CODIFICAREA INFORMAŢIEI ÎN NOILE MODELE DE CALCUL. CALCUL BAZAT PE ADN ŞI CALCULUL CU MEMBRANE CELULARE

LUCIANA MARIA MOROGAN
Universitatea Spiru Haret

Rezumat: Cercetări recente au considerat atât ADN-ul, cât şi membranele celulare, drept medii pentru calculul la scară mare dar şi pentru stocarea ultra compactă a informaţiei (ADN-ul acţionează drept mecanism de stocare şi codificare a informaţiei genetice, deşi nu este folosit direct în catalizarea proceselor celulare, ci, mai degrabă, ARN-ul şi proteinele actionează drept enzime ce catalizează reacţiile din interiorul celulelor). Lucrarea de faţă însumează un colaj de modele de codificare a informaţiei (colaj realizat în urma cercetării printre lucrările scrise până în prezent) privite din perspectiva a două noi modele de calcul inspirate de micro – biologie: calculul bazat pe ADN şi calculul folosind P-sisteme (realizat pentru reprezentarea unară a unui număr întreg pozitiv scris sub formă binară).

Cuvinte cheie: DNA computing, membrane computing, P systems.

Clasificare AMS: 94A60, 92D99, 74K15

1. INTRODUCERE

Tratamentul matematic al geneticii reprezintă un subiect vechi, cu o tradiţie onorabilă ce ne duce înapoi la Gregor Mendel, cercetător austriac care a demonstrat în 1865 cum caracteristicile plantei de mazăre au fost moştenite de la o generaţie la alta într-un mod previzibil. Recent însă, conexiunea dintre matematică şi biologie a devenit mai apropiată în urma analizării cantităţii de date biologice, aflate într-o continuă creştere, pe baza conceptelor matematice fundamentale, ceea ce face parte din "*revoluţia biologică, cea mai mare revoluţie ştiinţifică a timpurilor noastre, şi, posibil, a tuturor timpurilor*" (Michael C. Reed – matematician de renume).

Compresia şi transmiterea datelor nu este aplicabilă doar mediilor electronice: natura a "dezvoltat" un sistem capabil să ofere codajul instrucţiunilor necesare supravieţuirii celulare. Poate este corect să afirmăm că cel mai important cod existent

este *codul genetic*, dar, să nu omitem faptul că acesta reprezintă, în esenţă, doar un cod. Din aceste considerente, o aplicaţie o reprezintă *sistemele criptografice moleculare* (ce folosesc codificarea informaţiei). În stilul altor ramuri ale *calculului natural* (*Natural Computing*), învăţăm de la natura vie noi modele de calcul şi noi strategii (să le spunem… paradigme) bazate pe *calculul ADN-ului* (*DNA computing*) şi pe şi mai noul model de calcul: *calculul cu membrane* (*membrane computing*) – *P sisteme* (încercări timide se cunosc însă în aceste două direcţii).

2. REMARCI GENERALE

O *codificare* este o aplicaţie injectivă ce asociază fiecărui cuvânt sursă dintr-o mulţime X, un şir de simboluri ale unui alfabet Y. Presupunem că avem cuvintele sursă x_i, x_j cu $i \neq j$ şi $f(x_i) \neq f(x_j)$. Un *mesaj* se defineşte ca fiind orice şir de cuvinte sursă din $X = \{x_1, ..., x_m\}$. Numim *cuvinte cod* acele şiruri peste Y de forma $f(x_i), 1 \leq i \leq m.$ Mulţimea de cuvinte cod de forma $f(x_i)$ reprezintă un *cod* C. Operaţia de codificare *f* se poate extinde natural la mulţimea mesajelor : *f(xy) = f(x) f(y).*

O codificare f se numeşte a fi *unic decriptabilă* dacă nu există două mesaje $M_1 \neq M_2$ astfel încât $f(M_1) \neq f(M_2)$. O codificare f este un *cod instantaneu* (*cod prefix*) dacă nu există două cuvinte sursă $x_i \neq x_j$ astfel încât $f(x_i)$ să fie prefix pentru $f(x_j)$ (nu putem avea $f(x_j) = f(x_i)y, y \in Y$). Astfel, un cod prefix poate fi codificat în mod unic, de la stânga la dreapta, fără a "privi înainte". Codurile prefix sunt cunoscute şi sub numele de *coduri libere de prefix* (*prefix-free codes*) [13].

Lemă. Dacă f este un cod liber de prefix, atunci codificarea f este unic decriptabilă. Demonstraţia lemei se găseşte în [13].

Considerăm *q* un număr prim şi $n \in N^*$. Se numeşte *cod liniar* orice subspaţiu liniar al lui Z_q^n. Un subspaţiu *k* dimensional al lui Z_q^n se numeşte (*n*, *k*) – *cod liniar* peste alfabetul Z_q şi se notează $A_{n,q}$. Acesta admite o bază formată din *k* vectori cu *n* elemente. Dacă $A_{n,q}$ este un cod liniar cu baza $e_1, ..., e_k$, atunci matricea $G_{n,k} = \begin{pmatrix} e_1 \\ : \\ e_k \end{pmatrix}$ se numeşte matricea generatoare a codului. Operaţia de codificare, Φ, este definită prin matricea generatoare: $\forall u \in Z_q^k, \Phi(u) = uG$. Dacă $A_{n,q}$ este un cod liniar generat de matricea $G = G_{n,k}$, se numeşte *matrice de control* o matrice $H = H_{n-k,n}$ cu proprietatea ca $GH^T = 0$ de rang $n - k$.

Distanţa Hamming H(x,y) dintre două cuvinte $x = x_1x_2...x_n$ şi $y = y_1y_2...y_n$, reprezintă numărul indicilor *i* astfel încât $x_i \neq y_i$ (*distanţa Hamming* a unui cod liniar

este cea mai mică distanţă dintre elementele codului). Spunem că un (n, k) - cod liniar are distanţa minimă d dacă şi numai dacă poate detecta orice combinaţie de maxim $d - 1$ erori (dacă şi numai dacă poate corecta orice combinaţie de maxim $\left[\frac{d-1}{2}\right]$ erori) şi verifică relaţia $d \le n - k + 1$.

Se numeşte *extensie* a unui (n, k) – cod liniar A peste Z_q, (n + 1, k) – codul liniar A^* obţinut prin adăugarea la fiecare cuvânt cod $a_1,...,a_n$ a unui simbol nou a_{n+1} (în cazul binar numit *bit de paritate*) cu proprietatea că $\sum_{i=1}^{n+1} a_i = 0(\mathrm{mod}\, q)$. Pentru H matricea de control a codului $A_{n,q}$, considerăm matricea de control H^* a codului extins $A^*_{n+1,k}$ ca fiind de forma

$$H^* = \left(\begin{array}{ccccc|c} & & H & & & 0 \\ & & & & & \vdots \\ & & & & & 0 \\ \hline 1 & 1 & \dots & 1 & 1 & 1 \end{array}\right)$$

Dacă un cod liniar binar A are distanţa minimă impară d, atunci codul extins are distanţa minimă d + 1. Ultima linie a matricii de control reprezintă ecuaţia: $\sum_{i=1}^{n} x_i = 1$. Un cod A de lungime n corectează t erori şi detectează s erori simultan dacă orice cuvânt cod v are proprietatea:

$$\forall w \in Z_q^n [d(w,v) \le s \Leftrightarrow \forall a \in A \setminus \{v\}, d(w,a) > A].$$

Teoremă. Un cod corectează t erori şi detectează s erori simultan dacă şi numai dacă d $\ge t + s + 1$ [13].

Codul liniar binar în care coloanele matricii H sunt reprezentarea binară a numerelor $1, 2, ..., 2^r - 1$ se numeşte *cod Hamming binar*. Deci, pentru orice $r \in N$ $(r \ge 2)$ putem construi un (n, k) – cod liniar în care $n = 2^r - 1, k = 2^r - r - 1$. Cum un cod H are distanţa minimă 3, poate corecta o eroare şi detecta 2 (dar nu poate realiza simultan acest lucru). Codul H poate fi îmbunătăţit prin extensie, operaţie ce duce la un $\left(2^m, 2^m - m - 1\right)$ - cod liniar cu toate cuvintele cod de pondere pară. Un cod H extins este soluţia unui sistem liniar de n - k + 1 ecuaţii; ecuaţia suplimentară $\sum_{i=1}^{n+1} x_i = 0$ se numeşte *ecuaţia de control a parităţii*. Un cod H extins are d = 4. Codul H extins corectează o eroare simplă şi detectează două erori simultan.

Considerăm un cod de lungime $n = n_1 n_2$. Cuvintele codului pot fi reprezentate ca matrici cu n_1 linii şi n_2 coloane prin scrierea cuvântului cod **a** = $a_0 a_1 ... a_{n-1}$ ca o

matrice $X = [x_{i,j}], 0 \le i \le n_1 - 1, 0 \le j \le n_2 - 1$, unde $x_{i,j} = a_{in_2+j}$, numită *scriere canonică*. Dacă A_1, A_2 sunt două coduri liniare de lungime n_1 respectiv n_2, putem construi codul A de lungime $n_1 n_2$ folosind scrierea canonică. Spunem că $A = A_1 \times A_2$ este *codul produs* al lui A_1 cu A_2 dacă şi numai dacă A constă din toate cuvintele – cod în care reprezentarea matriceală verifică proprietăţile: (i) Fiecare coloană a unei matrici este un cuvânt – cod din A_1; (ii) Fiecare linie a unei matrici este un cuvânt – cod din A_2.

Structura ADN–ului. *ADN*–ul (*Acidul DezoxiriboNucleic*) [17] este un *polimer* (substanţă compusă din molecule cu masă moleculară mare, alcătuite din unităţi structurale repetate sau monomeri repetaţi, conectaţi prin legături chimice covalente) alcătuit din *monomeri* (cuvânt de provenienţă grecească, *mono* "unu" şi *meros* "parte" - moleculă mică ce poate fi legată chimic de alţi monomeri alcătuind astfel un polimer) numiţi *deoxiribonucleotide*. ADN-ul reprezintă molecula crucială a organismelor vii, având o structură fascinantă care suportă cele mai importante două funcţii ale acesteia: codarea pentru producerea proteinelor şi propria reproducere astfel încât o copie exactă a acesteia să fie trecută noilor celule ce urmează a fi create.

Speranţele înalte privind viitorul calculului ADN-ului se bazează pe două trăsături fundamentale [1]: (i) *paralelismul* masiv al catenelor de ADN şi (ii) *complementaritatea* Watson-Crick. Descriem pe scurt cele două trăsături.

(i) Majoritatea faimoaselor probleme de calcul se rezolvă printr-o căutare exhaustivă prin toate soluţiile posibile. Dificultatea incontestabilă constă în faptul că o asemenea căutare este prea vastă pentru a putea fi dusă la capăt folosind tehnologia actuală. Pe de altă parte, densitatea informaţiei, conţinută în moleculele de ADN, cât şi uşurinţa de construcţie a copiilor acestei informaţii pot face posibile aceste căutari exhaustive. Un exemplu îl constituie criptanaliza, într-un sistem clasic de criptare (cu chei relativ puţine – până la 2^{56}), unui text cifrat în care toate posibilele chei pot fi încercate simultan [1].

(ii) Complementaritatea Watson-Crick reprezintă o trăsătură furnizată "gratuit" de către natură. Când au loc legăturile dintre cele două catene ale moleculei de ADN, ştim că bazele opuse una celeilalte sunt complementare. Dacă cunoaştem unul dintre membrii legăturii, automat îl cunoaştem şi pe celălalt.

Structura membranei. *Calculul cu membrane* (*Membrane Computing - MC*) ([2] şi [3]) este o latură a informaticii bazată pe idei de calcul abstract şi modele ale structurii şi funcţionalităţii celulelor vii, cât şi pe modul în care celulele sunt organizate în ţesuturi sau alte structuri superioare ordonate. Dispozitivul de calcul cu membrane este numit *P sistem*. Ingredientul principal al unui P sistem îl reprezintă *structura de membrană* reprezentată de o mulţime de membrane aşezate ierarhic conţinute într-o membrană externă distinctă, care corespunde plasmei membranei

celulare, numită *piele (skin)*. În interorul acesteia pot fi plasate mai multe membrane, fiecare determinând un compartiment numit *regiune*. Compartimentele unei celule conţin substanţe ce înoată într-o soluţie apoasă. Aici nu există ordine, orice este închis la orice, ceea ce contează este doar concentraţia, adică populaţia, *numărul de copii ale fiecărei molecule*. Aceasta sugerează lucrul cu mulţimi de obiecte ale căror multiplicitate contează, numite *multiseturi*. Formal, fie U o mulţime arbitrară, un *multiset* peste U este o aplicaţie $M : U \to N$, $M(a)$ reprezentând multiplicitatea lui a în multisetul M, oricare ar fi a din U. Putem indica aceasta prin perechea $(a, M(a))$.

Obiectele evoluează prin intermediul unor *reguli de evoluţie* a căror localizare este asociată regiunilor unei celule. Există trei tipuri de astfel de reguli: reguli de rescriere multiset (membrane şi reguli de evoluţie), reguli de comunicare şi reguli de dezvoltare([2], [3]).

3. SCRIEREA INFORMAŢIEI ÎN ADN

O dată cu tehnologia care avansează, cererea pentru scrierea informaţiei în interiorul ADN-ului creşte. Domeniile de aplicabilitate imediată, asemenea biotehnologiilor convenţionale în sensul că încearcă să *codifice informaţia artificială în ADN*, sunt:

1. *calculul bazat pe ADN* (care încearcă să realizeze matematica biologică, adică, rezolvarea problemelor matematice prin aplicarea metodelor experimentale în biologia moleculară; aplicaţii se găsesc şi în domeniul criptografiei);
2. *memorarea datelor de tip ADN* (care advocă folosirea ADN-ului bacterial ca structură de memorare a datelor de densitate mare şi de lungă durată şi care poate fi rezistent la radiaţii);
3. *semnătura ADN-ului* (care este importantă pentru înregistrarea "copyright"-ului genomului bacterial şi a celui viral). Steganografia (semnatură invizibilă ascunsă în altă informaţie) este folositoare în acest sens.

3.1. CERINŢE PENTRU CREEAREA UNUI COD DE TIP ADN

Considerăm o mulţime de cuvinte de tip ADN pentru schimbul de informaţie. Cuvintele trebuie s[fie distincte şi de lungimi egale, astfel încât nici un cuvânt să nu producă hibridizări nedorite în ceea ce priveşte aranjamentul acestora. (*Hibridizarea* reprezintă procesul de combinare a acizilor nucleici monocatenari complementari într-o singură moleculă. Nucleodidele se vor lega la complementele lor în condiţii normale, proces numit *renaturare*[17]). În acelaşi timp, toate cuvintele trebuie să fie uniforme, din punct de vedere fizico-chimic, pentru a garanta, în experimentele biologice, o reacţie în care probabilitatea de emitere a valorii 1 este de exact $\frac{1}{2}$ (termenul folosit în limba engleză pentru o secvenţă de acest tip este cel de *unbiased* [11]).

În principiu, există două măsuri pentru evaluarea calitaţii cuvintelor ADN create: statistic termodinamice şi combinatoriale. Deşi metodele termodinamice redau

o estimare destul de clară, costul de calcul este destul de ridicat, motiv pentru care ne axăm pe estimările combinatoriale, care sunt în esenţă apropiate de cele termodinamice. În cele ce urmează vor fi introduse cerinţele formale pentru mulţimea de cuvinte de tip ADN.

3.1.1. CONSTRÂNGERI ASUPRA SECVENŢELOR

Fie $x = x_1 x_2 ... x_n$ un cuvânt ADN generat de cele patru baze {*A*, *C*, *G*, *T*} ({adenine, citosine, guanine, timine}). *Inversul* lui x este $x^R = x_n x_{n-1} ... x_1$ iar *complementul* lui *x*, obţinut prin înlocuirea bazelor A cu T şi C cu G în *x* şi vice-versa, îl notăm cu x^C. Pentru o mulţime *S* de cuvinte ADN, notăm complementarea cu secvenţele complementului opus cu

$$S^{RC} = \{x \mid x \in S \text{ sau } (x^R)^C \in S\}. \qquad (1)$$

Constrângeri Hamming [4]. La fel ca în teoria codurilor, cuvintele ADN proiectate trebuie să păstreze o distanţă Hamming mare între perechile de cuvinte. Ceea ce face proiectarea codului ADN-ului mai complicată decât teoria standard a codurilor corectoare de erori (a se vedea bibliografia [11], [12] şi [13]) este faptul că trebuie să considerăm nu doar *H*(*x*, *y*) ci şi $H(x^C, y^R)$ pentru a garanta *nepotrivirile* hibridizării cu alte cuvinte şi complementele acestora. (O bază necomplementară într-o catenă dublă nu poate forma legături de hidrogen stabile şi poartă numele de *nepotrivire* (a bazei). În engleză, (base) *mismatch* [13]).

Constrângeri de tip "liber de prefix" [4]. În proiectarea cuvintelor ADN se doreşte ca acestea să fie de tipul "libere de prefix". Proprietatea prin care orice cuvânt concatenat diferă de orice alt cuvânt prin cel putin *d* poziţii se numeşte *liber de prefix de index d*. Deci, codul ADN trebuie să fie de tipul "liber de prefix" cu index mare.

Constrângeri legate de energie [4]. Temperatura de topire a tuturor cuvintelor ADN trebuie să fie asemănătoare pentru a garanta comportamentul lor comun *in vitro*. Estimarea pe care ne putem baza este aproximarea cea mai apropiată la care temperatura trebuie calculată de la frecvenţa de16 *baze dimer* (de la *AA* la *TT*), unde *frecvenţa dimer* a unei secvenţe *x* este reprezentată de tripletul de intregi ce reprezintă, fiecare, frecvenţele a trei şabloane. Pentru a integra şi bazele terminale se presupune că *x* este ciclic în calculul frecvenţei. Arita şi Kobayashi [18] au propus această aproximare grupând [*GC*] şi [*AT*] ale căror temperaturi de topire depind de frecvenţa a trei şabloane: [*GC*][*GC*], [*GC*][*AT*] sau [*AT*][*GC*] şi [*AT*][*AT*].

Alte constrângeri [4] ce depind de modelul folosit sunt:

1. *Subcuvintele interzise* ce corespund locurilor de restricţie, repetărilor simple sau altor secvenţe de semnal biologice, nu trebuie să apară oriunde în cuvintele proiectate. Aceste constrângeri apar în cazul în care modelul de

codificare foloseşte secvenţe pre -determinate cum ar fi locurile de restricţie pentru enzime.

2. *Orice cuvânt de lungime k* nu trebuie să apară mai mult de o dată în cuvântul dorit. Numărul *k* este de obicei mai mare sau egal cu 6.
3. *O structură secundară* ce implică hibridizarea aşteptată a cuvintelor de tip ADN nu trebuie să existe. Pentru a găsi structura optimă pentru aceste cuvinte, minimul de energie liberă a catenelor este calculat de programarea dinamică. Această constrângere apare în cazul în care controlul temperaturii este important pentru modelul de codificare.
4. *Doar trei baze*, A, C şi T pot fi folosite în proiectarea cuvintelor. Această constrângere serveşte în principal la reducerea numărului de nepotriviri. Pentru cuvintele proiectate pe baza ARN-ului, constrângerea este importantă pentru stabilitatea ambelor perechi *G* - *C* şi *G* - *U* (echivalentul perechii *G* - *T* în ADN).

3.1.2. MODURI DE STOCARE A DATELOR

Abordarea bazată pe suprafaţă (faza solidă) [4] presupune plasarea cuvintelor ADN pe un suport solid. Cum una dintre cele două catene ale unei molecule dublu catenare este imobilizată, cuvintele cod pot fi separate de complementele lor, reducând astfel riscul unei agregări neaşteptate a cuvintelor – ceea ce reprezintă un avantaj al utilizării acestei metode. Un alt avantaj îl reprezintă eficacitatea etichetării fluorescente, deoarece este mai simplu de recunoscut cuvintele pentru citirea informaţiei.

Abordarea solubilă (faza lichidă) [4]. Accesul la informaţie din cazul anterior poate limita abilitaţile biomoleculare. Fragmente de ADN în soluţie sunt capabile să simuleze automate celulare drept purtători ai informaţiei, datorită flexibilităţii acestor fragmente. Un alt avantaj al abordării solubile îl constituie posibilitatea introducerii de ADN în microbi şi astfel, cuvintele să poată fi folosite pentru proiectarea unor nano-structuri.

În concluzie, problema proiectării este formulată astfel: fiind daţi doi întregi *l* şi *d* ($0 < d < l$), se poate proiecta o mulţime *S* de lungime *l* de cuvinte ADN astfel încât S^{RC} este liberă de prefix de index *d* şi pentru orice două secvenţe $x, y \in S^{RC}$,

$$H(x, y) \geq d$$

şi

$$H(x^C, y^D) \geq d.$$

3.2. METODE DE PROIECTARE A CODULUI DE CUVINTE ADN

Strategia şablon-plan. Tratamentul matematic al generării oligonucleotidelor se bazează pe strategia şablon-plan, strategie ce utilizează codurile Hamming [13] pentru a oferi şabloane şi plane necesare generării de secvenţe ADN. Un şablon

reprezintă o secvenţă nucleotidică asupra a doua baze, în timp ce un plan reprezintă un şir de variabile binare. Secvenţele de cuvinte ADN sunt generate de operaţia dintre fiecare plan, al unei mulţimi date, asupra fiecărui şablon din altă mulţime.

O mulţime de molecule de lungime l, în care toate distanţele Hamming sunt mai mari sau egale cu n, se notează ca mulţime de forma l:n. Considerând acestea, strategia funcţionează bine pentru cazul $l = 4k$ şi $n = 2k$, cu k număr întreg ($4k$:$2k$). Mulţimile de cuvinte ADN pot fi generate şi pentru cazurile $l \neq 4k$ în care strategia este folosită pentru creearea de mulţimi de tipul: ($4k$-1) : ($2k$-1), ($4k$+1) : $2k$ şi ($4k$-2) : ($2k$-1). În lucrarea de faţă prezentăm doar cazul $4k$:$2k$ (pentru mai multe detalii, a se urmări [19]).

În funcţie de numărul de plane şi şabloane, pot fi generate mulţimi de oligonucleotide de diferite dimensiuni. Fiecare pereche şablon-plan (t, m) generează o secvenţă ce satisface regula: pentru fiecare bit 1 al planului, bitul corespunzător din şablon se schimbă cu complementul său, şi pentru fiecare bit 0, bitul corespunzător din şablon rămâne neschimbat. De exemplu, o pereche şablon-plan de forma (ACAACCAA, 01100110) se foloseşte pentru generarea următoarei secvenţe: AGTACGTA (a se vedea figura 1).

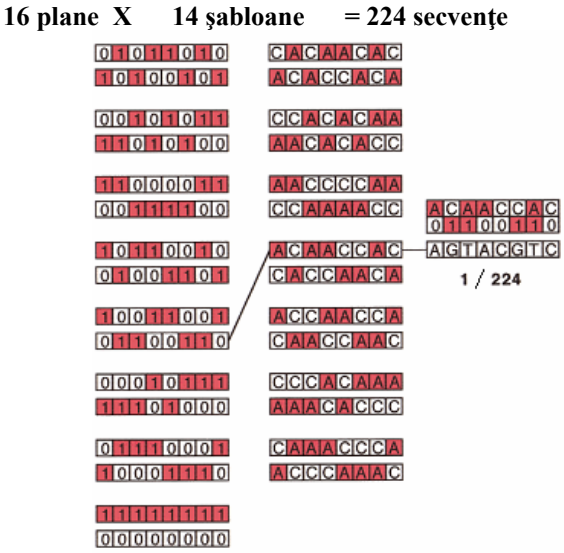

Figura 1. Mulţimea de plane şi şabloane folosite pentru a genera mulţimea de tip l:n = 8:4. Fiecare monomer de lungime 8 este generat prin încrucişarea fiecărui şablon cu fiecare plan, după regula menţionată.

În acest caz, mulţimea de plane şi şabloane folosite pentru generarea mulţimii de tip l:n = 8:4 conduce la generarea unei secvenţe de 224 4bm 8 meri (224 monomeri, de lungime 8, cu distanţă Hamming 4 – prescurtarea "bm" provenind de la expresia în limba engleză "*base mismatch*"). Fiecare plan are o distanţă Hamming de cel puţin 4 (diferă prin cel puţin 4 poziţii de celelalte plane).

Pentru determinarea mulţimii de plane, *M*, şi a mulţimii de şabloane, *T*, codului Hamming i se aplică două constrângeri: (i) conţinutul G/C al fiecărei mulţimi este fixat la aproximativ 50% pentru asigurarea unei stabilităţi termodinamice similare pentru fiecare catenă dublă de ADN, şi (ii) orice două oligonucleotide de lungime *l* din mulţime diferă printr-o distaţă Hamming de cel puţin *n* ($n \approx 1/2$). (Considerând parametrul distanţei *n*, pentru orice pereche de cuvinte distincte (*x*, *y*) din mulţime, trebuie ca $H(x, y) \geq n$). Fiecare secvenţă de ADN se distinge în mod unic de celelalte secvenţe prin hibridizarea complementului acesteia.

Deci considerăm astfel, codul Hamming binar de forma (*l*, *G*, *n*) ca fiind o mulţime de vectori *G* peste {0, 1}, de lungime *l*, astfel încât orice doi vectori deţin între ei o distanţă Hamming de cel puţin *n*. Pentru generarea unui număr maxim de secvenţe pentru mulţimi de tip 4*k*:2*k*, *k* număr întreg, se impun două condiţii: (i) mulţimea planelor, *M*, să fie reprezentată de codurile Hamming (*l*, 2*l*, *l*/2) şi (ii) mulţimea şabloanelor, *T*, să fie similară mulţimii *M*, mai puţin vectorii ce conţin doar 0 şi cel ce conţine doar 1, şi, în plus, înlocuim {0, 1} din codul Hamming cu perechi de baze, cum ar fi: {*A*, *C*}, {*A*, *G*}, {*T*, *C*}, sau {*T*, *G*} şi vice versa. În exemplul de mai sus, {0, 1} a fost înlocuit cu {*C*, *A*} pentru generarea mulţimii de şabloane *T*.

Cod ADN corector de erori cu constrângeri libere de prefix. Printre diferitele constrângeri asupra cuvintelor de cod ADN, cea mai dificilă de satisfăcut este cea de tipul "liber de prefix"[4]. Această proprietate reprezintă o condiţie necesară pentru proiectarea generală a unui cod ADN. Prezentăm în continuare un cod ADN de 112 cuvinte de lungime 12 având distanţa Hamming mai mare sau egală cu 4 în orice hibridizare nedorită (greşită). Cuvintele nu conţin mai mult de 6 subsecvenţe consecutive comune. Temperaturile de topire ale cuvintelor sunt similare.

În proiectare este implicată metoda lui Arita şi Komayashi [14] care generează sistematic o mulţime de cuvinte de lungime *l* astfel încât oricare dintre membrii săi va avea o distanţă Hamming de aproximativ *l*/3 în raport cu alte cuvinte şi cu complementele lor, şi, în plus, se suprapun cu concatenările acestora. La fel ca în strategia şablon-plan, s-au construit secvenţe ca produs de două tipuri de cuvinte binare, cu excepţia folosirii unui singur cuvânt binar, *T*, ca şablon. *T* este ales astfel încât la alinierea cu orice model, să deţină o distanţă Hamming mai mare sau egală cu *d*:

$$T^R \;\; TT^R \;\; T^RT \;\; TT \;\; T^RT^R$$

Şablonul specifică poziţiile *GC* ale cuvântului dezvoltat: [*GC*] va corespunde în şablon fie lui 0 fie lui 1. Cum T^R specifică şablonul *AT/GC* de complemente inverse, distanţa Hamming dintre *T* şi T^R garantează distanţa Haming a bazelor din catenele următoare şi din cele inverse ale ADN-urilor ce urmează a fi dezvoltate (proiectate). Alte şabloane de la *TT* la T^RT^R sunt resposabile de şabloanele shiftate (şabloane deplasate la stânga sau la dreapta cu un număr de biţi).

Pentru cuvintele de tip plan, se foloseşte orice cod binar corector de erori de *distanţă minimă d*. Astfel, orice pereche de cuvinte a codului rezultat induce o distanţa Hamming de cel puţin *d,* fără deplasări bloc de biţi datorită codului corector

de erori şi cu deplasări bloc datorită şablonului ales. Cum un singur şablon este folosit pentru specificarea poziţiilor *GC* pentru toate cuvintele, aranjamentul *GC* al codului rezultat va fi uniform şi, deci, vor rezulta temperaturi de topire similare pentru toate cuvintele.

4. CODIFICAREA INFORMAŢIEI SUB FORMĂ DE NUMERE BINARE ÎN STRUCTURA MEMBRANEI FOLOSIND MULTISETURI

Prezentăm în continuare o metodă simplă de codificare a numerelor binare folosind multiseturi (multiplicarea), şi o familie de P sisteme ce transformă aceste multiseturi în notaţie unară. Ideea codificării numerelor folosind multiseturi se poate extinde la codificarea informaţiei.

Fie $x_n x_{n-1}...x_1$ reprezentarea binară a unui întreg pozitiv x astfel încât

$$x = \sum_{i=1}^{n} x_i 2^{i-1}. \tag{2}$$

Vom folosi obiecte din următorul alfabet:

$$A_n = \{< b, j > | \, b \in \{0,1\}, j \in \{1,2,...,n\}\},$$

unde obiectul $< b, j >$ reprezintă bitul b de pe poziţia j în codificarea binară a unui număr întreg.
Pentru reprezentarea numărului x de mai sus, folosim multisetul :

$$< x_n, n >, < x_{n-1}, n-1 >, ..., < x_1, 1 > \tag{3}$$

(alfabetul A_n depinde de lungimea reprezentării binare a lui x şi deci folosind A_n putem reprezenta numere întregi din intervalul cuprins între 0 şi $2^n - 1$)

Pe de altă parte, reprezentarea unară a lui x se obţine prin alegerea unui simbol a dintr-un alfabet A' şi punând în multiset x copii ale acestui simbol: a^x. În concluzie, reprezentarea unară este exponenţial mai lungă decât cea binară. Astfel, pentru a oferi valorile de intrare pentru P sisteme, trebuie inserat în acestea un număr exponenţial de obiecte ceea ce duce la o cantitate exponenţială de lucru pentru pregătirea sistemului. În schimb, lucrul cu numere codificate binar, permite prepararea sistemului prin inserarea unui număr de obiecte limitat polinomial.

4.1. REPREZENTAREA UNARĂ A NUMERELOR BINARE

În cele ce urmează, este prezentată o familie de P sisteme ce permit reprezentarea, în notaţie unară, a unui număr întreg pozitiv dat x, exprimat în notaţie binară. Obiectele folosite de P sistemele din familie formează o submulţime a alfabetului A_n definit mai sus. Pentru reprezentarea lui x în notaţie binară se folosesc doar obiecte care corespund biţilor egali cu 1 ai reprezentării lui x. *De exemplu*, dacă x = 23, vom avea reprezentarea binară a lui x prin 10111 şi vom folosi obiectele: <1,

5>, <1, 3>, <1, 2> şi <1, 1>. Cum primul element al perechilor este întotdeauna 1, îl vom omite şi implicit vom omite şi parantezele unghiulare.

Familia P sistemelor care va produce transformarea se defineşte astfel:

$$\Pi = (A(n), \mu, w, R(n), i_{in}, i_{out}), \text{ unde:}$$

- $A(n) = \{1, 2, \ldots, m\} \cup \{a\}$ este alfabetul;
- $\mu = []_{skin}$ reprezintă structura membranei ce conţine doar membrana skin;
- $w = \varnothing$ - multisetul iniţial de obiecte prezente în regiunea marcată de membrană;
- $R(n)$ – mulţimea regulilor de evoluţie care sunt de forma:

$$[j \to (j-1)^2]_{skin}, \text{ pentru toti } j \in \{1, 2, ..., n\}$$
$$[j \to a]_{skin}$$

- $i_{in} = skin$ specifică membrana de intrare a lui $\Pi(n)$;
- $i_{out} = skin$ specifică membrana de ieşire a lui $\Pi(n)$.

Semantica regulilor este semantica obişnuită a regulilor de evoluţie (a se vedea bibliografia, [3] şi [6]) aplicate într-o manieră paralelă. Numărul paşilor celulari ai P sistemului este limitat de n şi calculul se opreşte când nici o altă regulă nu mai poate fi aplicată, iar multisetul plasat în membrana de ieşire reprezintă output-ul calculului.

Realizarea calculului. În general, un *calcul* al unui P sistem începe plecând de la o configuraţie iniţială a sistemului şi se termină atunci când nici o regulă de evoluţie nu mai poate fi aplicată. Rezultatul va reprezenta multisetul de obiecte conţinut în *membrana de ieşire* sau eliberat din membrana skin a sistemului în mediul înconjurător.

În cazul nostru, iniţial, în regiunea delimitată de membrana skin sunt introduse obiecte specificând poziţiile lui 1 din reprezentarea binară a lui x. Calculul începe şi regulile din $R(n)$ sunt aplicate. Se observă că prezenţa obiectului j ($j \in \{1, 2, ..., n\}$) va produce 2^{j-1} copii ale obiectului a. Deci, la sfârşitul calculului (când regulile din $R(n)$ nu se mai pot aplica), membrana skin va conţine x copii ale obiectului a, adică exact reprezentarea unară a lui x. Un exemplu de calcul va fi ilustrat în continuare, considerând, ca mai sus, pe 23 ca numărul a cărui reprezentare binară se doreşte.

Pentru $x = 23$, x va fi reprezentat de obiectele 5, 3, 2 şi 1 (copii unice).

Pas1: se aplică în paralel regulile $\begin{matrix} 1 \to a \\ 2 \to 1,1 \\ 3 \to 2,2 \\ 5 \to 4,4 \end{matrix}$ şi rezultă multisetul a,1,1,2,2,4,4.

Pas2: se aplică în paralel regulile $\begin{matrix} 1 \to a \\ 1 \to a \\ 2 \to 1,1 \\ 2 \to 1,1 \\ 4 \to 3,3 \\ 4 \to 3,3 \end{matrix}$ şi rezultă multisetul

$a,a,a,1,1,1,1,3,3,3,3 = a^3, 1^4, 3^4$.

Pas3: se aplică în paralel regulile $\begin{matrix} 1 \to a \\ 3 \to 2,2 \end{matrix}$ şi rezultă multisetul $a^3, a^4, 2^8 = a^7, 2^8$.

Pas4: se aplică în paralel regulile $2 \to 1,1$ şi rezultă multisetul $a^7, 1^{16}$.

Pas5: se aplică în paralel regula $1 \to a$ şi rezultă multisetul a^{23}.

5. CONCLUZII

Natura vie oferă idei fundamentale pentru noi modele de calcul. Lucrarea de faţă prezintă două strategii de codificare a informaţiei în stilul *calculului natural*. Direcţiile abordate se bazează scrierea informaţiei în ADN şi pe proiectarea unui cod ADN, dar şi pe mai noul model de calcul cu membrane celulare (P sistemele).

Paradox: asemănătoare şi totuşi atât de diferite! În ceea ce priveşte implementarea modelului de calcul cu membrane, nu este clar, la acest moment, dacă trebuie să ne orientăm spre biochimie – cum este cazul calculului bazat pe ADN, sau spre computerul electronic (în sensul celor create în scop general, sau a celor special creeate în acest sens) – cum este cazul reţelelor neurale şi a algoritmilor genetici.

În modelul de bază al calculului cu membrane, nu avem de-a face cu şiruri şi limbaje (aşa cum găsim în calculul bazat pe ADN), ci cu *multiseturi* de *obiecte atomice*. Iată unde intervin diferenţe între cele două abordări ale codificării informaţiei: nu sintaxa este cea care contează aici (vezi calculul bazat pe ADN - [1]), ci vectorii numerici care caracterizează multiplicitatea copiilor obiectelor ce aparţin unei anumite regiuni. Iată, deci, şi de ce apare un asemenea paradox : deşi sunt două abordări inspirate din natură, sunt totuşi atât de diferite! Atunci, teoria P-sistemelor

reprezintă o ramura a… ? *Teoria multiseturilor formale* sau *teoria limbajelor formale*, ca în cazul calculului bazat pe ADN?

Fundamental P sistemelor, aplicate codificării numerelor folosind multiseturi, nu este numai lucrul cu aceste multiseturi, ci şi *structura membranei*, împreună cu toate consecinţele acesteia (în special posibilitatea lucrului în *regiuni* separate, aranjate ierarhic, şi *comunicarea* dintre regiuni; este important să subliniem că membranele sunt *separatori* şi *canale de comunicaţie*) ducând la posibilitatea extinderii codificării numerelor la codificarea informatiei.

Calculul cu membrane provine din biologie, unde, împreună cu domeniul biochimiei, procesele sunt nedeterministe, rezultatul fiind doar aproximativ/adevărat din punct de vedere probabilistic. Până acum, în domeniul P sistemelor, a fost folosită doar matematica "pură", ceea ce este de altfel adevărat şi în domeniul calculului ADN-ului. Dar ce putem spune despre abordările matematice "aproximative", folosind probabilităţi? Ce putem spune despre calculul "aproximativ", orice ar însemna aceasta?

Toate acestea pot fi reformulate în termeni de *soft computing* care, se pare, aparţin *calculului molecular* (calcul bazat pe ADN şi calcul cu membrane) [2].

BIBLIOGRAFIE

1. Gh. Paun, G. Rozenberg, A. Salomaa, *DNA Computing. New Computing Paradigms*, Ed. Springer-Verlag Berlin Heidelberg, 1998.
2. Gh. Paun, *Computing with membranes (P systems): Twenty six research topics,*
3. C. S. Calude, M.J. Dinneen, Gh. Paun, *Pre-proceedings of the workshop on multiset processing (WMP-CdeA 2000)*, pages 203-217, CDMTCS-140, August 2000.
4. C. Calude, Gh. Paun, *Computing with cells and atoms*, Taylor and Francis, London, 2000 (Chapter 3: *Computing with membranes)*.
5. N. Jonoska, Gh. Paun, G. Rozenberg, *Aspects of molecular computing, A festschript in honor of the 70th birthday of Tom Head*, Ed. Springer-Verlag Berlin Heidelberg, 2003. (Chapter 2: M. Arita, *Writing information into DNA*)
6. N. Jonoska, Gh. Paun, G. Rozenberg, *Aspects of molecular computing, A festschript in honor of the 70th birthday of Tom Head*, Ed. Springer-Verlag Berlin Heidelberg, 2003 (Chapter 12: A. Gehani, T. LaBean, J. Reif, *DNA-based cryptography*)
7. G. Ciobanu, Gh. Paun, M. J. Perez-Jimenez, *Applications of membrane computing*, 2005, format elecronic.(Chapter 1: Gh. Paun, *Introduction to membrane computing*)
8. M. A. Gutierrez-Naranjo, A. Leporati, C. Zandron, *Converting integer numbers from binary to unary notation with P systems*, disponibil pe http://psystems.disco.unimib.it
9. A. Alhazov, C. Bonchis, G. Ciobanu, C. Izbasa, *Encodings and Arithmetic Operations in P systems*,disponibil pe http://psystems.disco.unimib.it
10. G. Michaels, *Cryptography and linguistic of macromolecules, Studying the codes and language of life*, EMSL Lecture notes, April 2004.
11. A. Gehani, T. LaBean, J. Reif, *DNA-based cryptography*, 5th annual DIMACS meeting on DNA based computers (DNA 5), MIT, Cambridge, MA, June 1999.

12. Aiden A. Bruen, Mario A. Forcinito, *Cryptography, Information Theory, and Error-Correction: a handbook for the 21st century*, Wiley-Interscience, John Wiley&Sons, Inc., Hoboken, New Jersey.
13. A. Atanasiu, *Criptografie*, *Note de Curs*, Disponibil pe: http://www.galaxyng.com/adrian_atanasiu/cript.htm
14. A. Atanasiu, *Teoria codurilor corectoare de erori*, Ed. Universitaţii Bucureşti, 2001.
15. A. Atanasiu, *P - systems and Arithmetic Calculus*, Research Group on Mathematical Linguistics, Universitat Rovira i Virgili Tarragona (Spania) 14/00 (2000).
16. A. Atanasiu, *Arithmetic with membranes*, ROMJIST 4 (2001), nr. 1-2, pp. 5-20.
17. Atanasiu, C. Martin – Vide, *Recursive Calculus with membranes*, Fundamenta Informaticae 49 (2001), 1-15.
18. Dr. A. M. Israil, *Biologie Moleculară. Prezent şi Perspective*, Ed. Humanitas Bucureşti, 2000.
19. M.Arita, S.Kobayashi, *DNA Sequence Design Using Templates*, New Generation Comput. 20(3), 263277(2002). lucrare disponibilă pe http://www.ohmsha.co.jp/ngc/index.htm.
20. M. Li, H. J. Lee, A. E. Condon, R. M. Corn, *DNA Word Design Strategy for Creating Sets of Non-interacting Oligonucleotides for DNA Microarrays*, Langmuir 18(3), 805812(2002).
21. http://psystems.disco.unimib.it
22. [www.dcs.shef.ac.uk

Abstract: Researchers considered the DNA and the cell membranes as devices for the storage of genetic material. Recently, they implemented the storage of huge quantities of information in DNA and more recently, in some abstract devices called membranes systems, known as well as P systems – called after their inventor, prof. dr. Gh. Păun. This paper presents the necessary terminology of understanding the new concepts in section 2, called "general remarks". Those two different categories: of writing the information into DNA and of coding the information into the membrane structure using multisets of atomic objects, are presented in sections 3 and 4, respectively. In section 3 are presented the solid and the liquid faze of writing the information in the DNA sequences along with the constraints necessary for such processes. A totally different aproach is illustrated in the next chapter, because in the base model we don`t have to deal with sequences, as in DNA computing, but with multisets of atomic objects, as we have already explained. This is where the differences between two nature based aproches, of coding the information, actually differ. What really maters here, is not the sintax of the membrane model, who totally differ of the DNA based model, but the numerical vectors, that characterized the multiplicity of object copies belonging of certains regions. Some of those conlusions are presented in the last section of this paper.