Symulowane wyżarzanie na mikroskopowych danych o chromatynie

Cezary Rybak, Jakub Sawicki, Jakub Seliga, Rafał Pyzowski

1. Wstęp

Cały kod był pisany w języku python z użyciem bibliotek: copy, math, random, numpy, networkx, scipy, pandas, matplotlib oraz seaborn.

1. Inicjalizacja grafu

W celu późniejszego utworzenia struktury, pierwszym krokiem była inicjalizacja grafu, przy wykorzystaniu pakietu networkx. Został on utworzony na podstawie dwóch plików z informacjami o krawędziach  
i wierzchołkach. Do każdej krawędzi w grafie zostały również dodane dwie dodatkowe informacje: waga oraz dystans.

1. Implementacja funkcji g

Funkcja g jako argumenty przyjmuje strukturę oraz graf G. Strukturą jest ścieżka w danym grafie. Jest ona przedstawiona jako lista kolejnych indeksów wierzchołków. Celem funkcji jest rozszerzanie otrzymanej struktury. Zwraca ona nową strukturę, rozszerzoną maksymalnie o jeden punkt (możliwe jest również, że struktura nie zostanie poszerzona,  
w skrajnych przypadkach może się zmniejszyć o jeden wierzchołek).

Ścieżka jest rozszerzana za pomocą jednej z dwóch funkcji pomocniczych.

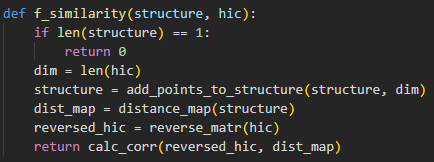
*add\_to\_end* - losuje jeden z wierzchołków będących końcem ścieżki, następnie wyszukuje jego sąsiadów, których jeszcze nie ma w ścieżce.  
W przypadku, gdy nie znaleziono żadnego takiego punktu, wybrany skrajny wierzchołek jest usuwany ze ścieżki i wykonywana jest funkcja *add\_in\_the\_middle*. W przeciwnym przypadku, kolejnym krokiem jest wylosowanie nowego wierzchołka, przy czym prawdopodobieństwa są przydzielane na podstawie wag odpowiednich krawędzi. Po losowaniu, nowy wierzchołek jest przypisywany do odpowiedniego miejsca w strukturze (w zależności od wybranego końca).

*add\_in\_the\_middle* - losuje dowolny punkt w strukturze i wybiera jego następnik w strukturze. Następnie, dysponując już dwoma sąsiednimi punktami, wyszukiwani są wszyscy wspólni sąsiedzi tych punktów w grafie, którzy nie występują w strukturze. Jeśli nie znaleziono żadnych takich wierzchołków, funkcja nic nie robi. W przeciwnym przypadku, spośród znalezionych punktów, losowany jest jeden, na podstawie wag odpowiednich krawędzi (iloczyn wag krawędzi łączących nowy punkt  
z wybranymi wierzchołkami w strukturze). Następnie nowy wierzchołek jest wstawiany w odpowiednie miejsce w strukturze, czyli pomiędzy dwoma wybranymi punktami.

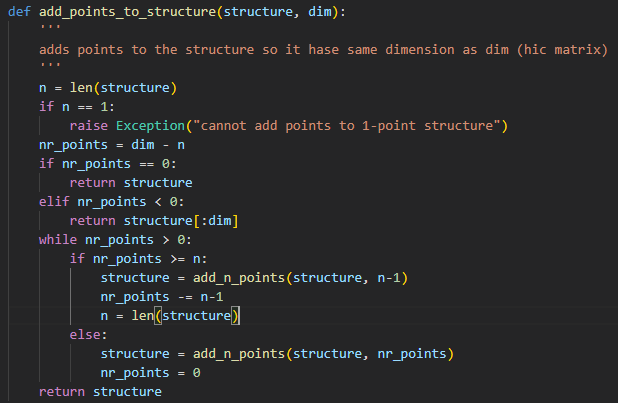
Gdy struktura ma mniej niż 10 elementów, wykonywana jest funkcja *add\_to\_end*. Gdy jest większa, losowana jest jedna z powyższych funkcji (z równym prawdopodobieństwem), która zostanie użyta. Po jej wykonaniu, zwracana jest nowa struktura.

Do późniejszych operacji konieczne jest przemapowanie struktury, w postaci indeksów wierzchołków, na trzy-elementowe listy, opisujące położenie danych punktów w przestrzeni trójwymiarowej. Realizowane jest to przez funkcję *get\_structure\_coordinates*. Przyjmuje ona strukturę oraz graf graf G, jako argumenty. Zwraca listę złożoną z wyżej wymienionych trzy-elementowych list.

1. Implementacja funkcji liczącej podobieństwo struktury do mapy hi-c



Ta funkcja przyjmuje jako argumenty strukturę punktów utworzoną za pomocą wcześniej opisanej funkcji oraz macierz mapy hi-c. Na początku oblicza rozmiar macierzy hi-c i zmienia rozmiar struktury, tak aby ilość punktów zgadzała się z rozmiarem tej macierzy. Robi to za pomocą funkcji *add\_points\_to\_structure*:



Ta funkcja, w zależności od potrzeby, dodaje bądź usuwa punkty ze struktury. Na początku oblicza różnicę aktualnej ilość punktów w strukturze i oczekiwanej ilości (dim). Jeśli ta wartość jest równa 0,  
to zwraca posiadaną strukturę, jeśli zaś ujemna, to zmniejsza posiadaną strukturę do oczekiwanego rozmiaru. W sytuacji, w której punkty należy dodać, liczy ich ilość i wykonuje jeden z dwóch kroków:

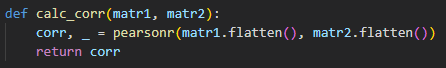
1. Jeśli liczba punktów do dodania jest mniejsza niż rozmiar struktury, to dodaje je równomiernie w miejsca środków odcinków
2. Jeśli liczba punktów do dodania jest większa lub równa rozmiarowi struktury, to tworzy nowy punkt w środku każdego odcinka  
   i powtarza algorytm aż do sukcesu.

Po dodaniu koniecznej ilości punktów do struktury tworzona jest mapa dystansów za pomocą funkcji *distance\_map* zaimportowanej z modułu *scipy.spatial*.

Wartości w mapie hi-c możemy uznać, że są odwrotnie proporcjonalne  
do odległości między odpowiednimi punktami, więc należy odwrócić w niej wartości. W tym celu została napisana funkcja *reverse\_matrix*:



Po odwróceniu wartości w macierzy hi-c można obliczyć korelację pearsona między macierzą hi-c, a macierzą dystansów struktury. Jest  
to wykonywane za pomocą funkcji *calc\_corr*, która wykorzystuje funkcję *pearsonr* zaimportowaną z moduły *scipy.stats*



Obliczona korelacja jest zwracana na końcu funkcji.

1. Implementacja algorytmu symulowanego wyżarzania

Za pomocą funkcji simulated\_annealing, przeprowadzony zostaje algorytm symulowanego wyżarzania. Funkcja za swoje argumenty przyjmuje trzy obiekty: liczbę całkowitą, stanowiącą liczbę iteracji algorytmu, graf G będący obiektem *graph* z biblioteki networkx  
oraz macierz hi-c. W ramach działania funkcji, na początku inicjalizowana zostaje pusta struktura za pomocą funkcji g z argumentami *None* oraz *G*. To do niej będą dodawane kolejne punkty w trakcie działania algorytmu. Przyjęta zostaje temperatura początkowa T\_0, której wartość została ustalona domyślnie na 1. Zainicjalizowana zostaje także pusta tablica, do której w trakcie działania algorytmu będą wpisywane korelacje obecnej struktury z właściwą macierzą hi-c - trzecim argumentem funkcji.

Następnie, przeprowadzone zostaje tyle kroków iteracyjnych, ile podano na wejściu funkcji. W ramach każdego z tych kroków przeprowadzone zostają następujące obliczenia: obecna temperatura, oznaczana jako zmienna T, zostaje obliczona ze wzoru T\_0 \* (1-i/iterations), gdzie i to obecny numer iteracji, zaczynając od liczby 1. Następnie następuje zaproponowanie nowej, rozszerzonej postaci struktury: od obiektu *new\_structure*, będącego kopią obecnie rozważanej struktury oraz grafu *G* wywołana zostaje funkcja *proponuj\_g*. Jeśli nowa struktura jest tożsama z obecną, pomijamy kolejne kroki wewnątrz obecnej iteracji i przechodzimy do następnej iteracji. Gdy natomiast uzyskaliśmy nową, rozszerzoną strukturę, obliczamy prawdopodobieństwo tego, że przyjmiemy ją jako obecną postać struktury. To prawdopodobieństwo p zostaje obliczone ze wzoru o postaci e^(-d/T), gdzie d to różnica podobieństw nowej i obecnej struktury z macierzą hi-c, obliczonych za pomocą funkcji *f\_similarity*, a T to obecna temperatura. Jeśli otrzymana liczba jest większa od 1, przyjmujemy zamiast tego prawdopodobieństwo równe 1. Następnie, za pomocą funkcji random() z biblioteki random wylosowana zostaje liczba z przedziału (0,1): jeżeli jest ona mniejsza niż wyliczona w poprzednim kroku liczba p, jako obecną strukturę przyjmujemy tą nowo zaproponowaną w tym kroku iteracyjnym. Na koniec każdego kroku iteracyjnego, dodajemy do zainicjalizowanej na początku tablicy korelację struktury z macierzą hi-c, co umożliwi przedstawienie zmiany tej korelacji względem ilości iteracji.

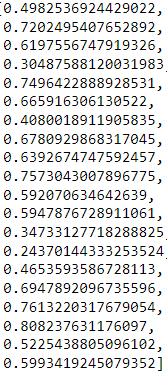
1. Wyniki

Wygenerowaliśmy ścieżki symulowanego wyżarzania i eksportowaliśmy je do plików .pdb. Te ścieżki reprezentują potencjalne konfiguracje chromatyny, które mogą wystąpić w komórce.

Następnie, za pomocą oprogramowania ChimeraX, porównaliśmy te ścieżki z modelem chromatyny .cmap. ChimeraX pozwolił nam na wizualizację i porównanie naszych symulowanych struktur  
z rzeczywistym modelem chromatyny.

Zauważyliśmy, że niektóre z naszych symulowanych ścieżek pokrywały się z modelem chromatyny do różnego stopnia. Część ścieżek zajmowała około połowy objętości modelu, podczas gdy inne pokrywały prawie cały model. To sugeruje, że nasz algorytm symulowanego wyżarzania jest  
w stanie generować realistyczne konfiguracje chromatyny, choć z różnym stopniem dokładności. Być może dlatego, że podczas generowania ścieżki nie została utworzona możliwość cofania się i ścieżka może “utykać” kiedy trafi na miejsce, gdzie nie będzie mogła się już dalej rozrastać.

W celu głębszego zbadania skuteczności działania zaimplementowanego przez nas symulowanego wyżarzania przeprowadziliśmy 20 symulacji,  
a następnie wyznaczyliśmy funkcją f korelacje między otrzymaną ścieżką, a mapą HiC. Oto wyniki



Prawdopodobnie niższe wyniki były otrzymane w przypadku, kiedy struktura rozrosła się jedynie na pół komórki, a te wyższe, gdy zajęła prawie całą.