Universitatea Babeș-Bolyai Cluj-Napoca
Facultatea de Matematică și Informatică
Departamentul de Limbaje și Metode de Programare

Rezumatul Tezei de Doctorat

Technici Noi de Căutare și Optimizare Competentă

Autor:

David-Andrei ICLĂNZAN

Conducător de doctorat: prof. dr. Dumitru DUMITRESCU

Cluj-Napoca, 2010

Cuprins

Capitolul 1

Introducere

Posibilitatea creării unor sisteme inteligente, capabile să reproducă inteligenţa biologică umană(al homo sapiensului), a fascinat omenirea încă din cele mai vechi timpuri. Cercetarea sistematică şi roditoare a sistemelor inteligente a început odată cu apariţia calculatorului, cu dezvoltarea teoriei informaţiei şi cu descoperirile majore din domeniul neurologiei. Inteligenţa artificială (IA) a fost înfiinţată în cadrul ştiinţelor calculatoarelor, ca un domeniu major de cercetare şi studiu, al cărui scop este crearea unor sisteme inteligente, capabile în a se angaja în comportamente considerate inteligente.

Marea parte a problemelor din domeniul IA pot fi rezolvate prin efectuarea unor căutări inteligente în mulţimea soluţiilor existente (Russel & Norvig, 1995). Alan Turing a subliniat trei tipuri de căutare care pot conduce la comportament inteligent: abordarea logică, abordarea culturală şi abordarea evolutivă a căutării. (Turing, 1969)

Majoritatea algoritmilor de învăţare automată utilizează algoritmi de căutare-optimizare. În scenariile din relitate, căuterea neinformată şi exhaustivă nu este practică datorită spaţiilor imense de căutare. Procedurile informate de căutare folosesc euristici, fac unele presupuneri în vederea ghidării căutării, astfel reducând spaţiul de căutăre, eliminând acele opţiuni care sunt puţin probabile. Majoritatea tehnicilor de bias în căutare sunt inspirate de mehanismele ce acţionează în natură.

Această disertaţie cercetează aspectele adiţionale ale căutării informate, unde biasul este introdus într-o manieră inteligentă, fiind rezultatul analizei datelor şi a exploatării experienţei de căutare. În particular, aplicăm tehnici de învăţare automată în cazul scenariilor de optimizare a unor funţii necunoscute (black box optimization). Aceste cunoştinţe învăţate sunt utilizate în conjuncţie cu tehnici de optimizare pentru rezolvarea rapidă, sigură şi exactă a problemelor de optimizare.

Această disertaţie mai analizează tehnici care îmbunătăţesc scalabilitatea în

vederea manevrării eficiente a problemelor complexe şi a optimizării cu multe variabile şi a propune principii de căutare generale, ce permit căutări cu rezultate deschise (open-ended solutions) pentru anumite probleme.

1.1 Problematica studiată

Una din cele mai interesante problematici şi totodată una din cele mai mari provocări provine din studiul sistemelor complexe. Acest domeniu studiază modul în care relaţia dintre anumite părţi-unităţi se leagă de comportamentul colectiv al unui sistem şi modul în care acest sistem interacţionează şi stabileşte legătura cu mediul său.

Aceste sisteme sunt caracterizate de interacţiunile multiple dintre componentele sale, unde fenomene locale şi globale interacţionează în mod complicat, şi deseori în mod neliniar(Rind, 1999). În numeroase sisteme complexe, ce ne înconjoară, aceste interacţiuni nu se manifestă la un singur nivel. Acestea manifestă o organizare ierarhică, sistemul fiind compus din multiple sub sisteme, care şi ei la rândul lor au organizare ierarhică (Simon, 1969). Pentru a abordarea problemelor la o scală largă, trebuie utilizată o descompunere corectă a problemei.

Scopul acestei lucrări este dezvoltarea unei metodologii bazate pe principii teoretice solide şi a unui cadru general capabile să exploateze experienţa de căutare prin analiza datelor şi prin tehnici de învăţare automate, permiţând identificarea şi exploatarea dependenţelor şi a modularităţii. Componenta învăţării şi a adaptării extinde-dezvoltă metodele simple, mărind eficienţa, robusteţea şi cel mai important, mărind scalabilitatea acestora. Detectarea regulilor şi a structurii problemei din mers, efectuarea decompoziţiei dinamice, poate ghida căutarea, şi poate transforma o problemă grea într-una abordabilă, permiţând rezolvarea eficientă a acesteia.

1.2 Contribuţii

Contribuţiile majore în domeniu includ:

- Introducerea structurii de vecinătate adaptivă în optimizarea bazată pe căutare locală, unde strategiile căutării locale pot fi extinse şi se pot adapta utilizînd analiza datelor şi tehnici de învăţare automată.
- Construirea cadrului căutării locale bazate pe modele (Model Based Local Search) şi demonstrarea faptului că acesta poate rezolva probleme complexe de tip ierarhic în timp linearitmic.
- Introducerea unor funcţii de referinţă pentru testarea influenţei neutralităţii, multimodalităţii masive şi a scalabilităţi în cazul optimizării cu multe variabile.

- Demonstrarea robusteţii, scalabilităţii şi fiabilitatea Căutării Locale Bazate pe Model în cazul unor problemelor intratabile prin metode clasice.
- Eliminarea procesului de căutare a modelelor costisitoare prin introducerea a unor metode noi automatice de detectare a cuplajelor, conexiunilor, bazate pe sisteme neuronale.
- Introducerea de tehnici de creare a modelelor on-line, eficiente din punct de vedere al memoriei.
- Propunerea augmentării eficienţei şi diminuarea complexităţii din metodele de construire a modelelor, prin introducerea pre-filtrării prin analiza corelaţiilor dintre variabile.
- Dezvoltarea unei tehnici de construire de modele nesupervizate, fără evaluarea calitaţii modelului, rapid, scalabil, şi uşor paralelizabil, bazat pe algoritmul de clustering Markov. Tehnica propusă poate fi utilizat în crearea a trei categorii de modele: reţele Bayesian, modele cu conexiuni suprapuse şi modele cu produse marginale nesuprapuse.
- Investigarea fundamentelor algoritmilor evolutivi şi schiţarea proprietăţilor topologice a problemelor, în cazul cărora operatorul de încrucişare este unul eficient şi necesar.
- Propunerea unui model competent de căutare neconvenţional, open-ended, bazat pe cooperaţie, specializare şi exploatarea experienţei de căutare. Sinteza deschisă este garantată de partea neconvergentă a căutării, care explorează la nesfârşit noile caracteristici, care dacă sunt detectate, sunt asimilate de mehanisme convergente.

Acesta se bazează pe următoarele publicaţii:

(**Iclanzan & Dumitrescu, 2010**) David Iclanzan and D. Dumitrescu. Graph clustering based model building. In *PPSN XI - to appear in Lecture Notes in Computer Science*, Krakow, Poland, 11-15 September 2010. Springer. (accepted).

(**Iclănzan et al., 2010**) D. Iclănzan, D. Dumitrescu, and B. Hirsbrunner. Pairwise Interactions Induced Probabilistic Model Building. *Exploitation of Linkage Learning in Evolutionary Algorithms*, pages 97–122, 2010.

(**Iclanzan et al., 2009a**) David Iclanzan, D. Dumitrescu, and Béat Hirsbrunner. Correlation guided model building. In *GECCO '09: Proceedings of the 11th Annual conference on Genetic and evolutionary*

computation, pages 421–428, New York, NY, USA, 8-12 July 2009. ACM.

(**Szilágyi et al., 2009**) László Szilágyi, David Iclanzan, Sándor M. Szilágyi, D. Dumitrescu, and Béat Hirsbrunner. A generalized c-means clustering model optimized via evolutionary computation. In *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'09, Jeju Island, Korea)*, pages 451 – 455, 2009.

(**Iclanzan et al., 2009b**) David Iclanzan, Béat Hirsbrunner, Michèle Courant, and D. Dumitrescu. Cooperation in the context of sustainable search. In *IEEE Congress on Evolutionary Computation (IEEE CEC 2009)*, pages 1904 – 1911, Trondheim, Norway, 18-21 May 2009.

(**Szilagyi et al., 2009**) Sandor M. Szilagyi, Laszlo Szilagyi, David Iclanzan, and Zoltan Benyo. A weighted patient specific electromehanical model of the heart. In *Proc. 5th International Symposium on Applied Computational Intelligence and Informatics (SACI 2009)*, pages 105–110, Timisoara, Romania, 28-29 May 2009.

(**Szilágyi et al., 2008**) László Szilágyi, David Iclanzan, Sándor M. Szilágyi, and D. Dumitrescu. Gecim: A novel generalized approach to c-means clustering. In José Ruiz-Shulcloper and Walter G. Kropatsch, editors, *CIARP*, volume 5197 of *Lecture Notes in Computer Science*, pages 235–242. Springer, 2008.

(**Iclanzan & Dumitrescu, 2008c**) David Iclanzan and D. Dumitrescu. Large-scale optimization of non-separable building-block problems. In Günter Rudolph, Thomas Jansen, Simon M. Lucas, Carlo Poloni, and Nicola Beume, editors, *PPSN*, volume 5199 of *Lecture Notes in Computer Science*, pages 899–908. Springer, 2008.

(**Iclanzan & Dumitrescu, 2008d**) David Iclanzan and D. Dumitrescu. Towards memoryless model building. In *GECCO '08: Proceedings of the 2008 GECCO conference companion on Genetic and evolutionary computation*, pages 2147–2152, Atlanta, GA, USA, 2008. ACM.

(**Iclanzan & Dumitrescu, 2008a**) David Iclanzan and D. Dumitrescu. Going for the big fishes: Discovering and combining large neutral and massively multimodal building-blocks with model based macromutation. In *GECCO '08: Proceedings of the 10th annual conference on Genetic and evolutionary computation*, pages 423–430, Atlanta, GA, USA, 2008. ACM.

(Iclanzan & Dumitrescu, 2008b) David Iclanzan and D. Dumitrescu. How can artificial neural networks help making the intractable search spaces tractable. In *2008 IEEE World Congress on Computational Intelligence (WCCI 2008)*, pages 4016–4023, Hong-Kong, 01-06 June 2008.

(Iclanzan & Dumitrescu, 2007c) David Iclanzan and D. Dumitrescu. Overrepresentation in neutral genotype-phenotype mappings and their applications. In *Symbolic and Numeric Algorithms for Scientific Computing, 2007. SYNASC. International Symposium on*, pages 427–432, Timisoara, Romania, 26-29 September 2007. IEEE Computer Society.

(Iclanzan et al., 2007) David Iclanzan, P.I. Fulop, and D. Dumitrescu. Neuro-Hill-Climber: A new approach towards more intelligent search and optimization. In *Symbolic and Numeric Algorithms for Scientific Computing, 2007. SYNASC. International Symposium on*, pages 441–448, Timisoara, Romania, 26-29 September 2007. IEEE Computer Society.

(Iclanzan, 2007a) David Iclanzan. The creativity potential within Evolutionary Algorithms. In Fernando Almeida e Costa et al., editor, *Advances in Artificial Life, 9th European Conference, ECAL 2007, Lisbon, Portugal, September 10-14, 2007, Proceedings*, volume 4648 of *Lecture Notes in Computer Science*, pages 845–854. Springer, 2007.

(Iclanzan, 2007b) David Iclanzan. Crossover: the divine afflatus in search. In Peter A. N. Bosman, editor, *Late breaking paper at Genetic and Evolutionary Computation Conference (GECCO'2007)*, pages 2497–2502, London, United Kingdom, 7-11 July 2007. ACM Press.

(Iclanzan & Dumitrescu, 2007b) David Iclanzan and D. Dumitrescu. Overcoming hierarchical difficulty by hill-climbing the building block structure. In Dirk Thierens et al., editor, *GECCO '07: Proceedings of the 9th annual conference on Genetic and Evolutionary Computation*, volume 2, pages 1256–1263, London, 7-11 July 2007. ACM Press.

(Iclanzan & Dumitrescu, 2007a) David Iclanzan and D. Dumitrescu. Exact model building in Hierarchical Complex Systems. In *Studia Universitatis Babes-Bolyai, Informatica Series*, volume Special Issue: KEPT 2007 - Knowledge Engineering: Principles and Techniques, Proceedings, pages 161–168, Cluj-Napoca, Romania, 6-8 June 2007. Universitas Napocensis, Presa Universitara.

(**Szilagyi et al., 2006c**) Sandor M. Szilagyi, Laszlo Szilagyi, David Iclanzan, and Zoltan Benyo. Unified neural network-based adaptive ECG signal analysis and compression. *SB-UPT TACCS*, 56(65)(4):27–36, 2006.

(**Iclanzan et al., 2006**) David Iclanzan, Sandor M. Szilagyi, Laszlo Szilagyi, and Zoltan Benyo. Advanced heuristic methods for ECG parameter estimation. In *CONTI 2006: Proceedings of the 7th International Conference on tehnical Informatics*, pages 215–220, Timisoara, Romania, 8-9 June 2006. Universitatea Politechica.

(**Szilagyi et al., 2006b**) Sandor M. Szilagyi, Laszlo Szilagyi, David Iclanzan, and Zoltan Benyo. Adaptive ECG signal analysis for enhanced state recognition and diagnosis. In *CONTI 2006: Proceedings of the 7th International Conference on tehnical Informatics*, pages 209–214, Timisoara, Romania, 8-9 June 2006. Universitatea Politechica.

(**Szilagyi et al., 2006a**) Laszlo Szilagyi, Sandor M. Szilagyi, David Iclanzan, and Zoltan Benyo. Quick ECG signal processing methods for on-line holter monitoring systems. In *CONTI 2006: Proceedings of the 7th International Conference on tehnical Informatics*, pages 221–226, Timisoara, Romania, 8-9 June 2006. Universitatea Politechica.

(**Iclanzan & Dumitrescu, 2006b**) David Iclanzan and D. Dumitrescu. ECG parameter estimation using advanced stochastic search methods. In Dorin Isoc and Eugen Stancel, editors, *2006 IEEE-TTTC International Conference on Automation, Quality and tesing, Robotics. Digest of Junior Section*, pages 17–22, Cluj-Napoca, Romania, 25-28 May 2006. Universitatea tehnica Cluj.

(**Iclanzan & Dumitrescu, 2006a**) David Iclanzan and D. Dumitrescu. Competitive vs. cooperative optimization. In *Proceedings of the 8th International Scientific Student Conference on tehnical Sciences*, pages 115–121, Timisoara, Romania, 7-9 April 2006. Universitatea de Vest. Distributed on CD.

(**Iclanzan, 2006**) David Iclanzan. Genetic engineering as an optimization paradigm. In *Proceedings of FMTU 2006*, pages 115–121, Cluj-Napoca, Romania, 24-25 March 2006. Transylvanian Musem Society.

(**Iclanzan, 2005**) David Iclanzan. Genetic engineering algorithm. In *Proceedings of the 7th International Scientific Student Conference on tehnical Sciences*, Timisoara, Romania, 22-24 April 2005. Universitatea de Vest. Distributed on CD.

1.3 Keywords

Îmbunătăţirea eficacităţii, construirea modelelor, învăţare automată, structură de vecinătate adaptivă, căutare locală bazată pe model.

Capitolul 2

Background and Related Work

Acest capitol oferă o introducere succintă în domeniul optimizării globale, accentul fiind pus pe algoritmii evolutivi. Sunt discutate insuficienţele metodelor clasice, care conduc la convergenţă prematură şi a metodelor cu reprezezentaţii şi operatori ficşi. Ultima parte prezintă ideile de bază ce se regăsesc în spatele metodelor competente, destinate să garanteze o evoluţie continuă. Se discută scurt costul şi modul în care construcţia modelelor poate ameliora problema disrupţiei blocurilor de construcţie.

2.1 Optimizarea globală

2.2 Algoritmi Evolutivi

2.2.1 Algoritmi Genetici

2.2.2 Programarea și Strategiile Evolutive

2.2.3 Programarea Genetică

2.2.4 Problemele prezentate de metodele clasice evolutive

2.3 Calcul Evolutiv sustenabil și Clasa Metodelor Competente

2.3.1 Convergența Prematură

2.3.2 Blocuri de Construcții și Construirea de Modele

Capitolul 3

Construirea de Modele Ghidată de Corelații

Acest capitol prezintă rezultate de bază, care demonstrează modul simplu în care datele de corelaţie pot fi extinse şi utilizate pentru a ghida eficient căutarea modelelor, scăzând numărul necesar de evaluare a modelelor cu ordini de magnitudine, fără a afecta semnificativ calitatea modelului. Ca şi un studiu de caz, înlocuim modelul $O(n^3)$ al Algoritmului Genetic Compact Extins cu o căutare ghidată de corelaţie, cu o complexitate lineară.

3.1 Introducere

3.1.1 Metrici generale pentru măsurarea nivelului de interacțiune dintre module

Fie M_R, conţinând valorile absolute ale matricei de corelaţie, corelaţia cu sine fiind setată la zero, adică $M_R(x,x) = 0$. Prima metrică utilizată face o medie al diferitelor interacţiuni pereche a variabilelor prezente în module:

$$d_1(X,Y) = \frac{\sum_{x \in X} \sum_{y \in Y} M_R(x,y)}{\binom{|X \bigcup Y|}{2}} \tag{3.1}$$

Calculul d_1 penalizează încorporarea componenţilor non corelate, astfel este influenţat de valori apropiate de 0. Pentru a cuantifica interacţiunile chiar şi printre subunităţi, introducem o funcţie, ce sumează interacţiunile:

$$d_2(X,Y) = \sum_{x \in X} \sum_{y \in Y} \sigma(x,y) \cdot M_R(x,y) \tag{3.2}$$

where

$$\sigma(x,y) = \begin{cases} 1 & \text{, if } M_R(x,y) \text{ is statistically significant;} \\ 0 & \text{, otherwise.} \end{cases} \tag{3.3}$$

În d_2 componentele necorelate nu pot afecta în mare măsură rezultatul. Acesta înseamnă, totodată, că formarea modulelor prea complexe nu va fi penalizată. Cele doua funcţii sunt complementare şi pot fi folosite concomitent. Dacă d_2 are valori înalte, iar d_1 are valori scăzute, trebuie luat în considerare, divizarea modelului în multiple părţi: există interacţiuni puternice, dar nu toate variabilele (x, y) sunt fi corelate.

3.2 Studiu de caz pe eCGA

Algoritmul Genetic Compact Extins (eCGA) (Harik, 1999) este o extensie multivariabilă al Algoritmului Genetic Compact bazat pe principiul fundamental, conform căruia, învăţarea unei bune probabilităţi de distribuţie a populaţiei, este echivalentă cu procesul de învăţare al conexiunilor. Măsura unei distribuţii bune poate fi cuantificată, utilizând principiul de Descriere de Lungime Minimă (LMD). LMD este bazat pe conceptul, că orice regularitate dintr-o serie de date poate fi utilizată pentru comprimarea datelor.

Pornind de la o populaţie aleatorie, eCGA aplică procese de evaluare, selecţie, construire de modele şi eşantionare, până ce criteriile de oprire sunt îndeplinite.

Construirea de modele în eCGA presupune un calcul costisitor pentru determinarea celui mai potrivit model. Căutarea modelului are o complexitate $O(n^3)$ în numărul de evaluări a modelului.

3.2.1 Hibridizarea eCGA

O metodă frecvent utilizată pentru creşterea eficienţei, este hibridizarea cu tehnicile de căutare locale. În cazul eCGA, încorporarea căutării locale în spaţiul de căutarea a subsoluţiei duce la obţinerea umor rezultate îmbunătăţite (Lima et al., 2005).

3.2.2 Construirea de model ghidată

Pentru determinarea unui model adecvat, eCGA utilizează o metodă de căutare greedy în spaţiul modelelor posibile, evaluând *toate* combinaţiile pereche dintre partiţii. Modelul este extins secvenţial cu fuziunea modulelor ce duc la cea mai mare îmbunătăţire. Ideea principală a căutării propuse este de a nu procesa în mod orb şi epuizant lista posibilelor extensii, ci a alege cele mai bune extensii pe baza analizei corelaţiei dintre partiţii. Căutarea se va opri imediat ce extinderea modelului nu va fi urmată de îmbunătăţirea criteriei de complexitate combinate. Motivaţia stă în faptul că toate fuziunile neanalizate, vor avea un grad mai mic de

Algorithm 1: Correlation guided model-building

```
1  Build initial model m where each variable is an independent partition;
2  Compute M_R and M_T;
3  if this is the first generation and there are no significant values in M_T then
4  |  Request a bigger population and suggest performing search for second
   |  order interactions;
5  |  Halt the search;
6  repeat
7  |  [p, q] ← StrongestInteraction(m, M_R, d);
8  |  Form new model m' based on m but with p and q merged into a joint
   |  partition;
9  |  Evaluate combined complexity criterion C_c(m');
10 |  if m' improves over m then
11 |  |  m ← m';
12 until No improvement was found;
```

interacţiune, decât ultima extensie propusă. Dacă această ultimă extensie a fost eliminată de criteriul combinat de complexitate(C_c), atunci nici cele rămase nu vor fi luate în considerare, deci nu mai există motiv pentru continuarea căutării.

Construirea de model, ghidată de corelaţie este prezentată în Algoritmul 1. În termenii evaluării complexităţii combinate, metoda propusă este foarte eficientă, fiind lineară.

3.3 Teste

Testăm eCGA cu construirea de model ghidată de corelaţie, prin două probleme, ce combină esenţa a două dimensiuni de dificultate bine cunoscute a problemelor:

- Dificultatea Intra-BB: funcţii capcană *deceptive*.
- Dificultatea Extra-BB: dependenţele non lineare datorită structurii ierarhice, care la un nivel simgular se manifestă ca şi *zgomot* exogen –generat de interacţiuni din nivele mai înalte.

3.3.1 Funcţia capcană k-Trap

3.3.2 XOR Ierarchic

3.4 Rezultate

Modul de funcţionare liniară a construirii modelelor, ghidate de corelaţie, din eCGA oferă un avantaj imens *calitativ* faţă de construirea de model clasic de complexitate $O(n^3)$. O construire de model euristică de complexitate $O(n^2)$ poate

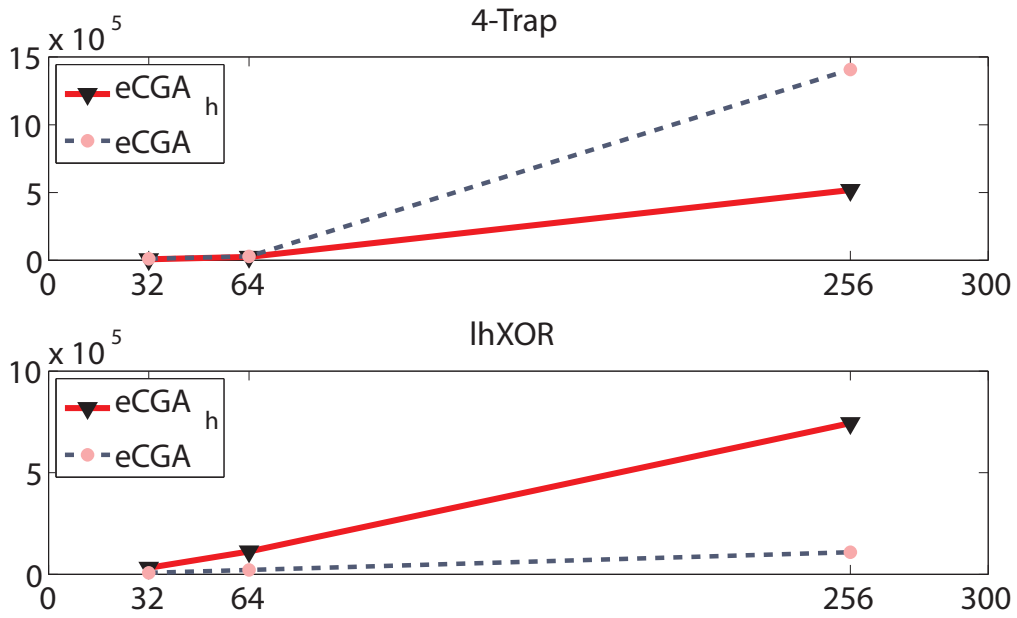


Figura 3.1: Scalabilitatea eCGA ghidată de corelaţii cu şi fără hibridizare cu tehnici de căutare locală (eCGA$_h$) în cazul funcţiei capcană concatenat 4-Trap şi lhXOR cu $l = 3$.

accelera eCGA-ul cu până la 1000 de ori. (Duque et al., 2008). Astfel, ne concentrăm asupra investigării empirice a scalării, calităţii modelului- numărul de generaţii până la apariţia convergenţei, şi efectele hibridizării, în loc să oferim o analiză cantitativă a timpilor de rulare.

3.4.1 Problemele test

Am testat eCGA ghidat de corelaţie cu şi fără hibridizare a căutării locale(denumită eCGA$_h$) pe funcţia capcană concatenat 4-Trap şi 1hXOR cu $l = 3$ pentru probleme de mărimea $psize = \{32, 64, 256\}$. Mărimile populaţiilor sunt $15psize$ pentru eCGA$_h$ şi $55psize$ pentru eCGA. Aceste valori nu sunt ajustate spre optimalitate. Pentru fiecare test s-au efectuat 10 rulări, calculându-se mediile. Rezultatele sunt prezentate prin Figura 6.1.

3.4.2 Analysis

Algoritmii au localizat structurile corecte şi au atins optimul global. Figura 6.1 ilustrează numărul de evaluări ale funcţiilor, necesare pentru fiecare problemă.

Capitolul 4

Căutarea Locală Bazată pe Modele

În acest capitol introducem cadrul pentru Căutarea Locală Bazată pe Modele, şi o căutare greedy, care operează în spaţiul blocurilor de construcţie, capabil de a aborda eficient probleme ierarhice, prin asimilarea-învăţarea structurii problemei din experienţa de căutare. Structura de vecinătate este adaptată ori de câte ori cunoştinţele noi referitoare la structurile de blocuri sunt descoperite, acestea fiind încorporate în căutare. Această metodă permite ascensiunea în structura ierarhică, prin dezvăluirea şi rezolvarea consecutivă a nivelelor ierarhice.

Demonstrăm că pentru probleme cu blocuri de construcţii organizate ierarhic, complet nedeceptive abordarea propusă poate rezolva aceste probleme într-un timp linearitmic, depăşind performanţele metodelor combinanativee bazate pe populaţii.

4.1 Motivaţie

Una dintre problematica de bază a Algoritmilor Evolutivi constă în determinarea clasei de probleme pentru care aceşti algoritmi pot fi utilizaţi-aplicaţi cu o mare eficacitate.

În ciuda existenţei unei vaste cantităţi de cunoştinţe acumulate în timp în acest domeniu, încă este neclar modul în care AE expolrează spaţiul de căutare. Este încă neclar şi pentru care funcţii acestea vor depăşi performanţele altor optimizatori.

4.2 Funcţii Decomposabile Ierarchic

4.2.1 Probleme Ierachice

Daca şi numai dacă ierarhic (hIFF)

XOR ierarhic (hXOR)

Funcţia capcană ierarhic(hTrap)

4.2.2 tehnici de Hill-climbing şi structuri de vecinătate

4.2.3 Căutarea în spaţiul blocurilor de construcţie

Problemele ierarhice sunt complet deceptive în spaţiul Hamming şi complet ne deceptive în spaţiul blocurilor de construcţie. Reprezentarea problemei concomitent cu structura vecinătăţii definesc topologia spaţiului de căutare. Recent, autori în domeniul căutării locale, au accentuat importanţa utilizării unui operator de vecinătate bun (Watson et al., 2003)

Cu o structură de vecinătate adecvată - care operează în spaţiul blocurilor- problemele de căutare pot fi transferate din spaţiul Hamming într-un peisaj de căutare agreabil, complet ne deceptiv, unde utilizarea urcării în direcţia celei mai bune schimbări ar fi facilă.

Această abordare aplică combinarea şi analiza sistematică a blocurilor, pe când AE exploatează structurile de blocuri prin recombinare probabilistică.

4.2.4 Adaptare online

Este important ca un AG (algoritm genetic) să prezerve blocurile pe parcursul evoluţiei simulate, aplicând operatorul de încrucişare. Studiile teoretice afirmă, că acei algoritmi care nu alterează structura blocurilor prin încrucişare, deţin numeroase avantaje faţă de AG simpli(Thierens & Goldberg, 1993). Pentru a atinge acest scop, se aplică învăţarea dependenţelor şi reprezentarea soluţiei este evoluată-schimbată concomitent cu populaţia.

n mod similar, pentru a putea aplica un algoritm hill-climbing pe blocuri de construcţie, modularitatea problemei trebuie cunoscută şi reprezentarea indivizilor trebuie evoluată, pentru a reflecta cunoşinţele despre blocurile acutale. Modificarea reprezentaţiei presupune adaptarea structurii vecinătăţii, care este cheia rezolvării problemelor ierarhice: prin explorarea vecinătăţii configuraţiei de blocuri curente, poate fi detectată următorul nivel de blocuri.

4.3 Hill-Climber pe blocuri de construcţii

Hillclimbing pe blocuri presupune 4 etape principale: (i) Iniţializarea algoritmului la fiecare locus ca un bloc- element, (II) urcarea în direcţia celor mai favorabile

modificări în concordanţă cu structura de vecinătate actuală,(III) optimizarea locală obţinută la (II) este utilizată pentru a detecta dependenţe şi pentru extragerea informaţiei despre blocuri, (IV) actualizarea configuraţiei blocurilor şi în mod implicit, a structurii de vecinătate. Aceast capitol descrie cadrul-contextul MBLS (Model Based Local Search) şi detaliile implementării al acesteia.

4.3.1 Cadrul de lucru a Căutarii Locale Bazată pe Modele

Figura 4.1 ilustrează cele două faze principale ale căutării locale bazate pe model, optimizarea MBLS. Prima se referă la acumularea experienţei de căutare, oferită de aplicarea căutării locale în mod repetat. A doua fază reprezintă exploatarea experienţei de căutare prin învăţarea dependenţelor şi actualizarea structurii blocurilor.

Datele de intrare a celei de a doua faze este reprezentată de experienţa căutării stocată în memorie. Legăturile sunt detectate şi datele de ieşire sunt reprezentate de actualizarea structurii blocurilor, care permit primei faze ansamblarea-combinarea noilor blocuri. În cazul problemelor ierarhice, modelate conform BBH, combinarea blocurilor de grad scăzut va produce crearea blocurilor de grad mai înalt. Astfel aplicarea secvenţială a fazelor poate învinge nivelele ierarhice succesive, incorporând cunoştinţele despre blocuri în procesul de căutare.

4.3.2 Hill-climbing pe blocuri de construcție

4.3.3 Detectarea conexiunilor

Problemele ierarhice studiate deţin structură nedeceptivă în spaţiul de blocuri, astfel pentru detectarea cuplărilor este propusă o metodă foarte simplă.

Setarea locusurilor în blocuri noi este efectuată prin căutarea aplicaţiilor bijective (corelaţie maximă).

Datorită tranzitivităţii aplicaţiilor bijective, toate blocurile semnificante sunt descoperite simultan. Algoritmul de detectare a cuplajului este prezentat in Figura 4.4.

Toate blocurile, cuplate de o aplicaţie bijectivă, sunt unite într-un bloc nou. Configuraţiile importante ale blocurilor noi sunt extrase din reprezentaţiile binare din memorie. Dacă un bloc nu poate fi cuplat-conectat cu alt loc, acesta îşi va păstra locaţia originală, şi configuraţiile sale posibile vor fi actualizate în aceiaşi manieră ca şi în cazul blocurilor noi.

4.3.4 Mărimea memoriei

Metoda propusă este sumarizată în Figura 4.5.

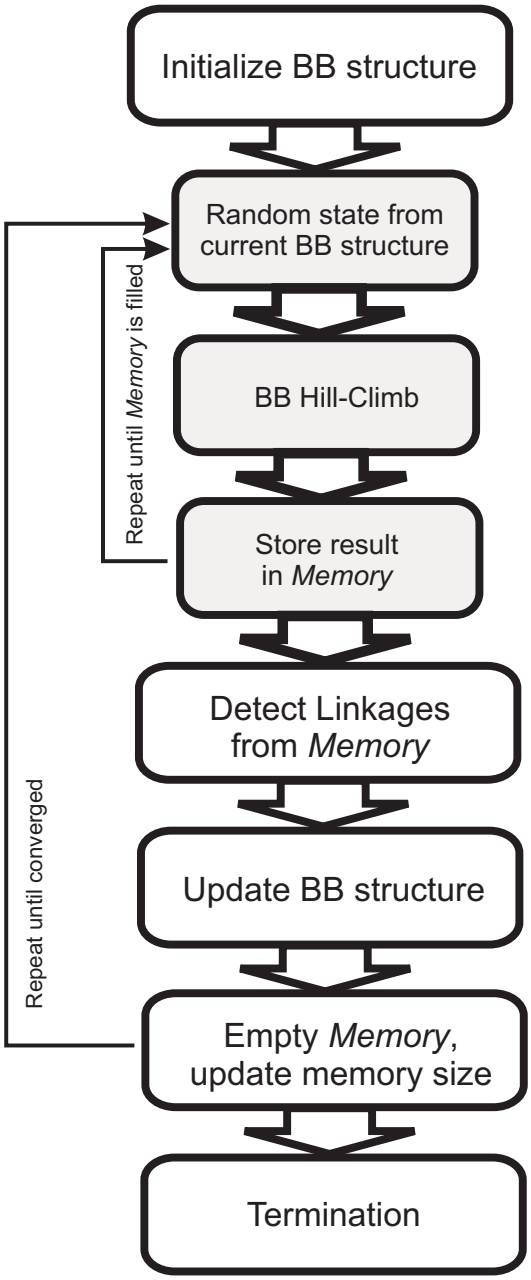


Figura 4.1: Cadrul de lucru general cu două mari etape: accumularea şi exploatarea experienţei de căutare.

4.4 Rezultate

Am testat scalabilitatea BBHC pe problemele hIFF şi hXOR de mărimi 128-biţi, 256-biţi, 512-biţi şi 1024-biţi, respectiv pe probleme hTrap cu mărimi de 81-biţi, 243-biţi şi 729-bişi. Problemele de mărimi mai mici nu au fost adresate deoarece

ar fi fost prea uşoare de rezolvat. Pentru fiecare test s-au efectuat în medie un număr de 100 rulări independente.

Rezultatul scalării aritmetice împreună cu proporţia dintre punctele învecinate sunt prezentate în Figura 6.1. În setul de teste, rezultatele metodei propuse au crescut aproape linear cu mărimea problemei, panta între două puncte vecine scăzînd spre 2 odată cu dublarea mărimii problemei în cazul hIFF is hXOR, şi în cazul hTrap a scăzut spre 3 odată cu triplarea marimii problemei.

Rezultatele experimentului au fost aproximate prin funcţii cu forma $f(x) = ax^b \cdot log(x)$ unde a şi b sunt determinate prin metoda erorii minime pătratice. În Figura 4.7 sunt reprezentate diagramele scalate ale rezultatelor testelor şi a funcţiilor de aproximaţie. Numărul aproximativ al evaluării funcţiilor obiectiv este de $O(l^{0.97} \cdot log(l))$ în cazul hIFF şi hXOR, şi $O(l^{0.91} \cdot log(l))$ în cazul hTrap, unde l reprezintă mărimea problemei. Aproximările sunt foarte apropiate de timpurile linearitmice aşteptate.

hBOA-ul, unul dintre cei mai buni optimizatori care operează prin decompoziţia ierarhică, cu parametrii ajustaţi manual, rezolvă problema hIFF de 256 biţi cu aproximativ 88000 evaluări. Performanţa BBHC pe acelaşi set de test este de 20666, aproximativ de 4 ori mai rapid decât hBOA. Datorită scalării linearitmice BBHC este capabil să rezolve versuinea de 512 biţi a problemei de 2 ori mai rapid decât hBOA o rezolvă pe cea de 256 biţi, efectuând doar 45793 de evaluări în medie. Similar altor metode, cum ar fi DSMGA++, BBHC folosesc metode explicite de descompunere, permiţând metodei transmiterea structurii problemei. În timp ce DSMGA++ şi alte metode stocastice se confruntă cu erorile eşantionării, ce pot cauza imperfecţiunii, BBHC poate detecta perfect structura problemei la fiecare ciclu, datorită abordării sistematice şi deterministice.

Capacitatea avansată de a dezvălui structura problemei este reflectată de faptul că hIFF şi hXOR sunt rezolvate în aproximativ aceiaşi număr de paşi, căci structurile de blocuri ce stau la baza lor (un arbore binar echilibrat) coincid. Pentru DSMGA++ timpul de optimizare a acestor două probleme diferă semnificativ, fiind $O(l^{1.84} \cdot log(l))$ pentru hIFF şi $O(l^{1.96} \cdot log(l))$ pentru hXOR.

4.5 Sumar

Testele de scalabilitate pentru metoda propusă indică faptul că BBHC deţine nu doar avantajul cantitativ faţă de alte metode ci şi un avantaj calitativ: scalează lineatrimic cu mărimea problemei.

Figura 4.2: Configuraţia blocurilor de construcţie pentru 8-biţi.

HC(s)

1. Choose randomly an unprocessed building block b_i from $B(s)$;
2. Choose randomly an unprocessed building block configuration $v_j \in V_i$;
3. Set $v(b_i)$ in s to v_j;
4. If the change results in a decrease of the objective function undo the change;
5. If there exists unprocessed building block configuration of V_i then *goto 2*;
6. If there exists unprocessed building block from $BB(s)$ then *goto 1*;

Figura 4.3: Căutarea greedy în spaţiul blocurilor de construcţie.

BBForm(s, M)

1. Choose randomly a building block b_i from $BB(s)$ which has not yet been clustered;
2. Compute the set of building blocks whose configuration from M are mapped bijectively to b_i and denote it by L;
3. If L is empty update the possible configurations V_i to the configurations encountered in M;
4. If L is not empty form a new building block $b_{new} = b_i \bigcup L$ from the union of loci from b_i and from building blocks in L. Also set the possible values V_{new} to all distinct configuration encountered, on the position defined by the b_{new}, operating on the binary representation of states from M.
5. Set b_i and the building blocks from L as clustered;
6. If there exists building blocks which have not been clustered *goto 1*;

Figura 4.4: Detectarea dependenţelor

BBHC(x, c, b, k) returns *best_state*

1. Initialize the building block knowledge with each single locus from x as a building block;
2. Initialize the memory size:
   $size[M] := c + log_b(length(x))$;
3. Generate a random state s according to the current building block structure knowledge $BB(s)$:
   $s := RandomState(BB(s))$;
4. building block hill-climb from s and store the result in memory: $M := M \bigcup HC(s)$;
5. If the resulted state is better then the best states seen so far, keep the new state:
   $s := best(s, best\_state)$
6. If M is not filled up *goto 3*;
7. Learn linkage from memory and update the building block configuration according to the detected linkages: $BBForm(s, M)$;
8. Empty memory: $M = \emptyset$;
9. Update the memory size:
   $size[M] := c + log_b(\aleph(BB(s)))$;
10. If there was an improvement in the last k epochs and the number of maximum function evaluations was not exceeded *goto 3*;

Figura 4.5: Outline of the hill-climbing enhanced with memory and linkage learning. In steps 3–6 we accumulate the search experience (phase 1) which is exploited in steps 7–9 (phase 2).

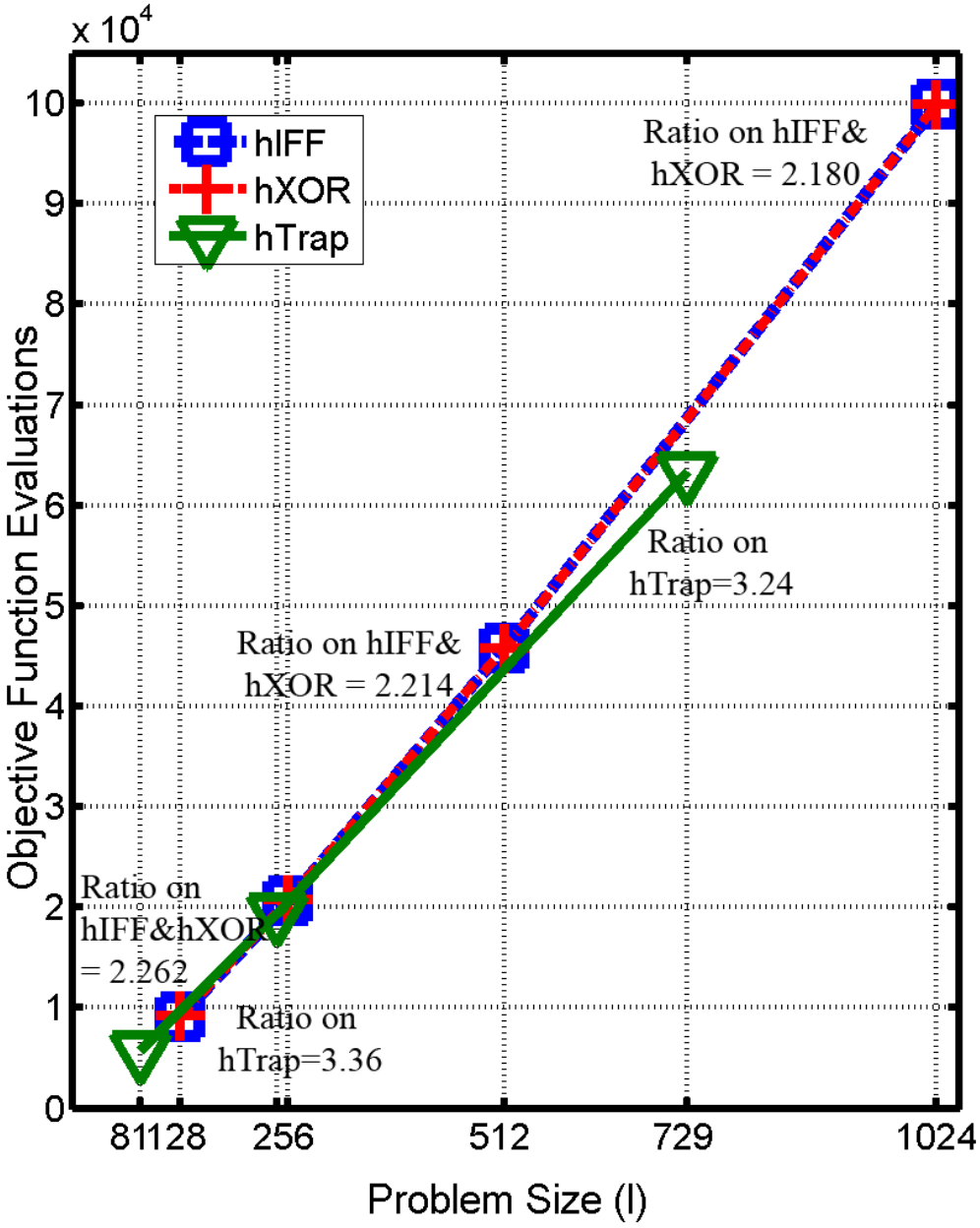


Figura 4.6: Scalarea BBHC pe hIFF, hXOR şi hTrap.

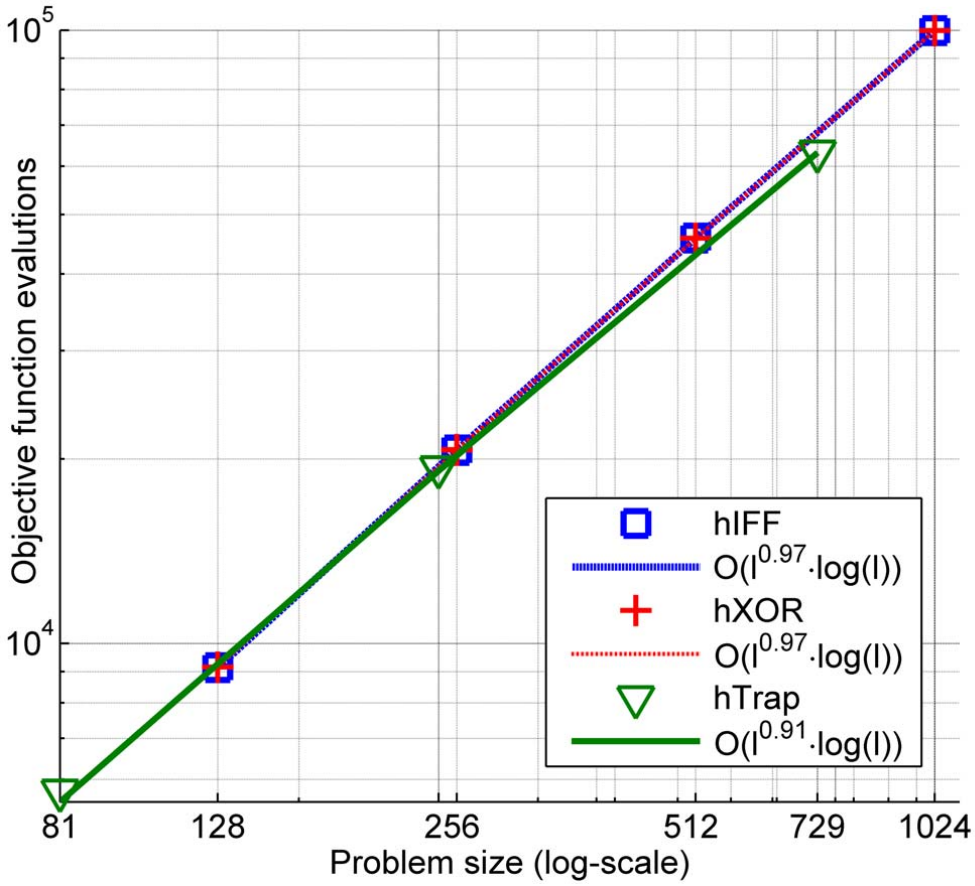


Figura 4.7: Numărul evaluărilor de funcţii obiectiv al BBHC creşte $O(l^{0.97} \cdot log(l))$ în cazul hIFF şi hXOR şi $O(l^{0.91} \cdot log(l))$ în cazul hTrap, unde l este mărimea problemei.

Capitolul 5

Soluții Pentru Neutralitate și Deceptivitate

În acest capitol prezentăm o metodologie competentă, capabil de combinarea şi detectarea eficientă a modulelor, chiar şi în cazul cuplajului genetic nefavorabil, fără gradient de adaptare intra-bloc, care să îndrume căutarea sau în cazul deceptivităţii. Metodologia este construită pe cadrul Căutării Locale Bazate pe Model, prezentat în capitolul anterior.

Învingerea dificultăţilor este realizată prin folosirea unei căutări bazate pe model, cu putere exploratorie marcată şi prin restricţionarea creării modelelor la un număr relativ de redus de mostre semi-convergente.

5.1 Introducere

5.2 Costuri

5.2.1 Mărimea Populațiilor și Costul Computațional al Modelelor în cazul PMBGAs

5.3 Funcții test

5.3.1 Funcția Shuffled Royal Road Extinsă

5.3.2 Funcția Ierarchical Masiv Multimodală Deceptivă

5.4 Macro-Mutație Bazată pe Modele

Căutarea se bazează pe cadrul MBLS (căutare locala bazată pe model) prezentat în capitolul 4, folosind pentru căutare un operator de macro mutaţie.

5.4.1 Reprezentație

5.4.2 Hill-Climber bazat pe Macro-Mutație

Hill-climbing bazat pe Macro Mutaţie(MMHC), este o metodă cu putere exploratorie mare, care depăşeşte performanţele AG chiar şi în cazul în care fiecare bloc corespunde unei funcţii capcană, cu condiţia ca problema să conţine dependenţe organizate în module compacte (Jones, 1995).

5.4.3 Învățarea structurii

Învățarea utilizând rețele Kohonen

Am utilizat o hartă cu organizare proprie - reţea neuronală Kohonen (SOM) pentru detectarea dependenţelor. SOM sunt antrenaţi prin *învăţare nesupravegheată* pentru a produce o reprezentaţie discretizată şi bidimensională a mostrelor de pregătire din spaţiul datelor de intrare, numită hartă. Caracteristica interesantă a SOM, exploatată în acest caz, este reprezentarea *prezervată a topologiei*, astfel date similare vor avea pondere similară. Legăturile sunt deduse din reprezentarea internă a SOM bazată pe idea euristică, conform căreia datele introduse similare produc tipare similare in asocierea lor cu ponderea, ex. datele de intrare subordonate tind să aibă valori similare în ponderi.

Algorithm 2: Macro Mutaţie bazată pe modele

Data: $M, V, n_S, z, c_2, @stopping\_cond, \epsilon_1$.

while *not* $@stopping\_cond$ **do**
  $n \leftarrow |M(s)|$;
  /* Phase I */
  for $i = 1, n_S$ **do**
    /* Generate a random state s according to the current building-block knowledge $M(s)$ */
    $s \leftarrow RandomState(M)$;
    /* Apply macro-mutation according to the current model for $c_2 n^2$ evals */
    $s \leftarrow MBMM(s, [M, V], c_2 n^2)$;
    $mem[i] \leftarrow s$;
  /* Phase II */
  $T \leftarrow NormalizeDataRanges(mem)$;
  $F \leftarrow 0_{n \times n}$;
  for $l = 1, z$ **do**
    $net \leftarrow InitializeSOM()$;
    /* Randomly select four samples */
    $S \leftarrow ChoseRandomSamples(T, 4)$;
    $net \leftarrow Train(net, S)$;
    /* Detect possible modules via weight analysis */
    $nm \leftarrow GetLinkages(net.Weights, \epsilon)$;
    /* Update the frequency matrix */
    $F \leftarrow Update(F, nm)$;
  /* Get modules from the frequency matrix */
  $b \leftarrow GetBaseModules(F, \epsilon_1)$;
  /* Merge overlapping modules */
  $b \leftarrow unique(\{b_i = b_i \bigcup b_j \text{ if } b_i \bigcap b_j \neq \emptyset; \forall i, j\})$;
  /* Collapse the search space and update the building-block configuration according to the detected modules */
  $[M, V] \leftarrow UpdateModuleKnowledge(b)$;

Filtrarea zgomotului

Construirea de model bazată pe Macro mutaţie este ilustrat pin algoritmul 2.

5.5 Rezultate

Performanţa metodei propuse a fost testată prin funcţia $ESRR$ cu modele de bază de mărime egală, mărimile variind de la 8 a 16 şi prin funcţia $HMMD$ clasă varia între 6 şi 8. În cadrul funcţiei $ESRR$ parametrii ce favorizau anumit configuraţii de blocuri au fost setate la valorile $r_1 = 1, r_2 = 2, r_3 = 4$. Funcţiile

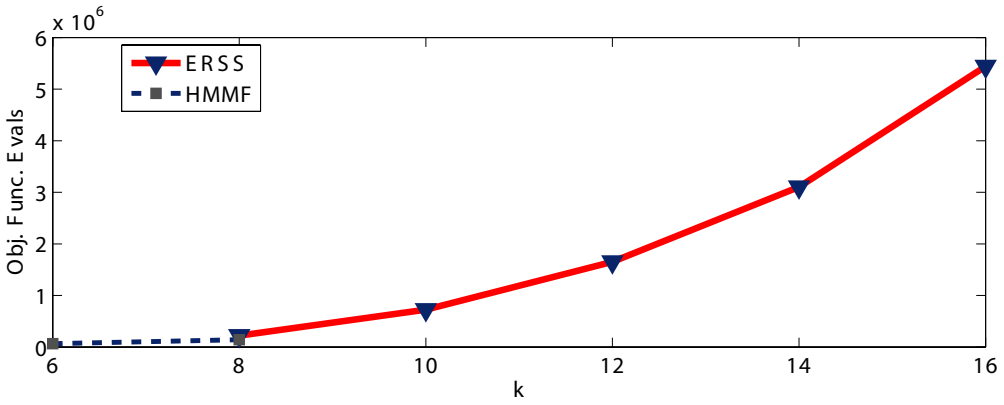


Figura 5.1: Performanţa şi scalarea MBMM pe problemele de test.

$HMMD$ au fost utilizaţi cu parametrii $r_1 = 1$ and $r_2 = 2$. Pragul ζ al variaţiei adaptabilităţii, în modelul bazat pe macro mutaţie, a fost determinat dinamic: s-au utilizat 100 probe, am estimat ameliorarea medie a adaptabilităţii δ_+ al mutaţiilor ce funcţionau în condiţii generate aleator. S-a considerat îmbunătăţire majoră a adaptabilităţii valori mai mari de 50%, deci $\zeta = 1.5 \cdot \delta_+$ Pentru seturile de teste, cu mărimea blocurilor până la 10, s-au efectuat în medie 100 cicluri. Pentru problemele cu mărimea blocurilor de bază până la 12 numărul mediu de cicluri efectuate a fost 30, respectiv 10 pentru mărimile de blocuri de 14, 16.

Numărul evaluărilor a căutării locale folosite intr-o fază (epoch) a fost setată la $5n^2$. Metoda a arătat un comportament foarte bun, cu o rată de succes de 100% la fiecare test, cu o construcţie de model precisă şi promptă. Performanţa şi scalabilitatea modelului este ilustrată în Figura 5.1. Odată cu creşterea valorii k, creşte exponenţial şi numărul eşantionării aleatorii necesare pentru descoperirea blocurilor. Chiar şi la valori k de 16, metoda propusă, este capabilă să localizeze şi să combine cu succes toate blocurile, cu un cost de $6e6$ evaluări funcţii obiectiv.

Am repetat aceste experimente pentru setul de teste $ESRR$, cu mărimi de blocuri până la 14, utilizând strategia simplă, oarbă a macro mutaţiei, fără analiza variaţiei a adaptabilităţii şi construcţiei preliminare a blocurilor. În acest caz, ori

k	LS - $c_2 n^{c_1}$ ($c_2 \leq k$)	Succ. rate	Avg. nr. obj. func. evals.
8	$5n^2$	100%	3.442e5
10	$5n^2$	100%	1.425e6
12	$10n^2$	100%	7.547e6
14	$5n^{2.39}$	100%	2.655e7

Tabela 5.1: Rezultate numerice pentru MBMM, mutaţie oarbă.

de câte ori se acceptă o stare nouă, se efectuează o căutare greedy, pentru a descoperi soluţii mai bune, dacă există, din vecinătatea noi stări. Rezultatele sunt schiţate în tabelul 5.1. Prima coloană reprezintă mărimile modulilor de bază. În coloana a doua este prezentată scalarea evaluării funcţiilor, referitoare la raportul blocurilor faţă de mărimea problemei(reţinem că $n = k^2$), determinate într-o singură rulare a căutării locale, cu scopul investigării structurii modulelor. Coloana a treia conţine ratele de succes a fiecărui set de teste, iar numărul mediu al evaluărilor de funcţii, până la atingerea optimului global, este raportat în ultima coloană.

$ESRR$ de mărimea 8 şi 10 a fost rezolvat cu uşurinţă în fiecare caz, prin alocarea de $5n^2$ evaluări de funcţie căutării locale. Datorită explorării profunde efectuate de către modelul cu bias bazat pe macro-mutaţie, neutralitatea $ESRR$, multimodalitatea masivă şi deceptivitatea medie pe a funcţiei $HMMD$ este depăşită, aşa cum o reflectă rezultatele. Metoda a identificat corect optimul global şi a asigurat construire de model exact la fiecare rulaj. Chiar dacă spaţiul căutării este extrem de dificil, datorită combinării aleatorii şi datorită numărului astronomic de optimi locali plasaţi, MBMM poate rezolva problema rapid, prin efectuarea şi exploatarea precisă a descompunerii problemei.

5.6 Concluzii

Investirea evaluării funcţiilor obiectiv într-o căutare locală puternic exploratorie poate facilita descoperirea modulelor mari. Deasemenea metoda este capabilă în a diferenţia între setările modulare corecte şi schema lor cea mai competitivă. Astfel, numărul total de mostre trebuie să fie doar suficient de mare, pentru a permite detectarea modulelor prin tehnici adecvate de învăţare automată.

Capitolul 6

Optimizarea la scală largă

Acest capitol prezintă rezultatele principale, ce demonstrează modul în care identificarea şi exploatarea dependenţelor prin construirea de modele în on-line, facilitată de Sistemele Neuronale Artificiale, combinat cu Căutare Locală Bazată pe Modele, deschide calea spre optimizare pe scală largă a problemelor grele cu blocuri, neseparabile, interdependente.

6.1 Introducere

Pentru rezolvarea problemelor neseparabile de ordinul milioanelor de variabile, vom avea nevoie de metode eficiente din punct de vedere computaţional în cazul construirii modelelor, totodată eficiente din punctul de vedere al utilizării memoriei. Pentru îndeplinirea acestor obiective, luăm în considerare extinderea Căutării Locale Bazate pe Model prezentat (Iclanzan & Dumitrescu, 2007b) cu un mehanism de învăţare şi construire de model în timp real (online) . În Cadrul Căutării Locale Bazate pe Model online (OMBLS) dependenţele sunt deduse dintr-o singură structură de date, astfel metoda este foarte eficientă din punctul de vedere a memoriei.

6.2 Funcţia test ierarhică mixtă

Pentru obţinerea unei singure probleme mari, scalabile, care să conţină toate caracteristicile regăsite în testele de până acum, considerăm combinarea a trei funcţii de test ierarhice bine cunoscute: IFF ierarhic (Watson et al., 1998), XOR ierarhic (Watson & Pollack, 1999) şi funcţia ierarhică capcană.(Pelikan & Goldberg, 2001).

6.3 Căutare Locală Bazată pe Modele cu Adaptare Online

6.3.1 Căutare locală utilizată

Folosim o căutare de tip greedy în cazul acestor configuraţii.

6.3.2 Detectarea conexiunilor

O reţea ce tinde a menţine proprietăţile spaţiului de date este harta autoorganizatorie - reţea neuronală Kohonen (SOM). Reţeaua este aantrenată folosind *învăţare nesupervizată* pentru producerea reprezentării discretizate şi bidimensionale ale spaţiului datelor de intrare, numită hartă.

Algoritmul de antrenare a SOM poate fi actualizat în mod repetat online cu date reale şi curente (live), introducând direct rezultate Căutării Locale Bazate pe Model,(stări de convergenţă locală), fără a utiliza un buffer pentru stocarea rezultatelor mostrelor de antrenare. Prin analiza ponderilor reţelei, putem decide care dintre variabilele curente sunt cuplate, dar pentru a extinde spaţiul căutării vom avea nevoie şi de setările lor context optimale. Ca şi consecinţă, dacă se furnizează doar informaţiile privind relaţia variabilelor, în cazul fiecărui bloc compus nou metoda va trebui să caute prin toate combinaţiile de sub module şi să reţină cele mai bune λ.

6.3.3 Reducerea spațiului de căutare

6.3.4 Algoritmul OMBLS

Pornind de la o reprezentare, care este în concordanţă cu problema originală, într-o fază incipientă experienţa căutării este acumulată, prin antrenarea SOM online, cu stări furnizate de căutări locale repetate. Căutarea trebuie să fie suficient de potentă, pentru localizarea modulelor complet optimizate la un nivel ierarhic singular. După convergenţa reţelei, într-o fază secundară, structura spaţiului datelor de intrare este dedusă din ponderea reţelei şi este exprimată prin conexiunile variabilelor. În continuare se efectuează o căutare exhaustivă, conform conexiunilor detectate, pentru detectarea celor mai bune setări context-optimale a modulelor formate.

Formal Căutarea Locală Bazată pe model Online este schiţată în Algoritmul 3.

6.4 Scalabilitatea, Complexitatea OMBLS

Presupunând că tehnica de învăţare automată utilizată, detectează cu succes legăturile corecte, convergenţa globală a OMBLS poate fi demonstrată pe problema

Algorithm 3: Căutarea Locală Bazată pe Modele Online

Data: n, n_S, ϵ,@stopping_cond.

```
   /* Initially each binary variable is a base module          */
1  for i = 1, n do
2      M[i][0] ← {0};
3      M[i][1] ← {1};
4  while not @stopping_cond do
       /* Phase I                                              */
       /* Build the SOM                                        */
5      net ← InitializeSOM(n);
6      for i = 1, n_S do
           /* Generate a random binary state of length n       */
7          s ← RandomState(n);
           /* Apply module-wise greedy search                  */
8          s ← GreedySearch(s, M);
           /* Train the network online using vector quantization */
9          net ← Train(net, s);
       /* Phase II                                             */
       /* Detect possible modules via weight analysis          */
10     nm ← GetLinkages(net.Weights, ε);
       /* Identify the two most fit schemata for new modules by
          exhaustive search                                    */
11     CO_set ← RetrieveBestTwoSettings(nm);
       /* Collapse the search space and update the building-block
          configuration according to the detected modules and their
          context-optimal settings                             */
12     n ← |CO_set|;
13     for i = 1, n do
14         if module i is new then
15             M[i][0] ← CO_set[i][0];
16             M[i][1] ← CO_set[i][1];
```

test, prin evidenţierea existenţei unei căi ce va duce la optimul global în cazul fiecărui component în parte, urmat cu uşurinţă de metodă.

În cazul căutării greedy prin module, la un anumit nivel al ierarhiei, numărul obiectiv al evaluărilor de funcţie va fi proporţional cu numărul modulelor l, numărul setărilor context optimale $\lambda = 2$ pentru fiecare modul, şi numărul perioadelor (epoch) utilizate pentru antrenarea reţelei n_S.

$$T_{LS} = 2 \cdot n_S \cdot l \tag{6.1}$$

Căutarea pentru setările context optimale va necesita un număr obiectiv de evaluări ale funcţiilor exponenţial în raport cu mărimea modulelor (simbolizat de

k_i) recent descoperite M' şi numărul setărilor context optimale:

$$T_{COS} = \sum_{i=1}^{|M'|} 2^{k_i} \tag{6.2}$$

Pentru nivele ierarhice p, limita superioară este dată de sumarea costurilor căutării locale bazate pe modele T_{LS} şi căutarea pentru setări optimale de context T_{COS} pentru fiecare nivel ierarhic.

$$T = \sum_{i=1}^{p} (T_{LS} + T_{COS}) \tag{6.3}$$

Cunoscînd faptul că mărime modulelor în cazul hIFF şi hXOR est eegal cu $k_1 = 2$ şi pentru hTrap este de $k_2 = 3$, am obţinut următoarea limită superioară pe hMIX pentru nivele ierarhice p:

$$T_{hMix_p} = 2 \cdot \sum_{\substack{l=k_1 \\ l=l*k_1}}^{k_1^p} (2 \cdot n_S \cdot l + \frac{l}{k_1} \cdot 2^{k_1}) + \sum_{\substack{l=k_2 \\ l=l*k_2}}^{k_2^p} (2 \cdot n_S \cdot l + \frac{l}{k_2} \cdot 2^{k_2}) \tag{6.4}$$

Pentru a confirma empiric acest rezultat şi pentru a verifica eficienţei SOM pentru detectarea dependenţelor online, am testat scalabilitatea OMBLS cu hMIX pe nivele ierarhice $p = \{4, 6, 8, 10, 11\}$, cu mărimi ale problemei $n = \{113, 857, 7073, 61097, 181243\}$

Metoda a localizat cu succes în fiecare caz unul dintre cele 4 optime globale, confirmând eficienţa învăţării cuplajelor bazate pe SOM online. Scalarea metodei pe hMix este prezentată în figura 6.1. Rezultatul experimental a fost aproximat cu o funcţie pe diagrama $f(x) = ax^b \cdot log_2(x)$ unde a şi b sunt determinate prin metoda erorii minime pătratice. OMBLS scalează pe hMix aproximativ cu $\theta(x^{0.909} \cdot log_2(x))$, unde x reprezintă mărimea problemei.

În cazul nostru, costul căutării de tip greedy este lineară în numărul de module, din ecuaţia (6.4), rezultând un timp de rulaj foarte eficient, sub-linearitmetic, confirmat empiric de experimentele noastre. Necesarul de memorie al OMBLS este foarte scăzut, fiind linear în mărimea problemei.

6.5 Sumar

OMBLS deschide calea spre optimizarea pe scală largă a problemelor cu blocuri neseparabile, grele. Lucrările viitoare se vor concentra asupra înregistrării probabilistice a informaţiei, pentru a aborda problemele cu structură şi mai complexă. O altă ramură a cercetării se va concentra asupra paralelizării cadrului propus: în locul căutării secvenţiale, numeroase căutări locale pot rula concomitent pe unităţi computaţionale diferite. Construirea de model poate fi paralelizat în mare măsură, prin rularea paralelă a căutării setărilor context optimale ale blocurilor individuale, cu calcularea distanţei cuplurilor a diferitelor module.

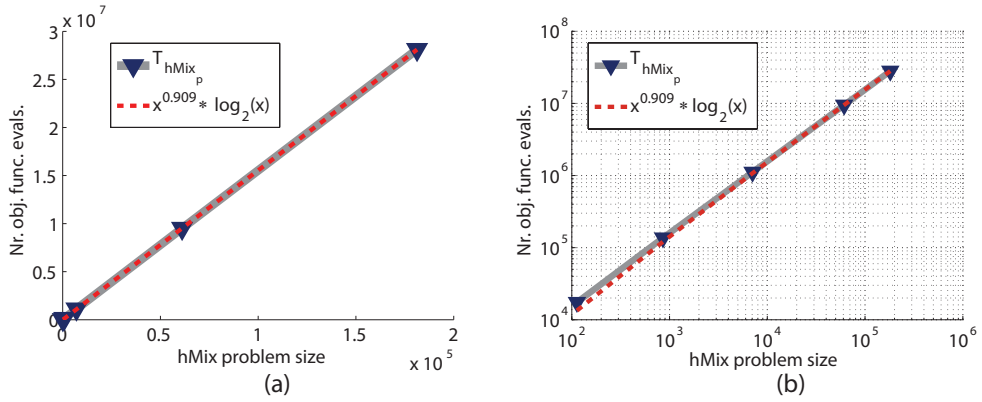


Figura 6.1: Scalabilitatea sub-linearitmică a OMBLS:(a) aritmetic; (b) logaritmic.

Capitolul 7

Construirea de Modele bazată pe tehnici de Clustering a Grafurilor

Capitolul descrie o nouă strategie de inducţie de model nesuprvegheat constituită pe o tehnică de clustering al grafurilor de flux maxim. Această abordare nouă oferă o metodă de evaluare a modelelor liberă, rapidă, scalabilă şi uşor paralelizabilă, capabilă de încorporarea structurilor complexe de dependenţe. Această metodă poate fi folosită pentru construirea unor clase de modele probabilistice diferite.

7.1 Introducere

Construirea modelelor poate fi foarte costisitor din punct de vedere computaţional, aşa cum s-a discutat în capitolul 3.

În acest capitol vom aprofunda explorarea confluenţei dintre algoritmii de clustering şi EDAs, unde algoritmii de graf clustering sunt aplicate unei matrice ce conţine o măsură a interacţiunilor dintre variabile. Am denumit această clasă de metode EDA asistat de graf clustering (CGEDA). Suntem interesaţi în a găsi algoritmi de clustering eficienţi care permit inducţia diferitelor clase de modele probabilistice.

Abordarea noastră oferă avantajul livrării automate a structurilor de dependenţă, fără a necesitat efectuarea unei căutări de model şi a evaluării performanţelor acestora foarte costisitoare.

7.2 Preliminarii

7.2.1 Graf Clustering și EDAs

Modelele probabilistice bune descriu care variabile interacţionează şi modul în care aceaste interacţiuni se manifestă. Metricile discriminative ale modelelor, utilizate în AE multivariaţi pentru ghidarea construirii aditive a modelelor, măsoară interacţiunile variabilelor, cuantificându-le pentru module.

7.2.2 Paradigma graf clustering, matrici stohastici și fluxuri

Algoritmii de clustering utilizând fluxul maxim se bazează pe următoarea idee principală: prin simularea unui flux în graf care promovează fluxul acolo unde curentul este puternic şi reduce fluxul acolo unde curentul este redus, va dezvălui structurile de grupare din interiorul acestui graf, precum graniţele dintre diferitele grupuri de flux vor dispărea în timp, dezvăluind grupurile interconectate.

7.2.3 Algoritmul Markov Clustering

Algoritmul Markov pentru Clustering (MCL) van Dongen (2000a) este un algoritm nesupravegheat de clustering a grafurilor rapid şi scalabil, bazat pe simularea fluxului stohastic din interiorul grafului. Oferă numeroase avantaje, cum ar fi: formulare matematică simplă şi elegantă, robusteţe faţă de perturbanţe topologiceBrohée & van Helden (2006), suport pentru paralelizăre şi a adaptării comportamentului prin setarea unui parametru, ce permite obţinerea unor clusteri de granularitate diferită. MCL simulează repetitiv fluxul într-un graf aplicând doi operatori, numiţi expansiune şi inflaţie, până la obţinerea convergenţei. La sfârşitul fiecărui pas de inflaţie se efectuează şi un pas de ajustare/truncare, pentru reducerea complexităţii computaţionale, păstrând densitate M.

7.2.4 Interpretarea rezultatelor MCL ca modele de dependență

Iteranţii procesului MCL M_t sunt în general matrici pozitive semi-definite. Utilizând caracteristica, conform căreia, minimele unei matrice semidefininte pozitive pe diagonală, sunt de obicei non negative, în van Dongen (2000b) s-a demonstrat că M_t-urile deţin o caracteristică structurală, care asociază un Graf Aciclic Direcţional(DAG) fiecărei valori. Aceste DAG-uri schematizează grafurile stea asociate limitelor MCL.

Vom prezenta o serie de abordări la modul în care informaţia provenită din MCL poate fi transmise modelelor de dependenţe capabile de a reprezenta şi exploata conexiunile.

Într-o *primă* abordare, DAG-urile reprezentate de M_t sunt direct utilizate pentru definirea structurii unei reţele Bayesian, ai căror extremităţi reprezintă dependenţele condiţionale dintre variabile. Extragerea parametrilor se va efectua

în mod similar cu BOA Pelikan et al. (1999) sau cu EBNA Etxeberria & Larranaga (1999). Reţeaua Bayaesiană obţinută astfel va reţine probabilitatea îmbinată a distribuţiei variabilelor, şi poate fi utilizată pentru eşantionarea/generarea următoarei generaţii.

Această abordare prezintă două impedimente minore. În primul rând, este necesară evaluarea modelelor, într-o anumită măsură pentru determinarea acelor M_t care sunt cele mai corespunzătoare construcţiei reţelei Bayaesiene. Al doilea impediment se referă la acele rare cazuri în care iterantul MCL va conţine cicluri. Înaintea extragerii parametrilor, aceste cicluri vor trebui eliminate.

Conform abordării *secunde* DAG-urile sunt interpretate ca şi grupuri-aglomerări de noduri. Nodurile terminale (sink) din DAG sint interpretate ca nuclee, cărora i se ataşează toate acele noduri, ce accesează aceste nuclee cu un flux ce depăşeşte un anumit prag. Această procedură poate rezulta în grupuri cu suprapuneri. Conexiunile extrase pot fi folosite pentru efectuarea incrucişării în blocuri cum ar fi DSMGA Yu et al. (2003) sau DSMGA++ Yu & Goldberg (2006), ori pot fi folosite pentru construirea unui model de conexiune suprapusă bazată pe distribuţia probabilităţilor.

Aborderea *a treia* este cea mai ieftină, pentru că studiază ultimul iterant, atunci când, matricea stohastica de flux M este complet convergentă. Aici nodurile şi-au localizat deja un nod de "atractor" în direcţia căruia este dirijat totalitatea fluxului, corespunzând unei singure intrări non zero pe coloană în M. Nodurile ce deţin atractor comun vor fi grupate în clustere. Această abordare este corespunzătoare modelelor de blocuri nesuprapuse, prin construirea produselor de model marginal, ca şi în cazul eCGA Harik (1999).

Cu excepţia primei abordări, deducerea structurilor de dependenţă este autonomă, nu necesită verificarea potrivirii modelului în ceea ce priveşte datele.

7.3 EDA asistat de MCL

Dorindu-se prezentarea unei EDA cu construcţie de model nesupervizată, capabilă de modelarea interacţiunii complicate ale variabilelor, vom utiliza a doua abordare a MCL, pentru obţinerea unui model probabilistic cu conexiuni suprapuse. Vom numi acest algoritm EDA Markov Cluster(MCEDA) Detaliile acestui algoritm vor fi prezentate în cele ce urmează.

7.3.1 Măsurarea interacţiunilor

În acest capitol, gradul dependenţei dintre perechile de variabile este calculat folosind informaţia mutuală dintre două variabile, înregistrată într-o matrice de vecinătate care va reprezenta datele de intrare a algoritmului de graf clustering.

Function ExtractBBs(Mlist) returns BBlist

```
1 BBlist ← ∅;
2 //each iterant from the MCL algorithm is processed
3 foreach M_t from Mlist do
4     //for all nodes find significant incoming flows
5     for i ← 1 to size(M_t) do
6         pBB ← find(M_t(i,:) > F_min(i));
7         if length(pBB) > 1 then
8             BBlist ← BBlist ⋃ pBB;
9 BBlist ← unique(BBlist);
```

Transformarea matricei A într-o matrice stohastică Markov va fi efectuată de algortimul MCL prin normalizarea coloanelor (acestea vor avea suma de 1).

$$M(i,j) = \frac{A(i,j)}{\sum_{l=1}^{n} A(l,j)} \tag{7.1}$$

7.3.2 Modelul cu interacțiuni suprapuse(OLM)

În MCEDA, interacţiunile multivariate ale variabilelor sunt modelate prin utilizare Modelelor cu conexiuni suprapuse (OLM), care se aseamănă puternic cu modelul produselor marginale utilizat de eCGA Harik (1999). Diferenţa dinte acestea este că variabilele modelelor OML sunt unite ca şi clustere, permiţând suprapuneri, în contrast cu partiţionarea variabilele în blocuri exclusive, fără elemente comune. Aceste grupuri pot reprezenta, natural blocuri, furnizând harta conexiunilor directe, astfel putem utiliza acesti termeni în mod alternativ în contextul OLM. Grupurile, împreună cu distribuţia marginală formează Modelele cu Conexiuni Suprapuse.

7.3.3 Construirea modelului de dependență

Grupurile care formează bazele OLM sunt extrase din iteranţii M_t ale algoritmului MCL.

Pentru fiecare nod se formează un bloc potenţial, prin gruparea tuturor nodurilor ce o accesează, ai căror flux depăşeşte pragul F_{min}. Grupurile de mărimea 1 sunt permise doar în cazul în care poziţia singulară descrisă nu se regăseşte în alt modul. După ce toate recurenţele au fost procesate, procedura returnează intrările unice a listei ce descrie blocurile posibile. Această organizare trebuie efectuată, căci un grup poate fi detectat în iteranţi diferiţi.

Extracţia blocurilor este ilustrată în Funcţia `ExtractBBs`

După ce aceste blocuri sunt determinate, este estimată distribuţia probabilităţii prin simpla numărare a frecvenţelor în date.

Algorithm 4: Algoritmul Markov Clustering EDA

```
pop ← RandomInit();
repeat
    ps ← Selection(pop); //select promising solutions
    {ps ← ReduceEntropy(ps)}; //optionally reduce entropy by LS
    A ← MutualInformation(ps); //extract global statistics
    Mlist ← MCL(A); //apply graph clustering
    BBlist ← ExtractBBs(Mlist); //extract dependency structure
    freq ← FrequencyCount(BBlist, ps); //compute marginal
    probabilities
    olm ← BuildMPM(BBlist, freq); //combine results into a OLM
    pop ← Sample(olm); //generate a new population using the model
until convergence criteria is met;
```

7.3.4 Markov Clustering EDA

Algoritmul 4 schiţează structura şi funcţionalitatea metodei.

7.4 Experimente

În cadrul acestui capito lucrăm cul Clasa Funcţiilor Decomposabile Aditiv (ADF) ,cu subprobleme deceptive tip capcană, cunoscute în literatura de specialitate ca probleme de referinţă Pelikan et al. (1999); Yu et al. (2005); Correa & Shapiro (2006).

7.4.1 Funcţii test

5-Trap concatenată Pelikan et al. (1999) este un ADF bazată pe măsurarea valorii de unitate (numărul valorilor 1 dintr-un şir binar), ilustrând un singur optim global în şirul format exclusiv din valori de 1.

Neseparabilitatea poate fi introdusă prin aplicarea lungimii fixe, a schemei suprapuse circulare dintre funcţiile capcană-5 Yu et al. (2005). De exemplu, la o problemă cu 3 subprobleme şi o lungime a suprapunerilor de $l = 2$, funcţia va fi $trap5(y1y2y3y4y5) + trap5(y4y5y6y7y8) + trap5(y7y8y9y1y2)$, unde yi este o permutaţie aleatorie al variabilelor xi, menită să disloce cuplajul strâmt. Fiecare bloc deţine $2l$ variable comune, l cu fiecare bloc vecin în parte.

7.4.2 Rezultate numerice

Experimentele sunt efectuate pe funcţii capcană-5 concatenate, fără suprapunere (simbolizate prin ctf5o0), cu suprapunere 1 (simbolizate prin ctf5o1) şi cu suprapunere de 2 (simbolizate prin ctf5o2). Pentru testarea scalabilităţii, pentru fiecare

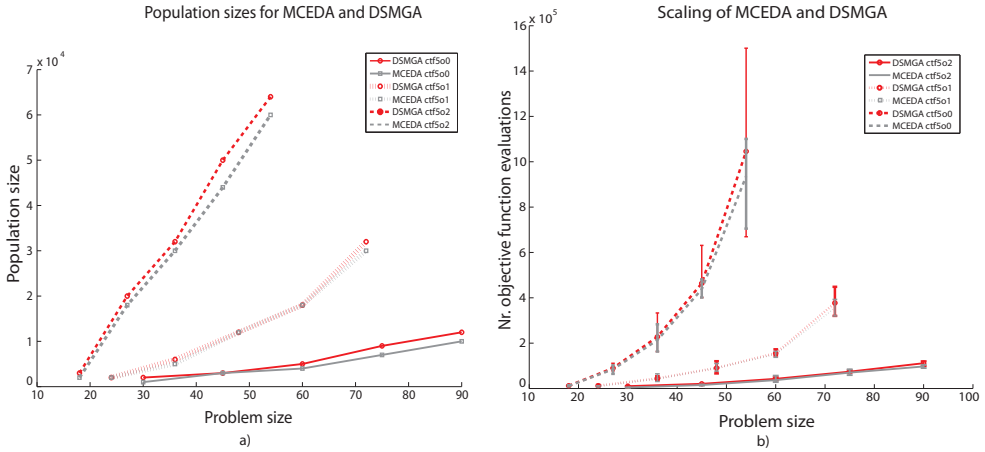


Figura 7.1: Scalabilitatea MCEDA şi DSMGA pe funcţiile test.

ADF numărul de subprobleme este variat de la 6 la 18, cu augmentare de 3, rezultând în mărimi de probleme variabile, cu până la 90 variabile.

Figura 7.1 b) prezintă performanţa metodelor în cazul problemelor de tipuri şi dimensiuni diferite. Rezultatele indică o scalare similară în cazul celor două metode. MCEDA utilizează mai puţine evaluări ale funcţiilor, şi funcţionează cu populaţii mai mici decât DSMGA. Această diferenţă de performanţă poate fi explicată prin doi factori:

- DSMGA utilizează descrie interacţiunea variabilelor prin valori crisp de 0 sau 1. Matricea de valori reciproce este transformată într-o matrice binară conform unui prag, când variabilele sunt considerate fie în completă interacţiune fie complet independente. În contrast, MCEDA lucrează direct pe o matrice cu valorile actuale reciproce normalizate, care descriu mai nuanţat ponderea nivelelor de interacţiune, facilitând descoperirea mai rapidă a modelelor corespunzătoare.

- MCEDA deţine un mehanism de menţinere a diversităţii mai bună, cu un număr de blocuri extrase ridicat ce rezultp din primii iteranţiale procesului MCL. Metoda crează mostrele conform frecvenţelor exacte observate din date. DSMGA se poate confrunta cu fenomenul de autostop (hitchhiking) Mitchell & Holland (1993).

Clusteringul MCL este mult mai rapid decât clusteringul bazat pe MDL, utilizat de DSMGA, avînd o complexitate de $O(nk^2)$ van Dongen (2000a), în cel mai rău caz, unde k este factorul de eliminare-pruning (cel mult câte valori non zero vor fi într-un rând al matricei stohastice- un număr mic în practic,$k << n$). Grupare a 5000 noduri prin algoritmul MCL durează câteva secunde, în comparaţie cu minutele necesare construirii de model în cazul DSMGA.

7.5 Sumar

Domeniul graf clustering oferă un set de oportunităţi larg şi flexibil pentru crearea modelelor, datorită faptului că numeroase tipuri de structuri de dependenţă pot fi deduse din rezultate. Ea mai oferă posibilitatea de a evita în totalitate evaluările de compatibilitate a modelelor cu datele sau acestea pot fi aplicate în mod controlat, când acestea doar ghidează căutarea. În al doilea scenariu, evaluarea modelelor este utilizată, dar rareori, cu rolul de a preveni adaptarea exagerată (overfitting), şi pentru a putea discrimina între diferite modele posibile.

Capitolul 8

Utilitatea operatorului de încrucișare

Din punct de vedere istoric, majoritate tentativelor de a reţine caracteristicile topologice a unei funcţii, care să exemplifice procesul intuitiv simplu descris de Ipoteza Blocurilor de Construcţie (BBH), au eşuat. Metodele recombinative bazate pe populaţii au fost depăşite în în repetate rânduro, de algoritmi bazaţi pe mutaţie, pe teste abstracte special create.

Pornind de la BBH, acest capitol caută să exemplifice utilitatea încrucişării din alt punct de vedere, accentuînd potenţialul creativ al operatorului de încrucişare. Am creat o clasă speciaală de teste abstracte, numite funcţii Trident, care exploatează capacitatea AG moderni de a amesteca soluţii bune dar *semnificant diferite*. Această abordare a fost neglijată, deoarece se presupune că împerecherea indivizilor mult prea diferiţi poate fi chiar şi nocivă. Însă, hibridizarea diferitelor concepţii induce o structură de vecinătate complexă, neabordabilă de metodelor bazate pe traiectorie, care ar putea masca solţii inedite.

În acest capitol vom demonstra că această clasă de probeme propusă, poate fi rezolvat eficient doar prin metode recombinative"panmictice" bazate pe populaţie, care utilizează mechanisme de prezervare a diversităţii.

8.1 Introducere

8.2 Scurtă istorie

8.3 Hibridizarea diferenţelor

Argumentul nostru este, ca pentru învingerea technicilor de hill-climbing, o problemă trebuie să conţină un amumit grad de deceptivitate, care nu poate fi învinsă prin utilizarea unui operator de căutare ce utilizează structuri de vecinătate simple. Acest lucru va stânjeni şi performanţa AG, pentru că mutaţia funcţionează în vecinătate unui individ, iar selecţia de scurtă durată va favoriza căutarea căilor deceptive. Însă, AG deţin un avantaj major, prin stuctura de vecinătate complexă generată de operatorul recombinant, care ia în considerare măcar doi indivizi. Acest lucru ajută metodele să evite optimul local, şi să învingă deceptivitatea. Reprezentarea problemei împreună cu structura de vecinătate definesc peisajul căutării. Argumentăm că există probleme, în cazul cărora, doar peisajele de căutare transformate prin încrucişare, pot fi exploatate eficient. În următoarele vom oferi un exemplu pentru clasei acestor probleme, numite Funcţii Trident(TF).

8.3.1 Funcţia Trident

FT acceptă stringuri de biţi de lungime $2k$ unde $k \geq 2$, şi foloseşte o funcţie deunitare(care depinde de numărul de valori 1 într-un string de biţi, şi nu de poziţia acestora) ca şi stuctură de bază:

$$base(x) = \|2 \cdot u(x) - |x|\| \tag{8.1}$$

unde $u(x)$ descrie numărul biţilor de 1 din x şi $|x|$ este lungimea x.

Valoarea minimă a funcţiei de bază este 0, care este generat de un string care conţine în mod egal 0 şi 1: $u(x) = |x| - u(x)$. Maximul se obţine din stringuri care sunt formate în totalitate din 0 sau 1 şi corespunde valorii$|x|$. Următoarea componentă an FT este o funcţie de contribuţie, care premiaza *anumite configuraţii* ai stringurilor, care au un număr egal de 1 şi 0. Să presupunem că $L = x_1, x_2, \ldots, x_{\frac{n}{2}}$ este prima parte a unui string binar x, de lungime n şi $R = x_{\frac{n}{2}+1}, x_{\frac{n}{2}+2}, \ldots, x_n$ a doua parte. Definim funcţia de contribuţie pe baza relaţiei OR exclusiv(XOR):

$$contribution(x) = \begin{cases} 2 \cdot |x| & \text{, if } L = \bar{R}; \\ 0 & \text{, otherwise.} \end{cases} \tag{8.2}$$

where $\bar{R}$ stands for the bitwise negation of R.

Accentuăm faptul că funcţia nu deţine un bazin de atracţie, acestea recompensează maximal o un bitstring, sau nu o recompensează deloc. Determinarea maximului acestei probleme este similar probelmei căutării *acului în carul cu fân*. Neexistînd metode mai bune de căutare pentru această clasă de funcţii,

decât căutarea aleatorie, înseamnă că aceste funcţii sunt rezistente căutării cu bias bazate pe mutaţie.

TF este definită ca şi suma funcţiei de contribuţie şi a funcţiei de bază:

$$trident(x) = base(x) + contribution(x) \tag{8.3}$$

Figura 8.1 prezintă reprezentarea grafică a TF.

Maximul TF se regăseşte în punctele recompensate de funcţia de contribuţie. În acest caz valoarea lui este de $2 \cdot |x|$ în timp ce funcţia de bază atinge minimul 0. TF este foare grea pentru algoritmii pe bază de mutaţie pentru că funcţia de bază ne îndeprtează de regiunea unde se regăseşte optimul global. Chiar dacă este generat o stare aleatorie în care regăsim în mod egal 0 şi 1, nu este probabil ca în cazul problemelor mari, fucţia de contribuţie să premieze acel string. Însă, dacă algoritmul efectuează o căutare cu bias, acesta se va îndepărta imediat de minimul funcţiei de bază, mergând spre o regine cu adaptabilitate mai ridicată.

TF poate învinge căţărătorii de macromutaţie, pentru că optima locală şi cea globală se află la distanţă mare în spaţiul Hamming. Şansa, ca un salt de la optima locală la optima globală să se petreacă, este minimă, datorită faptului că $\frac{n}{2}$ biţi trebuie schimbaţi simultan. Nu există sturucturi ascunse, uşor de exploatat. Blocurile L and R sunt recompensate doar în acel caz dacă în contextul în care se află, şi partrea compensatorie a stringului va fi compatibil. TF are $2^{\frac{n}{2}}$ număr de soluţii globale, probabilitatea petrecerii acestui lucru în cazul stringurilor generate în mod aleator este de $P_{hit} = \frac{2^{\frac{n}{2}}}{2^n} = \frac{1}{2^{\frac{n}{2}}}$.

Cum stă situaţia în cazu AG? optima globală poate fi localizată cu uşurinţă, dacă AG combină soluţii bune dar diferite. Luăm exemplul unde $n = 8$ şi avem două stringuri, fiecare la un optim local:$s_1 = 00000000$ şi $s_2 = 11111111$. Încrucişarea de punctul 1 va produce stringurile optimale: $s_3 = 00001111$ şi $s_4 = 11110000$, cu probabilitatea $P = \frac{1}{n-1} = \frac{1}{7}$. La folosirea unei încrucişări prin două puncte vom avea $\frac{n}{2} - 1 = 3$ cazuri favorabile. Punctele favorabile de încrucişare vor fi perechile $\{(1,7),(2,6),(3,5)\}$. Stringuri optime pot rezulta şi din încrucişarea stringurilor care nu sunt optime. De ex. o încreucişare cu un singur punct între stringurile $s_5 = 00100001$ şi $s_6 = 11111101$ între locusurile 4 şi 5 va produce o soluţie optimă $s_7 = 00101101$. Este important că se combină soluţii candidate diferite.

TF ilustrează acele probleme unde există numeroase soluţii bune dar diferite, iar hibridizarea acestora *poate* avea consecinţă producerea unei concepţii complet noi, valoroase. Chiar dacă încrucişarea nu produce indivizi extraordinari, în mod regulat, ocazional poate produce un organism excepţional. Astfel încrucişarea deţine un potenţial generativ, care credem că nu trebuie neglijat prin restricţionarea recombinaţiei la indivizi similari genotipic.

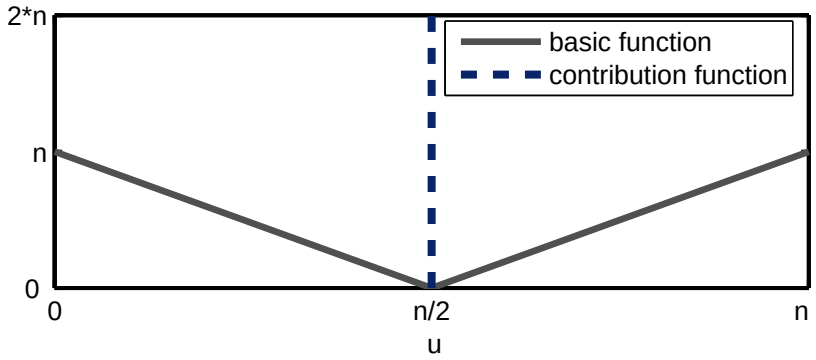


Figura 8.1: Funcţia Trident.

8.3.2 Metaforă naturală

8.4 Rezultate

8.4.1 Random Mutation Hill-Climber

8.4.2 Macro Mutation Hill-Climber

8.4.3 Algoritm Genetic Simplu

8.4.4 Deterministic Crowding

8.4.5 Rezultate Numerice

Rezultatele numerice ale experimentelor sunt ilustrate în tabelul 8.1. La versiunile de 64 and 128-biţi al FT, doar rezultatele DC sunt ilustrate, deoarece algoritmii hill-climbing SGA au eşuat la fiecare rulare a testului.

Cel mai slab rezultat pe TF a fost realizat de RMHC. Chiar şi la cea mai uşoară versiune de 16 biţi, rata de succes a fost doar de 85%. Similar cazului celuilalt căţărător, şi în acest caz, soluţiile sunt localizate la costuri foarte înalte, şi doar datorită mechanismului de resetare aleatorie. Numărul evaluărilor necesare identificării optimei locale a depăşit cu grade de magnitudine cantitatea necesară unei căutări randomizate. Concomitent creşterii mărimii problemei problema devine inabordabilă de technici hill-climbing prin eşantionare aleatorie, astfel aceştia eşuează datorită naturii deceptive a TF.

SGA a prezentat performanţe mai bune, acesta depăşind cu mult pe cea a căutării randomizate. Acesta demonstrează că în cazul algoritmilor recombinanţi simpli există potenţialul de a exploata caracteristicile peisajului TF. Dar cu creşterea mărimii problemei, acesta va eşua în localizarea optimei datorită eşantionării iniţiale insuficiente. Astfel populaţia este rapid mutată spre un bazin

Tabela 8.1: Performanţa metodelor.

TF size	**16**			
	Succ. rate	Avg. nr.	Max. nr	Nr. opt.
RMHC	**85%**	400020	-	/
MMHC	**100%**	4213	33528	/
SGA	**100%**	241	1390	1.46
DC	**100%**	250	2111	23.94

TF size	**32**			
	Succ. rate	Avg. nr.	Max. nr.	Nr. opt.
RMHC	**3%**	337136	-	/
MMHC	**37%**	471018	-	/
SGA	**25%**	8435	-	1.56
DC	**100%**	15771	23883	6.69

TF size	**64**			
	Succ. rate	Avg. nr.	Max. nr	Nr. opt.
DC	**100%**	46816	61366	6.02

TF size	**128**			
	Succ. rate	Avg. nr.	Max. nr.	Nr. opt.
DC	**100%**	112557	157890	4.32

de atracţie al unei singure optime locale. Creşterea semnificativă a populaţiei ar putea rezolva această problemă, dar succesul ar avea costuri ridicate.

Singurul algoritm competent în cazul TF a fost DC. Acesta a avut succes la fiecare rulaj, identificând optima globală într-un interval de 16% al evaluărilor permise. Succesul algoritmului derivă din mechanismul său de menţinre a diversităţii combinat cu populaţia panmictica. Accentuăm importanţa capacităţii de a combina concepţiile diferite, doar în aceste condiţii devine posibilă creativitatea. Un algoritm cu mechanisme de menţinere a diversităţii, dar cu încrucişare restricţionată, similar indivizilor, ar eşua pe TF.

8.5 Sumar

TF reprezintă acele probleme de design unde numeroasele concepte optime locale bune sunt uşor de localizat dominînd spaţiul căutării(decepţie) iar conceptele cu adevărat bune rezultă din hibridizarea diferitelor ciorne-prototipuri. Configuraţiile complexe ce definesc cea mai bună soluţie ies la iveală în zone reactive, unde caracteristicile specifice apar simultan, nu există secvenţe pentru îmbunătăţirea

acestor concepte,(acul în carul cu fîn). Totuşi încrucişarea posedă potenţialul creativ ex. structura de vecinătate mai complexă, care îi permite identificarea acestor soluţii şi combinarea caracteristicilor diferitelor subsoluţii, până ce configuraţia corectă este detectată.

Capitolul 9

Căutare non convergentă Bazată pe Cooperație și Specializare

Mulţi AE suferă de o tendinţă de a-şi pierde capacitatea da a integra material genetic nou, astfel stagnînd la un nivel suboptimal. Pentru aplicarea cu succes a acestor metode pe probleme cu complexitate şi dificultate crescătoare, este vitală abilitatea de a genera variaţii utile ce au ca urmare o îmbunătăţirea continuă. Totuşi există o dificultate majoră în găsirea extensiilor computaţionale pentru paradigma evoluţionară, care poate asigura emergenţa continuă a soluţiilor noi de calitate, căci esenţa paradigmei Darwiniene –selecţia naturală– acţionează ca o forţă stabilizatoare, asigurînd echilibrul evoluţionar al populaţiei.

În acest capitol propunem o abordare nouă, înlocuind paradigma supravieţuirii celui mai propice cu un cadru cooperativ, în care indivizii sunt puternic specializaţi in strategii de explorare şi exploatare diferite. Acesta va avea ca rezultat un proces de căutare foarte eficient, non convergent, şi robust, unde optima poate fi descoperit în mod continuu.

9.1 Introducere

O problemă majoră sesizată în timpul rulării AE pe probleme complexe de dimensiuni mari, este că odată cu cresterea presiunii selecţiei, populaţia tinde a convege spre optima locală, iar îmbunătăţiri nu se pot efectua, astfel metoda îşi perde abilitatea de a descoperii şi incorpora material genetic nou.

Algorithm 5: Individuali exploratorii

input : A state s and a set of exploring strategies p_{ES}
output: A new state

```
1 begin
      // Choose probabilistically an exploring strategy
2     @ES ← ProbabilisticallyChoose(p_ES);
      // Explore.  The method may or may not take use of the
         provided seed
3     s ← @ES(s);
      // Return the result
4     report s;
```

9.2 Paradigmă Cooperativă

AE imită schemele complexe implicate în transmisia informaţiei biologice din ecosistemele dinamice şi complexe. Dar, atunci când aplicaţiile AE cosideră peisajele de adaptare fixe, critica majoră primită la adresa AE este că metaaforele biologice pot fi inutil de complexe şi nu tocmai eficiente. O paradigmă cooperativă poate ameliora aceste fenomene.

9.2.1 Selecția

Selecția naturală nu este o forță puternică de optimizare

Selecția ca forță stabilizatoare

9.2.2 Căutare cooperativă

Fiecare abordare metaheuristică ar trebui fi concepută cu scopul de a explora spaţiul căutării eficient şi efectiv (Blum & Roli, 2003). În consecinţă, propunem un cadru unde exploatarea şi explorarea sunt considerate sarcini diferite dar complementare ale aceluiaşi proces, şi rezolvarea lor corectă este ţintită de diferite tehnici şi indivizi *specializaţi*.

Explorare

Căutarea poate conţine numeroase tehnici diferite de explorare, care sunt selectate cu o probabilitate anume, fixă sau adaptativă, în fiecare epoch, aşa cum este ilustrat în algoritmul 5.

Rolul indivizilor exploratori este de a localiza regiunile peste nivelul mediu şi nevizitate ale spaţiului căutării. Primul scop poate fi atins prin efectoarea unei căutări simple de spectru larg. Al doilea scop presupune tehnici care divizează spaţiul căutării sau determină topologia acestuia pentru a putea favoriza regiuni nevizitate.

Algorithm 6: Indivizi exploatatori

input : A starting state s and a set of exploiting strategies p_{IS}
output: State after exploitation
Data: s_{old}
`// Original state of the individual`
1 **begin**
`// Choose probabilistically an exploiting strategy`
2 @IS $\leftarrow$ `ProbabilisticallyChoose(`p_{IS}`)`;
`// Apply lamarckian operator starting from` s
3 $s' \leftarrow$ @IS(s);
`// Deterministic crossover between new and old state`
4 $s' \leftarrow$ `Merge(`s', s_{old}`)`;
5 report s';

Stările de incipit şi regiunile propuse de către indivizii exploratori sunt înregistrate într-o structură de date care furnizează un canal de comunicare, împărtăşesc informaţie cu indivizii exploatatori. Dacă timpul computaţional al funcţiei obiective necesită resurse numeroase, atunci subsarcina de explorate este foarte tolerantă la calcularea unei valori aproximative proxy, datorită faptului că scopul său este determinrea valorilor deasupra mediei. Aceast scop poate fi îndeplinit fără cunoaşterea precisă a valorilor. O funcţie de adaptare proxy, care este o funcţie corelată pozitiv cu funcţia originală, dar mult mai uşor de calculat poate fi suficientă.

Exploatare

Indivizii exploatatori aplică o strategie de căutare locală pentru identificarea rapidă a soluţiilor de calitate înaltă, pornind de la statele reportate de celălalt grup prin stucturile de date comune. Diferitele strategii de căutare locală sunt aplicate în acelşi mod probabilistic ca şi în cazul indivizilor exploratori, aşa cum este prezentat în Algoritmul 6.,

Cadrul general de cooperare prezentat conţine multe grade de libertate. Următoarea subsecţie prezintă metoda folosită în experimentele noastre.

Instanţierea paradigmei propuse

În vederea demonstrării faptului că, un comportament simplu, non convergent cooperativ este suficient pentru a îmbunătăţii calitativ performanţa căutării, nu vom folosi posibilităţile oferite de cadrul de lucru, care ar putea favoriza în mod inteligent căutarea prin efectuarea analizei de orice fel. Nu am implementat favorizarea regiunilor nevizitate, deci îmbunătăţirile din timpul căutării nu vor putea fi atribuite eşantionării înclinate spre teritorii neexploatate.

Algorithm 7: Îmcrucişare deterministică

input : Two states s_1 and s_2
output: Deterministically improved s_1

```
1 begin
2     for i=1:length(s_1) do
3         if s_1[i] == s_2[i] then
4             continue;
5         s' ← s_1;
6         s'[i] ← s_2[i];
7         if s' better than s_1 then
8             s_1 ← s';
9     return s_1;
```

Algorithm 8: Căutarea cooperativă, specializată

Data: $pop, pop\_size, p_{ES}, p_{IS}$,@stopping_cond.

```
1 begin
2     pop ← InitPop();
3     while not @stopping_cond() do
4         parfor i = 1, pop_size do
              // Apply Alg. 5 to explore
5             s ← Explore(pop[i], p_ES);
              // Apply Alg. 6 to exploit
6             s ← Exploit(s, p_IS);
7             if s better than pop[i] then
8                 pop[i] ← s;
          // Synchronization
9         pop ← MultiParentCrossover(pop);
```

Sunt aplicate consecutiv technici de explorare şi exploatare, aşa cum arată algoritmul 8. Acesta corespunde unei liste FIFO, utilizată pentru communicarea dintre indivizii ce explorează şi cei ce exploatează.

În procedura de sincronizare(Algoritmul 8, linia9) este folosită o încrucişare multi parentală. Acesta se aseamănă foartem mult cu Algoritmul 7, diferenţa fiind faptul că în acest caz este efectuată o căutare greedy sistematică în întreaga bază de gene al subpopulaţiei exploatate. Individul superior obţinut va înlocui cel mai adaptat individ din subpopulaţie.

Indivizii Exploatatori aplică una din aceste technici cu o probabilitate fixă: (I) eşantionare randomizată în limitele b_{rs} ale valuării obiective ala funcţiilor, (II) efectuarea unui răciri simulate rapide (Kirkpatrick et al., 1983) pentru identificării unui punct adecvat de pornire pentru exploatare. Ideea ce stă la baza metodei 2

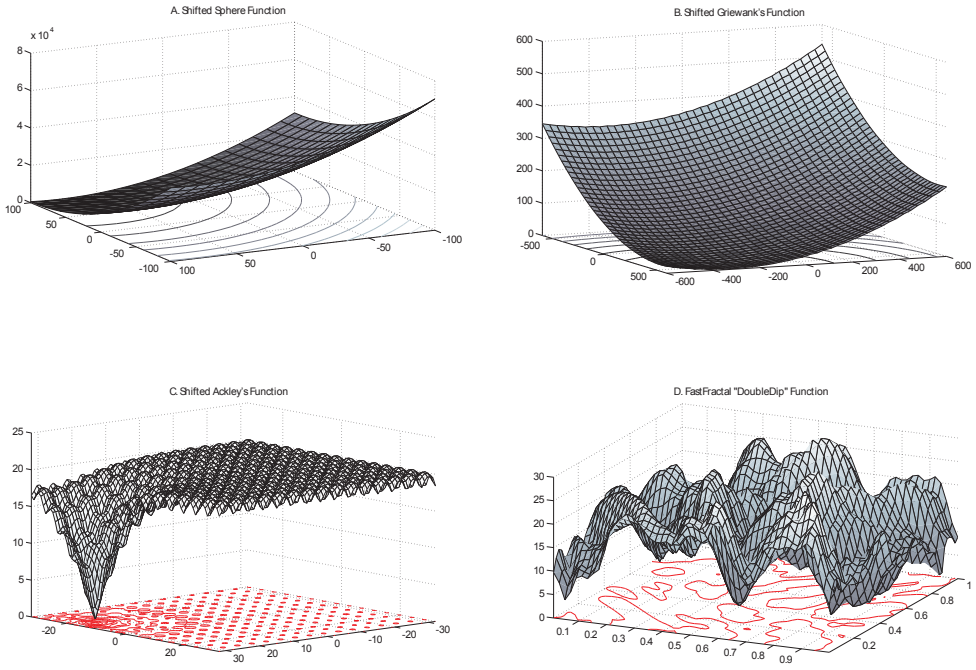


Figura 9.1: Funcţiile test pentru n=2.

este de a îngheţa rapid punctul de eşantionare în bazinul de atracţie al optimei locale(prin folosirea unui factor de răcire ridicat).

Indivizii exploatatori aleg în mod aleator cu probabilitate egală între folosirea (I) căutării de tipare (Torczon, 1997) şi Algoritmul-r Shor(Shor et al., 1985). Aceste tehnici sunt cunoscute a fi algoritmi buni de căutare locală pentru funcţii neuniforme şi nediferenţiabile.

9.2.3 Relaţia conceptuală cu alte metaheuristici şi paradigme de căutare

9.3 Experimente

9.3.1 Funcţia sferă

9.3.2 Funcţia Griewank

9.3.3 Funcţia Ackley

9.3.4 Funcţia FastFractal "DoubleDip"

9.3.5 Parameterizarea metodelor

În faza de explorare, algorimul utilizează 1000 evaluări de funcţii obiectiv prin căutare randomizată, respectiv 3500 prin călirea simulată rapidă pentru localizarea unui punct peste medie. Călirea simulată utilizează un program de răcire exponenţial cu un factor de răcire de 0.7.

Metodele de exploatare utilizate:

- Pattern search
- Shor r-algorithm
- Aceste technici sunt cunoscute ca fiind eficiente în cazul funcţiilor non diferenţiabile.

Exploatarea procedurilor de căutare locală rulează cu un maxim de 500 repetiţii de fiecare dată când sunt utilizaţi.

Pentru comparare vom folosi un algoritm genetic ce utilizează selecţia roţii de ruletă, cu o populaţii de 1000 indivizi, şi o rată de încrucişare de 0.8 şi cu mutaţie Gaussiană. Această populaţie a obţinut rezultate mai bune decît o populaţie mai mare de 10000 indivizi, furnizând un compromis bun dintre numărul de indivizi şi numărul de generaţii.

Am rulat algorimii pe problemele de test cu dimensiuni de 10, 100 respectiv 1000. Performanţa ambelor metode a fost înregistrează în limita a 1 milioa de evaluări de funcţie obiectiv.

9.4 Rezultate empirice

Rezultatele sunt prezentate în figura 9.2. Pentru funcţiile F_1, F_5 and F_6, unde optima globală exactă x^* este cunoscută, avem graficuri semilogaritmice cu $\log(F(x_{best}) - F(x^*))$ vs. FES, demonstrînd evoluţia în timp al rezultatului cel mai bun cunoscut până acum x_{best} pe o scală logaritmică. Pentru F_7 evoluţia simplă a rezultatului este prezentată.

Pentru dimensionalitatea problemei de 10, rezultate finale ale celor două metode sunt foarte apropiate pentru funcţiile F_1, F_6 and F_7. Calitativ aceste rezultate nu diferă, diferenţa lor absolută fiind mai mică decît 0.01, cun un avantaj mic în favoarea algoritmilor genetici, care exploatează mai amănunţit bazinul de atracţie al optimei.

Pentru funcţia F_5 algoritmul genetic nu poate identifica optima globală, nici în cazul a doar 10 dimensiuni, probabil datorită interdependenţei puternice ale variabilelor. Încrucişarea nu poate promva variabilele conectate, dat fiind că aceste legături nu sunt puternice.

Concomitent cu creşterea dimensionalităţii, se poate observa înrăutăţirea performanţei algoritmului genetic, care nu realizaează îmbunătăţiri pe durate de timp mari, chiar şi în cazul funcţiei sferice unimodale. Metoda, datorită presiunii crescute a selecţiei nu se poate deplasa în altă regiune a spaţiului, care ar putea conţine optime locale mai bune. Datorită exploziei dimensionalităţii variaţii utile sunt rar detectate. Această înrăutăţire a performanţei implică schimbarea cu câteva grade de magnitudine calitatea soluţiilor, cum este şi în cazul F_1, F_5.

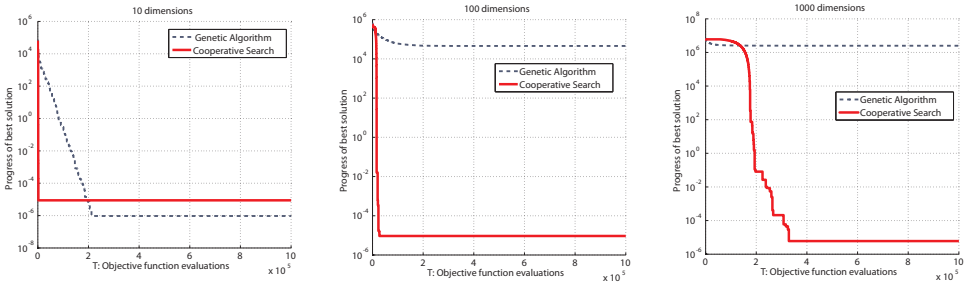
Pe funcţia F_7 FastFractal "Double Dip", putem observa o fază de ameliorare mai lungă al algoritmului genetic. Aici, cu apariţia noilor nivele de detalii la fiecare rezoluţie, metoda nu necesită translocarea la o altă regiune al căutării pentru a fi capabil de a efectua îmbunătăţiri.

Exploatarea este destul de eficientă datorită faptului că marea parte dacă nu fiecare membru a populaţiei va fi concentrată în bazinul de atracţie a opimei locale. Totuşi, cum rezoluţia funcţiei este finită, după un număr suficient de mare de evaluări ale funcţiilor, se manifestă aceaşi comportament. Exploatarea optimei locale este secată, algoritmul nu mai poate efectua îmbunătăţiri adiţionale, cum acest lucru ar presupune mutarea într-o nouă regiune a sapaţiului de căutare. Presiunea selecţiei permite acest lucru doar în cazul în care punctele generate(care se află în bazinul de atracţie a altei optime) deţine deja valoare competitivă de adaptare.

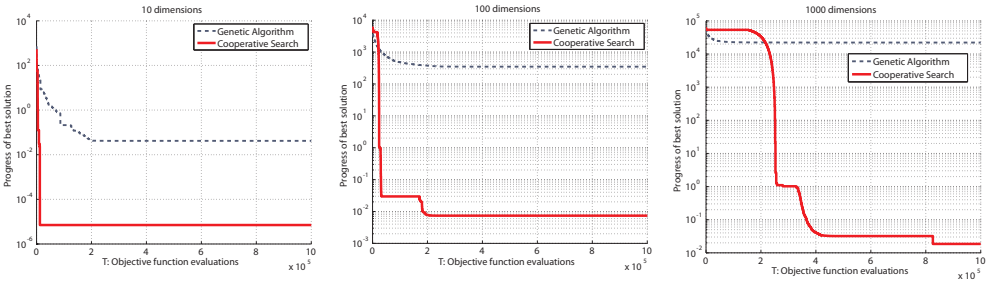
Pe altă parte, căutarea cooperativă afişază un comportament mai robust pe sprectrul întreg al problemei. În ciuda creşterii dimensionalitaţii este capabil de a identifica rapid soluţii şi poate face progres de durată spre optima globală, în majoritatea cazurilor. Pentru funcţia F_1, F_5, F_6 de dimensiuni de 10 şi 100 soluţiile sunt întotdeauna corespunzătoare optimei globale, sau sunt foarte aproape de aceasta, îndeplinind relaţia $F(x_{best}) - F(x^*) \leq 0.1$. Singurul caz în care căutarea cooperativă nu găseşte otima globală şi nu manifestă îmbunătăţire perpetuă este funcţia F_6 pentru 1000 dimensiuni. În acest caz numărul permis de evaluări nu este suficient de mare pentru ca metoda să învingă multimodalitatea funcţiei, chiar dacă noi regiuni sunt explorate în mod continuu. Nu putem afirma cu ceritudine ce se întâmplă în cazul funcţiei F_7, de dimensiune 10, unde nu se fac îmbunătăţiri, pentru că valoarea optimei globale este necunoscută pentru această

funcţie. Metoda noastră este capabilă de o îmbunătăţire susţinută, şi este foarte probabil ca şi în cazul funcţiilor mai grele, va găsi valoarea ce reprezintă optima globală, sau o optimă locală puternică. Funcţia F_7 de dimensiuni de 100 şi 1000 caraterizează metoda căutării cooperative în cel mai bun mod, căci ştim că optima globală nu se înregistrează în interiorul graniţelor celor 1 milion de evaluări.(experimentele mai lungi au obţinut rezultate mai bune). La acesste funcţii se observă că stările energetice mai scăzute identificate rapid, sunt îmbunătăşite pe parcurs, cu o pantă mereu descendentă. Acesta este în contrast cu linia constantă a convergenţei premature, care caracterizează performanţa AG pe funcţii de dimensiuni mai mari. Pe când algoritmul genetic se pote bloca la optime locale, datorită presiunii selecţiei, atunci căutarea cooperativă, specializată oferă o alternativă, confirmată şi de experimentele noastre.

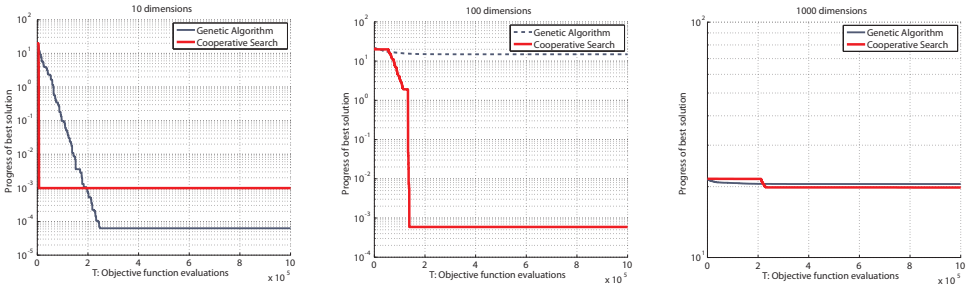
A. F_1 Shifted Sphere Function:



B. F_5 Shifted Griewank's Function:



C. F_6 Shifted Ackley's function:



D. F_7 FastFractal "DoubleDip" Function:

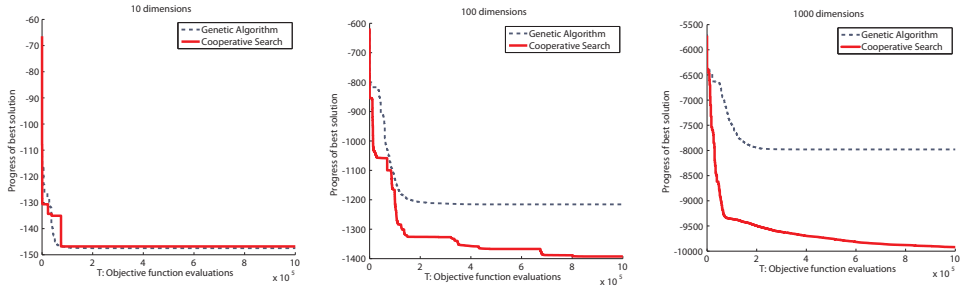


Figura 9.2: Results on the test suite for dimensions 10, 100 and 1000.

Capitolul 10

Concluzii și Posibile Extensii

Acest capitol prezintă sumarul investigaţiilor disertaţiei, în domeniul îmbunătăţirii robustneţii şi scalabilităţii algoritmilor evolutivi, precum şi identificarea eficientă a blocurilor şi combinarea a acestora.

10.1 Sumarul Rezultatelor

Această disertaţie este compusă din două părţi ce contribuie ambele în cadrul aceluiaşi context al îmbunătăţirii scalabilităţii metodelor competente de căutare şi metodelor de optimizare.

Prima parte introduce cadrul căutării locale asistate de învăţare automată şi analiza datelor. Asistat de tehnici de construire a modelelor inedite, bazate pe sisteme neuronale, automatice, online, cadrul de lucru poate rezolva probleme de mari dimensiuni, pentru că:

- Procesul de căutare a modelului este eliminat.
- Identificarea blocurilor este rezultatul unei căutări sistematice, nu doar al unei eşantionări iniţiale probabilistice.

A doua parte adresează una din problemele fundamentale ale AE, natura convergentă a cadrului tradiţional al calculului evolutiv, care este rădăcina fenomenului de convergenţă prematură. Pentru eliminarea acestei probleme, şi a furniza o dezvoltarea durabilă şi continuă, este propus un model de căutare competent şi neconvenţional, pe bază de cooperare, specializare şi exploatarea expetienţei de căutare.

În capitolul 3 recomandăm utilizarea analizei corelaţiilor dintre variabile pentru asistarea construcţiei modelelor. Atunci când alterăm un anumit model în vederea descoperirii unuia mai performant, analiza corelaţiilor poate indica cele

mai promiţătoare extensii. Acesta poate scădea costul căutării modelelor cu ordini de magnitudine, fără a afecta calitatea modelului descoperit. Într-un studiu de caz, complexitatea construcţiei modelelor este diminuată de la $O(n^3)$ la o căutare de timp liniar, ce deduce perfect structura problemelor, în cazul unor funcţii test ierarhice.

Capitolul 4 dezvoltă conceptul Căutării Locale Bazate pe Model, şi prezintă hill-climbing pe blocuri de construcţie. Aceasta este o metodă generică pentru rezolvarea problemelor printr-o descompunere ierarhic-recursivă.

În acest caz resursele sunt investite în găsirea soluţiilor bune, care în continuare sunt analizate prin tehnici de învăţare automată cu scopul identificării blocurilor. Combinarea adecvată a blocurilor este asigurată de o strategie de căutare locală, care operează în spaţiul de blocuri într-un mod sistematic şi exhaustiv. Actualizarea continuă a reprezentărilor blocurilor pe baza rezultatelor individuale va rezulta implicit în adaptarea structurilor de vecinătate la vecinătatea combinativă a blocurilor curente. În cazul problemelor ierarchice, unde ipoteza blocurilor de construcţie afirmă că, localizarea căutării în vecinătatea combinativa a reprezentaţiei curente a blocurilor facilitează descoperirea blocurilor noi, căci blocurile de rang mai scăzut pot fi combinate în blocuri de rang mai înalt.

Datorită faptului, că folosesc doar eşantionare randomizată pentru furnizarea blocurilor iniţiale, AG ce construiesc modele probabilistice clasice sunt limitate de mărimea populaţiei necesare, de ordine exponenţială. Pentru probleme mari, costul construirii de modele din populaţii ce cresc exponenţial, ar depăşi mult prea rapid limitele accesibilităţii şi ar depăşi praticalitatea economică.

În consecinţă, Capitolul 5 introduce o nouă paradigmă, un algoritm de căutare bazată pe modele şi macro mutaţie, unde blocurile iniţiale sunt descoperite prin utilizarea unor tehnici puternice de explorare, care urmăresc ghidajul funcţiilor obiective atunci când aceasta este disponibilă. Prin analiza variaţiei funcţiei obiectiv, metoda poate filtra mutaţiile nocive care ar putea distruge blocurile deja formate. Atribuirea evaluării funcţiilor unei căutări locale exploratorii, facilitează descoperirea modulelor mari. Metoda poate alege între setările cele mai adaptate şi schema cea mai competitivă a acestora. Astfel numărul mostrelor trebuie să fie suficient de mare încât să permită delimitarea diferitelor modele prin tehnici de învăţare automată.

Capitolul 6 analizează scalarea problemelor foarte mari în cadrul căutării locale bazate pe model. O variantă a cadrului căutării locale bazate pe model este descrisă, ce introduce o componentă online, care învaţă structura problemei cu ajutorul reţelelor neuronale Kohonen capabil să înveţe topologia spaţiului de intrare. OMBLS operează prin decompoziţie ierarhică, modulele detectate sunt folosite pentru colapsul spaţiului de căutare şi reformularea problemei de optimizare cu modulele descoperite şi setările context optimale ale acestora ca noi variabile ale căutării. În cazul problemelor limitate, dificile, cu blocuri ne

separabile, costul OMBLS este limitat superior de numărul nivelelor ierarchice multiplicate cu complexitatea necesară unei căutari locale pentru convergenţa la setări context optimale la un anumit nivel . Dacă setările context optimale pot fi descoperite de o strategie lineară în timp, şi ordinul dependenţelor este limitată la valori mici k, atunci cardrul propus deţine un avantaj calitativ asupra altor metode.

Un model nesupravegheat de învăţare, bazat pe algoritmi de graf clustering, este descris în capitolul 7. În afară de viteză, un avantaj mare al metodei este că permite deducerea-construirea diferitelor clase probabilistice.

Capitolul 8 re-examinează unele întrebări fundamentale ale AE. Insuficienţele funcţiilor test existente pentru delimitarea calitativă a operatorului de încrucişare sunt analizate, şi o idee ce promovează şi accentuează potenţialul generativ al AE este discutat. Conform acestui concept, forţa operatorului de încrucişare constă în capacitatea acestuia de a hibridiza trăsături diferite, în loc să promoveze similaritatea populaţiei.

În consecinţă este introdusă o nouă clasă de probleme, numite Funcţii TRIDENT. Acestea sunt dominate de o funcţie de bază complet deceptivă, iar optima globală corespunde minimei funcţiei de bază. Soluţiile optime discrete sunt definite de o funcţie de contribuţie care premiază punctele din spaţiul de căutare unde anumitele caracteristici genotipici apar concomitent. Datorită faptului că funcţia de contribuţie nu deţine un bazin de atracţie, deceptivitatea nu poate fi învinsă prin structuri de vecinătate simple.

Pentru ameliorarea problemei convergenţei premature Capitolul 9 propune un cadru cooperativ non convergent. Pentru o căutare echilibrată, explorarea şi exploatare sunt privite ca sarci bine definite a procesului de căutare, iar rezolvarea lor adecvată este ţintită de diferite metode şi indivizi specializaţi. Rezultatele testelor au confirmat abilitatea de a identifica soluţii de înaltă calitate, şi capacitatea de a genera şi exploata variaţiile utile în mod continuu, care eventual vor descoperii optimele globale.

Metodele constituite pe tehnicile competente de optimizare prezentate în această teză, au fost aplicate problemelor reale, cum ar fi aproximaţia semnalului ECG (Iclanzan & Dumitrescu, 2006b; Iclanzan et al., 2006), învăţarea parametrilor de clustering (Szilágyi et al., 2008, 2009) şi optimizarea modelelor de inimă (Szilagyi et al., 2009).

10.2 Posibile Extensii

Munca pe viitor va fi considera aplicarea ghidajului corelaţiilor pentru îmbunătăţire calitativă a construirii modelelor în alţi EDA competenţi. Este de mare interes crearea noilor tehnici care vor utiliza eficient informaţia obţinută, prin analizarea grupurilor restricţionare de variabile binare.

Cercetările următoare vor aprofunda studiul GCEDA prin interpretarea grupărilor ca şi grafuri aciclice dirijate, pentru construirea reţeleor Bayaesiene. Alt domeniu de cercetare se va concentra asupra dezvoltării construirii de model paralele şi optimizarea pe scală largă. Dorim să exploatăm construirea de model rapidă, automată şi cerinţe de memorie scăzute ale metodelor propuse în capitolul 6 pe probleme cu milioane de variabile, unde metodele clasice de rezolvare, bazate pe populaţie, implică costuri de memorie foarte mari.

Munca de viitoar se va concentra şi asupra exploatării avantajelor oferite de paradigma cooperativă descrisă în capitolul 9. Arii posibile de cercetare:

- Favorizarea explorării spaţiului căutării prin divizarea în sub regiuni sau învăţarea topologiei acesteia. O căutare inteligentă se poate dovedi eficient şi in cazul funcţiilor deceptive.
- Incorporarea analizei şi a tehnicilor învăţării automate care pot dezvălui şi exploata proprietăţile spaţiul căutării. Învăţarea cuplajelor şi construirea de model poate fi folosită sau strategiile de căutare utilizate pot fi alterate adaptativ.
- Paralelizarea cadrului.
- Folosireade evaluări proxy în fazele de explorare.

Bibliografie

Blum, C. & Roli, A. (2003). Metaheuristics in Combinatorial Optimization: Overview and conceptual comparison. *ACM Comput. Surv.*, 35(3), 268–308. [cited at p. 52]

Brohée, S. & van Helden, J. (2006). Evaluation of clustering algorithms for protein-protein interaction networks. *BMC Bioinformatics*, 7, 488. [cited at p. 38]

Correa, E. & Shapiro, J. (2006). Model complexity vs. performance in the bayesian optimization algorithm. *Parallel Problem Solving from Nature-PPSN IX*, (pp. 998–1007). [cited at p. 41]

Duque, T. S., Goldberg, D. E., & Sastry, K. (2008). Enhancing the efficiency of the ECGA. In *Proceedings of the 10th international conference on Parallel Problem Solving from Nature* (pp. 165–174). Berlin, Heidelberg: Springer-Verlag. [cited at p. 14]

Etxeberria, R. & Larranaga, P. (1999). Global optimization using Bayesian networks. In *Proceedings of the Second Symposium on Artificial Intelligence (CIMAF-99)* (pp. 151–173). [cited at p. 39]

Harik, G. (1999). *Linkage Learning via Probabilistic Modeling in the ECGA*. Technical Report IlliGAL Report no. 99010, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign. [cited at p. 12, 39, 40]

Iclanzan, D. (2005). Genetic engineering algorithm. In *Proceedings of the 7th International Scientific Student Conference on Technical Sciences* Timisoara, Romania: Universitatea de Vest. Distributed on CD. [cited at p. 6]

Iclanzan, D. (2006). Genetic engineering as an optimization paradigm. In *Proceedings of FMTU 2006* (pp. 115–121). Cluj-Napoca, Romania: Transylvanian Musem Society. [cited at p. 6]

Iclanzan, D. (2007a). The creativity potential within Evolutionary Algorithms. In F. A. e Costa et al. (Ed.), *Advances in Artificial Life, 9th European Conference, ECAL 2007, Lisbon, Portugal, September 10-14, 2007, Proceedings*, volume 4648 of *Lecture Notes in Computer Science* (pp. 845–854).: Springer. [cited at p. 5]

Iclanzan, D. (2007b). Crossover: the divine afflatus in search. In P. A. N. Bosman (Ed.), *Late breaking paper at Genetic and Evolutionary Computation Conference (GECCO'2007)* (pp. 2497–2502). London, United Kingdom: ACM Press. [cited at p. 5]

Iclanzan, D. & Dumitrescu, D. (2006a). Competitive vs. cooperative optimization. In *Proceedings of the 8th International Scientific Student Conference on Technical Sciences* (pp. 115–121). Timisoara, Romania: Universitatea de Vest. Distributed on CD. [cited at p. 6]

Iclanzan, D. & Dumitrescu, D. (2006b). ECG parameter estimation using advanced stochastic search methods. In D. Isoc & E. Stancel (Eds.), *2006 IEEE-TTTC International Conference on Automation, Quality and tesing, Robotics. Digest of Junior Section* (pp. 17–22). Cluj-Napoca, Romania: Universitatea Technica Cluj. [cited at p. 6, 63]

Iclanzan, D. & Dumitrescu, D. (2007a). Exact model building in Hierarchical Complex Systems. In *Studia Universitatis Babes-Bolyai, Informatica Series*, volume Special Issue: KEPT 2007 - Knowledge Engineering: Principles and Techniques, Proceedings (pp. 161–168). Cluj-Napoca, Romania: Universitas Napocensis, Presa Universitara. [cited at p. 5]

Iclanzan, D. & Dumitrescu, D. (2007b). Overcoming hierarchical difficulty by hill-climbing the building block structure. In D. T. et al. (Ed.), *GECCO '07: Proceedings of the 9th annual conference on Genetic and Evolutionary Computation*, volume 2 (pp. 1256–1263). London: ACM Press. [cited at p. 5, 31]

Iclanzan, D. & Dumitrescu, D. (2007c). Overrepresentation in neutral genotype-phenotype mappings and their applications. In *Symbolic and Numeric Algorithms for Scientific Computing, 2007. SYNASC. International Symposium on* (pp. 427–432). Timisoara, Romania: IEEE Computer Society. [cited at p. 5]

Iclanzan, D. & Dumitrescu, D. (2008a). Going for the big fishes: Discovering and combining large neutral and massively multimodal building-blocks with model based macro-mutation. In *GECCO '08: Proceedings of the 10th annual conference on Genetic and evolutionary computation* (pp. 423–430). Atlanta, GA, USA: ACM. [cited at p. 4]

Iclanzan, D. & Dumitrescu, D. (2008b). How can artificial neural networks help making the intractable search spaces tractable. In *2008 IEEE World*

Congress on Computational Intelligence (WCCI 2008) (pp. 4016–4023). Hong-Kong. [cited at p. 5]

Iclanzan, D. & Dumitrescu, D. (2008c). Large-scale optimization of non-separable building-block problems. In G. Rudolph, T. Jansen, S. M. Lucas, C. Poloni, & N. Beume (Eds.), *PPSN*, volume 5199 of *Lecture Notes in Computer Science* (pp. 899–908).: Springer. [cited at p. 4]

Iclanzan, D. & Dumitrescu, D. (2008d). Towards memoryless model building. In *GECCO '08: Proceedings of the 2008 GECCO conference companion on Genetic and evolutionary computation* (pp. 2147–2152). Atlanta, GA, USA: ACM. [cited at p. 4]

Iclanzan, D. & Dumitrescu, D. (2010). Graph clustering based model building. In *PPSN XI - to appear in Lecture Notes in Computer Science* Krakow, Poland: Springer. (accepted). [cited at p. 3]

Iclanzan, D., Dumitrescu, D., & Hirsbrunner, B. (2009a). Correlation guided model building. In *GECCO '09: Proceedings of the 11th Annual conference on Genetic and evolutionary computation* (pp. 421–428). New York, NY, USA: ACM. [cited at p. 3]

Iclănzan, D., Dumitrescu, D., & Hirsbrunner, B. (2010). Pairwise Interactions Induced Probabilistic Model Building. *Exploitation of Linkage Learning in Evolutionary Algorithms*, (pp. 97–122). [cited at p. 3]

Iclanzan, D., Fulop, P., & Dumitrescu, D. (2007). Neuro-Hill-Climber: A new approach towards more intelligent search and optimization. In *Symbolic and Numeric Algorithms for Scientific Computing, 2007. SYNASC. International Symposium on* (pp. 441–448). Timisoara, Romania: IEEE Computer Society. [cited at p. 5]

Iclanzan, D., Hirsbrunner, B., Courant, M., & Dumitrescu, D. (2009b). Cooperation in the context of sustainable search. In *IEEE Congress on Evolutionary Computation (IEEE CEC 2009)* (pp. 1904 – 1911). Trondheim, Norway. [cited at p. 4]

Iclanzan, D., Szilagyi, S. M., Szilagyi, L., & Benyo, Z. (2006). Advanced heuristic methods for ECG parameter estimation. In *CONTI 2006: Proceedings of the 7th International Conference on Technical Informatics* (pp. 215–220). Timisoara, Romania: Universitatea Politechica. [cited at p. 6, 63]

Jones, T. (1995). *Evolutionary Algorithms, Fitness Landscapes and Search*. PhD thesis, University of New Mexico, Albuquerque, NM. [cited at p. 26]

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220, 671–680. [cited at p. 54]

Lima, C. F., Sastry, K., Goldberg, D. E., & Lobo, F. G. (2005). Combining competent crossover and mutation operators: a probabilistic model building approach. In *GECCO '05* (pp. 735–742). NY, USA: ACM. [cited at p. 12]

Mitchell, M. & Holland, J. H. (1993). When will a genetic algorithm outperform hill climbing? In S. Forrest (Ed.), *Proceedings of the 5th International Conference on Genetic Algorithms* (pp. 647–647). San Mateo, CA, USA: Morgan Kaufmann. [cited at p. 42]

Pelikan, M. & Goldberg, D. E. (2001). Escaping hierarchical traps with competent genetic algorithms. In L. S. et al. (Ed.), *GECCO '01:* (pp. 511–518). San Francisco, California, USA: Morgan Kaufmann. [cited at p. 31]

Pelikan, M., Goldberg, D. E., & Cantú-Paz, E. (1999). BOA: The Bayesian optimization algorithm. In W. B. et al. (Ed.), *GECCO '99*, volume I (pp. 525–532). Orlando, FL: Morgan Kaufmann Publishers, San Fransisco, CA. [cited at p. 39, 41]

Rind, D. (1999). Complexity and climate. *Science*, 284(5411), 105–107. [cited at p. 2]

Russel, S. & Norvig, P. (1995). *Artificial Intelligence: a Modern Approach.* Prentice-Hall. [cited at p. 1]

Shor, N. Z., Kiwiel, K. C., & Ruszcayǹski, A. (1985). *Minimization methods for non-differentiable functions.* New York, NY, USA: Springer-Verlag New York, Inc. [cited at p. 55]

Simon, H. A. (1969). *The Sciences of the Artificial.* Cambridge, Massachusetts: MIT Press, first edition. [cited at p. 2]

Szilágyi, L., Iclanzan, D., Szilágyi, S. M., & Dumitrescu, D. (2008). Gecim: A novel generalized approach to c-means clustering. In J. Ruiz-Shulcloper & W. G. Kropatsch (Eds.), *CIARP*, volume 5197 of *Lecture Notes in Computer Science* (pp. 235–242).: Springer. [cited at p. 4, 63]

Szilágyi, L., Iclanzan, D., Szilágyi, S. M., Dumitrescu, D., & Hirsbrunner, B. (2009). A generalized c-means clustering model optimized via evolutionary computation. In *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'09, Jeju Island, Korea)* (pp. 451 – 455). [cited at p. 4, 63]

Szilagyi, L., Szilagyi, S. M., Iclanzan, D., & Benyo, Z. (2006a). Quick ECG signal processing methods for on-line holter monitoring systems. In *CONTI 2006: Proceedings of the 7th International Conference on Technical Informatics* (pp. 221–226). Timisoara, Romania: Universitatea Politechica. [cited at p. 6]

Szilagyi, S. M., Szilagyi, L., Iclanzan, D., & Benyo, Z. (2006b). Adaptive ECG signal analysis for enhanced state recognition and diagnosis. In *CONTI 2006: Proceedings of the 7th International Conference on Technical Informatics* (pp. 209–214). Timisoara, Romania: Universitatea Politechica. [cited at p. 6]

Szilagyi, S. M., Szilagyi, L., Iclanzan, D., & Benyo, Z. (2006c). Unified neural network-based adaptive ECG signal analysis and compression. *SB-UPT TACCS*, 56(65)(4), 27–36. [cited at p. 6]

Szilagyi, S. M., Szilagyi, L., Iclanzan, D., & Benyo, Z. (2009). A weighted patient specific electromechanical model of the heart. In *Proc. 5th International Symposium on Applied Computational Intelligence and Informatics (SACI 2009)* (pp. 105–110). Timisoara, Romania. [cited at p. 4, 63]

Thierens, D. & Goldberg, D. E. (1993). Mixing in genetic algorithms. In S. Forrest (Ed.), *Proc. of the 5th International Conference on Genetic Algorithms* (pp. 38–47). San Francisco, CA, USA: Morgan Kaufmann. [cited at p. 16]

Torczon, V. J. (1997). On the convergence of pattern search algorithms. *SIAM J. Optim.*, 7(1), 1–25. [cited at p. 55]

Turing, A. M. (1969). Intelligent machinery. In B. Meltzer & D. Michie (Eds.), *Machine Intelligence*, volume 5 chapter 1, (pp. 3–23). Edinburgh, UK: Edinburgh University Press. [cited at p. 1]

van Dongen, S. (2000a). *Graph Clustering by Flow Simulation.* PhD thesis, U. of Utrecht. [cited at p. 38, 42]

van Dongen, S. S. (2000b). *A stochastic uncoupling process for graphs.* Technical Report INS-R0010, National Research Institute for Mathematics and Computer Science in the Netherlands, Amsterdam. [cited at p. 38]

Watson, Beck, Howe, & Whitley (2003). Problem difficulty for tabu search in job-shop scheduling. *AIJ: Artificial Intelligence*, 143. [cited at p. 16]

Watson, R. A., Hornby, G., & Pollack, J. B. (1998). Modeling building-block interdependency. In *PPSN V: Proc. of the 5th International Conference on Parallel Problem Solving from Nature* (pp. 97–108). London, UK: Springer, LNCS. [cited at p. 31]

Watson, R. A. & Pollack, J. B. (1999). Hierarchically consistent test problems for genetic algorithms: Summary and additional results. In S. Brave (Ed.), *GECCO '99: Late Breaking Papers* (pp. 292–297). Orlando, Florida, USA. [cited at p. 31]

Yu, T.-L. & Goldberg, D. E. (2006). Conquering hierarchical difficulty by explicit chunking: substructural chromosome compression. In *GECCO '06:* (pp. 1385–1392). NY, USA: ACM Press. [cited at p. 39]

Yu, T.-L., Goldberg, D. E., Yassine, A., & Chen, Y.-P. (2003). Genetic algorithm design inspired by organizational theory: Pilot study of a dependency structure matrix driven genetic algorithm. In *Genetic and Evolutionary Computation – GECCO-2003*, volume 2724 of *LNCS* (pp. 1620–1621). Chicago: Springer-Verlag. [cited at p. 39]

Yu, T.-L., Sastry, K., & Goldberg, D. E. (2005). Linkage learning, overlapping building blocks, and systematic strategy for scalable recombination. In *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation* (pp. 1217–1224). New York, NY, USA: ACM. [cited at p. 41]