Opis algorytmu generującego consensusy

Założenia

Input

- *poagraph*
- *cutoff_search_range* - procentowy zakres compatibilities, wśród których będzie szukany próg odcięcia (krok 3)
- *multiplier* - mnożnik dla średniej odległości, która ma być przekroczona przy poszukiwaniu progu odcięcia (krok 7)
- *stop* - compatibility przy osiągnięciu którego nie rozdrabniać już węzła
- *re_consensus* - flaga, czy sprawdzać czy sekwencje już zakwalifikowane do węzła na obecnym poziomie drzewa nie powinny być zakwalifikowane do obecnie liczonego węzła

Output:

- lista consensusów
- drzewo, w którym każdy węzeł zawiera:
  - ID consensusu, któremu odpowiada ten węzeł
  - listę sekwencji, które zostały zakwalifikowane do tego consensusu
  - minimalne compatibility do consensusu wśród sekwencji w tym węźle

Konfiguracja używanego programu *poa*

poa -read_msa [po_file_path] -hb -po [hb_file_path] blosum80.mat -hbmin [hbmin]

- [hbmin] - zazwyczaj przujmuję niską wartość (np. 0.2), bo i tak samodzielnie obliczam compatibility aktualnie rozważanych sekwencji względem consensusu wyegenerowanego przez *poa* jako conensus0

Algorytm

Węzły są dzielone na coraz mniejsze węzły, tak długo, aż wszystkie sekwencje będą przydzielone do consensusu względem którego ich compatibility przekracza *stop*.

Dla pojedynczego węzła:

1. Uruchom *poa* na wszystkich sekwencjach w tym węźle, weź consensus0 jako **C**.
2. Policz compatibility **C** z wszystkimi sekwencjami w tym węźle.
3. Znajdź próg odcięcia **P1** wśród compatibilities policzonych w 2.:
   - weź compatibilities z przedziału *cutoff_search_range*
   - znajdź największą różnicę występującą pomiędzy dwoma kolejnymi compatibility **Ci**, **Cj**
   - **P1** = **Cj**
4. **max_sequences** - sekwencje, których compatibility do **C** przekracza **P1**
5. Uruchom *poa* na **max_sequences**, weź consensus0 jako **C_MAX** i dodaj go do poagraphu jako kolejny consensus.

6. Policz compatibility **C_MAX** z wszystkimi sekwencjami w tym węźle. Jeśli *re_consensus* == True to również compoatibilities do **C_MAX** dla sekwencji, które już zostały wpisane do rodzeństwa aktualnego węzła.
7. Znajdź próg odcięcia **P2** wśród posortowanych compatibilities policzonych w 6.:
   - policz średnią odległość między compatibilities
   - znajdź takie **Ci**, **Cj** pomiędzy odległość jest większa niż średnia odległość * *multiplier*
   - **P2** = **Cj**
8. **node_sequences** - sekwencje, których compatibility do **C_MAX** (policzone w 6.) przekracza **P2**
9. Parametry uzyskanego węzła:
   - ID consensusu **C_MAX**
   - sekwencje **node_sequences**
   - minimalne compatibility wśród compatibilities **node_sequences** do **C_MAX**