Sprawozdanie z Programów Komputerowych

Projekt końcowy - polecenie 6

Michał Humiński nr 117415
Daria Plewa nr 117359
Bioinformatyka rok 1
Studia licencjackie
rok 2019/2020

Polecenie

Utwórz transkryptor kodów nukleotydowych na aminokwasowe.
Dla danej sekwencji nukleotydowej program generuje odpowiadającą jej sekwencję aminokwasową, dokonując transkrypcji trójek na kody aminokwasów. Tłumaczenie takie zaczynane jest od trójki startowej.
Program operuje na pliku danych w formacie FASTA wskazanym przez użytkownika i generuje na jego podstawie inny plik FASTA. Do dobrego stylu należy opisanie dokonanych zmian w nagłówkach rekordu.

Pomysł

Naszym celem było napisanie kodu, który byłby w stanie przepisać sekwencję kodu genetycznego na odpowiadające mu aminokwasy. W pierwszej kolejności zdecydowaliśmy się na umieszczenie sekwencji sobie odpowiadających aminokwasów w słowniku (niektóre wielkości się powtarzały, co jest cechą kodu genetycznego - jest zdegenerowany). Następnie za pomocą określonych funkcji odczytywaliśmy trójkami odpowiednie litery w kodzie i porównywaliśmy je z wcześniej utworzonym słownikiem. Następnie zapisywaliśmy nowe wartości w liście, która była naszą nową przepisaną sekwencją aminokwasów.

Napotkane problemy oraz ich rozwiązania

Zanim rozpoczęliśmy pracę zebraliśmy informacje, które charakteryzują kod genetyczny. Kod genetyczny jest:

- trójkowy - jeden aminokwas jest określony przez 3 nukleotydy,
- bezprzecinkowy - trójki następują bezpośrednio po sobie bez dodatkowych znaków,
- niezachodzący - każda kolejna trójka oznacza kolejny aminokwas (jeden aminokwas nie nachodzi na poprzedni),
- jednoznaczny - jedna trójka koduje jeden odpowiadający jej aminokwas,
- zdegenerowany - aminokwas może być kodowany przez większą ilość trójek nukleotydów.

Z początku przepisywane przez nas dane miały postać trójliterowych symboli (np. Leu) aminokwasów, a w późniejszych próbach zmieniliśmy je na takie, które obowiązują w zapisie w formacie FASTA (czyli jednoliterowe symbole jak np. L).

Również w pierwszej kolejności cały skonstruowany przez nas kod znajdował się w jednym pliku, dopiero później zdecydowaliśmy się na rozdzielenie go do paru plików.

Po kolejnych rozmyślaniach doszliśmy do wniosku, że mogą zostać nam podane dwa rodzaje kodu do przetłumaczenia, jeden z wartością *U* zamiast *T* (RNA) a drugi z wartością *T* zamiast *U* (DNA), dlatego pozwoliliśmy sobie na dodanie funkcji która przepisuje w naszym kodzie wartość *T* (tymina) na *U* (uracyl), dzięki czemu program jest bardziej uniwersalny i jest w stanie odczytać oba formaty wpisu.

Kolejną edycją było założenie, że podany plik FASTA nie musi zawierać zapisu tylko jednej sekwencji nukleotydowej - może być ich więcej. Wszystkie takie sekwencje z pobranego pliku są odpowiednio translatowane.

Na sam koniec pozwoliliśmy użytkownikowi na utworzenie indywidualnej nazwy oraz miejsca zapisu dla przepisywanego pliku oraz zapisanie go pod tym właśnie tytułem w konkretnym miejscu (po podaniu jego ścieżki).

Funkcje, które umożliwiają odczyt pliku zawarliśmy w odrębnym pliku (tak jak słownik z zawartościami odpowiednich aminokwasów). Pierwsza linijka kodu dotyczy podstawowych informacji odczytywanego kodu i zaczyna się od znaku >, dzięki czemu mogliśmy rozpocząć przepisywanie kodu od linijki następnej.

Również z wykorzystaniem tych funkcji podzieliliśmy całą sekwencję na krótkie sekwencje 3-literowe (kod jest niezachodzący), które następnie za pomocą pętli były porównywane z naszym słownikiem.

Pod sam koniec także dopisaliśmy dodatkowe nukleotydy, które zawierał kod a jednak nie mogły zostać przetłumaczone (kod jest trójkowy).

Program końcowy wraz z krótkim opisem

Funkcje

```
#!/usr/bin/env python
# coding: utf-8
# In[1]:

'''
Funkcje użyte w transkryptorze kodu genetycznego na kod aminokwasowy.
'''

import Slownik

def odczyt(dane, naglowki, kod):
    for informacje in dane:
        informacje = informacje.rstrip('\n')
        if informacje[0] == '>':
            naglowki.append(informacje)
            kod.append('')
        else:
            kod[-1] += informacje

def TnaU(kod):
    licznik = 0
    dlugosc = len(kod)
    if licznik <= dlugosc:
        kod[licznik] = kod[licznik].translate(kod[licznik].maketrans('T', 'U'))
        licznik += 1

def translacja(kod, bialka, pominiete):
    for RNA in kod:
        bialko = ''
        while len(RNA) % 3 != 0:
            pominiete.append(RNA[-1])
            RNA = RNA[:-1]
        for i in range(0, len(RNA), 3):
            kodon = RNA[i:i + 3]
            bialko += Slownik.kod_genetyczny_RNA[kodon]
        bialka.append(bialko)

def zapis(utworz, naglowki, bialka):
    zapisz = open(utworz, 'w')
    numer = 0
    ilosc = len(bialka) - 1
    while numer < ilosc:
        zapisz.write(naglowki[numer] + '\n')
        zapisz.write(bialka[numer] + '\n')
        numer += 1
    zapisz.write(naglowki[numer] + '\n')
    zapisz.write(bialka[numer])
    zapisz.close()
```

W tym pliku zawarliśmy 4 funkcje.
Pierwsza służy do odczytania odpowiedniej sekwencji z pliku FASTA - omijając pierwszą linijkę pliku, która podaje źródło kodu (skąd pochodzi).
Druga pozwala na zamianę *T* na *U* dzięki czemu nasz program jest bardziej uniwersalny (analizujemy tylko RNA).
Trzecia dzieli całą sekwencję na trójki, które następnie są translatowane za pomocą słownika.
Czwarta funkcja zapisuje kod w pod określoną nazwą w wybranej lokalizacji.

Kod

```
#!/usr/bin/env python
# coding: utf-8

# In[ ]:


#!/usr/bin/env python
#coding: utf-8

'''
Transkryptor kodu genetycznego na kod aminokwasowy.
Program przyjmuje dane jako pliki .txt z zapisanym kodem w formacie fasta.
'''

import Funkcje

print('Witaj w programie do zamiany kodu DNA lub RNA na łańcuch aminokwasowy!')
plik = input('Wprowadź ścieżkę pliku: ')
dane = open(plik, 'r')

naglowki = []
kod = []
Funkcje.odczyt(dane, naglowki, kod)

Funkcje.TnaU(kod)

bialka = []
pominiete = []
Funkcje.translacja(kod, bialka, pominiete)

print('Jak chcesz nazwać docelowy plik?')
nazwa = input('Wprowadź nazwę docelowego pliku bez rozszerzenia: ')

print('Gdzie chcesz zapisać docelowy plik?')
lokalizacja = input('Wprowadź docelową ścieżkę bezwzględną: ')

utworz = lokalizacja + '/' + nazwa + '.txt'

Funkcje.zapis(utworz, naglowki, bialka)

print('Twój kod został pomyślnie przetłumaczony!')
pozostale = len(pominiete)
if pozostale > 0:
    pozostale = str(pozostale)
    print('Podczas pracy pominięto nukleotydy w liczbie: ' + pozostale)
```

Jest to nasz główny kod, który komunikuje się wraz z użytkownikiem i pobiera od niego wymagane dane. W pierwszej kolejności otrzymuje od użytkownika źródło pliku FASTA, który będzie służył jako kod, który program ma przepisać. Następnie korzysta on z funkcji, które są podane w drugim pliku i odczytuje odpowiednią sekwencję, zamienia *T* na *U* (aby program był bardziej uniwersalny), po czym dokonuje samej transakcji. Na sam koniec pyta się użytkownika o lokalizację zapisu oraz nazwę, którą ma otrzymać. Otrzymujemy informacje zwrotną, że nasz kod został przetłumaczony oraz dostajemy zwrotną ilość nukleotydów nieprzetłumaczonych, gdy długości badanych sekwencji nie były podzielne przez 3.

Lista aminokwasów

```
#!/usr/bin/env python
# coding: utf-8

# In[ ]:


'''
Kod genetyczny RNA->aminokwas wykorzystywany podczas translacji.
'''

kod_genetyczny_RNA = {'UUU':'F',
                      'UUC':'F',
                      'UUA':'L',
                      'UUG':'L',
                      'CUU':'L',
                      'CUC':'L',
                      'CUA':'L',
                      'CUG':'L',
                      'AUU':'I',
                      'AUC':'I',
                      'AUA':'I',
                      'AUG':'M',
                      'GUU':'V',
                      'GUC':'V',
                      'GUA':'V',
                      'GUG':'V',
                      'UCU':'S',
                      'UCC':'S',
                      'UCA':'S',
                      'UCG':'S',
                      'CCU':'P',
                      'CCC':'P',
                      'CCA':'P',
                      'CCG':'P',
                      'ACU':'T',
                      'ACC':'T',
                      'ACA':'T',
                      'ACG':'T',
                      'GCU':'A',
                      'GCC':'A',
                      'GCA':'A',
                      'GCG':'A',
                      'UAU':'Y',
                      'UAC':'Y',
                      'UAA':'*',
                      'UAG':'*',
                      'CAU':'H',
                      'CAC':'H',
                      'CAA':'Q',
                      'CAG':'Q',
                      'AAU':'N',
                      'AAC':'N',
                      'AAA':'K',
                      'AAG':'K',
                      'GAU':'D',
                      'GAC':'D',
                      'GAA':'E',
                      'GAG':'E',
                      'UGU':'C',
                      'UGC':'C',
                      'UGA':'*',
                      'UGG':'W',
                      'CGU':'R',
                      'CGC':'R',
                      'CGA':'R',
                      'CGG':'R',
                      'AGU':'S',
                      'AGC':'S',
                      'AGA':'R',
                      'AGG':'R',
                      'GGU':'G',
                      'GGC':'G',
                      'GGA':'G',
                      'GGG':'G'
                     }
```

Jest to klasyczny słownik. Na przykładzie jednej jego wartości możemy przedstawić jego zasadę działania. ‘GGU’:’R’ oznacza, że trójkowej sekwencji *GGU* będzie odpowiadał aminokwas o symbolu *R* (warto zwrócić uwagę, że nie będzie to działało w odwrotną stronę). Wartości w słowniku się powtarzają, gdyż kod genetyczny jest zdegenerowany - dla jednego aminokwasu mogą przypadać różne trójki nukleotydowe, jednak różne aminokwasy nie mogą być oznaczane przez jeden trójkowy kodon.