Escuela de Computación
Bachillerato en Ingeniería en Computación

TEC | Tecnológico de Costa Rica

Proyecto #2

Ensamblaje de Fragmentos

IC-8050 Introducción a la Biología Molecular Computacional

Tecnológico de Costa Rica, Campus Central Cartago

Escuela de Ing. en Computación, Ingeniería en Computación

I Semestre 2025

Prof. M.Sc. Esteban Arias Méndez

Un programa genera hileras aleatorias de ADN con ciertas características controladas y un programa que fragmentará un archivo de texto en forma aleatoria, con ciertos parámetros definidos por el usuario, para luego reconstruir el original.

DESARROLLO

A. Generador de Fragmentos (shotgun)

Este es un programa que debe tomar un archivo (que podría ser de ADN o un texto arbitrario) y generará fragmentos aleatorios siguiendo las especificaciones del usuario.

Al archivo de entrada se le podrá aplicar la inserción de errores en los fragmentos de acuerdo con los parámetros que brinde el usuario, para simular una toma de muestra de laboratorio. El usuario seleccionará los tipos de errores que desea y los porcentajes de probabilidad en cada caso usando como base los parámetros iniciales dados. Todos los fragmentos se grabarán en un archivo de texto plano, donde cada línea de texto será un fragmento, hasta el final del archivo.

Opciones generales

El usuario indicará cuantos fragmentos desea obtener (en realidad este es un número mínimo de fragmentos). Esto lo podrá hacer pidiendo explícitamente la cantidad de fragmentos o indirectamente estableciendo una cobertura promedio deseada. También indicará la longitud promedio de los fragmentos y la desviación estándar de los mismos. Para este proyecto basta con seguir una distribución normal de las longitudes de los fragmentos, pero *opcionalmente* se podrían ofrecer otras distribuciones (distribución uniforme, por ejemplo).

El usuario tendrá la opción de solicitar que los fragmentos **cubran totalmente** a la hilera base, *i.e.* que se garantice que para toda posición *i* de la hilera original hay al menos un fragmento que la contiene. Si no se pide esta opción, el programa no tiene la obligación de garantizar este requisito.

Opciones adicionales

El usuario podrá establecer algunos parámetros adicionales al programa:

- **Errores**: el usuario indicará cuales posibles errores de base podrían aparecer en cada fragmento (cambio de base o letra, borrado o inserciones) y para cada uno de ellos establece la distancia promedio entre errores. Cada tipo de error se parametriza de forma independiente. Si no se indica ninguno de estos errores o se indican en 0, significa que los fragmentos no serán alterados.
- **Orientación inversa**: con cierta probabilidad que el usuario establece el fragmento podría invertirse. Si esta probabilidad es 0, la colección completa de fragmentos mostrará siempre la misma orientación.
- **Quimeras**: este es un porcentaje que indica cuantos de los fragmentos generados serán quimeras (concatenación arbitraria de 2 o más fragmentos, los cuales a su vez podrán tener o no errores u orientaciones arbitrarias). El usuario indicará este porcentaje (de ser 0 no habrá quimeras) y la cantidad máxima de fragmentos a ser concatenados.

Archivo Descriptivo y Archivo de Fragmentos

Cuando el usuario solicite que los fragmentos sean generados, se crearán dos archivos: uno con los fragmentos propiamente y otro archivo con información descriptiva de la colección. Ambos archivos podrían tener el mismo nombre con extensiones o sufijos diferentes. Cada pareja puede definir el formato del archivo descriptivo, pero en todo caso deben documentarlo y buscar la mayor simplicidad posible. Por ejemplo, podría ser un archivo de texto plano o un XML. Entre la información a guardar e indicar en el archivo descriptivo está: la cantidad de fragmentos, longitud promedio, desviación estándar, cobertura promedio, cobertura total, tipos de errores con sus probabilidades, presencia de quimeras y sus valores dados, orientación, etc. Este archivo debe planearse para permitir la "reejecución" del mismo como valores de entrada para generar una nueva colección de fragmentos con características equivalentes, sin necesidad de volver a ingresar todos los datos manualmente.

Modo Batch (¡Opcional, extra!)

El programa generador ofrecerá un modo batch donde se generarán en forma no interactiva tantas colecciones de fragmentos viniendo del mismo archivo base como el usuario indique, siguiendo los parámetros que se le den. Estos parámetros se dan en forma interactiva o cargándolos de un archivo descriptivo creado en una sesión previa. Todos estos archivos tendrán un nombre genérico indicado por el usuario como un parámetro adicional, seguido de un número consecutivo.

B. Ensamblaje de Fragmentos

Su programa debe cargar un archivo de fragmentos, ya sea un archivo generado por ustedes o bien un archivo de fragmentos dado (del cual no se conoce el archivo origen) solo los fragmentos. El formato de este archivo, como se indicó antes, será un archivo de texto plano, donde cada línea de texto será un fragmento, hasta el final del archivo. Si embargo se debe plantear cómo se deben almacenar los caracteres tipo "cambio de línea" para el caso de archivos de fragmentos de textos genéricos. Se debe coordinar entre todos los miembros del grupo para definir un mismo formato de entrada para todos.

Cuando el usuario lo solicite, su programa deberá listar los fragmentos cargados donde debe ser posible para el usuario ordenarlos y filtrarlos de múltiples formas: orden alfabético, largo de hilera, ordenar fragmentos mayores o menores a un valor dado, buscar palabras claves, etc.

Utilizando el algoritmo de ensamblaje de grafo de traslapes para la obtener la *Superhilera Mínima Común*, tomar los fragmentos dados y procesarlos para generar el grafo completo con todos los pesos de traslapes de todos los fragmentos dados.

El usuario debe poder visualizar dicho grafo y explorarlo (navegando por él o visualizando todas sus partes: fragmentos y valores de traslapes (pesos) y la dirección de los arcos que indican el sufijo-prefijo correspondientes al traslape.

El usuario podría solicitar un grafo simplificado indicando el valor de traslape mínimo deseado. En este caso el programa deberá informar al usuario si el grafo es conexo o no.

Del grafo original su programa deberá generar el grafo conexo mínimo, es decir el que selecciona los mayores traslapes hasta que el grafo sea conexo. Adicionalmente el usuario podrá indicar un valor de traslape mínimo.

El usuario seleccionará de entre los posibles grafos anteriores, desde el cual su programa deberá generar a partir de éste y por orden de traslapes (greedy) la hilera de salida como ensamblaje de los fragmentos dados.

Se deberá informar al usuario si el ensamblaje produjo o no "islas" de hileras, en el caso que el grafo de trabajo fuera no conexo por ejemplo o bien, no todas las hileras lograron la cobertura esperada.

Con este ensamblaje se debe reconstruir la hilera original siguiendo el algoritmo. Si tiene la hilera o archivo original deberá hacer una comparación de similitud entre la hilera original y la hilera reconstruida para su evaluación. Adicionalmente si no tenía los datos del archivo original su programa podría solicitarlo para validar su efectividad.

Análisis y Pruebas

Documente y brinde múltiples ejecuciones (experimentos) del programa para valorar su efectividad, proveyendo cambios en los parámetros usados para los fragmentos, para medir su valor de convergencia de acuerdo con los cambios introducidos en los fragmentos.

Parámetros

Al iniciar, su programa deberá proveer valores iniciales o por default para todos los parámetros requeridos en las operaciones, pero el usuario debe poder modificarlos según sus necesidades.

Debe recibir los valores de probabilidad para cada uno de los 5 posibles errores: sustitución, inserción, deleción, quimeras, inversión. En caso de ser 0 significa que ese error no sucede.

Además, debe parametrizar el valor de % de cobertura deseado y el rango de valores de traslape mínimo a máximo, donde el valor dado entre estos límites puede ser aleatorio.

GENERALIDADES

1. Este trabajo es para ser desarrollado en parejas de 2 personas máximo.
2. En cualquiera de los casos se debe incluir un reporte de las actividades desarrolladas por cada miembro, dentro de su documentación.
3. Documente detalladamente su código.
4. Puede hacer uso de código de terceros siempre y cuando esté disponible para su uso legal y libre (open-source), documente su origen y los cambios introducidos para ajustarlos a sus necesidades. Indique con comentarios las fuentes de código abierto de las que haga uso dentro de sus programas, de lo contrario se considerará plagio.
5. De preferencia trabaje sobre el Sistema Operativo Linux. Se recomienda trabajar su código en módulos, incluso en varios archivos para facilitar el trabajo.
6. Puede trabajar en los lenguajes y herramientas de programación que prefiera.
7. Use la siguiente invitación de Git para crear su repositorio de trabajo: https://classroom.github.com/a/8C6UMfRj
8. Todos los programas deben ser parametrizables. Es decir, todos los valores que sean requeridos para los cálculos, tales como valores de comparación, tamaños, etc. deben ser parametrizables sin requerir que el programa deba ser recompilado o su código alterado para cambiar valores.
9. Está prohibido el uso de bibliotecas o código *open-source* que implementen de forma directa el trabajo solicitado para este Proyecto.
10. Ninguna estructura de datos asociada debería estar definida estáticamente, sino que serán creadas y destruidas en el momento que se necesiten.
11. Cualquier resultado debe incluir información para el usuario sobre los requerimientos de memoria y de tiempo de CPU que se usaron para atender la consulta del usuario.
12. El programa debe manejar correctamente entradas de cualquier tamaño. Por supuesto, si se exceden las capacidades de la máquina en particular, el usuario será informado de esta situación sin que el programa colapse.
13. En caso de requerir máquinas con Linux, se han dispuesto máquinas en los servidores de la Escuela de Computación, sobre la red de la Escuela. Los accesos a estos equipos serán remotos, usando el servicio gratuito VPN de la Escuela y mediante protocolos SSH u otros que. En caso de requerir accesar alguna deberá solicitarlo por correo o Telegram al profesor.
14. Puede hacer uso de los utilitarios programados previamente como Tarea, documente el uso de estos, y si debió hacerles cambios o ajustes.
15. El proyecto se calificará con los siguientes criterios:
    i. 80% - programas solicitados, interactividad, uso de parámetros, funciones de probabilidad usadas, generación de archivos descriptivos para su reejecución, exploración de los datos obtenidos mediante el grafo, reporte de errores introducidos, etc.
    ii. 20% - Documentación completa del trabajo.
    iii. 10% - Extra: modo Batch.
16. El proyecto debe resolverse, implementándolo de la mejor manera. Para optar a los puntos extras, todas las partes previas solicitadas deben estar completas y funcionales.
17. Se revisará el código del proyecto, para asegurar que se cumpla con lo solicitado de forma interna y no solo el comportamiento del programa sea como el solicitado. Se hará verificación de copias o similitud de código.

18. De forma global, se evaluará la presentación del trabajo según los parámetros solicitados, estrategias empleadas y la calidad, la entrega a tiempo del trabajo y la documentación completa correspondiente.
19. Para la documentación siga las instrucciones dadas en la presentación inicial del curso para entrega de Tareas y Proyectos.
    a. Portada completa con abstract en inglés.
    b. La introducción del documento es una descripción breve del trabajo realizado y herramientas usadas.
    c. Como marco teórico incluya las descripciones de los algoritmos implementados y las herramientas empleadas.
    d. Como desarrollo debe explicar, los procedimientos, rutinas, y métodos que utilizo para resolver el problema.
    e. Indicar los ejemplos de código que ha usado como guía para el desarrollo de estos usando las referencias bibliográficas correspondientes.
    f. Explicar el uso y funcionamiento de su proyecto.
    g. Para el análisis de resultados, explique el trabajo implementado, la forma de funcionamiento general, ejemplos documentados, problemas presentados, estructuras de datos empleadas, algoritmos usados, etc.
    h. Conclusiones, Observaciones y Recomendaciones sobre el proyecto.
    i. En una sección de Apéndices incluya el código fuente documentado del proyecto y su explicación. Explique la estructura del código empleada en su proyecto, organización de módulos, etc.
20. Al finalizar el proyecto, para su entrega, deberá preparar una carpeta con el nombre de ambos participantes, donde deberá incluir sus archivos fuentes y archivos requeridos. No incluya archivos ejecutables. Prepare un archivo comprimido .tgz de esta carpeta. NO incluya acá el archivo PDF de su documentación.
21. La entrega será mediante la plataforma Google Classroom, deberán entregar el PDF de su documentación y el archivo .tgz; en 2 archivos separados.
22. Para los nombres de los archivos use el formato:

BMC_Proyecto2_SusNombresCompletos

23. Atienda la fecha de entrega indicada en el sistema Google Classroom. Cada día de atraso serán 15pts menos de la nota final.
24. Si el trabajo no es entregado antes de la Fecha y Hora de revisión, el proyecto no será recibido, ya que se deben entregar las notas finales del curso.
25. Cualquier consulta puede hacerla mediante correo o Telegram.
26. Las revisiones serán mediante un demo de forma remota. Durante la revisión del proyecto deben estar presentes los miembros del trabajo, la no presentación les restará 10pts a cada uno de los ausentes.