0613	0.2612	0.8863
4	2	6	0	0	0.5309	12	0.0852	0.2651	0.8813
4	1	6	0	0	0.52	11	0.0938	0.2705	0.8843
6	13	12	1	0	0.6273	32	0.0376	0.2193	0.8842
11	4	14	4	10	0.6374	43	0.0654	0.2222	0.925

Figura 7.36. Tabelul din interfața grafică cu rezultatele hiper-euristicii NSGA-II-(RJG)-QNSNN

8. Concluzii finale

8.1 Concluzii referitoare la obiectivele rezolvate în teză

Având în vedere dificultățile ce apar la optimizarea proceselor din ingineria chimică prin metodele tradiționale, această teză de doctorat a propus introducerea unor abordări noi bazate pe instrumente ale inteligenței artificiale, şi anume *metodologii de optimizare ce includ algoritmi evolutivi şi rețelele neuronale artificiale*. Astfel, au fost propuse şi rezolvate următoarele obiective:

- Elaborarea unei *metodologii de optimizare a parametrilor unei rețele neuronale cu propagare înainte* care să conducă la obținerea unor modele neuronale performante

S-a dezvoltat un algoritm în şase paşi bazat pe metoda încercare şi eroare, urmărindu-se optimizarea parametrilor de control ai unei rețele neuronale de tip feed-forward. În procesul de elaborare a metodologiei de optimizare s-a ținut cont de o serie de considerente practice existente în literatura din domeniu referitoare la elementele caracteristice unei rețele neuronale. Parametrii optimizați au fost: numărul de straturi ascunse din rețea, numărul de neuroni din straturile ascunse, valoarea ratei de învățare, valoarea termenului moment, numărul de epoci de antrenare şi funcția de activare a stratului de ieşire. Metodologia de optimizare a parametrilor unei rețele neuronale cu propagare înainte s-a dovedit a fi eficientă şi uşor de folosit, deoarece utilizatorul nu necesită cunoştințe avansate despre modelarea neuronală sau despre alte tehnici ale inteligenței artificiale pentru a putea utiliza metodologia propusă. Algoritmul în şase paşi ce constituie structura de bază a metodologiei de optimizare s-a dovedit a fi concis şi transparent, oferind un control total asupra etapelor de optimizare a parametrilor rețelei neuronale. Deşi nu se poate afirma că metodologia dezvoltată constituie răspunsul univoc la problema complicată şi dificilă a determinării unei rețele neuronale optime, este totuşi clar faptul că acestă metodologie oferă informații importante şi utile pentru conducerea unei căutări logice şi sistematice.

- Adaptarea şi aplicarea *algoritmului evolutiv multi-obiectiv NSGA-II* la optimizarea unor procese complexe din ingineria chimică

Algoritmul evolutiv multi-obiectiv NSGA-II a fost adaptat pentru rezolvarea problemelor de optimizare din ingineria chimică, prin folosirea codificării cu numere reale pentru soluții şi a operatorilor genetici artificiali potriviți acestui tip de codificare. A fost inclus un nou operator pentru eliminarea soluțiilor duplicat din setul de soluții ne-dominate. Pentru calcularea funcțiilor multi-obiectiv de tip vectorial ale algoritmului, s-a introdus un modul funcțional bazat pe unul sau mai multe modele neuronale, în funcție de numărul de parametri ai procesului modelat. S-a observat faptul că parametrii de control ai algoritmului evolutiv NSGA-II au o puternică influență asupra performanței acestuia şi prin urmare, s-a dezvoltat o tehnică de evaluare a calității soluțiilor corelată cu timpul de convergență, în funcție de valorile folosite pentru parametri. Rețeaua neuronală ce modelează procesul supus optimizării, folosită pentru a calcula funcțiile de fitness ale algoritmului

NSGA-II, a condus la scurtarea timpului de convergență al algoritmului, consolidând astfel procedura de optimizare multi-obiectiv. Cu toate că metoda propusă a fost utilizată cu succes pentru probleme de optimizare cu doar două funcții obiectiv, aceasta poate fi, de asemenea, aplicată pentru probleme cu mai mult de două obiective. De asemenea, metoda poate fi aplicată diferitelor altor tipuri de procese, despre care informațiile disponibile sunt limitate, rețelele neuronale constituind un instrument util în această situație. Procedura de optimizare multi-obiectiv bazată pe NSGA-II s-a dovedit a fi generală şi uşor de adaptat la alte seturi de date (procese). Deoarece fiecare soluție optimă Pareto are proprietăți diferite, posibilitatea de a inspecta întregul front Pareto furnizat de către NSGA-II uşurează munca utilizatorului în a decide care dintre specificațiile obiectivelor pot fi atinse şi care nu, şi ce compromisuri se pot face în scopul realizării condițiilor optime de operare pentru procesul abordat.

- Îmbunătățirea soluțiilor oferite de algoritmul evolutiv NSGA-II prin introducerea *operatorului artificial de transpoziție pentru cromozomi codificați folosind numere reale* (NSGA-II-RJG)

Un nou operator genetic artificial a fost introdus în algoritmul NSGA-II în scopul contracarării scăderii diversității soluțiilor datorată folosirii elitismului. Noul operator imită operația de schimbare a poziției unei gene în cadrul aceluiaşi cromozom sau într-un cromozom diferit. Acesta se numeşte operator de transpoziție sau „jumping genes" şi a fost inclus înainte de operațiile de încrucişare şi mutație. Fiind codificat real şi implementat prin folosirea unor operatori de încrucişare şi mutație adaptați acestui tip de codificare, noul operator introdus s-a deosebit față de variantele deja existente în literatura din domeniu. Algoritmul rezultat, NSGA-II-RJG a fost aplicat la optimizarea multi-obiectiv a unui proces de polimerizare complex, iar rezultatele obținute au fost comparate cu cele furnizate de NSGA-II pe acelaşi studiu de caz. S-a observat că NSGA-II-RJG aduce o îmbunătățire atât la nivelul diversității soluțiilor, cât şi la nivelul vitezei de convergență spre soluția optimă, dacă parametrii de control ai acestuia sunt setați la valorile cele mai potrivite, prin intermediul tehnicii de evaluare propuse în acest scop.

- *Combinarea algoritmului de căutare globală NSGA-II cu algoritmul de căutare locală Quasi-Newton* în scopul dezvoltării unor rețele neuronale optime

Algoritmul hibrid rezultat este o hiper-euristică în cadrul căreia algoritmul evolutiv multi-obiectiv NSGA-II, combinat cu algoritmul de optimizare Quasi-Newton (QN), reprezintă metodologia de nivel înalt ce este utilizată pentru dezvoltarea de rețele neuronale optime care reprezintă, la rândul lor, euristicile de nivel scăzut. *Hiper-euristica evolutivă NSGA-II-QN* rezultată determină un întreg set de soluții optime Pareto ne-dominate care corespund celor mai bune compromisuri între complexitatea structurală a rețelei neuronale şi capacitatea de generalizare a acesteia. Se caută cele mai bune valori pentru parametrii structurali ai rețelelor neuronale astfel încât sa se obțină erori cât mai mici la antrenare şi testare, dar, totodată, să se reducă cât mai mult numărul de ponderi sinaptice (sau numărul total de neuroni ascunşi). Algoritmul NSGA-II-QN a fost particularizată pentru optimizarea unor rețele neuronale simple şi a unor rețele neuronale de tip stivă. Cele două variante rezultate, NSGA-II-QNN şi NSGA-II-QNSNN s-au diferențiat în primul rând prin variabilele de decizie care construiesc soluția căutată. Hiper-euristica evolutivă NSGA-II-QN este superioară metodei încercare şi eroare deoarece oferă o soluție globală optimă într-o singură rulare a aplicației software şi nu are nevoie de interacțiunea cu un utilizator de-a lungul procesului de optimizare. Utilizatorul trebuie doar să introducă valori pentru parametrii de control ai algoritmului NSGA-II-QN şi, după câteva ore, i se prezintă un set de rețele neuronale optime. Faptul că parametrii de control ai hiper-euristicii evolutive sunt setați de către utilizator face ca aplicația software să aibă un grad mare de generalitate, putând fi aplicată pentru o vastă diversitate de probleme de modelare.

- Realizare unor *aplicații software originale* prin implementarea algoritmilor de modelare şi optimizare

Pentru fiecare algoritm dezvoltat s-au implementat programe software originale folosind limbajul de programare C#. Interacțiunea cu utilizatorul s-a realizat prin intermediul consolei, rezultatele fiind înscrise atât în consolă, cât şi într-un fişier text. S-au dezvoltat şi testat module software pentru fiecare etapă şi operație a algoritmilor construiți. De asemenea, s-au creat

instrumente software noi pentru evaluarea performanțelor algoritmilor de modelare şi optimizare creați. Implementările pentru hiper-euristicile NSGA-II-QNN şi NSGA-II-QNSNN au fost reunite într-o aplicație software cu interfață grafică utilizator care oferă şi posibilitatea folosirii operatorului de transpoziție pentru NSGA-II. Deci utilizatorul poate decide dacă va folosi algoritmul NSGA-II sau algoritmul NSGA-II-RJG ca algoritm evolutiv de căutare globală în cadrul algoritmului de optimizare hibrid.

- Asocierea strategiilor de modelare şi optimizare cu diferite *procese din domeniul ingineriei chimice şi protecției mediului*, în vederea simulării diferitelor caracteristici de proces sau proprietăți ale produselor obținute.

Modelarea viscozității unor compuşi pe bază de poliazometină a reprezentat studiul de caz pe care s-a dezvoltat, aplicat şi testat metodologia de optimizare a parametrilor unei rețele neuronale cu propagare înainte. Măsurarea masei moleculare prezintă un interes special în industria polimerilor deoarece multe proprietăți fizice şi dinamice sunt legate de mărimea şi forma unei macromolecule. Viscozitatea soluțiilor diluate de polimeri este sensibilă la aceste caracteristici structurale, ceea ce face ca măsurarea viscozității să fie utilă stabilirii de corelații cu proprietățile fizice şi de procesare. Prin urmare, predicția viscozității pentru anumite concentrații şi temperaturi ale soluției este utilă practicii experimentale. O rețea neuronală de tip MLP optimizată prin metodologia de optimizare a parametrilor unei rețele neuronale a fost aplicată la modelarea viscozității reduse funcție de temperatură şi concentrația soluției, conform cu tipul copolimerului folosit. La fiecare pas al algoritmului de optimizare au fost testate diferite valori pentru parametrii de control ai rețelei neuronale şi au fost alese cele care conduc la obținerea celor mai mici erori la antrenare şi testare şi la cel mai mare coeficient de corelație liniară la testare. Cea mai performantă rețea neuronală, cu un indice de performanță de 0.76, a fost o rețea de tip MLP(3:5:1), având tangenta hiperbolică drept funcție de activare pentru stratul ascuns şi pentru stratul de ieşire, rate de învățare de 1 şi 0.1 şi termeni moment de 0.9 pentru straturile ascuns şi de ieşire.

Cea de-a doua problemă pe care a fost aplicată şi testată metodologia de optimizare a parametrilor unei rețele neuronale a fost *predicția viscozității intrinseci* a aceloraşi compuşi pe bază de poliazometină, funcție de temperatură şi concentrația soluției. Rețeaua neuronală care a obținut cel mai mare indice de performanță (0.9972) la modelarea viscozității intrinseci a fost o rețea MLP(2:2:1), având tangenta hiperbolică drept funcție de activare pentru stratul ascuns şi stratul de ieşire, rate de învățare de 1 şi 0.3 şi termeni moment de 0.6 şi 0.1 pentru straturile ascuns şi, respectiv, de ieşire.

Următorul studiu de caz considerat a fost bazat pe *procesul de sinteză a polisiloxanilor*. S-a urmărit stabilirea dependenței conversiei monomerului de condițiile de reacție (temperatură, timp, cantitate de catalizator, cantitate de apă) şi, pe de altă parte, a dependenței masei moleculare de aceleaşi condiții de reacție. Astfel, s-a aplicat metodologia de optimizare a parametrilor unei rețele neuronale pentru determinarea celor mai eficiente rețele neuronale care să modeleze procesul de polimerizare. Rețeaua care a modelat cel mai bine cei doi parametri ai procesului a fost o rețea MLP(4:9:2) cu indicele de performanță total egal cu 1.59. Rețeaua neuronală a fost apoi inclusă în algoritmul evolutiv multi-obiectiv NSGA-II pentru calcularea funcțiilor de fitness ale acestuia. Procedeul de optimizare bazat pe algoritmul NSGA-II a determinat un set de condiții de reacție optime (temperatura, timpul de reacție, cantitatea de catalizator şi cantitatea de co-catalizator), care au condus la obținerea, în acelaşi timp, a unei conversii maxime a monomerului şi a unei mase moleculare viscozimetrice impuse.

Alt studiu de caz tratat a propus modelarea şi optimizarea multi-obiectiv a *procesului de preparare a nanoparticulelor polimerice cu surfactanți siliconici*. În acest studiu a fost analizată influența concentrațiilor de surfactant şi de polimer şi a temperaturii de stocare asupra rezultatelor nanoprecipitării unui polidimetilsiloxan liniar (PDMS), folosind rețele neuronale artificiale determinate cu metodologia de optimizare a parametrilor unei rețele neuronale cu propagare înainte. Diametrul mediu al particulelor de surfactant siliconic şi indicele de polidispersitate au fost modelați cu o rețea de tip MLP(3:9:1) şi, respectiv, o rețea de tip MLP(2:6:1). Aceste modele neuronale au fost apoi incluse în algoritmii NSGA-II şi NSGA-II-RJG în scopul determinării

valorilor optime pentru concentrația de surfactant, concentrația de polimer şi temperatura de stocare care să minimizeze atât diametrul mediu, cât şi indicele de polidispersitate. Cei doi algoritmi evolutivi multi-obiectiv au îndeplinit cu succes obiectivele urmărite, într-un timp de execuție foarte scurt. Rezultatele au indicat că valori minime pentru diametru se obțin cu o concentrație de surfactant de 2 gL^{-1}, o concentrație de polimer cuprinsă între 0.9% şi 1.02% şi o temperatură de stocare de 4˚C. Pe de altă parte, cea mai mică valoare pentru indicele de polidispersitate a fost obținută utilizând o concentrație de surfactant de 0.5 gL^{-1}, o concentrație de polimer de 0.53% şi o temperatură de depozitare de 4˚C.

- *Polimerizarea radicalică a stirenului* efectuată prin tehnica în suspensie, constituie al cincilea studiu de caz pe care a fost aplicată metodologia de optimizare a parametrilor unei rețele neuronale cu propagare înainte. Predicția variației conversiei monomerului şi masei moleculare a polimerului în funcție de concentrația de inițiator, temperatura şi timpul reacției a fost realizată cu o rețea neuronală optimizată, cu două straturi ascunse, având un indice de performanță de 0.9.
- Optimizarea *procesului de sinteză a unor copolimeri siloxanici* a fost realizată prin aplicarea implementării software a algoritmului NSGA-II. Procedura de optimizare multi-obiectiv bazată pe NSGA-II a fost utilizată pentru a obține variabilele de control optime care să maximizeze conversia reacției şi să minimizeze diferența dintre compoziția obținută şi cea impusă copolimerului. Variația în timp a principalilor parametri ai procesului (conversia reacției şi compoziția copolimerului) a fost modelată cu o rețea neuronală feed-forward, cu arhitectura MLP(4:10:2), care a fost apoi utilizată în calcularea funcțiilor de fitness ale algoritmul evolutiv.
- Un alt studiu de caz a urmărit dezvoltarea şi optimizarea unei rețele neuronale de tip stivă pentru a fi utilizată la modelarea *procesului de sinteză a unor hidrogeluri multicomponente pe bază de poliacrilamidă*. În acest scop, a fost aplicată hiper-euristica evolutivă NSGA-II-QNSNN care a oferit un întreg set de soluții optime Pareto reprezetând rețele neuronale stivă cu performanțe optime la modelare şi o complexitate structurală scăzută. Aceste rețele neuronale stivă au stabilit influența condițiilor de reacție (concentrația de monomer, concentrația de inițiator, concentrația agentului de reticulare, cantitatea de polimer de incluziune, timpul de reacție, temperatura şi tipul de polimer inclus) asupra randamentului reacției şi gradului de umflare. Cel mai mare indice de performanță realizat la modelare a fost de 0.98. A fost obținută o eroare relativă medie de 1.92% la predicția randamentului polimerizării şi una din 8.96% la predicția gradului de umflare.
- Modelarea neuro-evolutivă a *procesului de ultrafiltrare pe suport polimeric / electrodepunere pentru recuperarea metalelor grele* a fost realizată prin aplicarea hiper-euristicii evolutive NSGA-II-QN. Aceasta a determinat un set de rețele neuronale optime prin optimizarea parametrilor de control ai acestora, şi anume numărul de straturi ascunse, numărul de neuroni din fiecare strat ascuns, funcțiile de activare ale straturilor ascunse şi ale stratului de ieşire şi ponderile sinaptice (incluzând biasurile) ale rețelei neuronale. Cel mai mare indice de performanță la modelare (0.99) a fost obținut de o rețea neuronală simplă, având 395 de ponderi sinaptice.

8.2 Aspecte originale

Contribuțiile originale ale tezei sunt evidențiate în următoarea enumerare:

- Dezvoltarea unei *metodologii de optimizare a parametrilor unei rețele neuronale cu propagare înainte* sub forma unui algoritm în şase paşi, concis şi uşor de înțeles şi aplicat. Această metodologie reprezintă o soluție eficientă la problema dezvoltării unei rețele neuronale care să ofere performanțe bune la modelarea proceselor din ingineria chimică.
- Crearea unor *criterii şi formule pentru cuantificarea performanțelor rețelelor neuronale* atât la antrenare, cât şi la testare, reprezintă un punct esențial pentru metodele de optimizare a unei rețele neuronale propuse în acestă lucrare.
- Adaptarea algoritmului NSGA-II la optimizarea multi-obiectiv a proceselor din ingineria chimică prin folosirea unor *operatori dedicați codificării cu numere reale* şi introducerea unui *operator nou pentru eliminarea soluțiilor duplicat* din frontul Pareto.

- Adăugarea unui *nou operator genetic* în algoritmul NSGA-II adaptat codificării cu numere reale – operatorul de transpoziție („jumping genes") – pentru evitarea scăderii diversității soluțiilor oferite de acesta. Noul algoritm evolutiv rezultat a fost denumit NSGA-II-RJG. Operatorul de transpoziție a fost implementat prin combinarea operatorului de mutație aleatorie cu cel de încrucișare aritmetică.
- Includerea modelelor neuronale în cadrul algoritmilor evolutivi NSGA-II şi NSGA-II-RJG pentru *calculul funcțiilor de fitness* corespunzătoare fiecărei soluții. Rețelele neuronale folosite pentru modelarea procesului optimizat facilitează procesul de căutare a unei soluții optime.
- Dezvoltarea unei *tehnici de evaluare a performanțelor unui algoritm evolutiv multi-obiectiv* pe baza metricilor de acoperire a setului şi de spațiere. Această tehnică a fost creată sub forma unei comparații bazate pe relația de dominanță Pareto.
- Realizarea unei *metode de optimizare a parametrilor de control* (mărimea populației, numărul maxim de generații, probabilitățile operațiilor de încrucișare, mutație şi transpoziție) ai algoritmului evolutiv NSGA-II şi ai celorlalți algoritmi dezvoltați pe baza acestuia.
- Construirea unei *hiper-euristici evolutive* pentru determinarea unor rețele neuronale optime destinate modelării unor procese din ingineria chimică şi protecția mediului. S-a realizat hibridizarea dintre un algoritm de căutare globală şi un algoritm de căutare locală, iar algoritmul rezultat a fost folosit în optimizarea multi-obiectiv a unor modele neuronale. Hiper-euristica evolutivă propusă a fost particularizată pentru două tipuri de rețele (simple şi de tip stivă), rezultând două variante ce au fost denumite NSGA-II-QNN şi NSGA-II-QNSNN.
- Implementarea prin *aplicații software originale* a noilor algoritmi evolutivi de optimizare propuşi în această lucrare.

8.3 Direcții de continuare a cercetărilor

Cercetările începute în cadrul acestei teze de doctorat vor fi continuate în sensul perfecționării algoritmilor şi tehnicilor dezvoltate, aplicării şi testării acestora în cazul altor procese sau prin abordarea unor noi probleme din domeniul inteligenței artificiale. Referitor la această ultimă direcție, se au în vedere probleme cum ar fi:

- cuantificarea şi clasificarea tuturor elementelor caracteristice unei rețele neuronale care influențează în mod semnificativ performanța acesteia la modelarea proceselor din ingineria chimică;
- extinderea metodologiei de optimizare a unei rețele neuronale la alte tipuri de rețele;
- crearea şi testarea unor funcții obiectiv alternative pentru algoritmii de optimizare propuşi în această lucrare;
- includerea parametrilor de control ai algoritmilor evolutivi în cadrul procesului de optimizare multi-obiectiv prin realizarea unor algoritmi evolutivi auto-adaptivi. O altă alternativă în sensul automatizării procesului de determinare a parametrilor de control optimi ar fi folosirea tehnicilor bazate pe logica fuzzy pentru controlul dinamic al algoritmilor evolutivi sau pentru monitorizarea comportamentului acestora;
- introducerea unor metode pentru selecția anumitor soluții din frontul optim Pareto generat de algoritmii evolutivi multi-obiectiv, bazate pe prioritățile utilizatorului. Una dintre aceste metode ar fi includerea preferințelor utilizatorului în funcțiile obiectiv sau în codificarea variabilelor de decizie pentru a restrânge spațiul de căutare al soluțiilor;
- testarea altor algoritmi evolutivi şi a altor algoritmi de antrenare pentru o eventuală îmbunătățire a hiper-euristicii neuro-evolutive propusă în această lucrare.

Bibliografie selectivă

Aguirre, H., Tanaka, K., *Selection, drift, recombination, and mutation in multiobjective evolutionary algorithms on scalable MNK-Landscapes*, Evolutionary multi-criterion optimization, 3rd international conference, EMO 2005, Lecture notes in computer science, 3410, 355 – 369, 2005.

Brizuela, C.A., Aceves, R., *Experimental genetic operators analysis for the multi-objective permutation flowshop evolutionary multi-criterion optimization*, in: Proceedings of the 2nd international conference on Evolutionary multi-criterion optimization, 2632, Springer-Verlag Berlin, Heidelberg, 578 – 592, 2003.

Cazacu, M., Marcu, M., *Silicone rubbers. IX: Contributions to polydimethylsiloxane-α,ω-diols synthesis by heterogeneous catalysis*, Macromolecular Reports, 32, 1019 – 1029, 1995.

Coello Coello, C.A., Lamont, G.B., Van Veldhuizen, D.A., *Evolutionary Algorithms for Solving Multi-Objective Problems*, in: D.E. Goldberg, J.R. Koza (Eds.), Genetic and Evolutionary Computation Series, Springer, New York, 2007.

Curteanu, S., *Modeling and simulation of free radical polymerization of styrene under semibatch reactor conditions*, Central European Journal of Chemistry, 1, 69 – 90, 2003.

Curteanu, S., Cazacu, M., *Neural networks and genetic algorithms used for modeling and optimization of the siloxane-siloxane copolymers synthesis*, Journal of Macromolecular Science, Part A: Pure and Applied Chemistry, 45 (1), 23 – 36, 2008.

Curteanu, S., Dumitrescu, A., Mihăilescu, C., Simionescu, B., *Neural network modeling applied to polyacrylamide based hydrogels synthesized by single step process,* Polymer-Plastics Technology and Engineering, 47, 1061 – 1071, 2008a.

Curteanu, S., Dumitrescu, A., Mihailescu, C., Simionescu, B., *The synthesis of polyacrylamide-based multicomponent hydrogels. A neural network modeling*, Journal of Macromolecular Science. Part A: Pure and Applied Chemistry, 46 (4), 368 – 380, 2009.

Deb, K., Agrawal, S., Pratab, A., Meyarivan, T., *A Fast Elitist Non- Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II*, in: M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J.J. Merelo, H.-P. Schwefel (Eds.), Proceedings of the Parallel Problem Solving from Nature VI Conference, Springer, Lecture Notes in Computer Science, 1917, 849 – 858, 2000.

Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., *A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II*, IEEE Transactions on Evolutionary Computation, 6(2), 182 – 197, 2002.

Fausett, L., *Fundamentals of Neural Networks: Architectures, Algorithms and Applications*, Prentice-Hall Inc., New Jersey, 1994.

Fernandes, F.A.N., Lona, L.M.F., *Neural network applications in polymerization processes*, Brazilian Journal of Chemical Engineering, 22 (3), 323 – 330, 2005.

Fyfe, C., *Artificial Neural Networks*, The University of Paisley, 1996.

Horn, D., Rieger, J., *Organic Nanoparticles in the Aqueous Phase–Theory, Experiment, and Use*, Angewandte Chemie International Edition, 40, 4330 – 4361, 2001.

Hornik, K., Stinchcombe, M., White, H., *Multilayer feedforward networks are universal approximators*, Neural Netwrks, 2 (5), 359 – 366, 1989.

Kasat, R.B., Gupta, S.K., *Multi-objective optimization of an industrial fluidized-bed catalytic cracking unit (FCCU) using genetic algorithm (GA) with jumping genes operator*, Computers and Chemical Engineering, 27, 1785 – 1800, 2003.

Khaw, J.F.C., Lim, B.S., Lim, L.E.N., *Optimal design of neural networks using the Taguchi method*, Neurocomputing, 7, 225 – 245, 1995.

Leon, F., *Perceptronii multistrat*, in: F. Leon (Ed.), Inteligența artificială – principii, tehnici, aplicații, Editura Tehnopress, Iași, 125 – 134, 2006.

Leon, F., Piuleac, C.G., Curteanu, S., *Stacked Neural Network Modeling Applied to the Synthesis of Polyacrylamide Based Multicomponent Hydrogels*, Macromolecular Reactor Engineering, 4, 591 – 598, 2010.

Llanos, J., Pérez, A., Rodrigo, M.A., Cañizares, P., *Electrochemical regeneration of partially ethoxylated polyethylenimine used in the polymer-supported ultrafiltration of copper*, Journal of Hazardous Materials, 168 (1), 25 – 30, 2009.

Maren, A., Harston, C., Pap, R., *Handbook of Neural Computing Applications*, Academic Press Inc., San Diego, California, 1990.

Nawaz Ripon, K.S., Kwong, S., Man, K.F., *A real-coding jumping gene genetic algorithm (RJGGA) for multiobjective optimization*, Information Sciences 177, 632 – 654, 2007.

Vasiliu, M., Cazacu, M., Marcu, M., Ioan, S., *Modified polysiloxanes I. New siloxane-azomethine alternating copolymers*, Polymer-Plastics Technology and Engineering, 44 (5), 993 – 1002, 2005.

Publicații ce vizează obiective rezolvate în teză

• Lucrări publicate în reviste de specialitate cotate ISI

1. Furtună, R., Curteanu, S., Cazacu, M., *Optimization Methodology Applied to Feed-Forward Artificial Neural Network Parameters*, **International Journal of Quantum Chemistry**, 111, 539 – 553, 2011a.

2. Furtună, R., Curteanu, S., Leon, F., *An elitist non-dominated sorting genetic algorithm enhanced with a neural network applied to the multi-objective optimization of a polysiloxane synthesis proces*", **Engineering Applications of Artificial Intelligence**, 24 (5), 772 – 785, 2011b.

3. Furtună, R., Curteanu, S., Racleş, C., *NSGA-II-JG Applied to Multi-objective Optimization of Polymeric Nanoparticles Synthesis with Silicone Surfactants*, **Central European Journal of Chemistry**, 9 (6), 1080 – 1095, 2011c.

4. Piuleac, C.G., Furtună, R., Pîslaru, M., Curteanu, S., *Neural networks based models applied to an electrocoagulation process*, **Environmental Engineering and Management Journal**, 10 (3), pag. 375 – 380, 2011.

5. Furtună, R., Curteanu, S., Leon, F., *Multi-Objective Optimization of a Stacked Neural Network Using an Evolutionary Hyper-Heuristic,* **Applied Soft Computing**, 12 (1), 133 – 144, 2012.

• Lucrări indexate în baze de date internaționale

1. Furtună, R., Curteanu, S., Leon, F., *Non-dominated Sorting Genetic Algorithm Used for Optimization of the Siloxane-Siloxane Copolymers Synthesis*, Petrol Gas University of Ploieşti Bulletin, 61 (3), 161 – 166, 2009.

2. Curteanu, S., Leon, F., Furtună, R., Dragoi, E.N., Curteanu, N., *Comparison between Different Methods for Developing Neural Network Topology Applied to a Complex Polymerization Process*, Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN), 1293 – 1300, iulie 2010, Barcelona, Spania.

3. Piuleac, C.G., Furtună, R., Curteanu, S., *Soft-computing Methodology Based on Neural Network and Genetic Algorithm Applied to a Photocatalytic Decomposition Process*, Bulletin of the Polytechnic Institute of Iaşi (acceptat spre publicare).

• Lucrări trimise spre publicare

1. Furtună, R., Piuleac, C.G., Curteanu, S., *Multi-Objective Optimization of a Stacked Neural Network Used in Modeling an Electrolysis Process*, Environmental Engineering and Management Journal, 2011d.

• Participări la manifestări stiințifice (comunicări, postere)

1. Furtună, R., Curteanu, S., Leon, F., *Non-dominated Sorting Genetic Algorithm Used for Optimization of the Siloxane-Siloxane Copolymers Synthesis,* Simpozionul „The 6th International Symposium on Process Control SPC 2009", organizat de Universitatea „Petrol-Gaze" din Ploieşti, Facultatea de Automatică şi Calculatoare, 1-3 iunie 2009.

2. Furtună, R., Curteanu, S., Racleş, C., *NSGA-II-JG Applied to Multi-objective Optimization of Polymeric Nanoparticles Synthesis with Silicone Surfactants,* Zilele Facultății de Inginerie Chimică şi Protecția Mediului, Editia a VI-a, „Noi frontiere in chimie şi inginerie chimică", Iaşi, 18-20 noiembrie 2009.

3. Furtună, R., Curteanu, S., Leon, F., *Pachet software cu trei module pentru rezolvarea problemelor de optimizare multi-obiectiv*, Catalogul Expoziției Europene a Creativității şi Inovării – Euroinvent, Iaşi, 7-9 mai 2010, pag. 129. ISBN: 978-973-702-769-6

4. Curteanu, S., Leon, F., Furtună, R., Dragoi, E.N., Curteanu, N., *Comparison between Different Methods for Developing Neural Network Topology Applied to a Complex Polymerization Process,* Conferința „2010 IEEE World Congres on Computational Intelligence" (WCCI 2010, IJCNN 2010), Barcelona, Spania, 18-23 iulie 2010.

• Distincții şi premii

1. Diplomă de participare la Simpozionul „The 6th International Symposium on Process Control SPC 2009", organizat de Universitatea „Petrol-Gaze" din Ploieşti, Facultatea de Automatică şi Calculatoare, 1-3 iunie 2009.

2. Diplomă de participare la Zilele Facultății de Inginerie Chimică şi Protecția Mediului, Editia a VI-a, „Noi frontiere in chimie şi inginerie chimică", Iaşi, 18-20 noiembrie 2009.

3. Diplomă de participare la Expoziția Europeană a Creativității şi Inovării – Euroinvent, Iaşi, 7-9 mai 2010.

• Proiecte finanțate

1. Contract tip „Idei", Id_592, nr. 59/2007, „Configuratii hibride softcomputing cu aplicații in chimie", 2009-2010, membru.

2. Contract PC nr. 71 – 006/2007, „Modelare şi conducere automată utilizând instrumente ale inteligenței artificiale pentru aplicații în chimie şi inginerie de proces", 2009-2010, membru.

• Brevete

1. Popa - Furtună, R., Curteanu, S., *Algoritm evolutiv multiobiectiv proiectat pentru optimizarea unei stive de rețele neuronale. NSGA-IIQNSNN*, înregistrat la O.R.D.A. (Registrul Național de Opere) sub numărul 14646/22.11.2010.