UNIVERSIDAD DE COSTA RICA

FACULTAD DE INGENIERÍA

ESCUELA DE CIENCIAS DE LA COMPUTACIÓN E INFORMÁTICA

CI-0124 Computabilidad y Complejidad

Grupo 2

Profesora Mauren Murillo Rivera

Tarea 2 - Primer Avance - Propuesta de proyecto

Grupo ZzZ:

Nathalie Alfaro Quesada B90221

José Pablo Mora Cubillo B75044

II Semestre 2024

Descripción e importancia del problema:

Las aneuploidías son alteraciones cromosómicas que se caracterizan por la ganancia o pérdida de cromosomas completos o segmentos cromosómicos. Este tipo de alteraciones son muy frecuentes en cáncer, hasta el 88% de las muestras del proyecto "The Cancer Genome Atlas" exhiben algún grado de aneuploidía. En la actualidad, se cuenta con tecnologías que permiten medir el número de copias (CN) a nivel de gen, en condiciones normales este valor debería ser de 2.

Se cuenta con un conjunto de datos que contiene datos de CN de 718 genes implicados en cáncer para 1398 líneas celulares cancerígenas. En este proyecto se busca agrupar estas líneas celulares en grupos de acuerdo a similitudes en los valores de CN, para lo que se va a utilizar algoritmos de agrupamiento.

La importancia de lo anterior radica en que se ha observado que los distintos patrones de CN tienen cierto grado de relación con el tipo de cáncer, tejido de origen, estadio y presiones selectivas a las cuales ha estado sujeto el tumor. Debido a esto, se puede hipotetizar que las líneas celulares con patrones parecidos de CN tienen comportamientos similares, por ejemplo, sensibilidad al mismo tipo de fármacos.

Ejemplo de los datos:

symbol model_id	ABCB1	ABI1	ABL1	ABL2	ACKR3	ACSL3	ACVR1	ACVR2A	AFDN	AFF1	...	ZNF208	ZNF331	ZNF384	ZNF429	ZNF521	ZNF626
SIDM00001	3.0000	2.0000	3.0000	2.0000	1.903779	2.0000	2.0000	2.5000	1.0000	2.0000	...	1.5000	1.0000	2.0000	1.5000	2.0000	1.5000
SIDM00002	3.0000	3.0000	3.0000	3.0000	3.000000	3.0000	3.0000	3.0000	2.0000	3.0000	...	3.0000	3.0000	3.0000	3.0000	3.0000	3.0000
SIDM00003	4.0000	3.0000	3.0000	3.0000	3.000000	3.0000	3.0000	3.0000	2.0000	2.0000	...	3.0000	3.0000	3.0000	3.0000	3.0000	3.0000
SIDM00008	5.0000	3.0000	2.0000	4.0000	4.000000	4.0000	4.0000	5.0000	3.0000	3.0000	...	3.0000	3.0000	3.0000	3.0000	4.0000	3.0000
SIDM00011	7.0000	3.0000	3.0000	4.0000	4.000000	4.0000	4.0000	4.0000	3.0000	3.0000	...	2.0000	3.0000	3.0000	2.0000	3.0000	2.0000
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
SIDM02076	3.8307	2.0521	4.2114	3.2612	3.079400	3.0782	3.1249	3.1249	2.9867	3.1873	...	3.3713	3.7299	3.8227	3.3713	3.6534	3.3713
SIDM02077	2.9821	2.4088	2.1983	1.6317	1.636200	1.6362	1.6053	0.9731	1.6669	1.0377	...	0.9191	1.9538	2.0136	1.8329	1.9477	1.9803
SIDM02078	2.9376	1.9788	2.8560	2.0221	2.882000	2.8820	2.8694	2.8694	2.0289	2.9833	...	1.9748	3.7649	4.9985	1.9748	1.0058	1.9748
SIDM02079	2.9775	2.7970	3.8654	3.1729	1.998400	4.0418	1.9879	1.9879	2.0014	1.1020	...	1.9649	3.8902	3.6761	1.9649	3.0050	1.9649
SIDM02080	1.8506	1.9958	2.0053	2.1766	2.071200	2.0117	2.0267	2.0267	2.0100	1.0082	...	1.9783	2.0057	1.0433	1.9783	3.8952	1.9783

1398 rows × 718 columns

Algoritmos de agrupamientos:

El agrupamiento, también conocido como clustering o análisis de grupo, es una técnica utilizada en el análisis de datos para agrupar un conjunto de objetos en grupos (o clusters) de manera que los objetos dentro de un mismo grupo sean más similares entre sí que con los objetos de otros grupos. Los algoritmos enfocados en resolver este problema reciben el nombre de algoritmos de agrupamiento. Los problemas de agrupamiento son considerados *NP-Hard*, pero existen heurísticas y metaheurísticas a partir de las cuales se pueden obtener buenos resultados empleando una cantidad de recursos computacionales limitada.

Heurística:

Como heurística se utilizará el algoritmo *k-means* (también llamado algoritmo de Lloyd), el cual es un algoritmo iterativo, específico para problemas de agrupamiento y que se puede quedar atrapado en óptimos locales (versión clásica del algoritmo). Generalmente el algoritmo recibe por parte del usuario los datos y un valor *k* que representa la cantidad de grupos a generar. A partir de lo anterior, se generan en una primera iteración *k* centroides aleatorios y cada entrada de los datos se asigna al *cluster* cuyo centroide se encuentra más cerca, para esto se emplea una métrica de distancia.

El proceso de asignación de un dato a un *cluster* es fundamental y lo utilizaremos tanto en *k-means* como en el algoritmo genético que se va a emplear como metaheurística. Existen múltiples métricas de distancia, pero la más común es la distancia euclidiana, esta consiste en:

$$d = \sqrt{(X1 - Y1)^2 + (X2 - Y2)^2 + \ldots + (Xn - Yn)^2}$$

Donde *Xi* corresponde al valor del centroide en la dimensión *i* y *Yi* corresponde al valor de la entrada en la dimensión *i*. Si tenemos *K = 4* y 3 dimensiones (a veces también llamadas *features*), los cluster van a tener la siguiente forma:

C1 = (X1, Y1, Z1); C2 = (X2, Y2, Z2); C3 = (X3, Y3, Z3); C4 = (X4, Y4, Z4)

Con base en lo anterior, *K-means* consiste en los siguientes pasos:

1) Inicializar los centroides, ya sea aleatoriamente o seleccionando puntos al azar del conjunto de datos.
2) Asignar cada punto al cluster cuyo centroide esté más cercano, utilizando la distancia euclidiana.
3) Recalcular los centroides de los clusters, donde el nuevo centroide estará formado por los promedios por dimensión de cada entrada presente en el cluster.
4) Repetir los pasos de (2) y (3) hasta que las asignaciones de los puntos de datos ya no cambien significativamente.

Metaheurística:

Como metaheurística utilizaremos los algoritmos genéticos, basándonos en las descripción de Maulik & Bandyopadhyay (2000) para *clustering* con algoritmos genéticos. Por practicidad, al igual que en *K-means*, se asume que el usuario va a brindar la cantidad *k* de *clusters* en los que se deben agrupar los datos. El proceso de asignación se realiza empleando una distancia euclidiana de la misma manera que se realiza en *k-means*.

En relación a lo visto en clases para algoritmos genéticos, se debe puntualizar que el espacio de búsqueda consiste en todos los posibles valores que van a tomar los *k* centroides. Por lo tanto, el cromosoma va a estar formado por estos mismos valores. Por ejemplo, en el caso de *k = 4* y 3 dimensiones, un posible cromosoma sería:

[(X1, Y1, Z1), (X2, Y2, Z2), (X3, Y3, Z3), (X4, Y4, Z4)]

De manera que para la mezcla de la etapa de cruce, se pueden intercambiar *clusters* y/o valores dentro de los clusters.

Para calcular el valor de *fitness* de cada cromosoma, se puede escoger una función objetivo que maximice la diferencia entre los *clusters* o una que minimice la diferencia entre las entradas pertenecientes a cada grupo. En nuestro caso,

utilizaremos esta última, la cual va a consistir el inverso de la sumatoria de las distancias euclidianas para cada muestra dentro de cada *cluster*:

$$Fitness = \frac{1}{\sum_{i=1}^{Nk} \sum_{j=1}^{Ei} dij}$$

Donde *Nk* es el número de *clusters, Ei* es el número de entradas en el cluster *i* y *dkj* es la distancia euclidiana de la entrada *j* en el cluster *i* con el centroide *i.*

Fuerza bruta:

A partir de lo anterior, existen dos posibilidades de fuerza bruta. (1) probar todos los valores que pueden tomar los centroides. (2) Probar todas las posibles combinaciones que se pueden realizar entre todos los datos y los *k* clusters.

REFERENCIA:

Maulik, U., & Bandyopadhyay, S. (2000). Genetic algorithm-based clustering technique. Pattern recognition, 33(9), 1455-1465.