Uniwersytet Warszawski
Wydział Matematyki, Informatyki i Mechaniki

Aleksander Mućk
Nr albumu: 382184

Algorytmy do klastrowania duplikacji genomowych

Praca licencjacka
na kierunku BIOINFORMATYKA I BIOLOGIA SYSTEMÓW

Praca wykonana pod kierunkiem
dra hab. Pawła Góreckiego

Sierpień 2019

Oświadczenie kierującego pracą

Potwierdzam, że niniejsza praca została przygotowana pod moim kierunkiem i kwalifikuje się do przedstawienia jej w postępowaniu o nadanie tytułu zawodowego.

Data Podpis kierującego pracą

Oświadczenie autora (autorów) pracy

Świadom odpowiedzialności prawnej oświadczam, że niniejsza praca dyplomowa została napisana przeze mnie samodzielnie i nie zawiera treści uzyskanych w sposób niezgodny z obowiązującymi przepisami.

Oświadczam również, że przedstawiona praca nie była wcześniej przedmiotem procedur związanych z uzyskaniem tytułu zawodowego w wyższej uczelni.

Oświadczam ponadto, że niniejsza wersja pracy jest identyczna z załączoną wersją elektroniczną.

Data Podpis autora pracy

Streszczenie

Niniejsza praca przedstawia propozycje rozwiązań algorytmicznych dla problemów klastrowania duplikacji genomowych w oparciu o scenariusze ewolucyjne. W części pierwszej wprowadzane są podstawowe pojęcia dotyczące drzew genów, gatunków, modeli ich uzgadniania oraz tworzenia scenariuszy ewolucyjnych. Omówiony został również problem przeliczania i klastrowania duplikacji genomowych. W części drugiej opisana została proponowana heurystyka wraz z przykładowymi testami oraz jej implementacją w języku Python.

Słowa kluczowe

duplikacja genu, drzewo genów, drzewo gatunków, analiza filogenetyczna, drzewo uzgadniające, Python, scenariusz ewolucyjny, strata genu, minimalizacja kosztu ewolucyjnego

Dziedzina pracy (kody wg programu Socrates-Erasmus)

11.9 Inne nauki matematyczne i informatyczne

Klasyfikacja tematyczna

Computional biology, Applied computing, Life and medical sciences

Tytuł pracy w języku angielskim

Algorithms for the clustering of genomic duplication

Spis Treści

Wprowadzenie

Uzgadnianie drzew filogenetycznych jest, przez rozmiar danych i coraz bardziej skomplikowane modele, niezwykle złożone zarówno obliczeniowo jak i koncepcyjnie. Badania drzew genów i gatunków, a w szczególności zależności między nimi może odpowiedzieć na pytania w jaki sposób wyodrębniały się gatunki przez pryzmat zmian w ich genomie. Mimo wszystko jednak należy pamiętać, że pokrewieństwo gatunków nie zawsze implikuje pokrewieństwo genów, których drzewo ewolucyjne nie musi pokrywać się z drzewem zawierających je gatunków, które samo w sobie nie jest tak bardzo bardzo zróżnicowane jak drzewo genów. Tworzenie scenariuszy ewolucyjnych, dzięki którym możemy poznać w jaki sposób ewolucja genów wpływała na ewolucję gatunków jest zadaniem nietrywialnym. Potrzebne są narzędzia, które potrafiłyby ocenić scenariusze pod kątem ilości epizodów ewolucyjnych. Epizody, takie jak duplikacje genomowe, mogą być wyznacznikami prawdopodobieństwa danego scenariusza. Zagadnienie to jest jeszcze bardziej wymagające od uzgodnienia pojedynczego drzewa, ponieważ jeden scenariusz zawiera wiele drzew genów co wpływa na poziom złożoności obliczeń. W niniejszej pracy proponowany jest algorytm, który ocenia zbiór scenariuszy tworząc na ich podstawie jeden, którego koszt, liczony jako ilość duplikacji, będzie możliwie najmniejszy.
Praca składa się z czterech rozdziałów i dodatków. W rozdziale 1 przedstawiono podstawowe pojęcia dotyczące drzew genów, drzew gatunków oraz modeli i scenariuszy ewolucyjnych. Rozdział 2 przedstawia propozycję heurystyki wraz z jej testami na rzeczywistych danych. W rozdziale tym opisano również implementację i sposób użycia programu napisanego na podstawie przybliżonej we wcześniejszej sekcji heurystyki. Ostatni rozdział zawiera przemyślenia dotyczące możliwego użycia algorytmu i perspektyw jego rozwoju. W dodatkach umieszczono fragmenty kodu, przykładowe dane wejściowe i wyniki działania algorytmu.

Rozdział 1

Podstawowe pojęcia

W tym rozdziale poruszane są pojęcia i definicje niezbędne do zrozumienia problematyki klastrowania duplikacji genomowych.

1.1. Wstęp biologiczny

Ewolucja biologiczna jest procesem zmian w trakcie których organizmy stopniowo nabywają lub tracą pewne cechy. Jest to element kluczowy dla powstawania nowych gatunków: specjacji. Śledzenie w jaki sposób kształtowały się nowe gatunki i w jaki sposób zachodziły na Ziemi procesy ewolucyjne jest zadaniem niezwykle złożonym i wymagającym specyficznego podejścia. Jednym z możliwych sposobów przedstawienia historii ewolucyjnej gatunków jest drzewo filogenetyczne, które przedstawiają zależności ewolucyjne pomiędzy umieszczonymi na nim gatunkach lub genach.

Początkowo za wyznacznik pokrewieństwa gatunków służyło podobieństwo morfologiczne, jednak obecnie często stosuje się metody polegające na badaniu podobieństwa danych rodzin genów. Można założyć, że im większe podobieństwo genomu danych organizmów tym bliżej są one spokrewnione. Z punktu widzenia tej pracy genom jest niczym więcej jak zbiorem genów obecnym w danym organizmie.

1.2. Drzewa genów i gatunków

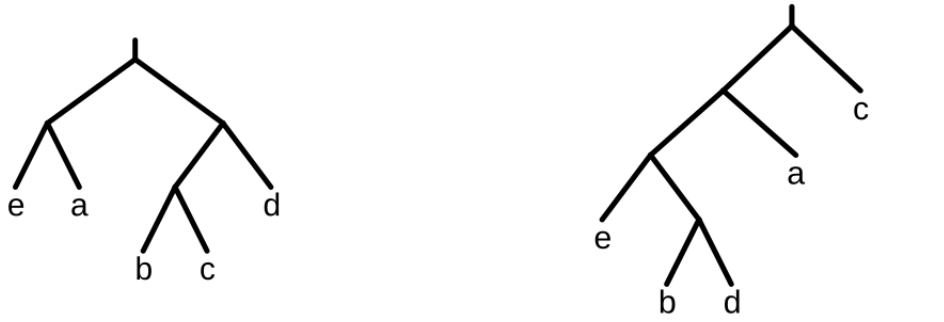
W pracy tej T niech będzie ukorzenionym, binarnym drzewem o zbiorze krawędzi E_T i zbiorze węzłów V_T:

$$\mathrm{T} = <V_T\ ,\ E_T>.$$

E_T zawiera pary węzłów (v_x , v_y) takie, że $V_T \ni v_x$,v_y. Węzły z których nie wychodzą żadne krawędzie nazywane są liśćmi, a korzeń jest węzłem do którego nie prowadzą żadne krawędzie (nieposiadającym rodzica). Węzeł v_x jest przodkiem węzła v_y jeśli istnieje ścieżka skierowana z węzła v_x do węzła v_y. Liczba krawędzi w ścieżce od węzła v_x do węzła v_y jest nazywana długością. Poddrzewem węzła v_x jest drzewo oznaczone T(v_x) w którym węzeł v_x jest korzeniem.

Definicja 1.2.1 *Związki między gatunkami przedstawia się za pomocą drzewa T, zwanego drzewem gatunków S. W drzewie S każdy z liści reprezentuje inny gatunek Sp, a węzły wewnętrzne są specjacjami.*

Definicja 1.2.2 *Związki między genami w danej rodzinie przedstawia się za pomocą drzewa T, zwanego drzewem genów G. W drzewie G każdy z liści reprezentuje przynależność danego genu do gatunku Sp.*



(a) Przykładowe drzewo genów G

(b) Przykładowe drzewo gatunków S

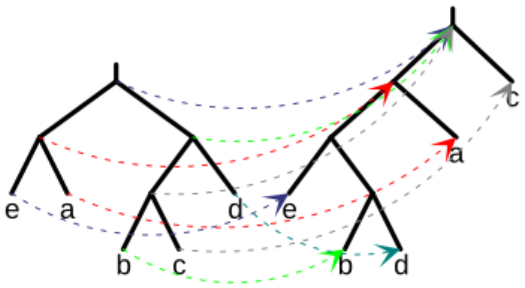
Rysunek 1.1: Przykładowe drzewa T [1]

1.3. Uzgodnienie drzew

Bardzo częste różnice struktury drzewa genów w stosunku do historii ewolucyjnej opisanej drzewem gatunków wymagają mapowania węzłów drzewa G na węzły znajdujące się w drzewie S. Jest to krok niezbędny by zrozumieć w jaki sposób ewolucja gatunków wpływała na strukturę ich genomów.

1.3.1. Mapowanie LCA

Podstawowym algorytmem dla tego typu uzgodnień jest **algorytm LCA** (ang. *Lowest Common Ancestor*; pl. *Najniższy Wspólny Przodek*). Najniższym przodkiem węzłów v_x i v_y jest taki węzeł v_{anc}, który jest przodkiem obu węzłów i którego długość od korzenia drzewa jest największa.



Rysunek 1.2: Uzgodnienie LCA dla przykładowych drzew genów G i gatunków S [1]

Definicja 1.3.1 *Uzgodnienie drzew jest taką funkcją MAP_{LCA}: $G \to S$ gdzie dla każdego węzła v_g z drzewa genów G przyporządkowuje się węzeł v_s z drzewa gatunków S tak, że węzeł v_s jest węzłem v_{anc} i węzły z poddrzewa T(v_g) są obecne w T(v_s).*

1.3.2. Drzewa DLS

1.4. Koszt scenariusza ewolucyjnego

1.5. Modele scenariuszy ewolucyjnych

Podstawowy opis idei. Podział ruchów:

Definicja 1.5.1 *TMOVE.*

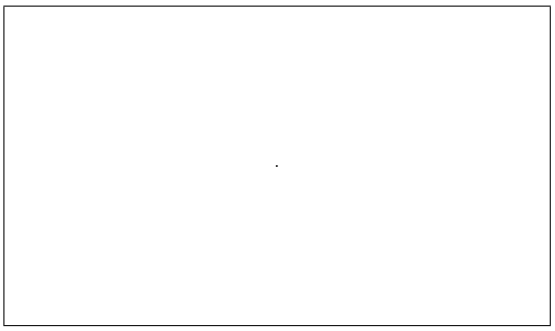
Definicja 1.5.2 *CLOST.*

1.6. Opis modeli

Dozwolone modele:

1. PG

2. FHS;



Rysunek 1.3: Obrazek scenariusza

Rozdział 2

Heurystyka

2.1. Opis algorytmu

Algorytm zakłada, że na wejściu dostępne są dane:

- Drzewa genów G_1 ... G_k,
- m_i scenariuszy drzewa G_i opisanych jako wektory v_{i1}, v_{i2} ... v_{im} z wyliczonymi wartościami kosztu ewolucyjnego, mierzonego jako ilość duplikacji dla każdego węzła.

Istnieje wektor przybliżający rozwiązanie ME, który nazwijmy V* i który pasuje do każdego scenariusza.
Jednym z takich wektorów jest wektor V_{max}, który na każdej współrzędnej x zawiera maksymalną wartość $v_{im}[x]$ dla każdego scenariusza m w każdym drzewie genów G_i. Oczywiście nie jest to rozwiązanie najlepsze, ponieważ jest one kosztowne, ale pasuje do każdego scenariusza.

Bazując na wyliczonym wektorze V_{max} heurystyka wylicza wektor V* i w pętli poprawia go w następujący sposób:

- Obniż jedna z wartości w wektorze V*,
- Zaakceptuj w/w zmianę jeśli dla każdego drzewa genów istnieje scenariusz, który jest zgodny z takim wektorem epizodów,
- Zakończ działanie jeśli nie da się poprawić żadnej współrzędnej.

Wybór współrzędnej może, zależnie od potrzeby, może być dokonywany w inny sposób:

- od końca wektora (od korzenia),
- od początku (od liści),
- losowo.

2.2. Testy algorytmu na danych rzeczywistych

Zbiorem danych dla testu na danych rzeczywistych był zbiór Guigo zawierający 53 ukorzenione drzewa genów pochodzące od 16 eukariontów. Do porównania wyników użyty został program napisany przez dra Jarosława Paszka. Program ten korzystając z dostępnych metod algorytmicznych wylicza dokładne i najniższe możliwe wartości kosztu ewolucyjnego dla podanych modeli.

2.3. Testy algorytmu na danych symulowanych

sdsd

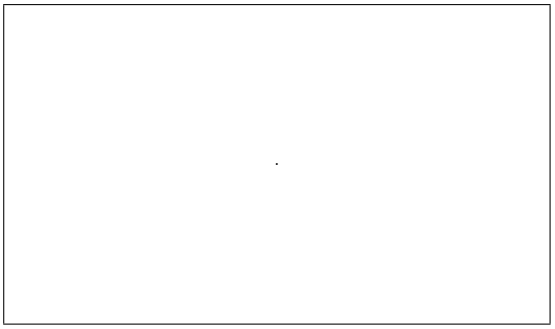
2.4. Dokumentacja użytkowa i opis implementacji

Opisana wyżej heurystyka zaimplementowana została w języku Python w wersji 3.7.4 z użyciem paradygmatu obiektowego, gdzie zbiór wszystkich scenariuszy dla wszystkich drzew, pojedynczy diagram redukcyjny otrzymany z jednego drzewa genów, a także pojedynczy scenariusz stanowią oddzielne klasy. Program pythonowy przyjmuje na wejściu listę plików w których zawarte są wyliczone scenariusze dla danego drzewa genów.

Pliki przekazane do programu muszą być w określonym formacie gdzie w każdej linii dla każdego drzewa muszą znajdować się podane niżej pola w określonym porządku:

- id drzewa,
- wysokość drzewa na diagramie redukcyjnym,
- drzewo zawierające liczbę duplikacji dla danej pozycji w drzewie w porządku prefiksowym,
- id drzew,
- typ zmiany w kolejności drzew.

Algorytm ten, dla wygody użycia, został obudowany skryptem napisanym w języku bash, który pozwala na wyliczenie scenariuszy w modelu FHS i PG z wykorzystaniem programu DLSgen autorstwa dra hab. Pawła Góreckiego. Program ten wylicza dla danego drzewa genów i drzewa gatunków niezbędne dane w podanym powyżej formacie.



Rysunek 2.1: Przykład pliku wejściowego

Rozdział 3

Podsumowanie

W pracy przedstawiono pierwszy i dosyć intuicyjny pomysł na ocenę scenariuszy ewolucyjnych pod kątem ilości duplikacji. Należy jednak wspomnieć, że bazując samą idę da się znacząco usprawnić i zejść ze złożonością nawet do liniowej oceny, bez poprzedzającego kroku, w którym wyliczane są wszystkie scenariusze. Testy pokazują również, że opisany algorytm zwraca wyniki, które różnią się w bardzo niewielkim stopniu (o ile w ogóle) od rzeczywistego minimalnego kosztu ewolucji.

3.1. Perspektywy rozwoju

Trudno przewidzieć wszystkie możliwości rozwoju, ale te bardziej oczywiste można wskazać już teraz. Są to:

- opisy naszych algorytmów optymalizacyjnych,

- opisy naszych algorytmów optymalizacyjnych,

- opisy naszych algorytmów optymalizacyjnych,

3.2. Perspektywy wykorzystania

DOPYTAĆ

Dodatek A

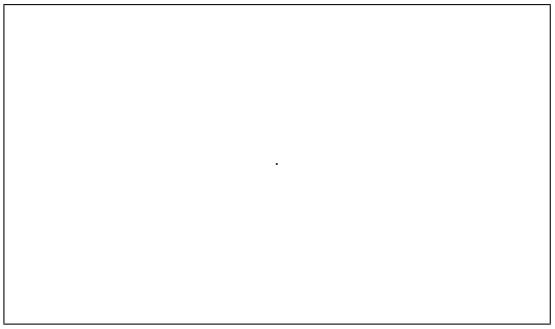
Pętla programu zapisana w języku Python wykonywana dla losowego wybierania indeksów

```
max_trees = []
        for scenario in self:
            all_dup_pref = [tree.duplication_prefix for tree n scenario]
            max_trees.append(self.rate_scenario(all_dup_pref))
        max_tree = self.rate_scenario(max_trees)

        if select_type == "random":

            index_list = [x for x in range(len(max_tree)) if x != 0]

            while index_list:
                index_list_position = random.randint(0, len(index_list) - 1)
                index = index_list[index_list_position]

                max_tree_temp = max_tree[:]
                max_tree_temp[index] -= 1

                for scenario in self:
                    for tree in scenario:
                        for i in range(len(tree.duplication_prefix)):
                            if max_tree_temp[i] - tree.duplication_prefix[i] < 0:
                                break
                        else:
                            break
                    else:
                        index_list.pop(index_list_position)
                        break
                else:
                    max_tree = max_tree_temp
            return max_tree, sum(max_tree)
```

Dodatek B

Przykładowe drzewo gatunków

(prot,(fung,((chlo,embr),(arth,((acoe,anne),(echi,(chon,(oste,(amph,(moll,((mamm,(aves,rept)),agna)))))))))

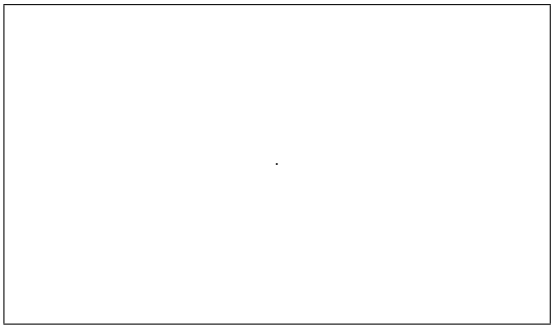


Rysunek B.1: Wizualizacja drzewa gatunków

Dodatek C

Przykładowe drzewa genów

(((amph,aves),mamm),chon)
    ((((acoe,mamm),chlo),fung),prot)
    (((((echi,arth),mamm),embr),fung),prot)



Rysunek C.1: Wizualizacja drzewa genów

Dodatek D

Przykładowy wynik działania programu (dla zbioru guigo)

```
————————FHS————————————
Data loaded. 0.0total random
([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 2, 3, 4], 18)
Done in 2.3286948204040527 .
start
([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 5], 5)
Done in 0.0676581859588623 .
end
([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 3], 4)
Done in 0.24111151695251465 .
index random
([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 3], 5)
Done in 0.0971217155456543 .
————————PG————————————-
Data loaded. 0.0total random
([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 3], 8)
Done in 0.18230891227722168 .
start
([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 3], 6)
Done in 0.006891012191772461 .
end
([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 3], 6)
Done in 0.006652355194091797 .
index random
([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 3], 6)
Done in 0.007452487945556641 .
```

Bibliografia

[1] Obraz wygenerowany za pomocą serwisu http://gsevol.azor.mimuw.edu.pl