Odtwarzanie historii ewolucyjnej sekwencji

Program zaliczeniowy nr 2 z przedmiotu *Algorytmy i Struktury Danych*

semestr letni 2017/2018

Wstęp

Historię ewolucji organizmów zazwyczaj reprezentuje się za pomocą *drzew filogenetycznych*, w których liście reprezentują gatunki współczesne, zaś wierzchołki wewnętrzne wspólnych przodków liści znajdujących się w zawieszonych w nich poddrzewach. W podobny sposób modeluje się ewolucję sekwencji poszczególnych genów bądź białek.

Zazwyczaj badacz dysponuje jedynie informacją na temat obiektów współczesnych i stara się na jej podstawie zrekonstruowac przodków i odtworzyć strukturę drzewa filogenetycznego. Często przyjmuje się przy tym, że im prostsza jest rozważana historia ewolucji, tym bardziej jest prawdopodobna. Do porównywania złożoności służy pojęcie *kosztu ewolucyjnego*. Za koszt ewolucji jednego obiektu w drugi można uznać minimalną liczbę potrzebnych do tego *elementarnych zdarzeń ewolucyjnych*, do których w przypadku sekwencji można zaliczyć następujące modyfikacje:

- *substytucja* – zmiana pojedynczego symbolu na inny,
- *delecja* – usunięcie pojedynczego symbolu,
- *insercja* – wstawienie pojedynczego symbolu.

Koszt ewolucyjny dla całego drzewa filogenetycznego to łączny koszt ewolucji wszystkich przodków w ich bezpośrednich potomków (wierzchołki odpowiadające takim parom przodek – potomek połączone są w drzewie krawędziami).

Zadanie

Należy napisać moduł *ImieNazwisko.py* zawierający implementację następujących obiektów:

- `Alignment(sequence1,sequence2)` – funkcja zwracająca parę:
  - minimalny koszt ewolucji `sequence1` w `sequence2`, czyli łączna liczba potrzebnych substytucji, insercji i delecji,
  - uliniowienie odpowiadające temu kosztowi, w postaci pary sekwencji otrzymanych z `sequence1` i `sequence2` przez wstawienie w wybrane miejsca symboli `_`.
- `BinTree`, `Leaf` i `BinNode` – klasy implementujące drzewa filogenetyczne dla sekwencji biologicznych oraz etykietowane sekwencjami ich liście i wierzchołki wewnętrzne.

  Klasa `Leaf` powinna zawierać następujące metody:
  - `__init__(sequence)` – utwórz liść z etykietą `sequence`
  - `is_leaf()` – zwróć `True`
  - `label()` – zwróć etykietę

  Klasa `BinNode` powinna zawierać następujące metody:
  - `__init__(left,right)` – utwórz wierzchołek wewnętrzny z synami `left` i `right`
  - `is_leaf()` – zwróć `False`
  - `son(which)` – zwróć lewego syna, gdy `which='L'`, a prawego, gdy `which='R'`
  - `set_label(sequence)` – nadaj wierzchołkowi etykietę `sequence`
  - `label()` – zwróć etykietę, jeśli wierzchołek ją posiada; w przeciwnym razie zwróć `None`

Klasa `BinTree` powinna zawierać następujące metody:

  - `__init__(node)` – utwórz drzewo z wierzchołkiem `node` w korzeniu
  - `root()` – zwróć korzeń drzewa
  - `history_cost(cost=Alignment)` – zwróć koszt ewolucyjny dla danego drzewa (lub `None`, jeśli nie wszystkie wierzchołki posiadają etykiety)
  - `reconstruct_ancestors(cost=Alignment)` – nadaj wewnętrznym wierzchołkom etykiety reprezentujące sekwencje przodków starając się zminimalizować koszt ewolucyjny drzewa

- `Reconstruct_History(sequences,cost=Alignment)` – funkcja zwracająca drzewo reprezentujące historię ewolucyjną o możliwie najmniejszym koszcie dla listy sekwencji współczesnych `sequences`.

Koszt ewolucyjny w metodach `history_cost` i `reconstruct_ancestors` oraz w funkcji `Reconstruct_History` powinien być liczony względem zadanej parametrem `cost` (domyślnie funkcja `Alignment`) funkcji obliczającej koszt ewolucji dla pary sekwencji i odpowiadające mu uliniowienie.

Pamięć potrzebna do analizy danych składających się z kilkunastu sekwencji o długości kilkadziesiąt symboli nie powinna prekraczać 0.5GB, a czas wykonania poszczególnych operacji na przeciętnym laptopie nie powinien znacząco przekraczać:

- 1 minuty dla `Reconstruct_History`,
- 5 sekund dla `reconstruct_ancestors`,
- 1 sekundy dla pozostałych operacji.

Rozwiązanie zadania powinno zawierać kod programu z komentarzami.

Ocena

Za pełne rozwiązanie można otrzymać 15 pkt., w tym:

5 pkt. implementacja funkcji `Alignment` oraz wszystkich metod klas `Leaf`, `BinNode` i `BinTree` za wyjątkiem `reconstruct_ancestors`

5 pkt. jakość drzew wyznaczanych przez metodę `reconstruct_ancestors`

5 pkt. jakość drzew wyznaczanych przez funkcję `Reconstruct_History`

Uwaga

Jakość rekonstruowanych drzew mierzona jest ich kosztem ewolucyjnym.

W przypadku metody `reconstruct_ancestors` 3 punkty można otrzymać za koszt zbliżony do otrzymanego w wyniku następującej strategii: sekwencję wierzchołka wewnętrznego wyznaczamy wykonując uliniowienie sekwencji jego synów, a następnie dla każdej kolumny w uliniowieniu wybieramy losowo symbol z pierwszego lub drugiego wiersza; na koniec z otrzymanej sekwencji usuwamy symbole `_`.

W przypadku funkcji `Reconstruct_History` 2 punkty można otrzymać za koszt zbliżony do otrzymanego w wyniku następującej strategii: generujemy losowe drzewo o liściach etykietowanych sekwencjami z zadanej listy, a następnie wykonujemy na nim metodę `reconstruct_ancestors`.