Alineando secuencias genéticas: COVID y complejidades ¿¿bajas??

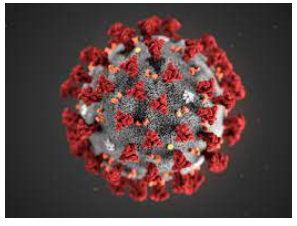
Elvira Mayordomo

Universidad de Zaragoza

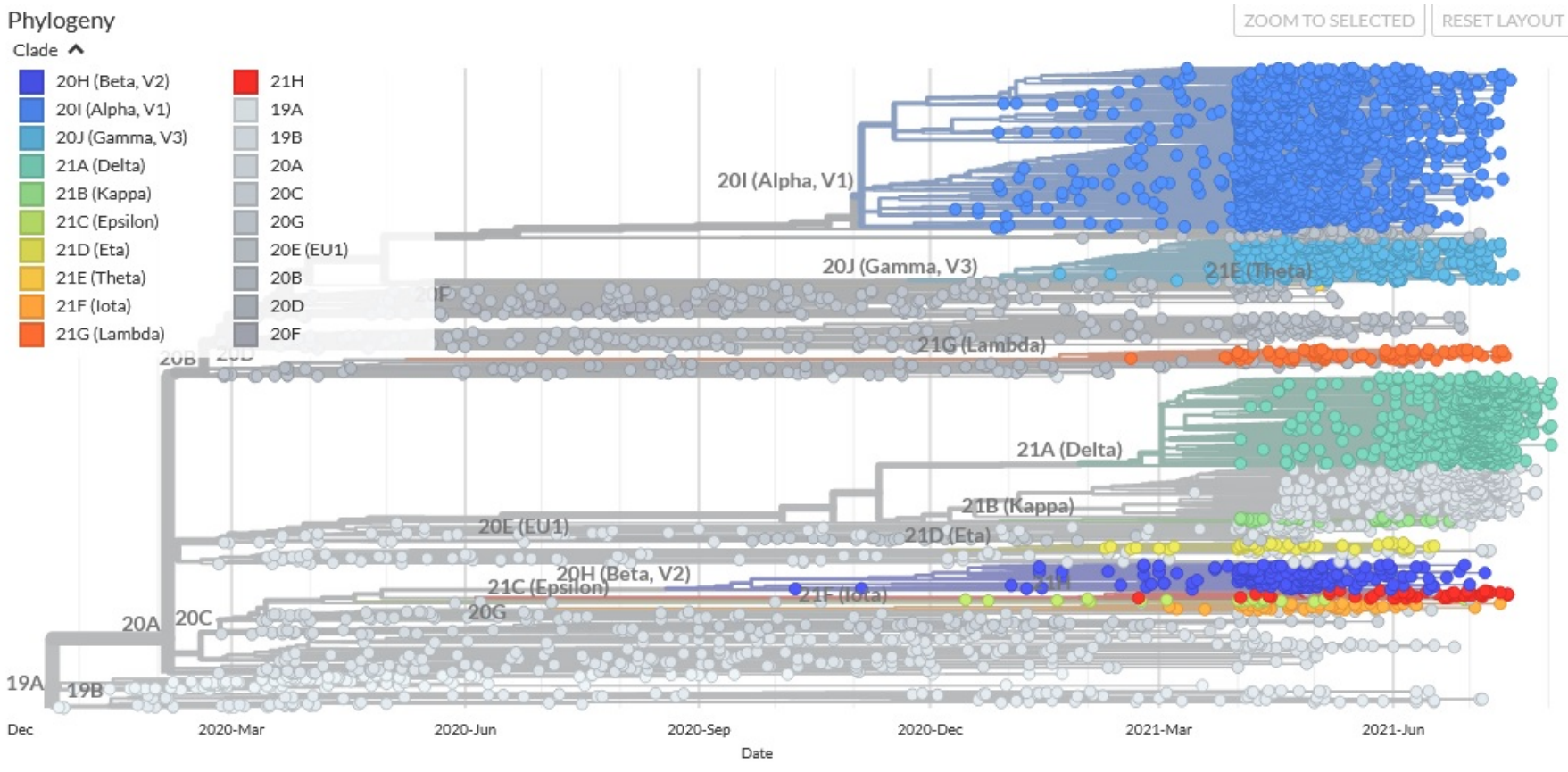
9 de agosto de 2021

Unos cuantos datos

- Más de 203.000.000 enfermos de COVID-19 diagnosticados en el mundo
- Unas 2.672.000 secuencias de genoma completo del virus



Árbol filogenético

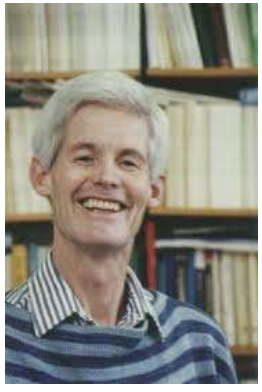


- https://nextstrain.org/ncov/global

¿Por qué hay sólo 2.672.000 genomas completos de 203 millones de casos

- Cómo se diagnostica
- Utilidad del genoma completo
- ¿Y por qué?

Qué es un problema intratable



Stephen Cook

Leonid Levin

Richard Karp

Estos tres investigadores son los responsables iniciales de haber conseguido entender los límites prácticos de un ordenador

Problemas intratables

- Desde principios de los 70 Cook, Levin y Karp (2 premios Turing y un premio Knuth) empezaron a darse cuenta de que muchos problemas cuyas soluciones computacionales era muy ineficientes se parecían mucho
- Ahora incluyen miles de problemas fundamentales de
  - optimización,
  - inteligencia artificial,
  - combinatoria,
  - lógica,
  - bases de datos,
  - localizaciones en mapas,
  - biología molecular,
  - etc

¿Cómo son los problemas intratables?

En general los problemas intratables:

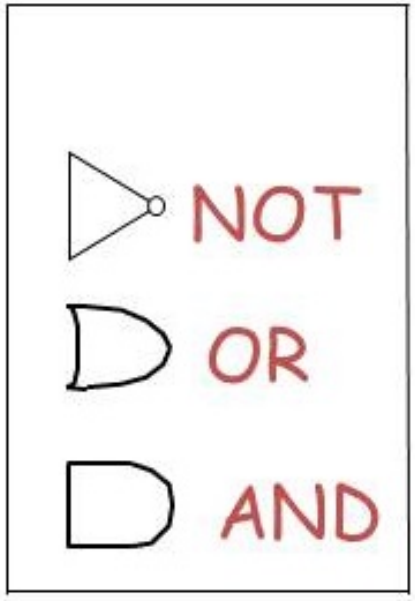
- sólo tienen algoritmos de fuerza bruta o sencillos pero muy lentos
- hace décadas que no se han conseguido algoritmos más rápidos
- Son los famosos problemas NP-completos

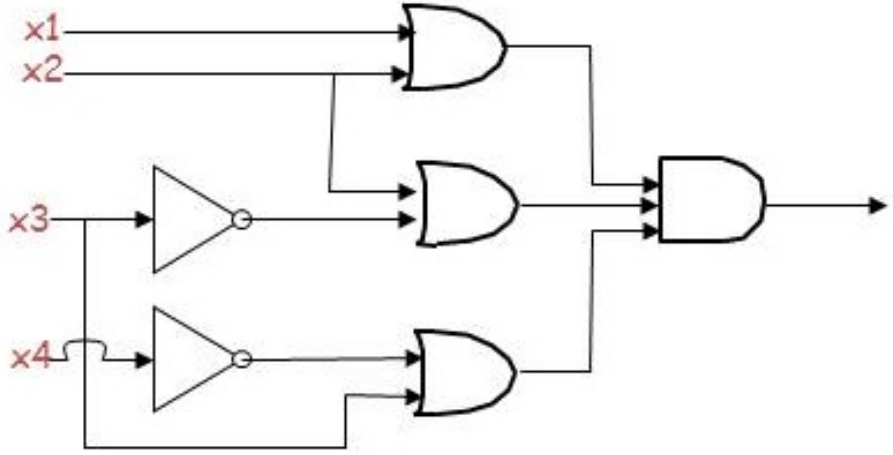
La madre de todos los intratables

- Es un problema de lógica o circuitos booleanos que se llama Satisfacibilidad (SAT)

SAT

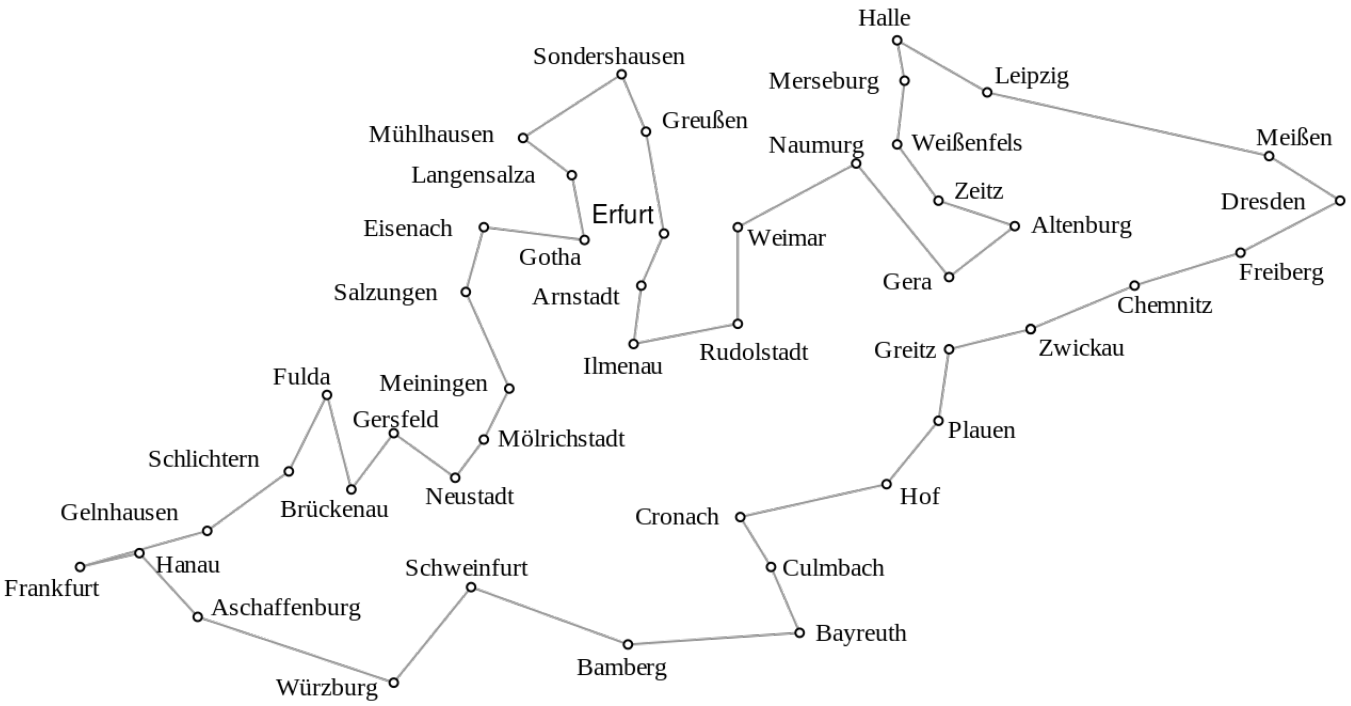
Problema: SAT
Datos de entrada: Un circuito booleano C con una única salida.
Buscamos: Una asignación de las entradas de C que da salida Cierto





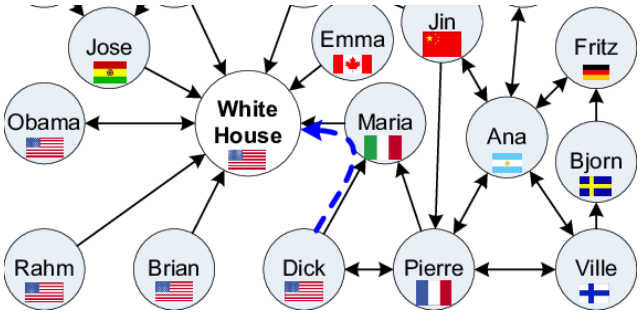
Otros intratables

El problema del viajante: Dadas una serie de ciudades y la distancia entre ellas encontrar el recorrido más corto por todas ellas



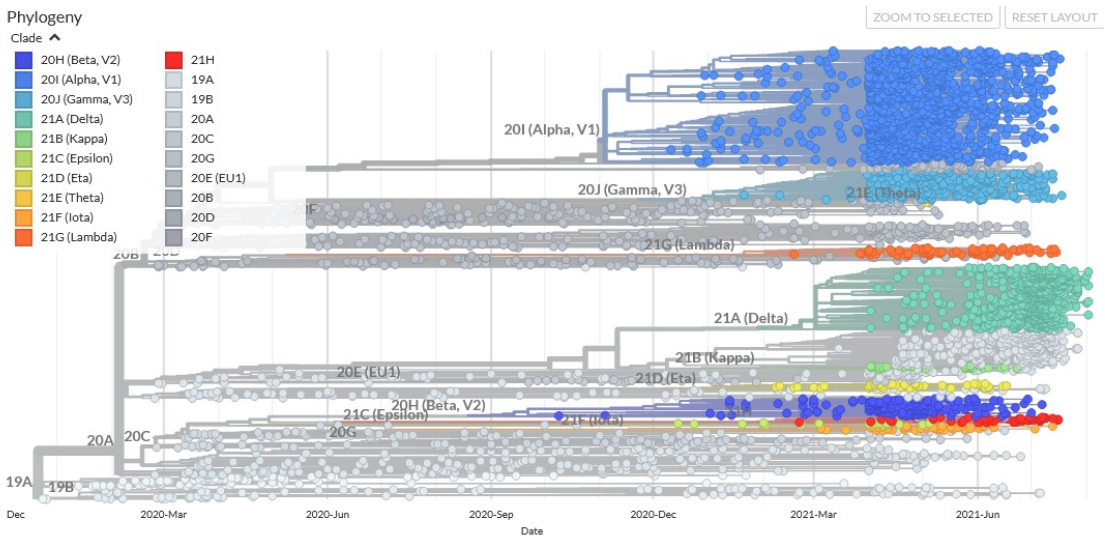
Otros intratables

Clique: Dado un conjunto de perfiles en facebook encontrar el grupo más grande de entre ellos que están completamente conectados (todos son amigos de todos)



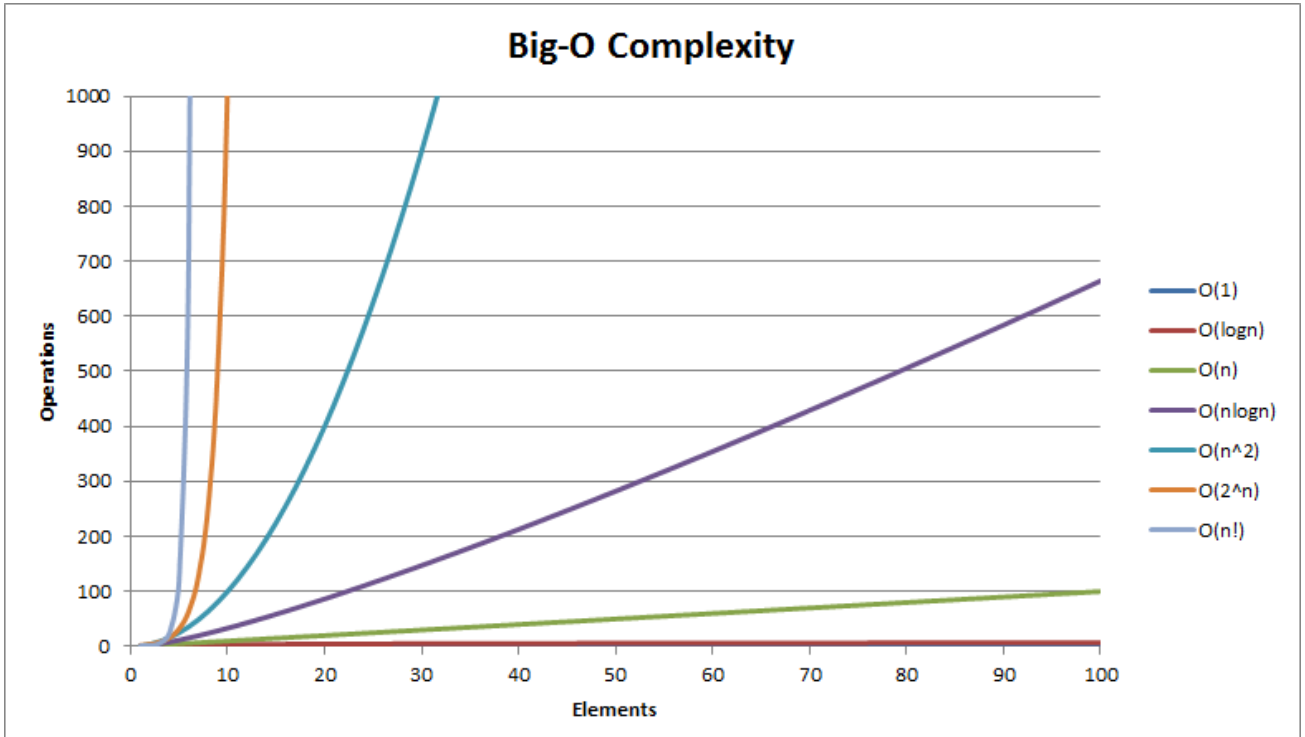
Otros intratables

Filogenia: Dadas unas secuencias de ADN reconstruir el árbol que representa su historia evolutiva



Qué sabemos de los intratables

- Son todos **algorítmicamente equivalentes** en un sentido bastante fuerte: si consigues un programa eficiente para resolver uno de ellos lo puedes transformar fácilmente en un programa eficiente para otro
- Para ninguno de ellos tenemos realmente soluciones satisfactorias, los mejores algoritmos que funcionan tardan tiempo exponencial

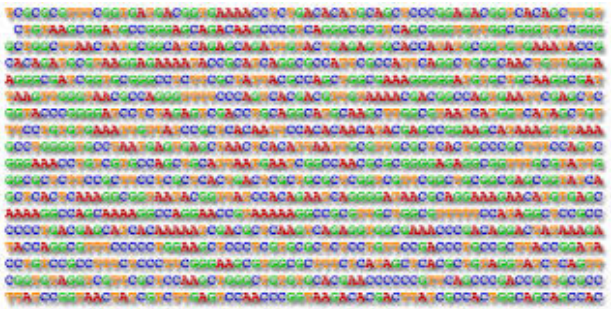


Complejidad de grano fino

- Tiempo exponencial es demasiado pero es que para los casos enormes que queremos resolver **tiempo cuadrático es demasiado**
- Vamos a presentar los **intratables débiles** que son problemas que no sabemos resolver en tiempo esencialmente menor que cuadrático
- Al conseguir compararlos unos con otros los hacemos **algorítmicamente equivalentes**: un algoritmo subcuadrático para uno de ellos nos daría uno para todos

Alinear 2 secuencias

- Los humanos tenemos alrededor de 3 Gigas de ADN en cada célula (una palabra de $3 * 10^9$ letras)
- Como va cortado en cromosomas vamos a pensar que nos conformamos con trabajar con uno de ellos, 1 mega (palabra de 10^6 letras)
- Vamos a pensar en el problema de **comparar** el mismo cromosoma de 2 individuos



Supongamos que queremos comparar nuestro fragmento de un mega con todos los de una base de datos como Genbank (1.600 millones de secuencias)

Alineamiento

Datos de entrada: Dos palabras o cadenas
Buscamos: Una optimización de las inserciones, borrados y cambios que me han llevado de una a otra

```
GAATTCAGTTA
GGATCAGTA
```


```
GAATTCAGTTA
GGA-TCAGTA-
```

Coste de un alineamiento

- Cada inserción tiene un coste (por ejemplo 3)
- Cada sustitución tiene un coste (5)
- Objetivo: encontrar el alineamiento que minimiza el coste total

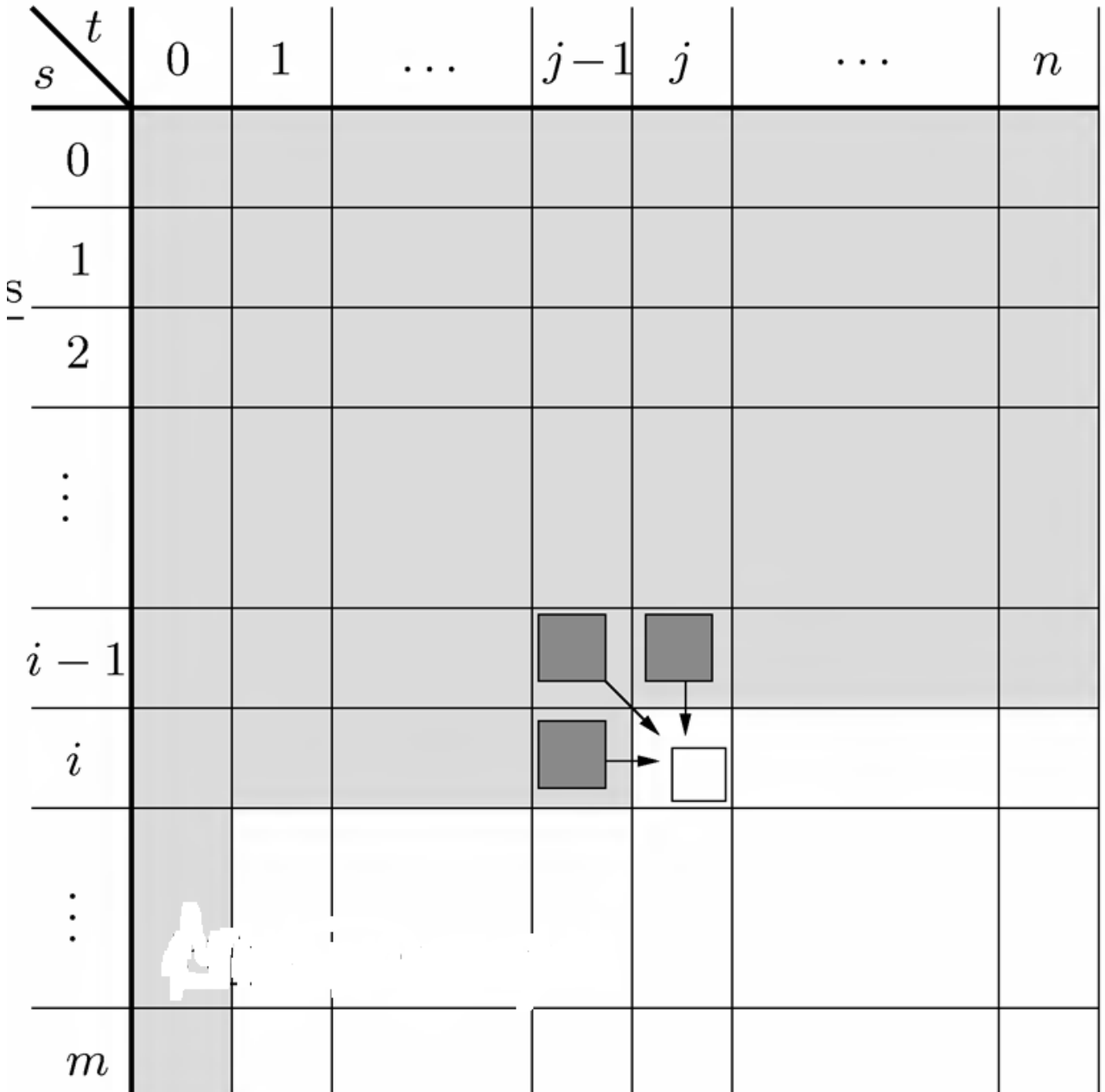
```
GAATTCAGTTA
GGA-TCAGTA-
```

Algoritmo para alineamiento

- Queremos alinear $s = s_1 \dots s_m$ con $t = t_1 \dots t_n$
- ¿Algoritmo cuadrático?
- Ir calculando incrementalmente el coste de alinear $s_1 \dots s_i$ con $t_1 \dots t_j$
- **Programación dinámica**
- $M(i,j)$ =coste de alinear $s_1 \dots s_i$ con $t_1 \dots t_j$
- El objetivo es $M(m,n)$

$$M(i,j) = \text{mín}\{M(i,j-1)+3, M(i-1,j)+3, M(i-1+j-1)+5\}$$

Programación dinámica



Alineamiento

- Los mejores algoritmos que se conocen para resolver **realmente** este problema tardan tiempo $n^2/\log n$
- para nuestro cromosoma de $n = 10^6$ estamos hablando de 10^{12} instrucciones (1 minuto en un solo procesador)
- Lo queremos hacer para un número enorme de casos (1.600 millones)

¿Por qué no bajamos de tiempo cuadrático?

- Compararlo con otros problemas que tampoco sepamos mejorar de tiempo cuadrático nos puede ayudar a entender las razones
- Aquí **la madre de los intratables débiles** es el problema de **Vectores Ortogonales (OV)**

Vectores ortogonales

Problema: OV

Datos de entrada: Dos conjuntos S, T de n cadenas binarias cada uno de longitud $d \geq \log n$ ($S, T \subseteq \{0,1\}^d$)

Buscamos: ¿Existen $a \in S$ y $b \in T$ que sean ortogonales ($\sum_i a_i b_i = 0$)?

- El mejor algoritmo conocido es esencialmente probar todos con todos (tiempo cuadrático)

Alineamiento es tan difícil como Vectores ortogonales

$\mathrm{OV}(S, T, n, d)$

1 $x =$ CONVERSION1(S)
2 $y =$ CONVERSION2(T)
3 **if** Alineamiento$(x, y) \leq 5d + X$ Resultado True
4 **if** Alineamiento$(x, y) \geq 15d + X$ Resultado False

- CONVERSION1 cambia cada 0 por 0111 y cada 1 por 0001
- CONVERSION2 cambia cada 0 por 0011 y cada 1 por 1111
- Es más complicado (bloques entre una cadena y otra, etc)
- **Si tuviera un algoritmo con coste $n^{2-\epsilon}$ para Alineamiento me saldría tiempo $O(n^{2-\epsilon})$ para OV**

Otros problemas similares

- **Colinearidad**: Dados n puntos en el plano, comprobar que no hay 3 alineados
- **Suma de 3**: Dados n números enteros entre $-n^4$ y n^4, encontrar 3 de ellos x, y, z que cumplan $x + y = z$
- Les pasa lo mismo que al de antes, sólo los sabemos resolver en tiempo cuadrático

Algunos punteros

- Tenemos así una nueva familia de intratables débiles qué son problemas que no sabemos resolver en tiempo subcuadrático
- Son intratables cuando queremos resolver casos masivos
- De todo esto se ocupa un reciente campo de investigación que es **la complejidad de granularidad fina**
- Si queréis saber algo más de ello podéis echar un vistazo a la página de la profesora Virginia Williams del MIT `https://people.csail.mit.edu/virgi/`
- Un survey algo técnico `http://people.csail.mit.edu/virgi/eccentri.pdf`

Fin

- Queríamos conocer los límites de la computación y así llegamos al concepto de intratable
- Pero ha sido esa misma computación la que nos ha dado suficientes datos como para tener que replantearnos esa intratabilidad, necesitamos algoritmos realmente rápidos para poder trabajar con todos los datos que hemos conseguido
- Y una gran fuente de datos es la genómica ... y estamos en medio de una pandemia mundial con un virus cuyas mutaciones nos interesa mucho controlar ...