

XXXII. OTDK Biológia Szekció

RNS-MOLEKULÁK TÉRSZERKEZETÉNEK GLOBÁLIS ÉS LOKÁLIS ÖSSZEHASONLÍTÁSA ATOMI TÁVOLSÁGELOSZLÁSOKON ALAPULÓ MÓDSZEREKKEL

OTDK DOLGOZAT

Fodor Mátyás



Konzulens: Gáspári Zoltán

Molekuláris Bionika BSc III. évfolyam

Pázmány Péter Katolikus Egyetem

Információs Technológiai és Bionikai Kar

2013.

Tartalomjegyzék

Tartalomjegyzék	1
1. Bevezetés.....	3
2. Irodalmi áttekintés	4
2.1.Az RNS.....	4
2.2.Az RNS szerkezete	5
2.3.Az RNS típusai.....	8
2.4.Az RNS evolúciója.....	9
3. Célkitűzések	10
4. Módszerek és metódusok	11
4.1.A SARA program.....	11
4.2.A PRIDE program	12
4.3.A CD-hit program	13
5. Eredmények és értékelésük	15
5.1.Az adatkészlet létrehozása.....	15
5.1.1. A keresés paraméterei	15
5.1.2. A cd-hit szűrés	15
5.2.A fájlok további szűrése, formázása szkript segítségével	16
5.2.1. A PDB fájlok szerkezete	16
5.2.2. A fájlok szűrése.....	17
5.2.3. A szűrő szkript működése	17
5.3.Az EDIRP implementációja	19
5.3.1. A hisztogramok kiszámítása	19
5.3.2. Az EDIRP pontok kiszámítása, a hisztogramok összevetése	19
5.3.3. Kontingencia analízis	20
5.3.4. Súlyfüggvény alkalmazása.....	22
5.4.Egyéb szkriptek megvalósítása:	23
5.4.2. Burkoló szkript a SARA programhoz	23

5.5.A hasonlóságkeresés eredménye	24
5.5.1. Az eredmények statisztikai értékelése	24
5.5.2. A legjobb EDIRP találatok	25
5.5.3. A SARA legjobb találatai az EDIRP eredményeivel összevetve	28
5.5.4. Az eredmények összegzése	28
5.6.Távlati lehetőségek	29
6. Összefoglalás	30
7. Irodalomjegyzék	31

1. BEVEZETÉS

Az RNS információhordozó molekulaként került be a köztudatba, ám több olyan felfedezés is született, amely az RNS egyéb, térszerkezetével kapcsolatba hozható szerepét igazolja. Ezek közül talán a legismertebb a riboszómát alkotó riboszómális RNS (rRNS) és a tRNS. A gének expressziója egy meglehetősen összetett, finomhangolt folyamat, melyben részt vesz például a mikroRNS és a small interfering RNS (siRNS). Ezek a nukleinsavak képesek egymással, a DNS-sel és fehérjékkel interakcióba lépni, emiatt szerepük pontos megismeréséhez nem szabad figyelmen kívül hagyni térszerkezetüket sem.

Dolgozatom célja egy olyan algoritmus fejlesztése, mely a bázissorrend figyelmen kívül hagyásával, pusztán a térszerkezetek adataiból számított távolságtartományokhoz tartozó szekvenciális távolságeloszlás segítségével képes összehasonlítani az RNS láncokat. Az algoritmus lehetőséget nyújt arra, hogy a későbbiekben osztályozásra is alkalmazható legyen. Amennyiben sikerül az eltérő bázissorrendű, ellenben hasonló térszerkezetű molekulákat azonosítani, korábban nem ismert funkcionális összefüggéseket fedezhetünk fel.

2. IRODALMI ÁTTEKINTÉS

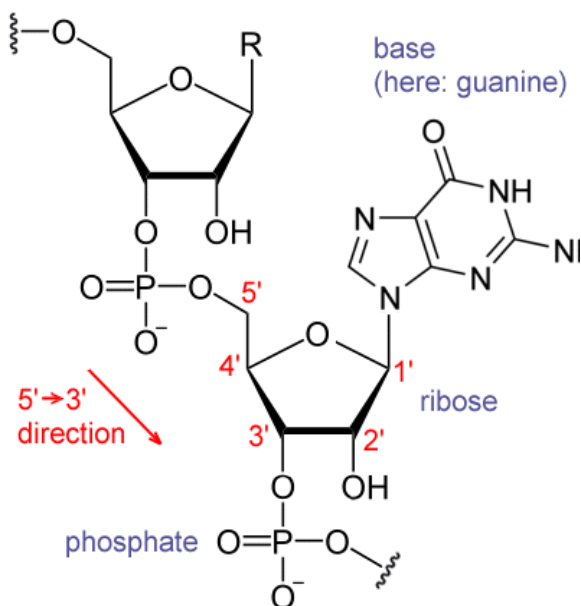
2.1. AZ RNS

Az RNS a biológiai makromolekulák családjába tartozik, a transzlációban, a transzkripcióban, a gének szabályozásában és expressziójában játszik fontos szerepet. Felépítését tekintve a DNS-hez hasonlóan nukleotidok láncából áll, de a DNS-től eltérően többnyire egyszálú. Egyes RNS molekulák a sejten belül aktív szerepet töltenek be a biológiai reakciók katalízisében, a génkifejeződés szabályozásában, a sejten belüli szignalizáció érzékelésében és továbbításban. Ezen folyamatok egyike a fehérjeszintézis, melynek során az mRNS molekula segítségével a riboszóma fehérjét szintetizál. Ebben a folyamatban tRNS szállítja az aminosavakat a riboszómához, ahol riboszómális RNS-ek (rRNS) kapcsolják össze azokat. [2,3,13]

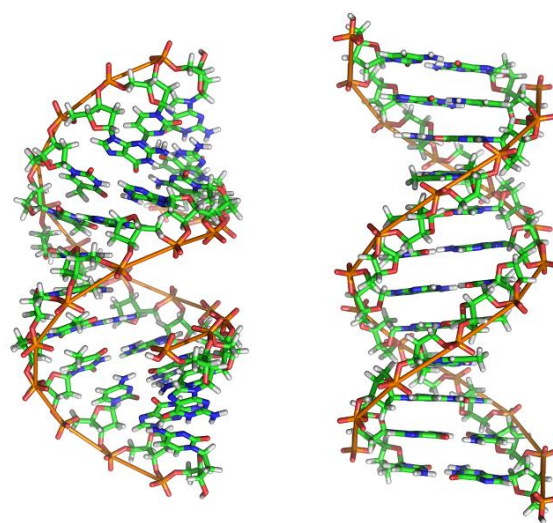
Az RNS kémiai szerkezete nagyon hasonlít a DNS-éhez, azonban több dologban különbözik tőle:

- A kétszálú DNS-sel ellentétben a sejten található RNS többnyire egyszálú, és sokkal rövidebb láncot alkot.
- Míg a DNS dezoxiribózt tartalmaz, az RNS-ben ribóz található (a dezoxiribóz pentóz gyűrűjének 2' szénatomjához nem csatlakozik OH csoport) (1. ábra). Ez a hidroxil csoport teszi az RNS-t a DNS-nél kevésbé stabillá, mivel így könnyebben tud hidrolizálni.
- Az RNS-ben az adenin bázispárja a DNS-sel ellentétben nem timin, hanem uracil, mely a timin metilálatlan formája. Továbbá a DNS-sel szemben az RNS-re jellemző, hogy módosított bázisokat tartalmaz.

A legtöbb biológiailag aktív RNS (mRNS, tRNS, rRNS, snRNS és egyéb nem kódoló RNS-ek) ún. palindrom (itt: önmagával bázispárosodásra képes) szekvenciákat tartalmaznak, melynek köszönhetően az egyszálú RNS-en belül kettős hélix alakulhat ki. Ezen nukleinsavak vizsgálata megmutatta, hogy meglehetősen strukturáltak. A DNS-sel ellentétben ezek a hélixek rövidek, és a fehérjék másodlagos szerkezetéhez hasonlóan a harmadlagos szerkezet vázát adják. Ennek a szerkezetnek köszönhetően az RNS képes kémiai folyamatokat katalizálni, akár csak az enzimek.



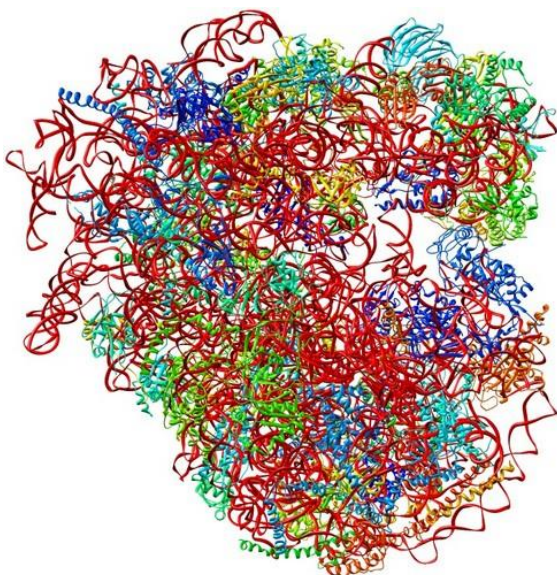
1. ábra. Az RNS lánc kémiai szerkezete. Az ábrán látható, hogy a ribóz molekula 2' szénatomján hidroxil csoport található, valamint a 3' és 5' szénatomokhoz kötődő foszfátcsoporttal csatlakozik a szomszédos nukleotidokhoz. Forrás: Wikipédia



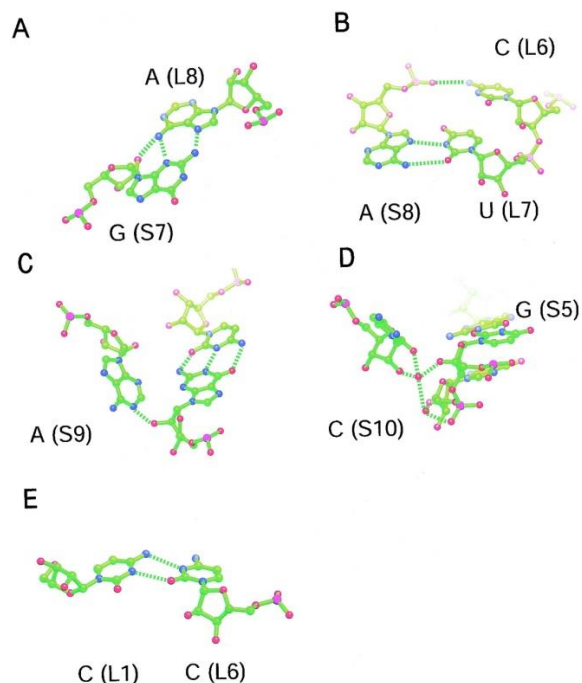
2. ábra. A baloldalon látható az A típusú , a jobb oldalon a B típusú hélix. Az RNS-re általában az előbbi, míg a DNS-re az utóbbi szerkezet kialakítása jellemző. Az A típusú elrendeződésre jellemző a széles, sekély kisírok valamint a mély és szűk nagyírok a B típussal összehasonlítva. Forrás: Wikipédia

2.2. AZ RNS SZERKEZETE

Az RNS polimerizálódott nukleotidokból áll. Ezek mindegyike tartalmaz egy ribóz gyűrűt, melynek szénatomjai 1'-től 5'-ig meg vannak számozva (1. ábra). Az 1' szénatomhoz kapcsolódik a nukleotid bázisa, mely adenin, citozin, guanin vagy uracil lehet. Az adenin és a guanin purin, míg az uracil és a citozin pirimidin alapú bázisok. A 3' szénatomhoz csatlakozik egy foszfátcsoport, melyen keresztül a nukleotid egy másik nukleotid ribóz gyűrűjének 5' szénatomjához kapcsolódik. A foszfátcsoportnak fiziológiás pH értékeken negatív töltése van, ezért a teljes RNS molekula negatívan töltött. A DNS és az RNS közötti fontos szerkezeti különbség a hidroxil csoport jelenléte a 2' szénatomon. Ennek köszönhetően a DNS-sel szemben az RNS általában „A” típusú hélixet alkot (2. ábra). A hidroxil csoportnak köszönhető az is, hogy az RNS flexibilis részein a csoport kölcsönhatásba lép a foszfodiészter kötással, ezzel a láncot elhasítva. Ennek köszönhetően az RNS a DNS-nél könnyebben elhasad.[7]



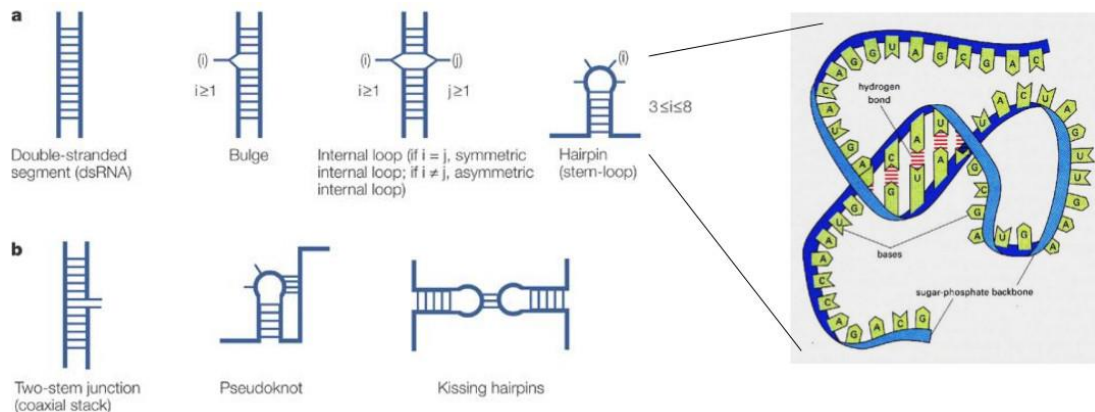
3. ábra A *Trypanosoma brucei* 80S riboszomális alegységének szerkezete szalagos reprezentációval ábrázolva (PDB kód: 3ZEX). Az ábrán látható, hogy a 8 RNS láncból álló alegység igen komplex harmadlagos szerkezettel rendelkezik.



4. ábra nem Watson-Crick bázispárok. A: egy adenin és egy citozin között kialakuló bázispárosodás; B: egy adenin, egy uracil és egy citozin között kialakuló hidrogénkötések; C: egy adenosin kapcsolódik egy guanin-citozin bázispárhoz; D: Egy citozin és egy guanin kapcsolódása víz molekula segítségével; E: egy citozin-citozin bázispár. Forrás: Wikipédia

Bizonyos bázisok egymással hidrogénkötést alakíthatnak ki: általában citozin és guanin, adenin és uracil valamint guanin és uracil között jön létre kötés. A bázispárosodáson kívül még sokféle kötés kialakulhat, gyakran több bázis is részt vesz az interakció kialakításában (4. ábra).

Az enzimatis funkcióval rendelkező egyszálú RNS a fehérjékéhez hasonló specifikus harmadlagos szerkezetet követel meg (3. ábra). A harmadlagos szerkezet vázát a másodlagos elemek alkotják, melyeket a molekulán belüli hidrogénkötések stabilizálnak. Ennek köszönhetően számos másodlagos szerkezeti elemet megkülönböztetünk, például: hajtűhurkot, „kidudorodást” (angolul bulge) és belső hurkot. [11] Mivel az RNS töltéssel rendelkezik, fémionok (pl.: Mg^{2+}) szükségesek a térszerkezet stabilizálásához.



5. ábra. Különböző RNS másodlagos szerkezetek. Balról jobbra haladva:

- a) Kétszálú RNS; „kidudorodás” — a kétszálú RNS egy szakaszán az egyik szál egy szakasza nem párosodik, hanem kifelé fordul; Belő hurok — $i=j$ esetén szimmetrikus, $i \neq j$ esetén aszimmetrikus; hajtú hurok — egy RNS szakasz két vége önmagának fordított sorrendű komplementere
- b) Közös tengely menti elrendeződés; Álcsozó — a hajtú hurok ívében kifelé forduló bázisok egy másik láncszakasszal párosodnak; Csókolózó hajtú hurkok: ez előzőhöz hasonló, azzal a különbséggel, hogy itt két hajtúhurok ívének bázisai párosodnak.

Az RNS gyakori típusai, azok szerepe és pár nevezetes RNS		
Rövidítés	Angol név	Szerep
mRNS	messenger RNA	Fehérje szekvenciát kódol
tRNS	transfer RNA	Aminosavak szállítását végzi
rRNS	ribosomal RNA	Enzimatisz folyamatokat lát el a riboszómában
snRNS	small nuclear RNA	A szplájszingban van szerepe
snoRNS	small nucleolar RNA	Néhány RNS kémiai módosulását irányítja
miRNS	microRNS	RNS alapú génszabályozás
siRNS	small interfering RNS	RNS alapú génszabályozás
lncRNS	long noncoding RNS	Génszabályozás RNS közreműködésével
7S RNS	SRP RNA	A sejt belső jelazonosításában van szerepe
TERC	telomerase RNS component	A telomeráz holoenzim része, templátot alkot a szintetizálódó telomerikus DNS számára
Rnase P	ribonuclease P	Egy ribozim, mely a tRNS érésben játszik szerepet

1. táblázat az RNS főbb funkciói. A nem kódoló RNS-ek közül mindnél fontos szerepet tölt be a harmadlagos szerkezet.

2.3. AZ RNS TÍPUSAI

A messenger RNS szállítja a genetikai információt a DNS-től a riboszóma felé, ahol a fehérjeszintézis végbemegy. Az mRNS kódoló szakasza határozza meg a képződő fehérje aminosav-sorrendjét. Az RNS molekulák nagy részéről (eukariótákban az átíródó RNS kb. 97%-a) nem szintetizálódik fehérje, ezeket nevezzük nem kódoló RNS-nek (non-coding, ncRNS) (1. táblázat).[1]

A nem kódoló RNS-ek egy része a DNS-ről szintetizálódik, a saját génjéről (RNS gének), de az mRNS intronok is ide tartozhatnak. Ezen nem kódoló RNS-ek közé tartozik a transzfer RNS (tRNS), és a riboszómális RNS (rRNS), melyek a fehérjeszintézis alappillérei. A génszabályozásban és az RNS-feldolgozásban is szerepet játszanak a nem kódoló RNS-ek. Bizonyos RNS molekulák képesek kémiai reakciókat katalizálni: például más RNS láncokat elvágni, vagy a peptid kötést kialakítását segíteni a riboszómában. Ezeket az enzimatikus tulajdonsággal rendelkező RNS-eket ribozimeknek nevezzük.

2.4. AZ RNS EVOLÚCIÓJA

Mára elfogadottnak tekinthető az az elmélet, amely szerint a DNS és fehérje alapú biokémiát egy ún. RNS világ előzte meg. Ekkor a biológiai információ tárolását, reprodukálását, a molekulák szintézisét kizárólag RNS szerkezetek hajtották végre. Később kialakult egy ribozim komplex, a riboszóma, mely képes volt aminosavak polimerizációjára a peptidkötések katalízise révén. Ez a korai „ős riboszóma” feltételezhetően még nem a genetikai információ alapján, csupán véletlenszerűen kapcsolta sorba az aminosavakat, így alakultak ki az első fehérjék. A szelekció eredményeként a jellemzően stabilabb szerkezettel rendelkező fehérjék töltötték be az enzimatikus szerepeket, valamint a korai riboszóma fejlődésével egy új funkciót nyert, RNS alapján szintetizálta a fehérjét. Mára a riboszóma egy összetett RNS – fehérje komplexszé vált. Feltételezhető, hogy evolúciója során akkor őrzött meg egy-egy motívumot, ha az nem gátolta funkciója betöltésében, ugyanakkor stabilabbá, vagy hatékonyabbá tette. K. Bokov és S. V. Steinberg cikkükben ismertettek egy RNS evolúció modellt.[1] Ennek alapja, hogy a riboszómáról egyesével választják le a különböző egybefüggő térbeli egységet alkotó hurkokat. A folyamat során figyelembe vették azt, hogy a riboszómának minden állapotban stabilnak kellett maradnia. Egy RNS láncdarabot akkor lehetett leválasztani a komplexumról, ha annak 5' és 3' vége térben közel volt egymáshoz. Ez azt jelenti, hogy az RNS szerkezet az evolúció során úgy változik, hogy az eredeti térszerkezetbe hurokként inzertálódik az új láncrészlet. Ezt a modellt sikerült alátámasztaniuk, 10 „réteget” sikerült leválasztaniuk a riboszómáról, és a megmaradt, feltételezett ősriboszóma tartalmazta a PTC-t, ez megerősíti a modell helyességét.[1]

Ha a riboszóma fejlődésére valóban igaz, hogy az inzerció során a molekula megtartja eredeti szerkezetét, és az új lánc egy hurokként csatlakozik, akkor érdemes megvizsgálni ezeket a molekulákat olyan módszerrel, amely a szekvenciális távolságot és a térbeli távolságot hasonlítja össze, mivel lokálisan illesztve a különböző RNS láncokat igazolhatjuk a rokonságot az egyes molekulák között.

3. CÉLKITŰZÉSEK

Célom egy olyan EDIRP nevű algoritmus fejlesztése volt, mely képes az RNS térszerkezeteket összehasonlítani a különböző távolságtartományokban található nukleinsavak szekvenciális távolságeloszlása alapján. A bemeneti PDB fájlokból egy olyan kimenetet kell adnia, melyben az egyes láncpárok térbeli távolságtartományokhoz tartozó szekvenciális távolságeloszlásán alapuló összehasonlításának pontszáma szerepel. Az algoritmussal szemben támasztott egyéb elvárások: legyen gyors, hatékony, kizárólag a térszerkezet (és ne a bázissorrend) alapján dolgozzon, legyen RNS-re optimalizált.

A programnak el kell végeznie a fájlok elő feldolgozását, transzformációját, illetve el kell végeznie a láncok összehasonlítását.

Az algoritmus működésének teszteléséhez szükség van egy adatkészlet létrehozására, mely megfelelő minőségű adatokból, és megfelelő hosszúságú láncokból áll, valamint minél kevésbé redundáns.

Az eredmények értékeléséhez szükség van egy eszközre, melynek segítségével megvizsgálhatom, hogy az adatok statisztikailag szignifikánsnak tekinthetők-e, illetve segít a program finomhangolásához szükséges paraméterek meghatározásában.

Az EDIRP eredményeit egy térszerkezet illesztő program, a SARA eredményeivel össze kell hasonlítanom, ehhez szükséges automatizálni azt. A legjobb eredményeket, valamint a két programban jelentős eltéréseket mutató szerkezeteket a CHIMERA program segítségével láthatóvá kell tennem.[15]

4. MÓDSZEREK ÉS METÓDUSOK

Az RNS-térszerkezetek összehasonlítására már léteztek korábban is eszközök, ezek közül a SARA programmal ismerkedtem meg részletesebben. Az EDIRP előzményének számít a PRIDE, mellyel korábbi munkám során már végrehajtottam RNS térszerkezet összehasonlításokat. Az algoritmus verifikációjához szükséges szűrt adatbázis létrehozásához a cd-hit programot használtam, mely rendkívül gyorsan képes osztályozást és – ennek melléktermékeként – szűrést végrehajtani.

4.1. A SARA PROGRAM

A SARA programot 2006-ban fejlesztették, az EDIRP-hez hasonlóan általános RNS összehasonlítás céljából.[12]

A SARA az RNS szerkezeteket összevetését egységvektorok összehasonlításával hajtja végre, melyeket a láncban egymást követő vázatomokból számol ki (az atomot a felhasználó választja ki, lehet foszfor vagy C3-as szénatom, alapértelmezetten az utóbbi). Egy egyszerűsítő reprezentációt használ az RNS láncok páronkénti összehasonlítására. Ez kulcsfontosságú a strukturálisan megegyező atomok gyors megtalálásához két merev testnek tekintett lánc között. Az algoritmus a következőképpen jár el:

1. Atom-nyomvonalat számol az egymást követő (C3 vagy P - előre meghatározott) vázatom típusból.
2. A nyomvonal alapján kiszámol k db vektort a szekvenciában egymástól k távolságra lévő atomok között, majd azokat normálja. „ k ” egy paraméter, ajánlott értéke: 7.
3. A k db egységvektort leképzi az egységgömbre.
4. Egy mindent mindennel összevető pontmátrixot számol a két szerkezet egységgömbjeiből; az egyes egységgömbön található egységvektorok eltérésének négyzetátlagának gyöke alapján (RMSD – root mean square deviation).
5. A pontmátrixból dinamikus programozás segítségével 0 end gap büntetést alkalmazva meghatároz egy illesztést a két szerkezet között.
6. A Max Sub algoritmus egy változatával maximalizálja az atomok számát $3.5 \text{ \AA}$ távolságon belül a két struktúra között.[16]

7. Az illesztés szignifikanciájának megítéléséhez P-értéket számol, valamint annak negatív logaritmusát.

A fehérjeszerkezet-illesztéssel szemben, ahol az illesztett részletek számának maximalizálása és az RMSD (Root-mean-square deviation) minimalizálása a fő cél, RNS szerkezetek esetén a másodlagos szerkezet kialakításában résztvevő bázispárok pontos illesztése is fontos szempont. Ezért az 1. és 6. lépés opcionálisan felhasználja a lehetséges másodlagos szerkezetekről elérhető információt a 3DNA program segítségével. Ezt az opciót választva a SARA csak azokat az atomokat veszi figyelembe, melyek csak a bázispárosodásban vesznek részt, és a bázispár atomok számát maximalizálja a 3.5 Å távolságon belül.

4.2. A PRIDE PROGRAM

A fehérjékben szekvenciálisan 3 és 30 aminosav távolságra lévő alfa szénatomok térbeli távolságának eloszlása meglehetősen jellemző a térszerkezetre, ezért jogosan merül fel az ötlet, hogy ennek alapján hasonlítsuk össze őket. Emellett szól az is, hogy az így képezhető távolságeloszlás-hisztogramokat könnyű összehasonlítani. A PRIDE program az összehasonlítást kontingencia analízissel hajtja végre, mely megadja a minták azonosságának valószínűségét (PRIDE pont). Ennek értéke 0 és 1 között van. Egymáshoz közel álló szerkezetek esetén a PRIDE pont erősen összefügg az alfa-szénatomok RMSD távolságával (az atomok közötti távolságok négyzetes átlagával). Azonban e módszer korrekt klasszifikációt nyújt egymástól távol eső struktúrák esetén is, amikor a szerkezetillesztésnek nincs jelentősége. A CATH adatbázis analízise azt mutatta, hogy a térszerkezetek 98,8%-a a megfelelő homológ szupercsaládba esik, ha a legmagasabb PRIDE pont szerint van osztályozva. A szerkezetek illesztése és a másodlagos szerkezet ismerete nem szükséges a PRIDE pont számításához, mely így megfelelő sebességű nagy adatbázisok vizsgálatához is.[6]

4.3. A CD-HIT PROGRAM

A CD-hit egy általános felhasználású fehérje- és nukleinsav-szekvenciaosztályozó program, működését tekintve egy mohó algoritmus (a lokális optimumok alapján keresi a globális optimumot). Alkalmazása széleskörű, hiszen nagyon gyorsan képes nagy adatkészletek osztályozására, melynek során az egyes osztályok reprezentatív szekvenciáiból egy nem redundáns adatkészlet jön létre. Az algoritmus lépései:

1. Első lépésben hossz szerinti csökkenő sorrendbe rendezi a szekvenciákat.
2. A leghosszabb lánc lesz az első klaszter reprezentatív szekvenciája.
3. Ezt egy iteratív eljárás követi, melynek során minden szekvenciát összehasonlítunk a reprezentatív szekvenciákkal, és ha nincs olyan reprezentatív szekvencia, amellyel összehasonlítva egy bizonyos küszöbérték feletti pontszámot kapunk, akkor a vizsgált lánc egy új reprezentatív szekvencia lesz.

A páronkénti összehasonlítás a legidőigényesebb lépése az osztályozásnak, ezért az összehasonlítások számának csökkentésére egy rövidszó-szűrőt (rövid szekvenciadarabokon alapuló heurisztikus szűrőt) alkalmaz. Ha két fehérje egy bizonyos szinten hasonlít, akkor megadható, hogy minimum hány dipeptidnek, tripeptidnek és pentapeptidnek kell megegyeznie a két szekvenciában. Ezért azokat a párokat, melyek ezt a feltételt nem teljesítik, nem kell összehasonlítani, ez pedig gyorsítja az adatbázis osztályozást.

Az algoritmus fejlesztői egy új szűrő eljárást alkottak, amely rövidebb szavak összehasonlításán (2 és 5 betű hosszúságú szavak) és indextáblán alapszik. A rövid szavak előnye, hogy könnyen indexelhetőek. Pentapeptidek esetén összesen 21^5 féle szó lehetséges (20 féle aminosav és X), az indextábla 4 millió bejegyzést tartalmaz. Így egy olyan indextábla, melyben minden bejegyzés egy egyed pentapeptidet tárol, elfér egy általános számítógép memóriájában. A fejlesztők módszerében minden bejegyzés tartalmazza azon reprezentatív szekvenciák számát, melyek tartalmazzák ezt a pentapeptidet.

Amikor egy új szekvenciát feldolgoz az algoritmus, a rövidszó-szűrő a következő lépéseken halad végig:

1. Meghatározza az új szekvencia pentapeptideinek számát és típusát
2. Átfésüli az indextábla azon bejegyzéseit, amelyekben ezek a pentapeptidek megtalálhatóak

3. Minden korábbi reprezentatív szekvenciára kiszámolja azokat a pentapeptideket, amelyek megegyeznek az új és a korábbi szekvenciában.
4. Csak akkor hajt végre páronkénti összehasonlítást, ha ez a szám nagyobb egy bizonyos küszöbértéknél.

Ezzel a szűréssel az algoritmus nagyon sok fölösleges lépést elkerül. Az indestábla feldolgozása sokkal gyorsabb, mint egy look-up tábla alkalmazása. A rövidszó-szűrő lehetővé teszi az adatbázis klasszifikációját viszonylag alacsony küszöbértékek esetén is. A pentapeptid és tetrapeptid szűrők 80 és 85% feletti küszöbértékek esetén érvényesek, míg a tripeptid és dipeptidszűrők 75 és 65% között.

A program bemeneti formátuma egy FASTA fájl, amely összefűzve tartalmazza az egyes szekvenciákat. Elsődleges kimenete egy, a reprezentatív szekvenciákat tartalmazó FASTA fájl. A másik kimenete a klasztereket tartalmazó fájl.

A CD-hit programot az adatkészlet szűrésére használtam, hogy a 70%-nál nagyobb hasonlóságot mutató szerkezetek közül csak az egyik maradjon benne az adatkészletben. A fent leírt algoritmust eredetileg fehérjék osztályozására alkották, de 2006 óta nukleinsav-szekvenciák összehasonlítására is képes.[4,5]

5. EREDMÉNYEK ÉS ÉRTÉKELÉSÜK

5.1. AZ ADATKÉSZLET LÉTREHOZÁSA

A megfelelő adatkészlet elengedhetetlenül fontos az eredmények értékelése szempontjából, mivel egy nem redundáns biológiai adatkészletből kapott eredmények eloszlása alapján következtetni tudunk a program működésének helyességére.

5.1.1. A keresés paraméterei

A PDB.org weboldal ADVANCED SEARCH lehetőségét választva a következő szűrőfeltételt állítottam be:

Macromolecule Type:	
Contains Protein	Ignore
Contains DNA	Ignore
Contains RNA	Yes
Contains DNA/RNA Hybrid	Ignore

Ennek a lekérdezésnek az eredményében azok a szerkezeti fájlok találhatóak, melyek tartalmaznak RNS-t, de mellett egyéb makromolekulákat is tartalmazhatnak. A különböző szerkezeti fájlok neveit 2013 augusztusában kérdeztem le, összesen 2697 eredményt adott. Bár a szerkezetek letölthetőek a webes felületről, célszerűbbnek tartottam ezt egy szkript segítségével végrehajtani.

5.1.2. A cd-hit szűrés

Adatkészletemből egy perl szkript segítségével létrehoztam a FASTA fájlokat, majd összefűztem őket. A cd-hit-et 3-as szóhosszal, 70%-os azonossági küszöbértékkel futtattam le a fájlon. A reprezentatív szekvenciákat tartalmazó kimeneti fájlból könnyen ki tudtam nyerni a láncok nevét.

5.2. A FÁJLOK TOVÁBBI SZŰRÉSE, FORMÁZÁSA SZKRIPT SEGÍTSÉGÉVEL

A PDB fájlok meglehetősen sok adatot tartalmaznak, melyek nagy részét nem használja fel egyik program sem, tehát ezek kiszűrése gyorsítja azok futását, illetve megkönnyíti a beolvasást. A bemeneti fájlt úgy terveztem meg, hogy az kompatibilis legyen a PRIDE programmal, ezzel egyszerűvé téve az eredmények összehasonlítását.

5.2.1. A PDB fájlok szerkezete

A PDB fájlok részletes dokumentációja a <http://pdb.org/> webcímen elérhető. Ebből számomra az “ATOM” jelzésű sorok voltak fontosak, hiszen ezek tartalmazzák az egyes atomok térkoordinátáit. Ezen sorok felépítése a 2. táblázatban láthatóak.

2.

Karakterek száma:	Típus:	Tartalma:
1 – 6	szöveg	“ATOM”
7 - 11	szám	atom sorszáma
13 - 16	atom	atom típusa
17	karakter	alternatív hely jelzése
18 - 20	szöveg	monomer típusa (nukleotid/aminosav)
22	karakter	lánc jelzése
23 - 26	szám	monomer sorszáma
27	karakter	inzerció jelzése
31 - 38	valós szám	X koordináta
39 - 46	valós szám	Y koordináta
47 - 54	valós szám	Z koordináta
55 - 60	valós szám	az adott hely elfoglalásának
61 - 66	valós szám	hőmérsékleti tényező
77 - 78	szöveg	atom vegyjele
79 - 80	szöveg	töltés

táblázat: A PDB fájlok atom sorainak tartalma

5.2.2. A fájlok szűrése

A PDB fájlok szűrését az általam írt FETCHV5.PY szkript hajtja végre, melynek nincs bemeneti argumentuma, automatikusan felismeri a mappában található „pdb.tar.gz” kiterjesztésű fájlokat. A kimeneti fájl formátuma egy összefűzött PDB struktúra, melyben minden lánc külön szerkezetként (külön HEADER és END címkék között) szerepel. A láncok megkülönböztetésére a [PDB fájl neve]_[Lánc jelzése] elnevezést használtam. Csak azokat a láncokat tartottam meg, melyek minden monomere „A”, „U”, „G” vagy „C” jelzésűek. Az ATOM szekcióban fel van sorolva a makromolekulát alkotó összes atom, programomban azonban minden monomert egyetlen atom koordinátaival jellemeztem, ezért csak a nukleotidhoz tartozó foszfátcsoport foszforatomjának elhelyezkedését vettem figyelembe. A szűrés során elhagytam a 30 bázisnál rövidebb és a 3 Ångströmnél rosszabb felbontású szerkezeteket.

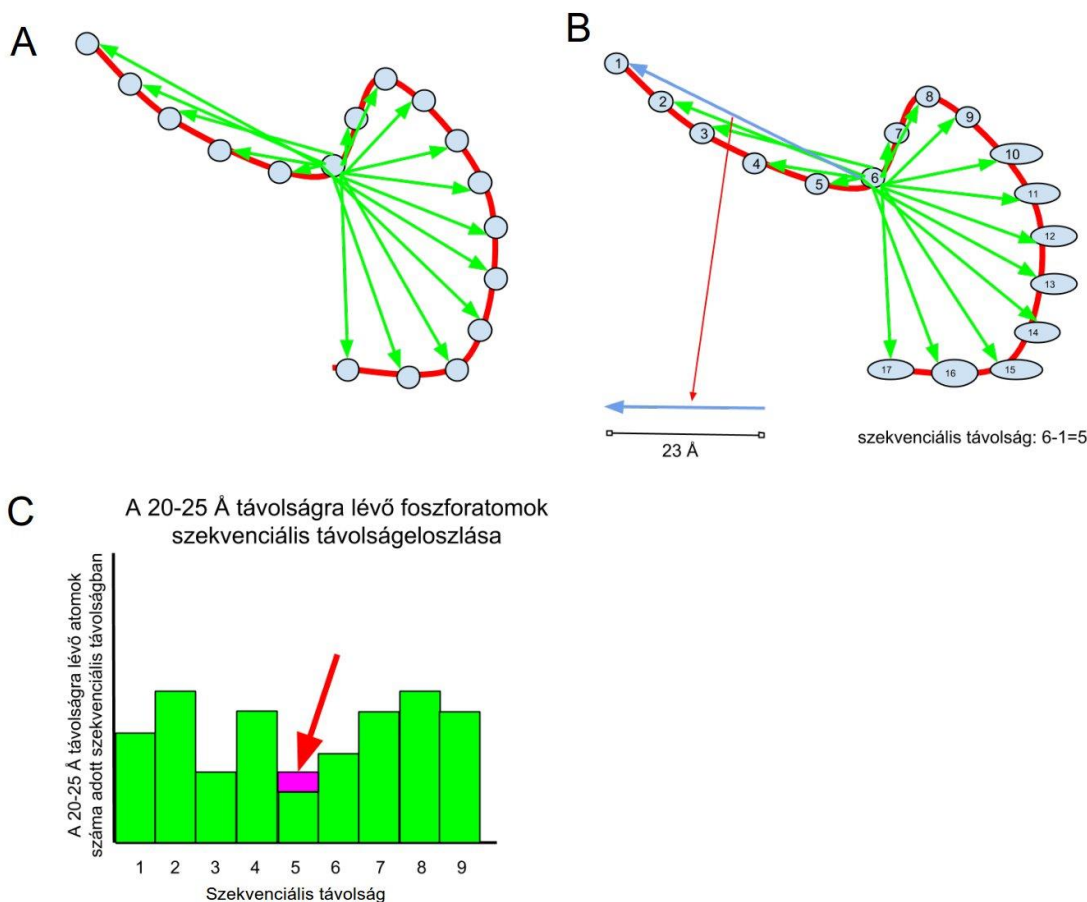
5.2.3. A szűrő szkript működése

A szkriptet az (egyenként) tömörítve letöltött pdb.tar.gz fájlok mappájába kell helyezni, és terminálból a “python fetchv5.py” paranccsal indítható. Összesen 6 kimeneti fájlt generál:

- gzfiles.txt: a “.tar.gz” fájlok listája, ez pusztán technikai megoldás, illetve az ellenőrzés szempontjából fontos
- errorlist.txt: azon fájlok listája, melyek felbontását nem tudta értelmezni a szkript. Ennek szintén ellenőrző funkciója van
- DiscardedList.txt: azon fájlok listája, melyek felbontása nem volt megfelelő
- distell.txt: ellenőrző fájl, azokat a sorokat tartalmazza, ahol több pozíció is meg van jelölve a fájlban.
- prideInput.txt: a kimeneti összefűzött PDB fájl
- dataset.txt: a kimeneti fájlban található szerkezetek listája [PDB azonosító]_[láncazonosító] névvel ellátva

A program először inicializálja a kimeneti fájlokat (törli tartalmukat, ha megtalálhatóak a mappában), mivel futás közben nincsenek folyamatosan megnyitva a fájlok, hanem fájlba íráskor hozzáfűzi az új sort. A mappában található “.tar.gz” kiterjesztésű fájlok neveit kimentí a “gzfiles.txt”-be.

A "gzfiles.txt"-n soronként végighaladva megnyitja a fájlnevvvel jelölt szerkezetet, és beolvassa annak sorait. A RESOLUTION attribútum alapján eldönti, hogy megfelelő-e a szerkezet felbontása. Amennyiben a felbontás megfelelő, elkezdi lánc szerint szétválogatni a foszforatomokat (egy hash táblába). A láncokból ezután kiválogatja az alternatív hellyel rendelkező atomokat, és az az alacsonyabb "occupancy" (betöltöttség) értékkel rendelkezőt elveti. Megvizsgálja, hogy a lánc RNS-e (csak A, U, G és C szerepelhet a sor 22. oszlopában) és hossza legalább 30 bázis. A szűrés után megmaradt láncokat hozzáfűzi a prideInput.txt fájlhoz.



6. ábra: Az EDIRP algoritmus lépései. a: minden atom távolságát kiszámítja a többi atomtól (az ábrán csak egy atom többi atomtól számított távolsága látható az átláthatóság érdekében); **B:** Meghatározza az egyes atompárokhoz tartozó szekvenciális távolságot, valamint eldönti, hogy melyik távolságtartományba esik az atompár; **C:** A megfelelő távolságtartományhoz tartozó hisztogram adott szekvenciális távolságértékhez tartozó értékét növeli.

5.3. AZ EDIRP IMPLEMENTÁCIÓJA

5.3.1. A hisztogramok kiszámítása

Az összefűzött PDB fájlból a STRUCTURELIST.PY szkript generálja a hisztogramokat tartalmazó pickle formátumú fájlkat.[14] Azért ezt a megoldást alkalmaztam szöveg alapú fájl generálása helyett, mert a pickle formátumú fájlok python kódból gyorsan beolvashatóak, és gyakorlatilag azonnal létrejönnek a benne reprezentált adatstruktúrák a memóriában. A szkriptnek egyetlen bemenete van, a prideInput.txt, kimenetét a [PDB azonosító]_[láncazonosító].p névvel ellátott, hisztogramokat tartalmazó pickle fájl, valamint a structurelist.txt alkotja. Az utóbbi csupán egy listát készít a létrehozott fájlokból, ellenőrzés céljából.

A szerkezetek adatai a HEADER és az END jelzésű sorok között találhatóak, a HEADER kezdetű sorban található a szerkezet neve. Először egy listát készít a program, melyben kizárólag az egyes nukleotidok foszforatomjának térkoordinátái szerepelnek szekvencia szerinti sorrendben, majd a szerkezet neve, és a lista alapján létrehoz egy structure típusú objektumot. Ennek a COMPUTEPRIDEHISTOGRAM() tagfüggvénye számolja ki a hisztogramot, és menti azt el pickle fájlba. Ennek során kiszámítom az összes foszforatom többi foszforatomtól mért térbeli távolságát, és távolságtartományonként hisztogramban tárolom az atomok szekvenciális távolságeloszlását.

Előzetes vizsgálatok során azt az eredményt kaptam, hogy a tömegközépponttól legfeljebb 160 Å távolságra találhatóak foszforatomok, ezért úgy döntöttem, hogy a 0-300 Å-ös távolságot 5 Å-ként osztom fel, így legfeljebb 60 hisztogramot készítve lánconként. Mivel az adatkészlet elemeinek mérete eltérő, a futásidő és fájl méret szempontjából előnyösnek vélt megoldásként a hisztogramokat egy hash táblában tároltam el. Elvetettem azokat a hisztogramokat, melyek 15-nél kevesebb elemet tartalmaztak. A hisztogramok is hash táblaként tárolódnak el, ezáltal nincs felső határa a feldolgozható szekvenciák hosszának, de a fölösleges nulla értékek nem tárolódnak el.

5.3.2. Az EDIRP pontok kiszámítása, a hisztogramok összevetése

Az elkészült hisztogramokat többféleképpen felhasználhatjuk, egy adatbázisként funkcionál. Futtathatunk rajta egyetlen szerkezet – összes szerkezet típusú összehasonlítást, és mindent – mindennel összehasonlító keresést is. Az algoritmus helyes működésének igazolására a legcélravezetőbb a kimerítő keresés megvalósítása volt, mivel így az algoritmust egyszer lefuttatva az eredményeket tartalmazó kimeneti fájlból könnyen kiszűrhetők a vizsgálandó adatok. Az

objektum orientált tervezésmódnak köszönhetően pár sor megváltoztatásával lehetőség nyílik más típusú összehasonlítás elvégzésére is.

5.3.3. Kontingencia analízis

Az előző szkript segítségével olyan hisztogramokat kaptunk, melyek RNS lánconként a bizonyos távolságtartományra lévő nukleinsavak szekvenciális távolságeloszlását reprezentálják.

Ebben a lépésben a hisztogramokat páronként összehasonlítjuk egy statisztikai módszer, a kontingencia analízis segítségével. Ez a módszer megadja annak a valószínűségét, hogy a két eloszlás között van-e összefüggés.

Első lépésként a hisztogramokat reprezentáló sorvektorokat egymás alá illeszttem, így egy mátrixot kapok. A következő elnevezéseket vezetem be:

$N_{(i,j)}$: az i . sor j . eleme, az i . (1. vagy 2.) hisztogram j . osztályának elemszáma

N_i : az i . sor összege, az i . (1. vagy 2.) hisztogram elemeinek összege

N_j : az j . sor összege, az i . (1. vagy 2.) hisztogram elemeinek összege

N : a mátrix elemeinek összege, a hisztogramok összes elemének száma

$n_{(i,j)}$: az i . sor j . elemének várható értéke

Az a nullhipotézisünk, hogy a két hisztogram független, az analízis során pedig megkapjuk annak a valószínűségét, hogy mekkora a valószínűsége annak, hogy összefüggőek.

$N_{(i,j)}$ várható értéke sor és oszlopösszegekből számolható a következő módon:

$$\frac{n_{(i,j)}}{N_j} = \frac{N_i}{N}, \text{ ebből következik, hogy } n_{(i,j)} = \frac{N_i * N_j}{N}$$

Ezek alapján látszik, hogy ha az egyik sor vagy oszlopösszeg 0, akkor az összes hozzá tartozó várható érték 0, így azok az oszlopok, melyek értéke mindig 0, egyszerűen eltávolíthatóak az analízisből. A következő lépésben a várható értékek alapján kiszámítjuk a két hisztogramhoz tartozó Khi-négyzet statisztikát, a következő képlettel:

$$\chi^2 = \sum_{i,j} \frac{(N_{(i,j)} - n_{(i,j)}^2)}{(n_{i,j})}$$

A Khi-négyzet értékének és a szabadságfokok ismeretében egy táblázatból meghatározható a két hisztogram összefüggőségének valószínűsége. A minták szabadságfoka megegyezik a minták (a mátrix értékeinek száma) és a megkötések számának összegével. Megkötés a szélsőértékek száma, amely eggyel kevesebb, mint az oszlopok és sorok számának összege. Ennek képlete esetünkben

$$n * 2 - (n + 2 - 1)$$

ahol n a hisztogram osztályainak száma.

A kontingencia analízist oly módon hajtottam végre, hogy a két hash táblában tárolt hisztogram kulcsértékeinek unióját vettem, és egy 2 dimenziós tömböt készítettem belőlük. Így elkerültem, hogy üres oszlopok szerepeljenek a kontingencia táblázatban. (Üres sorok nem szerepelhettek benne, mivel a 15-nél kevesebb elemet tartalmazó hisztogramok automatikusan elvetésre kerültek.) A marginális értékek és a várható érték kiszámítása után a scipy könyvtár `SCIPY.STATS.CHI2_CONTINGENCY` függvényével számítottam ki a hisztogramok összefüggőségének valószínűségét. A kontingencia analízisnél fontos, hogy lehetőleg ne szerepeljenek benne alacsony értékek, és a várható értékek között nem szerepelhet 0, mivel ekkor a Khi-négyzet számításakor nullával kellene osztanunk. A beépített függvény 0 várható érték esetén hibát jelez. Ezt a hibát úgy kezeltem le függvényemben, hogy ilyenkor 0-nak vettem az adott hisztogram pár összefüggésének valószínűségét.[10]

Az egyes hisztogram párokhoz tartozó valószínűség-értékeket átlagoltam, majd beszoroztam egy büntető együtthatóval, amelyet úgy kaptam, hogy a hisztogramokhoz tartozó kulcsérték halmazok metszetének és a nagyobbik kulcsérték halmaz számosságát elosztottam egymással.

5.3.4. Súlyfüggvény alkalmazása

A számtani átlaggal számított EDIRP pontok nem feleltek meg az elvárásoknak, a szerkezetek többsége 0 vagy 1 pontszámot kapott. Ekkor merült fel az ötlet, hogy érdemes lenne megvizsgálni a különböző távolságtartományban található hisztogramok statisztikai minőségét. Ezt úgy valósítottam meg, hogy ismét lefuttattam a mindent – mindennel összehasonlítást, azonban ezúttal nem átlagoltam a távolságtartomány pontszámait, hanem minden hisztogramhoz tartozó pontszámot fájlba mentettem, majd ezek Shannon-féle entrópiáját kiszámítottam. Ezt az adatsort vettem alapul a súlyozás elkészítéséhez, melynek kritériuma az volt, hogy összege 1 lehet. Az így kapott számsort beillesztettem az EDIRP-be és a továbbiakban a különböző távolság tartományokhoz tartozó pontszámokat ezzel a súlyozással összegeztem. Az így kapott eredmények statisztikai vizsgálata megfelelt az elvárásoknak.

5.4. Egyéb szkriptek megvalósítása:

Munkám során több rövidebb szkriptet írtam, melyek többnyire az adatok rendezését, illetve vizualizációját segítették.

5.4.1. A Súlyfüggvény kiszámítása

Amint fentebb említettem, az algoritmus első verziójában a szerkezetek nagy része (több mint 95%) 0-ás és 1-es pontszámot kapott, ekkor újrafuttattam az algoritmust, és a különböző távolságtartományokhoz tartozó valószínűségek pontszámait nem átlagoltam, hanem fájlba mentettem. A későbbiekben ezt a fájlt a CONVERTER.PY szkripttel dolgoztam fel. A program a fájl megnyitása után a távolságtartományokhoz tartozó pontszámokat hisztogramokba rendezi, melyekben 100 osztály van a 0-1 tartományon.

A Shannon-féle entrópia az információelméletben használt mérőszám, mely egy üzenet információtartalmának várható értékét adja meg. Úgy számítjuk, mint a valószínűségi változó átlagos megjósolhatatlanságának mértékét, amely megegyezik az információtartalom várhatóértékével. [8]

A hisztogramokat normáltam, majd Shannon-féle entrópiát számoltam belőlük a következő képlettel:

$$H(X) = \sum_i P(x_i) * \log_b(P(x_i))$$

A kapott eredményeket pedig egy listában helyeztem el, és bemásoltam az összehasonlító szkriptbe, melyben súlyfüggvényként funkcionál.

5.4.2. Burkoló szkript a SARA programhoz

A SARA alkotói ingyenesen hozzáférhetővé tették programjukat, annak átírása meglehetősen bonyolult lett volna, ezért írtam egy Python szkriptet, amely egy ciklusban parancssorból hívja meg a SARA-t a különböző szerkezetek összehasonlítására. A SARA kimenete minden illesztésre egy PDB fájl, mely tartalmazza az illesztett láncokat, illetve annak REMARK-kal jelzett soraiban megtalálhatók az illesztés pontszámai. Ezeket egy másik szkript segítségével exportáltam egy „.csv” fájlba, hogy rendezni és formázni tudjam az adatokat.

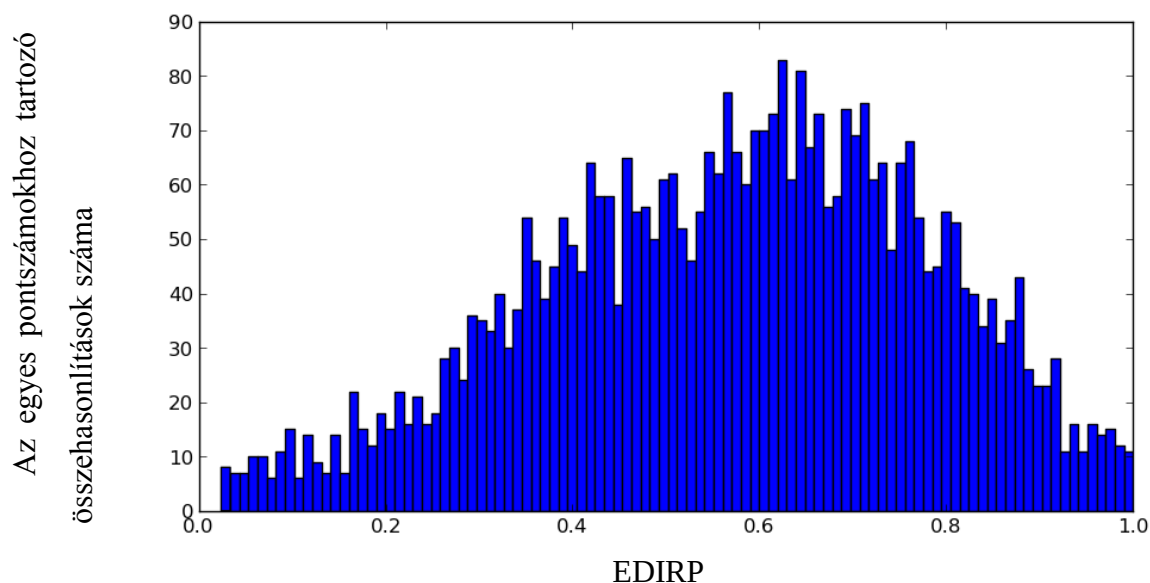
5.5. A HASONLÓSÁGKERESÉS EREDMÉNYE

Az EDIRP eredményeinek validációját úgy végeztem, hogy megvizsgáltam a nem redundáns adatkészlet pontszámainak eloszlását, illetve összevettem a SARA eredményeivel.

5.5.1. Az eredmények statisztikai értékelése

Az elv helytállóságának és az algoritmus helyességének feltétele az, hogy az eredmények eloszlása megfelelő legyen (7. ábra). Az eloszlásfüggvény maximuma kb. 0.6-nál van, körülbelül 10 láncpár tartozik a 0.99-1.0 tartományba, a legalacsonyabb pontszámmal rendelkező szerkezetekből is körülbelül 10 van. Megfigyelhető, hogy egyik szerkezet sem kapott 0.0-0.03 közötti pontszámot. Ennek háttérében valószínűleg az áll, hogy a legelső távolságtartományok hisztogramjai minden szerkezet esetén megegyeznek. Az adatkészletből ki vannak szűrve a szekvenciálisan 70%-nál jobban hasonlító láncok, mégis vannak szerkezetpárok, melyek magas pontszámot kaptak. Ezekről elmondható, hogy bázissorrendjük alapján nem számítanak hasonlóknak, de térszerkezetük az EDIRP alapján hasonló, érdemes őket további vizsgálat alá vetni.

[9]



7. ábra: A cd-hit szűrt EDIRP pontok eloszlása. A vízszintes tengelyen az eredmények értékkészlete, a 0-1 tartomány látható, 100 rekeszre osztva, a függőleges tengelyen pedig az egyes rekeszekbe tartozó EDIRP pontok száma.

5.5.2. A legjobb EDIRP találatok

A **Hiba! A hivatkozási forrás nem található.**ban láthatóak a legmagasabb EDIRP pontszámmal rendelkező láncpárok, és a hozzájuk tartozó részletes SARA értékek. 9-ből hat esetben mind az EDIRP, mind a SARA értékek magasak. A SARA által készített illesztések képei a következő oldalon találhatóak. Megfigyelhető, hogy mindegyik esetben látszólag hasonló szerkezetek kaptak magas pontszámot.

Összesen 11 lánc alkotja a legjobb 9 találatot, mivel egyes láncok (a 3KNH W ötször, az 1UOB A háromszor, a 3ZJV B kétszer, a 3KNH kétszer) többször is előfordulnak a listában, egymással kapcsolatba állítva. Ezeket a láncokat a SARA is hasonlóan találta egy kivétellel, az 1UOB A láncának és a 3ZJV B láncának az illesztésére alacsony, 2.486-os log P értéket számolt, a szerkezetillesztés volt sikertelen.

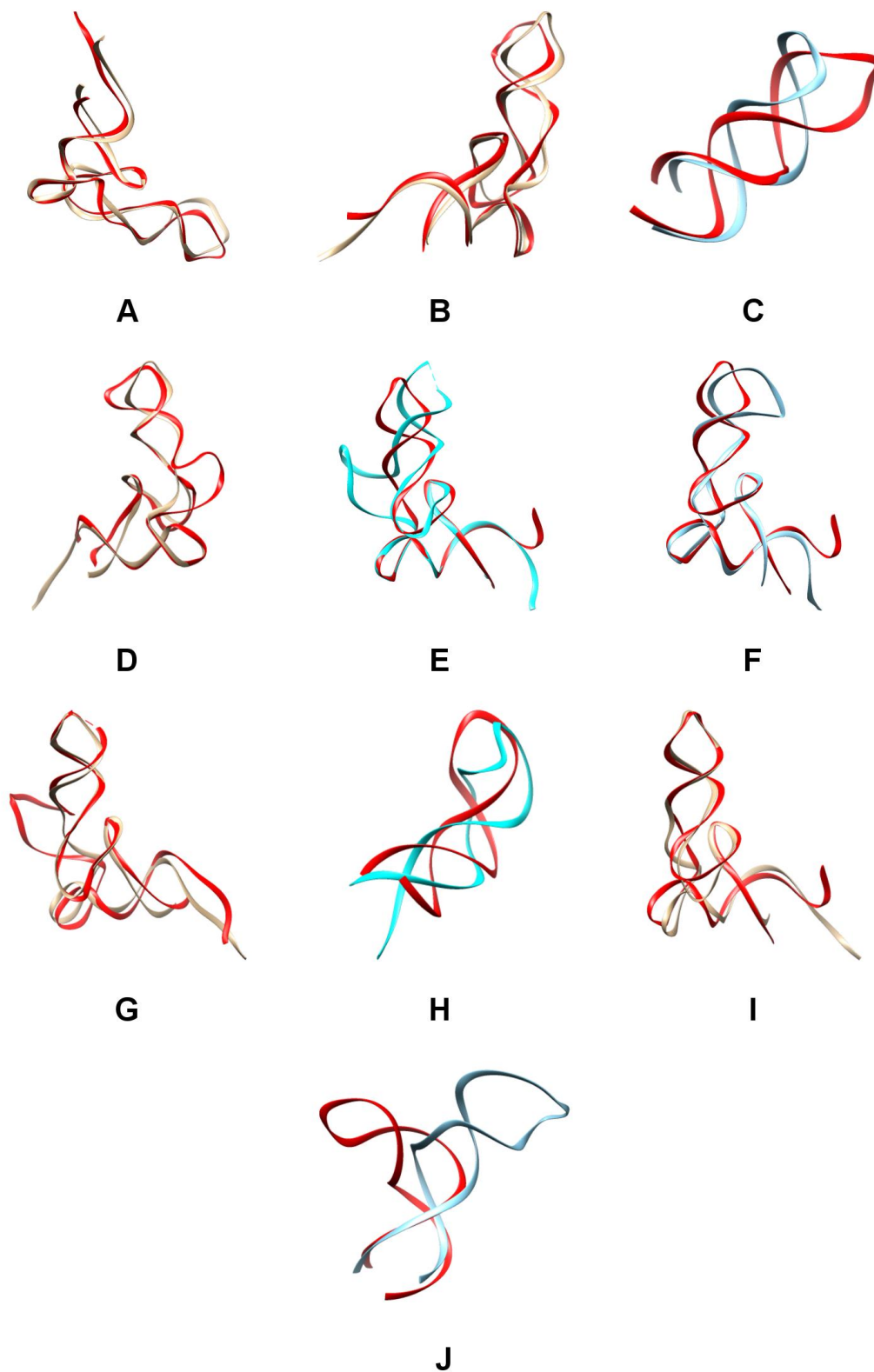
A legjobb találatok között a legalacsonyabb SARA pontszámot elérő 2LHP A lánc és az 1T4L A lánc közötti illesztés az ábrán látható módon tényleg nem volt sikeres. Ennek a két szerkezetnek az egymás mellé illesztett szalagmodellje **Hiba! A hivatkozási forrás nem található.**n látható. Továbbá a szintén rövid 2FDT A láncának és a 2LWK A láncának illesztése is alacsony pontszámot kapott. Az utóbbi két szerkezetpár esetében nehéz hasonlóságot megállapítani, hiszen mind a kettő egy rövid hajtússzerkezet, azaz "stem loop". A **Hiba! A hivatkozási forrás nem található.**ban látható, hogy a két lánc szekvenciáját sem sikerült illesztenie a SARA-nak.

A találatok valóban hasonlóak, azonban többnyire tRNS-ek, illetve rövid hurkok. Ezáltal az algoritmus működését igazolják, hiszen nagyfokú szekvenciális eltérés esetén is megtalálja a hasonló szerkezeteket, azonban nem jelent biológiai szempontból új eredményt.

Compared structures				SARA				EDIRP	RMSD	ID
PDB azonosító 1	Lánc 1	PDB azonosító 2	Lánc 2	Sequence Identity:	Structure Identity:	Sequence Identity Log(P- value):	Structure identity Log(P- value):	Average Log(P- value):	score	
3AMT	B	3KNH	W	45.30%	84.00%	8.08	6.94	6.72	0.999	1,75 Å A
3KNH	W	2ZUE	B	46.70%	77.30%	8.34	6.17	6.36	0.999	1,81 Å B
2FDT	A	2LWK	A	34.40%	87.50%	2.27	2.16	2.23	0.996	1,94 Å C
3KNH	W	2ZZM	B	52.00%	78.70%	9.39	6.32	6.85	0.994	1,46 Å D
1U0B	A	3ZJV	B	27.00%	45.90%	4.41	2.50	2.49	0.992	2,33 Å E
1U0B	A	3KFU	K	41.90%	77.00%	7.29	6.02	5.67	0.992	2,13 Å F
3KNH	W	3ZJV	B	42.70%	86.70%	7.55	7.25	6.76	0.991	2,16 Å G
2LHP	A	1T4L	A	28.10%	56.20%	1.83	0.88	1.26	0.99	1,57 Å H
3KNH	W	1U0B	A	40.50%	85.10%	7.02	6.95	6.58	0.989	2,32 Å I

Compared structures				SARA				EDIRP	RMSD:	
PDB azonosító 1	Lánc 1	PDB azonosító 2	Lánc 2	Sequence Identity:	Structure Identity:	Sequence Identity Log(P- value):	Structure identity Log(P- value):	Average Log(P-value):	score	
3ZJV	B	2BTE	B	62.80%	92.30%	12.05	8.32	8.72	0.98	1,93 Å
3AMT	B	2ZUE	B	61.80%	88.20%	11.50	7.55	7.58	0.98	2,14 Å
2ZZM	B	2BTE	B	50.00%	82.10%	9.42	7.08	7.11	0.71	1,87 Å
2ZZM	B	2ZZN	C	57.70%	83.10%	9.88	6.35	7.02	0.98	1,60 Å
3KNH	W	2ZZM	B	52.00%	78.70%	9.39	6.32	6.85	0.99	1,46 Å
3KNH	W	3ZJV	B	42.70%	86.70%	7.55	7.25	6.76	0.99	2,16 Å
3AMT	B	3KNH	W	45.30%	84.00%	8.08	6.94	6.72	1.00	1,75 Å
3ZJV	B	2ZZM	B	36.90%	77.40%	7.33	7.19	6.46	0.97	1,81 Å
3AM1	B	3ADB	C	48.10%	72.80%	9.44	6.28	6.19	0.66	2,18 Å
3KNH	W	3KFU	K	36.50%	79.70%	6.24	6.33	5.52	0.98	1,93 Å

3. táblázat: A, A legmagasabb EDIRP pontszámmal rendelkező szerkezetek SARA pontjai, EDIRP pontjai, szekvencia azonossága, és RMS távolsága. A táblázat utolsó sora tartalmazza a 8 ábrán található azonosítót. B, A legmagasabb SARA pontszámot kapott láncpárok, és azok EDIRP pontja, szekvencia azonossága valamint RMS távolsága.

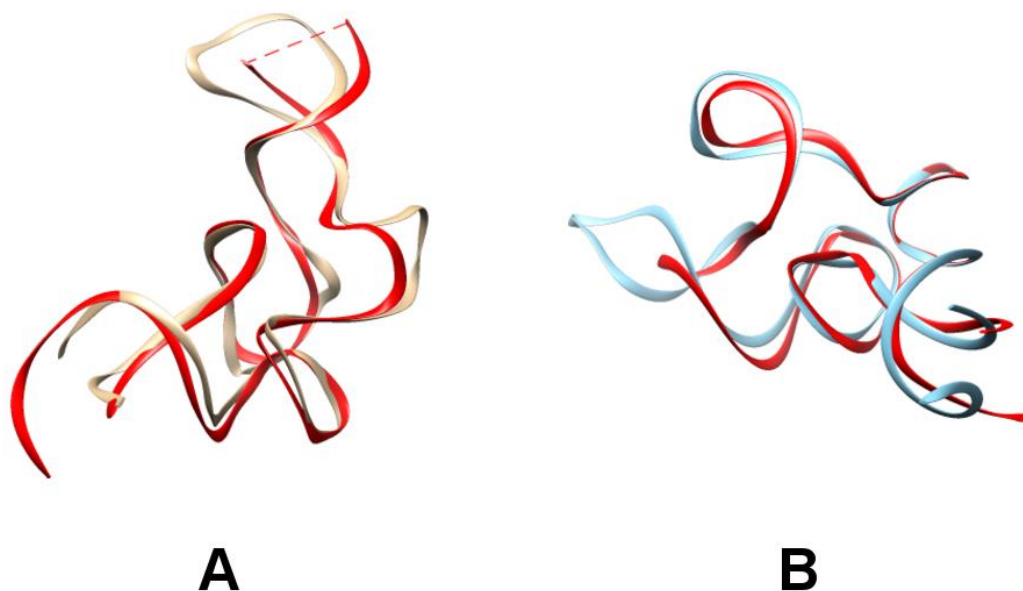


8. ábra: A legmagasabb EDIRP pontszámmal rendelkező láncpárok illesztése

5.5.3. A SARA legjobb találatai az EDIRP eredményeivel összevetve

A 3. táblázatban található az automatizált SARA-val végzett összehasonlítások legjobb 10 eleme, és azok EDIRP pontszámai. A 10 illesztésből 8-nál az EDIRP pontszám is magas.

Az egyik alacsony pontszámú összehasonlítás a 2ZZM B láncá, illetve a 2BTE B láncá között történt, a 2BTE láncában van egy szakadás, valószínűleg ez okozza az alacsony EDIRP pontot. A 3AM1 B láncának és a 3ADB C láncának vannak hasonló régiói, azonban ahogy a **Hiba! A hivatkozási forrás nem található.** mutatja, jelentős szerkezeti eltérések is találhatóak köztük.



6. ábra: Magas SARA, ellenben alacsony EDIRP pontszámmal rendelkező illesztések: A, A 2ZZM B láncának és a 2BTE B láncának illesztése; B, A 3AM1 B láncának és a 3ADB C láncának illesztése

5.5.4. Az eredmények összegzése

Az EDIRP pontszámok eloszlása, és a SARA pontjaival történő összehasonlítás alapján elmondható, hogy a program valóban képes hasonlóságot találni az RNS térszerkezetek között, érdemes alaposabb vizsgálatot végezni a helytállóságát és az alkalmazási területét illetően.

5.6. TÁVLATI LEHETŐSÉGEK

Az EDIRP már tartalmaz optimalizációkat (pl.: pickle fájlok használata, hash táblák használata), azonban még számtalan lehetőség van a futtatás gyorsítására, a kezelés konfigurálhatóvá tételére, az eredmények pontosítására is. Ezek a következők:

- Az EDIRP és a PRIDE közötti nagy különbség, hogy az EDIRP-ben hisztogramok adatait “elcsúsztatva” az x tengely mentén inzerciót, illetve deléciót tudunk vizsgálni. Erre a legjobb módszer a Needleman-Wunsch eljárás alkalmazása a hisztogramok összehasonlításakor. Ez a módszer alkalmazható homológ szerkezetek keresésére is.
- Extrém-érték eloszlást illeszteni a pontszámok eloszlására: a program statisztikai helytállóságát ezzel lehet a legjobban bizonyítani.
- Paraméterezhetővé lehet tenni azokat a távolság-tartományokat, amelyekhez a hisztogramokat hozzárendeljük, így a távolabbi tartományok (200 Å fölött) nagyobbak lennének, a 60 Å körüliek pedig kisebbek. Érdeemes megfontolni az átfedő távolságtartományok használatát.
- Parancssor paraméterek: Jelenleg konstans fájlnevekkel működnek a szkriptek, azonban így csak az tudja használni, aki ismeri a kódot. Ha a bemeneti paramétereket és a konfiguráció paramétereket parancssorból kapná meg a program, könnyedén beágyazhatóvá válna, és “felhasználóbarátabb” is lenne.
- Implementálás C++-ban: A jelenlegi verzió Python nyelven íródott, melynek előnye a gyors fejleszthetőség és a sok beépülő könyvtár, azonban sokkal lassabb, mint ha C++-ban íródott volna.
- RNS osztályozás: Az egyszerűsített, hisztogram alapú reprezentáció előnye, hogy kumulálva is tud adatokat tárolni. Ennek legegyszerűbb módja az adatok átlagolása, melynek hasznát osztályozás esetén látjuk: minden klasztert egy “átlaghisztogram” reprezentálna, így az algoritmus nagyon gyorsan lefutna, hiszen csak annyi összehasonlítást kell elvégezni, ahány osztályunk van

6. ÖSSZEFOGLALÁS

Munkám során megismerkedtem az RNS térszerkezetének jellemzőivel, enzimátikus jelentőségével, a különböző bioinformatikai eszközök használatával. Létrehoztam egy RNS adatkészletet a PDB adatbázisból, és ennek alapján egy szűrt RNS adatkészletet is a cd-hit 70%-os szekvencia-azonossági szűrője segítségével. Írtam egy szkriptet, melynek segítségével a PDB fájlokat átalakítom egy olyan konkatenált PDB fájlra, mely nukleotidonként csak a foszforatom koordinátáját tárolja el, illetve lánconként szétválasztja a szerkezeteket. Egy másik szkripttel létrehoztam a részeredményeket tároló fájlokat, melyekben a szerkezetekhez egy egyszerűsített reprezentáció tartozik. Ebben a reprezentációban az egymástól bizonyos távolságtartományban található nukleinsavak szekvenciális távolságának hisztogramjai szerepelnek. A teljes adatkészlet minden elemét összehasonlítottam minden elemével, és az így kapott adatokat statisztikailag megvizsgálva meghatároztam egy súlyfüggvényt, melynek következtében statisztikailag jobb eredményeket kaptam. Az EDIRP által meghatározott pontszámokat összevettem a SARA program eredményeivel, azokat ábrázoltam a CHIMERA programmal.

7. IRODALOMJEGYZÉK

- [1] K. Bokov and S. V. Steinberg, “A hierarchical model for evolution of 23S ribosomal RNA,” *Nature*, vol. 457, no. 7232, pp. 977–980, Feb. 2009.
- [2] J. M. Berg, J. L. Tymoczko, and L. Stryer, *Biochemistry*. New York: Freeman, 2003.
- [3] P. J. A. Cock, T. Antao, J. T. Chang, B. A. Chapman, C. J. Cox, A. Dalke, I. Friedberg, T. Hamelryck, F. Kauff, B. Wilczynski, and M. J. L. de Hoon, “Biopython: freely available Python tools for computational molecular biology and bioinformatics,” *Bioinformatics*, vol. 25, no. 11, pp. 1422–1423, Jun. 2009.
- [4] W. Li and A. Godzik, “Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences,” *Bioinformatics*, vol. 22, no. 13, pp. 1658–1659, Jul. 2006.
- [5] W. Li, L. Jaroszewski, and A. Godzik, “Clustering of highly homologous sequences to reduce the size of large protein databases,” *Bioinformatics*, vol. 17, no. 3, pp. 282–283, Mar. 2001.
- [6] K. V. Zoltán Gáspári, “Efficient recognition of folds in protein 3D structures by the improved PRIDE algorithm,” *Bioinformatics (Oxford, England)*, vol. 21, no. 15, pp. 3322–3, 2005.
- [7] I. Tinoco and C. Bustamante, “How RNA folds,” *Journal of Molecular Biology*, vol. 293, no. 2, pp. 271–281, Oct. 1999.
- [8] *Information Theory for Continuous Systems*. World Scientific, 1993.
- [9] J. D. Hunter, “Matplotlib: A 2D Graphics Environment,” *Computing in Science Engineering*, vol. 9, no. 3, pp. 90–95, 2007.
- [10] T. E. Oliphant, “Python for Scientific Computing,” *Computing in Science & Engineering*, vol. 9, no. 3, pp. 10–20, May 2007.
- [11] P. G. Higgs, “RNA secondary structure: physical and computational aspects,” *Q. Rev. Biophys.*, vol. 33, no. 3, pp. 199–253, Aug. 2000.
- [12] E. Capriotti and M. A. Marti-Renom, “RNA structure alignment by a unit-vector approach,” *Bioinformatics*, vol. 24, no. 16, pp. i112–118, Aug. 2008.
- [13] P. E. Bourne and H. Weissig, *Structural Bioinformatics*, Illustrated edition. Wiley-Liss, 2003.
- [14] „Pickle — Python object serialization,” <https://docs.python.org/2/library/pickle.html>
- [15] E. F. Pettersen, T. D. Goddard, C. C. Huang, G. S. Couch, D. M. Greenblatt, E. C. Meng, and T. E. Ferrin, “UCSF Chimera—a visualization system for exploratory research and analysis,” *J Comput Chem*, vol. 25, no. 13, pp. 1605–1612, Oct. 2004.
- [16] J. Bentley, “Programming Pearls: Algorithm Design Techniques,” *Commun. ACM*, vol. 27, no. 9, pp. 865–873, Sep. 1984.