Tarea 4: Predicción de Estructuras Biológicas

José Benavente, Héctor Ayala

1. Predicciones de Estructura en el Área Biológica

La predicción de estructuras es fundamental en biología para entender la función de diversas macromoléculas. Existen varios tipos de predicciones de estructura biológica:

1. **Predicción de estructura de proteínas**: Determina la conformación tridimensional de una proteína a partir de su secuencia de aminoácidos.
2. **Predicción de estructura de ARN**: Identifica la conformación tridimensional de moléculas de ARN, que juegan roles críticos en la regulación génica y procesos celulares.
3. **Predicción de estructura de ADN**: Modela la estructura tridimensional del ADN, incluyendo conformaciones no canónicas como cuádruplex-G o estructuras cruciformes.
4. **Predicción de interacciones proteína-proteína**: Determina cómo interactúan diferentes proteínas entre sí.
5. **Predicción de estructura de complejos macromoleculares**: Predice la estructura de grandes ensamblajes de proteínas, ácidos nucleicos y otras biomoléculas.

2. Aplicaciones para Predicción de Estructura y sus Lenguajes

1. **AlphaFold2** (DeepMind)
   - *Lenguaje*: Python, JAX
   - *Descripción*: Sistema de IA para predecir estructuras de proteínas con precisión atómica
2. **RoseTTAFold** (Baker Lab)
   - *Lenguaje*: Python, PyTorch
   - *Descripción*: Método de aprendizaje profundo para predicción rápida y precisa de estructuras proteicas
3. **I-TASSER** (Universidad de Michigan)
   - *Lenguaje*: C++, Python
   - *Descripción*: Plataforma jerárquica para predicción de estructura y función de proteínas
4. **Phyre2** (Imperial College London)
   - *Lenguaje*: C++, JavaScript
   - *Descripción*: Servidor web para predicción y análisis de estructuras proteicas
5. **SWISS-MODEL** (SIB Swiss Institute of Bioinformatics)
   - *Lenguaje*: Python, C++
   - *Descripción*: Servicio automatizado de modelado por homología de estructuras proteicas
6. **ESMFold** (Meta AI)
   - *Lenguaje*: Python, PyTorch
   - *Descripción*: Modelo de predicción de estructura basado en transformers pre-entrenados
7. **MODELLER** (UCSF)
   - *Lenguaje*: Python, Fortran
   - *Descripción*: Software para modelado comparativo de estructuras 3D de proteínas
8. **trRosetta** (Universidad de Washington)
   - *Lenguaje*: Python, TensorFlow
   - *Descripción*: Método basado en redes neuronales para predicción de estructura protéica
9. **ColabFold** (Steinegger Lab)
   - *Lenguaje*: Python
   - *Descripción*: Implementación accesible de AlphaFold2 y RoseTTAFold en Google Colab
10. **OpenFold** (Columbia University)
    - *Lenguaje*: Python, PyTorch
    - *Descripción*: Reimplementación de código abierto de AlphaFold2

3. DeepMind y su Relación con la Predicción de Estructuras

DeepMind es una compañía de investigación en inteligencia artificial fundada en 2010 y adquirida por Google en 2014.

- **Enfoque general de DeepMind**:
  - Desarrolla sistemas de IA que pueden aprender a resolver problemas complejos sin instrucciones específicas.
  - Utiliza técnicas de aprendizaje profundo y redes neuronales.
  - Ha logrado avances significativos en diversos campos como juegos (AlphaGo, AlphaStar), ciencia climática (nowcasting de precipitaciones), matemáticas y biología estructural.
- **DeepMind y la predicción de estructuras**:
  - En 2018, participó en el CASP13 (Critical Assessment of protein Structure Prediction) con AlphaFold1, superando significativamente a otros métodos.
  - En 2020, presentó AlphaFold2 en el CASP14, alcanzando gran precisión en la predicción de estructuras proteicas, considerado un avance revolucionario.
  - En 2021, publicó la base de datos AlphaFold Protein Structure Database en colaboración con EMBL-EBI, proporcionando acceso gratuito a predicciones estructurales de casi todas las proteínas conocidas.
  - En 2022, expandió sus capacidades con AlphaFold-Multimer para predecir complejos proteicos.
  - En 2023-2024, ha continuado mejorando sus modelos para predecir interacciones proteína-ADN y otros complejos biomoleculares.

4. Diferencia entre Modelar y Predecir Estructuras

- **Modelado de estructuras**:
  - Se basa en información estructural existente (estructuras experimentales conocidas).
  - Utiliza principalmente técnicas de modelado por homología/comparativo.
  - Requiere proteínas "plantillas" con estructuras resueltas experimentalmente y similitud de secuencia.
  - Más confiable cuando existe alta similitud con estructuras conocidas.
  - Ejemplos: SWISS-MODEL, MODELLER, metodologías tradicionales que dependen de alineamientos y plantillas.
  - Limitación: menos efectivo para proteínas sin homólogos estructurales conocidos.
- **Predicción de estructuras**:
  - Determina la estructura tridimensional directamente a partir de la secuencia primaria.
  - Utiliza métodos *ab initio* o *de novo* que no dependen exclusivamente de plantillas.
  - Emplea principios físicos, estadísticos y/o de aprendizaje profundo.
  - Puede predecir estructuras incluso para proteínas sin homólogos estructurales conocidos.
  - Ejemplos: AlphaFold2, RoseTTAFold, métodos más recientes basados en aprendizaje profundo.
  - Los modelos actuales combinan información evolutiva, aprendizaje profundo y principios físico-químicos.

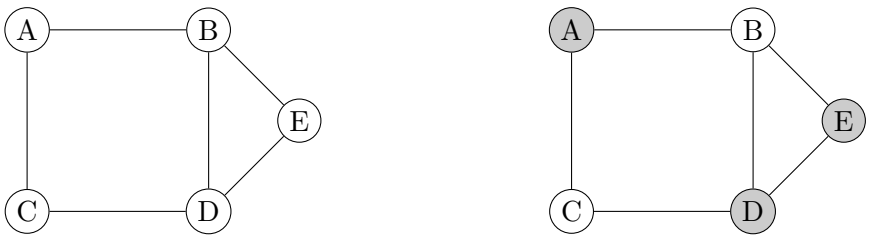
5. Cuadro Comparativo de Aplicaciones de Predicción de Estructura

Apps	**Lenguaje**	**Uso (descarga/online)**	**Tamaño de estructura que soporta**
Rosetta	C++, Python	Descarga (licencia académica gratuita)	Hasta 10.000 residuos en protocolos rígidos; óptimo $\leq$ 500 residuos en plegamiento ab initio
MODELLER	Python, Fortran	Descarga (gratuito para académicos)	Hasta 3,000 residuos por cadena; ideal $\leq$ 1.000
I-TASSER	C++, Python	Online y descarga (versión ligera)	Hasta 1.500 residuos (web); hasta 5.000 local
Phyre2	C++, JavaScript	Principalmente online	Hasta $\sim$1.000 residuos
Swiss-Model	Python, C++	Online	Hasta $\sim$2.000 residuos por cadena; >3.000 total posible
RoseTTAFold	Python, PyTorch	Online y descarga (GitHub)	Hasta 1.400 residuos (limitado por memoria GPU)
trRosetta	Python, TensorFlow	Online y descarga (GitHub)	Hasta $\sim$1.000 residuos
OpenFold	Python, PyTorch	Descarga (GitHub)	Hasta $\sim$3.000 residuos (depende del hardware)
OmegaFold	Python, JAX	Descarga (GitHub)	Hasta 2.000–2.500 residuos (GPU moderna)
ESMFold	Python, PyTorch	Online y descarga (GitHub)	Hasta $\sim$1.400 residuos
AlphaFold-Multimer	Python, JAX	Descarga (GitHub), Colab	Hasta poco más de 3.000 residuos totales; ideal $\leq$ 5 cadenas grandes o 8 medianas
RaptorX	Python, PyTorch	Descarga (GitHub)	Hasta $\sim$1.200 residuos

Cuadro 1: Comparativa de aplicaciones de predicción de estructura proteica

6. Vertex Cover

Un Vertex Cover (cobertura de vértices) es un concepto fundamental en teoría de grafos que consiste en un conjunto de vértices tal que cada arista del grafo incide en al menos uno de los vértices de dicho conjunto. En otras palabras, es un subconjunto de vértices que "cubre" todas las aristas del grafo.
El problema de encontrar el Vertex Cover mínimo (con el menor número posible de vértices) es un problema NP-completo clásico en ciencias de la computación, con aplicaciones en la biología computacional.



Grafo original Vertex Cover {A, D, E}; V = 3

Figura 1: Ejemplo de un Vertex Cover mínimo en un grafo simple. Los vértices sombreados (A, D, E) forman un conjunto que cubre todas las aristas del grafo.

En el ejemplo, el conjunto {A, D, E} representa un Vertex Cover mínimo porque:

- Cada arista del grafo está conectada a al menos uno de estos vértices
- No existe un conjunto con menos de 3 vértices que cubra todas las aristas
- El vértice A cubre las aristas (A,B) y (A,C)
- El vértice D cubre las aristas (B,D), (C,D) y (D,E)
- El vértice E cubre la arista (B,E)

La búsqueda de Vertex Cover mínimos se utiliza en aplicaciones bioinformáticas como el diseño de experimentos para interacciones proteína-proteína, validación de redes de interacción molecular, y problemas de ensamblaje de fragmentos en secuenciación de ADN/ARN.

7. Aplicaciones que Implementan el Algoritmo Vertex Cover

Existen diversas herramientas y bibliotecas que implementan el algoritmo de Vertex Cover para diferentes aplicaciones:

- **Library for Efficient Modeling and Optimization in Networks (LEMON)**: Biblioteca de C++ que proporciona implementaciones eficientes para problemas de optimización en grafos, incluyendo Vertex Cover.

- **Open Graph Drawing Framework (OGDF)**: Biblioteca de C++ con algoritmos de grafos que incluye optimizaciones para Vertex Cover.

- **Graph-tool**: Biblioteca de Python/C++ para análisis y manipulación eficiente de grafos con algoritmos para problemas NP-completos.

- **Cytoscape**: Software para visualización y análisis de redes biomoleculares que incluye plugins para encontrar Vertex Covers en redes biológicas.

Estas herramientas son ampliamente utilizadas en investigación bioinformática para analizar redes de interacción proteína-proteína, identificar objetivos de fármacos y estudiar rutas metabólicas críticas.