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 podstawowe propozycje rozwiązań algorytmicznych dla problemów klastrowania duplikacji genomowych w oparciu o scenariusze ewolucyjne. W części pierwszej wprowadzane są pojęcia dotyczące drzew genów, gatunków, modeli ich uzgadniania oraz tworzenia scenariuszy ewolucyjnych. Omówiony został również problem przeliczania i klastrowania duplikacji genomowych w scenariuszach ewolucyjnych. 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 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 3 opisano implementację i sposób użycia programu napisanego na podstawie przybliżonej we wcześniejszym rozdziale 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 problemistyki klastrowania duplikacji genomowych.

1.1. Drzewa genów i gatunków

Opis drzewa pojawiający się w danych przez Pana pracach.

Związki między genami przedstawia się za pomocą ukorzenionego, binarnego drzewa, nazywanego drzewem genów, którego liśćmi są geny opisane przynależnością do danego gatunku.

Definicja 1.1.1 *Definicja matematyczna.*

Związki między gatunkami przedstawia się za pomocą ukorzenionego, binarnego drzewa, nazywanego drzewem gatunków, którego węzły wewnętrzne są specjacjami, a liście to gatunki.

Definicja 1.1.2 *Definicja matematyczna.*

1.2. Uzgodnienie drzewa

Podstawowy opis idei.

1.3. Modele scenariuszy ewolucyjnych

Podstawowy opis idei. Podział ruchów:

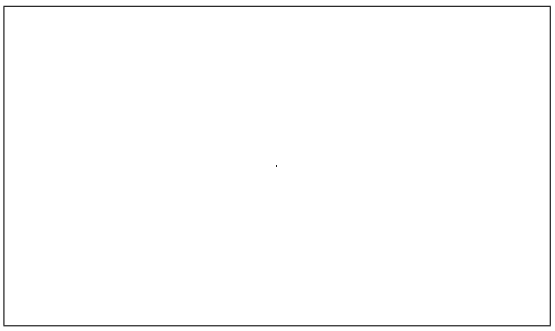
Definicja 1.3.1 *TMOVE.*

Definicja 1.3.2 *CLOST.*

Dozwolone modele:

1. PG
2. FHS;

1.4. Opis modelu ME



Rysunek 1.1: 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

Rozdział 3

Dokumentacja użytkowa i opis implementacji

Opis działania programu.

Rozdział 4

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.

4.1. 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,

4.2. Perspektywy wykorzystania

DOPYTAĆ

Dodatek A

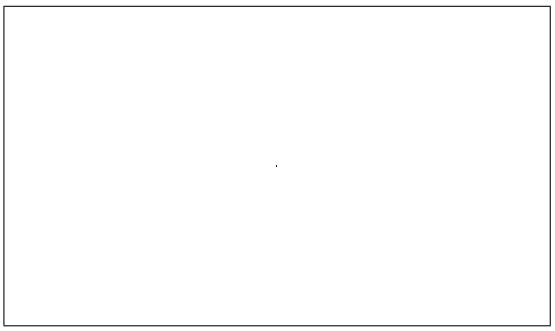
Pętla programu zapisana w języku Python wykonywana dla losowego wybierania indeksów

```
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)))))))))

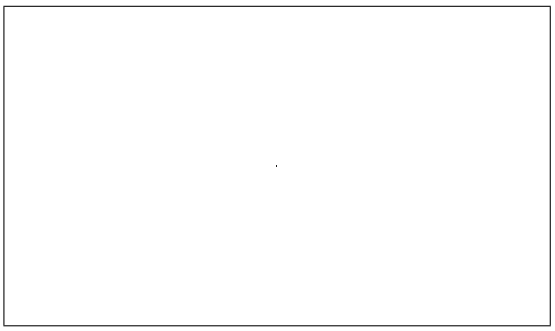


Rysunek B.1: Wizualizacja drzewa gatunków

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, 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, 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, 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, 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

[Zen69] Zenon Zenon, *Użyteczne heurystyki w analizie*, Młody Technik, nr 11, 1969.