Анализа одступања појаве сва 4 типа понављајућих секвенци у патогеним острвима бактерије Escherichia coli

Семинарски рад у оквиру курса
Истраживање података 2

Немања Ршумовић
Стефан Митровић

Ментор: Проф. Др. Ненад Митић

Математички факултет
Универзитет у Београду

мај 2024.

Садржај

1 Увод

Геномска анализа бактерија представља кључни корак у разумевању њихове патогености и механизама вируленције. У овом раду фокусираћемо се на бактерију Escherichia coli као једну од најистраживанијих бактерија због њене распрострањености и улоге у изазивању различитих инфекција. Посебну пажњу посветићемо патогеним острвима у геному ове бактерије, која садрже гене одговорне за њену способност изазивања болести.

У циљу прикупљања и обраде података користићемо софтвер StatRepeats и програмски језик Python. Овај рад ће се бавити идентификацијом и анализом четири типа понављајућих секвенци у патогеним острвима бактерије Escherichia coli. Прикупљени подаци биће анализирани и складиштени у бази података, што ће нам омогућити дубље разумевање њихове структуре и функције.

Истраживање се ослања на податке из PAIDB базе података. Ова база ће пружити детаљне информације о геномима бактерије и она је кључна за добијање података о патогеним острвима. Конкретно, за анализу понављајућих секвенци у патогеним острвима користићемо претходно поменути алат StatRepeats из пакета RepeatsPlus, који је дизајниран за овакву врсту генетичких анализа.

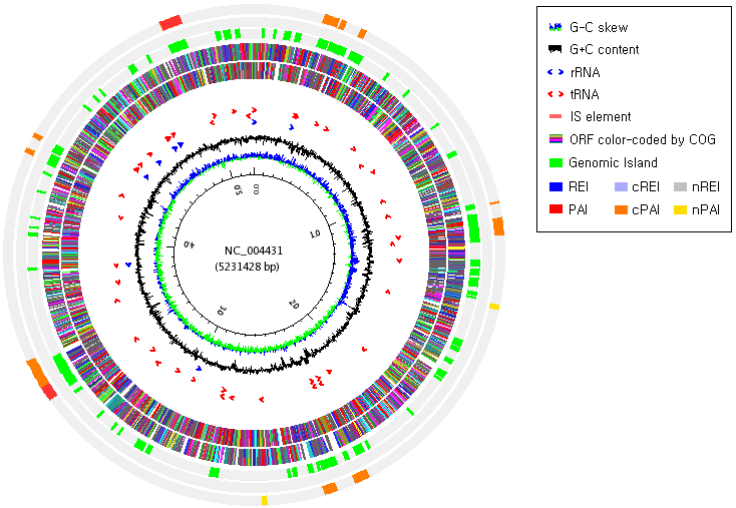
2 Патогена острва

Патогена острва (Pathogenicity Islands - PAI) су специфични сегменти генома бактерија који садрже гене одговорне за вируленцију [1]. Ови генетички елементи су обично присутни у патогеним, али не и у непатогеним сојевима исте бактеријске врсте. Истраживањем и детаљним проучавањем генетске основе E. coli откривени су кључни делово њеног генома одговорни за изазвање болести и они се класификују у три главне категорије: PAI, cPAI и nPAI.

PAI (Патогена острва): Потврђени делови генома бактерије E. coli који су повезани са патогеним својствима. У њима се налазе генске секвенце које кодирају вирулентне факторе, као што су токсини или адхезини, који омогућавају бактерији да изазове болест. Анализом PAI могу се идентификовати кључни гени и генски путеви одговорни за патогеност бактерије, омогућавајући развој циљаних терапија и превентивних стратегија.

cPAI (Кандидати за патогена острва): Делове генома за које се сумња да имају улогу у патогености, али још увек нису потврђени као такви. Они садрже гене који су слични онима у PAI, што их чини потенцијално вирулентним. Истраживачи посвећују посебну пажњу cPAI јер њихова анализа може открити непознате аспекте патогености бактерије.

nPAI (Мало вероватна патогена острва): Иако мање вероватна да су директно повезана са патогеношћу, nPAI још увек могу садржати гене који су слични онима у PAI или cPAI, али њихова улога у патогености још увек није јасна или потврђена.



Слика 1: Приказ генома Escherichia coli NC_004431 из базе података PAIDB са патогеним острвима (спољашњи круг)

[1] степен патогености или способност неког патогена да изазове инфекцију и болест

3 Прикупљање и обрада података

3.1 Прикупљање података

Скуп података обухвата 90 генома бактерије Escherichia coli који су преузети са националне базе података за биоинформатику (NCBI[1]) у FASTA формату. Ови геноми представљају богат извор информација за детаљну анализу и истраживање генетске структуре ове бактерије.

За сваки од ових 90 генома, коришћењем података из PAIDB[2] базе патогених острва, идентификована су и издвојена патогена острва (укључујући три главне категорије: PAI,cPAI и nPAI). Ова база је кључни ресурс за идентификацију и карактеризацију генетских елемената одговорних за патогеност бактерија.

За претходно наведене групе патогених острва креиране су табеле које укључују информације о почетним и завршним позицијама у геномској секвенци. Ови атрибути су есенцијални за разумевање како су патогена острва распоређена унутар генома и какву улогу играју у патогености бактерије. На пример, анализа може открити да одређена патогена острва имају сличне дужине и позиције у различитим сојевима, што може указивати на конзервиране регије [2] које су важне за вируленцију.

Јединствени идентификатори који се користе за означавање и референцирање специфичне геномске секвенце бактерије Escherichia coli у нашем скупу података су:

NC_011739, NC_017657, NC_009788, NC_018654, NC_018666, NC_009790, NC_013354, NC_018662, NC_013366, NC_009786, NC_017639, NC_013369, NC_018659, NC_017642, NC_007414, NC_017907, NC_002128, NC_013010, NC_011419, NC_017665, NC_017637, NC_017627, NC_007941, NC_017630, NC_011749, NC_013655, NC_010488, NC_011747, NC_017659, NC_017645, NC_009837, NC_013365, NC_009838, NC_012892, NC_012971, NC_012947, NC_012759, NC_017638, NC_012967, NC_017625, NC_000913, NC_009800, NC_007779, NC_010473, NC_011741, NC_013654, NC_010468, NC_017634, NC_011993, NC_020163, NC_022364, NC_011415, NC_017664, NC_017635, NC_008253, NC_011601, NC_009801, NC_022370, NC_017632, NC_017660, NC_011742, NC_017651, NC_017652, NC_007946, NC_010498, NC_008563, NC_017628, NC_017631, NC_011750, NC_017633, NC_011748, NC_017641, NC_011751, NC_011745, NC_004431, NC_017626, NC_018650, NC_017656, NC_018658, NC_018661, NC_017646, NC_013364, NC_017906, NC_013941, NC_013353, NC_002695, NC_013008, NC_002655, NC_011353, NC_013361

[2] очуване регије, делови генома који остају готово непромењени кроз еволуцију различитих сојева и врста

На примеру генома NC_004431 можемо видети распоред патогених острва са придодатим заглављем који садржи атрибуте ознака генома, почетак и крај патогеног острва:

- PAI острва

id	genome	start	end
1	NC_004431	3406225	3450866
2	NC_004431	4913367	4971660

Слика 2: NC_004431 PAI

- cPAI острва

id	genome	start	end
1	NC_004431	240329	288201
2	NC_004431	299622	310638
3	NC_004431	370224	378731
4	NC_004431	1127702	1135961
5	NC_004431	1183572	1241149
6	NC_004431	2216841	2264257
7	NC_004431	2331944	2343014
8	NC_004431	2349503	2367447
9	NC_004431	3406225	3552913
10	NC_004431	4274308	4290240
11	NC_004431	4329644	4344787
12	NC_004431	4910097	4971660

Слика 3: NC_004431 cPAI

- nPAI острва

id	genome	start	end
1	NC_004431	1476306	1485684
2	NC_004431	2571169	2585152

Слика 4: NC_004431 nPAI

FASTA фајл је текстуални формат који се користи за приказивање секвенци нуклеотида или протеина. Свака секвенца у FASTA фајлу почиње линијом која почиње са “>”, након чега следи назив секвенце. Секвенца нуклеотида или аминокиселина приказује се у наредним линијама. Овај формат је широко коришћен у биоинформатици због своје једноставности и читљивости. FASTA фајлови омогућавају лаку размену и анализу биолошких секвенци у истраживањима и апликацијама. Поред тога, они су компатибилни са бројним софтверским алатима за биоинформатичке анализе. Следећа слика представља преглед FASTA фајла генома NC_004431.

```
>NC_004431.1 Escherichia coli CFT073, complete sequence
AGCTTTTCATTCTGACTGCAACGGGCAATATGTCTCTGTGTGGATTAAAAAAAGAGTGTCTGATAGCAGC
TTCTGAACTGGTTACCTGCCGTGAGTAAATTAAAATTTTATTGACTTAGGTCACTAAATACTTTAACCAA
TATAGGCATAGCGCACAGACAGATAAAAATTACAGAGTACACAACATCCATGAAACGCATTAGCACCACC
ATTACCACCACCATCACCATTACCACAGGTAACGGTGCGGGCTGACGCGTACAGGAAACACAGAAAAAAG
CCCGCACCTGACAGTGCGGGCTTTTTTTTTGTGCACAGAAAACCCCCAGCTAGGCTGGGGGTTCCGGAAAG
CTTTCAGCTTTGAGCCAGTTATTAAAACCCCTTTTGATTTGTTAAAACACCTTGCGGTCTGGCAACTGCA
AGTGTCAAACAAGAAATCAAAAGGGGGTCCCAATGGGGAACGAAAAGAGCTTAGCGCACACCCGATGGAA
CTGTAAATATCACATAGTATTTGCGCCAAAATACCGAAGACAGGTGTTCTACAGAGAGAAGCGTAGAGCA
ATAGGCTGTATTTTGAGAAAGCTGTGTGAGTGGAAAAGTGTACGGATTCTGGAAGCTGAATGCTGTGCAG
ATCATATCCATATGCTTGTGGAGATCCCGCCCAAAATGAGCGTATCAGGCTTTATGGGATATCTGAAAGG
GAAAAGCAGTCTGATGCCTTACGAGCAGTTTGGTGATTTGAAATTCAAATACAGGAACAGGGAGTTCTGG
TGCAGAGGGTATTACGTCGATACGGTGGGTAAGAACACGGCGAAGATACAGGATTACATAAAGCACCAGC
TTGAAGAGGATAAAATGGGAGAGCAGTTATCGATTCCCTATCCGGGCAGCCCGTTTACGGGCCGTAAGTA
ACGAAGTTGGATGCAAATGTCAGATCGTGTGCGCCTGTTAGGGCGCGGCTGGTAAGAGAGCCTTATAGGC
```

Слика 5: NC_004431 FASTA фајл

3.2 Обрада података

Прикупљене податке је потребно обрадити за даљи наставак рада. Употребићемо софтверски алат StatRepeats и програмски језик Python[3] користећи алат Jupyer Notebook[4].

3.2.1 StatRepeats

StatRepeats[5] је напредни софтверски алат намењен за проналажење максималних понављајућих секвенци у улазним нуклеотидним или протеинским секвенцама. Развијен је од стране истраживачког тима на Математичком факултету Универзитета у Београду, са циљем да олакша анализу геномских података.

Софтвер користи статистичку процену за екстракцију само статистички значајних понављајућих секвенци или свих понављајућих секвенци, у зависности од потреба корисника. Филтрирање се може вршити на основу задате p-вредности, што омогућава да се у резултатима појављују само понављајуће секвенце које задовољавају одређени праг значајности.

StatRepeats је дизајниран да екстрахује четири типа понављајућих секвенци:

1. **Директне некомплементарне понављајуће секвенце**[3] - секвенце које се понављају у истом редоследу.
2. **Директне комплементарне понављајуће секвенце**[4] - секвенце које се понављају и њихови комплементи су у истом редоследу.
3. **Инверзне некомплементарне понављајуће секвенце**[5] - секвенце које се понављају у обрнутом редоследу.
4. **Инверзне комплементарне понављајуће секвенце**[6] - секвенце које се понављају у обрнутом редоследу са комплементарним базама.

Што се улазних аргумената тиче, основни су:

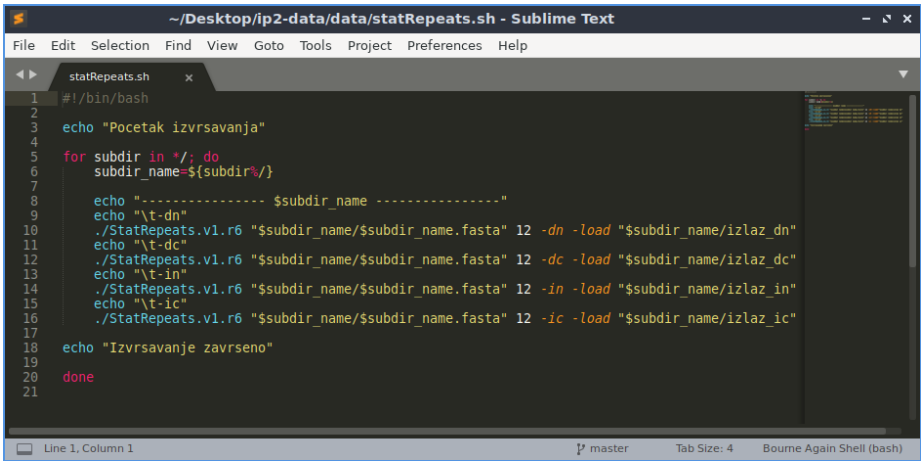
1. **Назив улазног фајла**: Име FASTA фајла који садржи секвенце за обраду. Фајл може садржати више FASTA секвенци које ће бити обрађене појединачно.
2. **Минимална дужина понављајуће секвенце**: Специфицира минималну дужину понављајућих секвенци које се екстрахују. Ако је наведена дужина прениска за коректно статистичко филтрирање, програм ће предложити минималну дужину. Ако је задато нула, StatRepeats ће аутоматски изабрати најмању могућу вредност за коју је процена вероватноће валидна.

Поред основних, овај алат пружа и неколико опционих аргумената, али навешћемо само оне које смо користили:

1. ***-loa[d] name*** - генеришу се три излазна фајла. Назив секвенце се записује у фајл *name.id*, статистика се записује у фајл *name.stat*, а резултати се записују у фајл *name.load*.
2. **Тип понављајућих секвенци**: Корисник може изабрати тип понављајућих секвенци које жели да претражује (-dn, -dc, -in, -ic). Ако ниједна опција није специфицирана, подразумевана опција је -dn (директне некомплементарне понављајуће секвенце).

[3] dn - direct non-complementary repeats
[4] dc - direct complementary repeats
[5] in - inverse non-complementary repeats
[6] ic - inverse complementary repeats

Зарад једноставније и брже обраде података написана је bash скрипта за покретања овог алата на целом скупу података (Слика 6). Као аргументи приликом покретања прослеђени су назив FASTA фајла, минимална дужина секвенце 12, сва 4 типа понављајућих секвенци и назив за генерисање излазних фајлова.



Слика 6: bash скрипта за покретање алата

3.2.2 Обрада података помоћу Python-а

Након што смо успешно идентификовали и екстраховали понављајуће секвенце користећи StatRepeats, следећи корак јесте анализа и даља обрада ових података користећи Python. Од интереса су нам фајлови са екстензијом *.load*, јер садрже информације о називу генома, почетак и крај понављајуће секвенце, почетак и крај њене одговарајуће понављајуће секвенце, њихову дужину, као и саме секвенце. Приказ једног *.load* фајла са придодатим заглављем можемо видети на наредној слици.

id	start_left	end_left	start_right	end_right	len	niska	niska2
NC_004431.1	1196109	1196122	3478619	3478632	14	AAAAAAACAAAGAC	TTTTTTTGTTTCTG
NC_004431.1	5172493	5172506	5181257	5181270	14	AAAAAAACAAATTA	TTTTTTTGTTTAAT
NC_004431.1	1279069	1279082	4830923	4830936	14	AAAAAAACGCTGCG	TTTTTTTGCGACGC
NC_004431.1	3906991	3907004	4829853	4829866	14	AAAAAAACGCTGCT	TTTTTTTGCGACGA
NC_004431.1	39241	39256	1248442	1248457	16	AAAAAAAGCCCTCTTC	TTTTTTTCGGGAGAAG
NC_004431.1	195338	195353	2879910	2879925	16	AAAAAAAGGGGGAAAA	TTTTTTTCCCCCTTTT
NC_004431.1	435891	435904	628847	628860	14	AAAAAAAGTAATCA	TTTTTTTCATTAGT
NC_004431.1	435891	435904	1372476	1372489	14	AAAAAAAGTAATCA	TTTTTTTCATTAGT
NC_004431.1	1806376	1806389	2395842	2395855	14	AAAAAAAGTAATTT	TTTTTTTCATTAAA

Слика 7: izlaz_dc.load генома NC_004431

За сваки геном из скупа података имаћемо четири Python скрипте у зависности од понављајућих секвенци за које су намењене. У наредним примерима показаћемо за директне комплементарне понављајуће секвенце генома NC_004431.

За почетак, неопходно је учитати сва три типа патогених острва чиме добијамо информације о њиховим почецима и крајевима (ако она постоје за дати геном), а потом и одговарајући *.load* фајл.

```
[2]: df_cpai = pd.DataFrame()
try:
    df_cpai = pd.read_csv("./NC_004431.cpai", sep=' ')
    if df_cpai.empty:
        print("Datoteka je prazna.")
except FileNotFoundError:
    print("Datoteka ne postoji.")
except pd.errors.EmptyDataError:
    print("Datoteka je prazna.")
```

Слика 8: Учитавање једног од три типа патогених острва за геном NC_004431

```
[9]: df_fasta=pd.read_csv("./izlaz_dc.load")
df_fasta
```

[9]:

	id	start_left	end_left	start_right	end_right	len	niska	niska2
0	NC_004431.1	1719039	1719050	2455661	2455672	12	AAAAAAAACATA	TTTTTTTTGTAT
1	NC_004431.1	295293	295304	3526060	3526071	12	AAAAAAAACATA	TTTTTTTTGTAT
2	NC_004431.1	1719039	1719050	3526060	3526071	12	AAAAAAAACATA	TTTTTTTTGTAT
3	NC_004431.1	1975705	1975716	2623377	2623388	12	AAAAAAAACCGG	TTTTTTTTGGCC
4	NC_004431.1	2385412	2385423	3460434	3460445	12	AAAAAAAACGGG	TTTTTTTTGCCC
...	...	...	...	...	...	...	...	...
401174	NC_004431.1	2185476	2185487	4277554	4277565	12	TTTTTTTTGGCC	AAAAAAAACCGG
401175	NC_004431.1	2185476	2185487	4936168	4936179	12	TTTTTTTTGGCC	AAAAAAAACCGG
401176	NC_004431.1	204875	204886	5128324	5128335	12	TTTTTTTTGTAT	AAAAAAAACATA
401177	NC_004431.1	3526060	3526071	5128324	5128335	12	TTTTTTTTGTAT	AAAAAAAACATA
401178	NC_004431.1	204875	204886	295293	295304	12	TTTTTTTTGTAT	AAAAAAAACATA

401179 rows × 8 columns

```
[10]: df_fasta.size
```

```
[10]: 3209432
```

Слика 9: Учитавање *.load* фајла и приказ количине података у фајлу

У наредном сегменту се врши поређење добијених података, тј. почетак и крај одговарајуће секвенце се пореди са сваким од почетака и крајева патогених острва добијајући информацију да ли се та секвенца налази унутар, у пресеку или ван тог патогеног острва.

```
[7]: if (not df_cpai.empty):
         for index_fasta, row_fasta in df_fasta.iterrows():
             dodat_presek=False
             dodat_unutra=False
             for index_cpai, row_cpai in df_cpai.iterrows():
                 if ((row_fasta['end_left']<=row_cpai['end'] and row_fasta['end_left']>=row_cpai['start'] and row_fasta['start_left'] < row_cpai[
                     presekcpai.append(row_fasta.iloc[6])
                     dodat_presek=True
                     break
                 elif (row_fasta['start_left'] > row_cpai['start']) and (row_fasta['end_left'] < row_cpai['end']):
                     unutracpai.append(row_fasta.iloc[6])
                     dodat_unutra=True
                     break
             if (dodat_presek==False and dodat_unutra==False):
                 vancpai.append(row_fasta.iloc[6])

         for index_fasta, row_fasta in df_fasta.iterrows():
             dodat_presek=False
             dodat_unutra=False
             for index_cpai, row_cpai in df_cpai.iterrows():
                 if ((row_fasta['end_right']<=row_cpai['end'] and row_fasta['end_right']>=row_cpai['start'] and row_fasta['start_right'] < row_cp
                     presekcpai.append(row_fasta.iloc[7])
                     dodat_presek=True
                     break
                 elif (row_fasta['start_right'] > row_cpai['start']) and (row_fasta['end_right'] < row_cpai['end']):
                     unutracpai.append(row_fasta.iloc[7])
                     dodat_unutra=True
                     break
             if (dodat_presek==False and dodat_unutra==False):
                 vancpai.append(row_fasta.iloc[7])
```

Слика 10: Разврставање секвенци на унутар/пресек/ван у зависности од њихових позиције за CPAI

Након разврставања следи пребројавање колико се пута та ниска (секвенца) јавила унутар, у пресеку или ван за свако од патогених острва PAI, CPAI и NPAI.

[1]:
```
counts = {}

# Iterirajte kroz cpai, npai i pai i brojite koliko
#se puta pojavljuje 6. atribut u svakom od njih
for count in unutracpai:
    if count in counts:
        counts[count][0] += 1
    else:
        counts[count] = [1, 0, 0, 0, 0, 0, 0, 0, 0]

for count in unutranpai:
    if count in counts:
        counts[count][1] += 1
    else:
        counts[count] = [0, 1, 0, 0, 0, 0, 0, 0, 0]

for count in unutrapai:
    if count in counts:
        counts[count][2] += 1
    else:
        counts[count] = [0, 0, 1, 0, 0, 0, 0, 0, 0]

for count in vannpai:
    if count in counts:
        counts[count][3] += 1
    else:
        counts[count] = [0, 0, 0, 1, 0, 0, 0, 0, 0]

for count in vancpai:
    if count in counts:
        counts[count][4] += 1
    else:
        counts[count] = [0, 0, 0, 0, 1, 0, 0, 0, 0]

for count in vanpai:
    if count in counts:
        counts[count][5] += 1
    else:
        counts[count] = [0, 0, 0, 0, 0, 1, 0, 0, 0]

for count in preseknpai:
    if count in counts:
        counts[count][6] += 1
    else:
        counts[count] = [0, 0, 0, 0, 0, 0, 1, 0, 0]

for count in presekcpai:
    if count in counts:
        counts[count][7] += 1
    else:
        counts[count] = [0, 0, 0, 0, 0, 0, 0, 1, 0]

for count in presekpai:
    if count in counts:
        counts[count][8] += 1
    else:
        counts[count] = [0, 0, 0, 0, 0, 0, 0, 0, 1]
```

Слика 11: Пребројавање за сваку од ниски

Те информације потом нормализујемо и смештамо у одговарајућу структуру. Уколико се јави случај да за неки геном не постоји неко од патогених острва целокупну ниску сматрамо да се она налази ван тог острва.

```
podaci={}
# Ispišite rezultate
for count, (unutra_cpai, unutra_npai, unutra_pai, vannpai_count, vancpai_count,vanpai_count,
            presek_npai, presek_cpai, presek_pai) in counts.items():

    sumacpai=unutra_cpai+vancpai_count+presek_cpai
    sumanpai=unutra_npai+vannpai_count+presek_npai
    sumapai=unutra_pai+vanpai_count+presek_pai
    suma=sumacpai+sumanpai+sumapai
    podaci[count] = {
        "Tip:":df_fasta['id'][0],
        "CPai": unutra_cpai / sumacpai if sumacpai!=0 else 0,
        "NPai": unutra_npai / sumanpai if sumanpai!=0 else 0,
        "Pai": unutra_pai / sumapai if sumapai!=0 else 0,
        "VANCpai": vancpai_count / sumacpai if sumacpai!=0 else 1,
        "VANNpai": vannpai_count / sumanpai if sumanpai!=0 else 1,
        "VANPai": vanpai_count / sumapai if sumapai!=0 else 1,
        "PRESEKCpai": presek_cpai / sumacpai if sumacpai!=0 else 0,
        "PRESEKNpai": presek_npai / sumanpai if sumanpai!=0 else 0,
        "PRESEKPai": presek_pai / sumapai if sumapai!=0 else 0
    }
```

Слика 12: Нормализација података

Пошто су подаци нормализовани и смештени у структуру података можемо их извести у CSV датотеку. За сваки од генома имаћемо укупно по четири овакве датотеке за dn, dc, in и ic.

```
[11]: (pd.DataFrame.from_dict(data=podaci, orient='index')
          .to_csv('NC_004431_dc.csv', header=True))
      store_data = pd.read_csv("NC_004431_dc.csv", header=0)
      store_data.rename(columns = {'Unnamed: 0':'Niska'}, inplace = True)
      store_data.to_csv('NC_004431_dc.csv', header=True,index=False)
      store_data = pd.read_csv("NC_004431_dc.csv", header=0)
```

Слика 13: Креирање CSV фајла за одговарајући геном и понављајућу секвенцу

4 Складиштење и анализа података

Целокупно обрађене податке желимо да сместимо у базу података. Како је задатак био одредити одступања појаве сва четири типа понављајућих секвенци у патогеним острвима бактерије Escherichia coli претходно добијене CSV датотеке за све геноме смо разврстали по понављајућим секвенцама и извршили њихово спајање[7] добијајући по једну главну датотеку за сваку од тих секвенци. [8]

За складиштење података користили смо IBM DB2[8] систем за управљање базама података и IBM Data Studio[9] алат за рад са базама података.

Након креирања базе податак потребно је креирати и различите табеле за одговарајуће понављајуће секвенце у које ће се сместити подаци из CSV датотека.

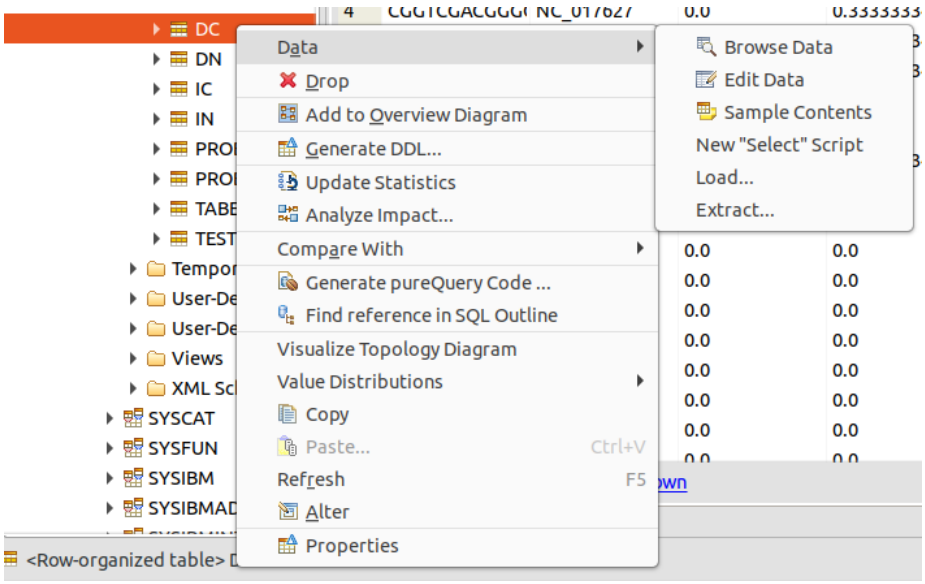
```
create table DC (Niska varchar(255),
Tip varchar(25),
CPAI float(5),
NPAI float(5),
PAI float(5),
VANCPAI float(5),
VANNPAI float(5),
VANPAI float(5),
PRESEKCPAI float(5),
PRESEKNPAI float(5),
PRESEKPAI float(5))
```

Слика 14: Креирање табеле DC

[7] Овај поступак смо извршили на Веб страни *Mighty Merge*[6]

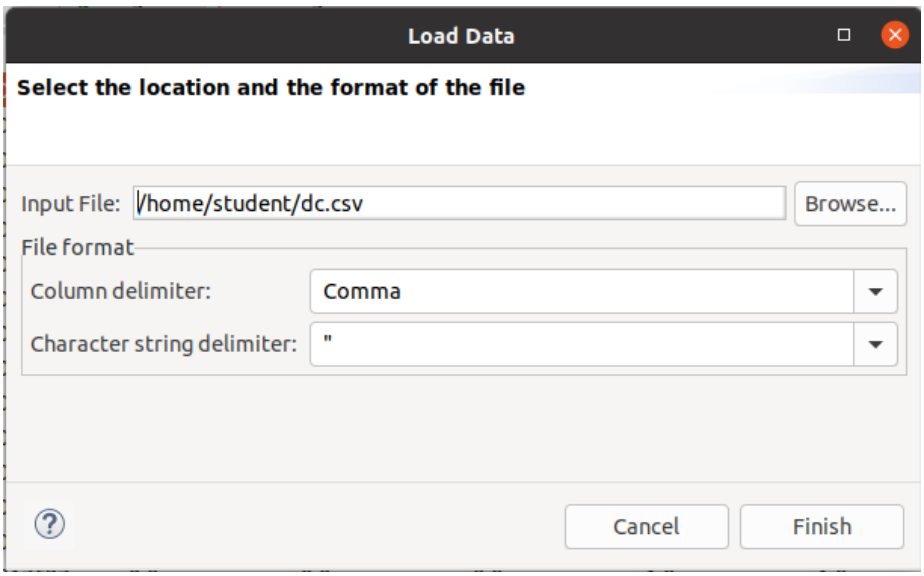
[8] Због количине и размене података и коришћења виртуелне машине коришћено је складиштење у облаку Mega [7]

Пошто су табеле успешно креиране вршимо увоз података. То ћемо одрадити следећим корацима. У директоријум *Tables* се налазе наше четири табеле. Десним кликом на једну од табела добијамо падајући мени на коме бирамо *Data* –> *Load*.



Слика 15: Увоз података у табеле 1

Добијамо искакајући прозор *Load Data* у коме у првом пољу бирамо датотеку коју желимо да увеземо, док остала поља остављамо подразумевано (као на слици). Овим су подаци успешно смештени у табелу.



Слика 16: Увоз података у табеле 2

Приказ података из табеле DC које смо увезли помоћу SQL упита:

SELECT * FROM DC

	NISKA	TIP	CPAI	NPAI	PAI	VANCPAI	VANNPAI	VANPAI	PRESEKCPAI	PRESEKNPAI	PRESEKPA
1	ACCTGGCACGGC	NC_017627	0.0	1.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0
2	AGTCAAGATAAA	NC_017627	0.0	0.5	0.0	1.0	0.5	1.0	0.0	0.0	0.0
3	CCGTGCCGGACC	NC_017627	0.0	0.33333334	0.0	1.0	0.6666667	1.0	0.0	0.0	0.0
4	CGGTCGACGGGC	NC_017627	0.0	0.33333334	0.0	1.0	0.6666667	1.0	0.0	0.0	0.0
5	CTTTTTACCCCG	NC_017627	0.0	0.33333334	0.0	1.0	0.6666667	1.0	0.0	0.0	0.0
6	GGTCGTACGGAC	NC_017627	0.0	0.33333334	0.0	1.0	0.6666667	1.0	0.0	0.0	0.0
7	TGCCGCCCACTC	NC_017627	0.0	0.5	0.0	1.0	0.5	1.0	0.0	0.0	0.0
8	TGCGCGTCTGCT	NC_017627	0.0	0.5	0.0	1.0	0.5	1.0	0.0	0.0	0.0
9	TGGACCGTGCCC	NC_017627	0.0	0.33333334	0.0	1.0	0.6666667	1.0	0.0	0.0	0.0
10	ACGGCGGGTGAC	NC_017627	0.0	0.0	0.0	1.0	1.0	1.0	0.0	0.0	0.0
11	CAAAACGCCCCT	NC_017627	0.0	0.0	0.0	1.0	1.0	1.0	0.0	0.0	0.0
12	CACTGTCTTAGT	NC_017627	0.0	0.0	0.0	1.0	1.0	1.0	0.0	0.0	0.0
13	CCAGCATGCCTC	NC_017627	0.0	0.0	0.0	1.0	1.0	1.0	0.0	0.0	0.0
14	CGGACCGCTTCC	NC_017627	0.0	0.0	0.0	1.0	1.0	1.0	0.0	0.0	0.0
15	GAAAAATGGGG	NC_017627	0.0	0.0	0.0	1.0	1.0	1.0	0.0	0.0	0.0
16	GCCAGCTGCCCC	NC_017627	0.0	0.0	0.0	1.0	1.0	1.0	0.0	0.0	0.0
17	GCCTGGCGAAGC	NC_017627	0.0	0.0	0.0	1.0	1.0	1.0	0.0	0.0	0.0
18	GGCACGGCCTGC	NC_017627	0.0	0.0	0.0	1.0	1.0	1.0	0.0	0.0	0.0
19	GTGACAGAATCA	NC_017627	0.0	0.0	0.0	1.0	1.0	1.0	0.0	0.0	0.0

Слика 17: Приказ малог дела података из табеле

Циљ пројекта јесте анализа одступања сва четири типа понављајућих секвенци. Одређујемо ниске такве да се оне у целости налазе унутар одређеног патогеног острва или у његовом пресеку, али да се не налазе ван њега. Такве ниске називамо карактеристичне ниске одговарајуће понављајуће секвенце.

Наредним SQL упитом добијамо карактеристичне ниске табеле DC.

```
select NISKA,
    sum(CPAI) AS "CPAI",sum(NPAI) AS "NPAI",sum(PAI) AS "PAI",
    sum(VANCPAI) AS "VANCPAI",sum(VANNPAI) AS "VANNPAI",sum(VANPAI) AS "VANPAI",
    sum(PRESEKCPAI) AS "PRESEKCPAI",sum(PRESEKNPAI) AS "PRESEKNPAI",
    sum(PRESEKPAI) AS "PRESEKPAI"
from DC
group by NISKA
having  ((SUM(CPAI)>1 or SUM(PRESEKCPAI)>0) and SUM(VANCPAI)=0) or
        ((SUM(NPAI)>1 or SUM(PRESEKNPAI)>0) and SUM(VANNPAI)=0) or
        ((SUM(PAI)>1 or SUM(PRESEKPAI)>0) and SUM(VANPAI)=0);
```

Слика 18: SQL упит за добијање карактеристичних ниски

Извршавањем овог упита добијају се следећи резултати:

	NISKA	CPAI	NPAI	PAI	VANCPAI	VANNPAI	VANPAI	PRESEKCPAI	PRESEKNPAI	PRESEKPA
43	AAAAATAAAATTC	7.0	0.0	1.0	0.0	7.0	6.0	0.0	0.0	0.0
44	AAAAATAAAGATT	7.0	0.0	2.0	0.0	7.0	5.0	0.0	0.0	0.0
45	AAAAATGGAAACC	7.0	0.0	0.0	0.0	7.0	7.0	0.0	0.0	0.0
46	AAAAATGGTCGCA	7.0	0.0	0.0	0.0	7.0	7.0	0.0	0.0	0.0
47	AAAACAGTGCGAC	7.0	0.0	1.0	0.0	7.0	6.0	0.0	0.0	0.0
48	AAAACGAAACCGC	7.0	0.0	1.0	0.0	7.0	6.0	0.0	0.0	0.0
49	AAAACGGTCGTTT	7.0	0.0	0.0	0.0	7.0	7.0	0.0	0.0	0.0
50	AAAAGAACAACGC	7.0	0.0	1.0	0.0	7.0	6.0	0.0	0.0	0.0
51	AAAAGTCGGTCG	7.0	0.0	0.0	0.0	7.0	7.0	0.0	0.0	0.0
52	AAAATATTTGCTT	7.0	0.0	0.0	0.0	7.0	7.0	0.0	0.0	0.0
53	AAAATGGCCTTTC	7.0	0.0	0.0	0.0	7.0	7.0	0.0	0.0	0.0
54	AAACAACTGGATT	7.0	0.0	0.0	0.0	7.0	7.0	0.0	0.0	0.0
55	AAAAAACATCTAA	6.0	0.0	1.0	0.0	6.0	5.0	0.0	0.0	0.0
56	AAAAAATAAACTT	6.0	0.0	2.0	0.0	6.0	4.0	0.0	0.0	0.0
57	AAAAAATAAAGAT	6.0	0.0	1.0	0.0	6.0	5.0	0.0	0.0	0.0
58	AAAAAATTGTAAT	6.0	0.0	1.0	0.0	6.0	5.0	0.0	0.0	0.0
59	AAAAACAGGTAA	6.0	0.0	0.0	0.0	6.0	6.0	0.0	0.0	0.0
60	AAAAACATTCAGC	6.0	0.0	0.0	0.0	6.0	6.0	0.0	0.0	0.0

Слика 19: Резултати упита

А како бисмо видели који је број карактеристичних ниски за дату понављајућу секвенцу извршава се SQL упит:

```
select count(*)
from
(select NISKA,
    sum(CPAI) AS "CPAI",sum(NPAI) AS "NPAI",sum(PAI) AS "PAI",
    sum(VANCPAI) AS "VANCPAI",sum(VANNPAI) AS "VANNPAI",sum(VANPAI) AS "VANPAI",
    sum(PRESEKCPAI) AS "PRESEKCPAI",sum(PRESEKNPAI) AS "PRESEKNPAI",
    sum(PRESEKPAI) AS "PRESEKPAI"
from DC
group by NISKA
having  ((SUM(CPAI)>1 or SUM(PRESEKCPAI)>0) and SUM(VANCPAI)=0) or
        ((SUM(NPAI)>1 or SUM(PRESEKNPAI)>0) and SUM(VANNPAI)=0) or
        ((SUM(PAI)>1 or SUM(PRESEKPAI)>0) and SUM(VANPAI)=0));
```

Слика 20: Пребројавање карактеристичних ниски

Упоредимо број ниски које се јављају у табелама, као и број карактеристичних ниски које се добијају претходним упитом за сваку од понављајућих секвенци.

	Број ниски	Број карактеристичних ниски	Проценат карактеристичних ниски
DC	9 963 150	26 805	0,26904
DN	24 861 684	6665	0,02681
IC	29 998 713	70 658	0,23554
IN	10 154 302	27 255	0,26841
$\sum$	74 977 849	131 683	0,17563

Табела 1: Поређење укупног броја ниски и броја карактеристичних ниски

Исто на примеру табеле DC за коју смо добили да је број карактеристичних ниски 26 805, наредним упитом можемо да видимо број карактеристичних ниски по њиховим дужинама и да закључимо да се највише јављају карактеристичне ниске дужине 12 и 14. Исти поступак обављамо и за остале табеле, а њихове резултате остављамо у прилогу.

```
SELECT
    LENGTH(subquery.NISKA) AS DUZINA_NISKI,
    COUNT(*) as BROJ_NISKI
FROM (
    SELECT
        NISKA,
        SUM(CPAI) AS CPAI,
        SUM(NPAI) AS NPAI,
        SUM(PAI) AS PAI,
        SUM(VANCPAI) AS VANCPAI,
        SUM(VANNPAI) AS VANNPAI,
        SUM(VANPAI) AS VANPAI,
        SUM(PRESEKCPAI) AS PRESEKCPAI,
        SUM(PRESEKNPAI) AS PRESEKNPAI,
        SUM(PRESEKPAI) AS PRESEKPAI
    FROM DC
    GROUP BY NISKA
    HAVING
        ((SUM(CPAI) > 1 OR SUM(PRESEKCPAI) > 0) AND SUM(VANCPAI) = 0) OR
        ((SUM(NPAI) > 1 OR SUM(PRESEKNPAI) > 0) AND SUM(VANNPAI) = 0) OR
        ((SUM(PAI) > 1 OR SUM(PRESEKPAI) > 0) AND SUM(VANPAI) = 0)
) AS subquery
GROUP BY LENGTH(subquery.NISKA);
```

Слика 21: SQL упит за добијање броја карактеристичних ниски по њиховим дужинама

Дужина карактеристичних ниски	Број карактеристичних ниски
12	10 732
13	3309
14	9072
15	2777
16	671
17	192
18	39
19	7
20	6
$\sum$	26 805

Табела 2: Број карактеристичних ниски по њиховим дужинама за табелу DC

5 Закључак

У овом раду анализирали смо присуство патогених острва у геномима бактерије Escherichia coli користећи податке из базе PAIDB и алат StatRepeats из пакета RepeatsPlus. Наша анализа идентификовала је кључне секвенце у патогеним острвима које су одговорне за вируленцију бактерија. Резултати су показали да се највећи број карактеристичних низа појављује у дужинама од 12 и 14, што указује на специфичну структуру секвенци у овим регионима.

Осим тога, категоризовали смо патогена острва на PAI, cPAI и nPAI, где су cPAI кандидати за патогена острва, а nPAI представљају мало вероватна патогена острва. Иако nPAI нису директно повезани са патогеношћу, њихова анализа може открити потенцијалне нове аспекте бактеријске вируленције.

Коришћење нормализованих података и креирање база података омогућило је дубљу анализу и складиштење информација о патогеним острвима. Наши резултати пружају значајан увид у генетичку основу патогености бактерије E. coli и могу послужити као основа за будућа истраживања.

Рад наглашава важност детаљне анализе генома патогених бактерија и показује како комбинација биоинформатичких алата може допринети бољем разумевању механизама патогености.

На основу резултата добијених у овом раду, можемо закључити следеће:

1. Специфичне дужине карактеристичних ниски указују на то да ове ниске могу имати јединствене структурне особине које су конзистентне кроз различите патогене геноме. То би значило да ове ниске могу бити даље коришћене у мапирању генетичких маркера патогености, односно служити за даљу идентификацију и проучавање патогених острва.

2. Класификација патогених острва на PAI, cPAI и nPAI омогућава прецизнију дијагностику и боље разумевање потенцијала ових острва за изазивање болести, што је важно за развој дијагностичких метода.

3. Наша методологија нормализације података и креирања база података може бити примењена и на друге бактеријске врсте, пружајући универзални алат за анализу патогених генома.

Литература

[1] National Center for Biotechnology Information

[2] Pathogenicity Island Database (PAIDB)

[3] Python

[4] Jupyter Notebook

[5] StatRepeats

[6] Mighty Merge

[7] Mega

[8] IBM DB2

[9] IBM Data Studio

A Додатак

Сви приложени фајлови и резултати истраживања налазе се на адреси: https://mega.nz/folder/hJhQGZIR#39jOtDPh5FUUgeyQ324LFw