

# Bioinformática y análisis de datos ómicos

## TEMA 8: CARACTERIZACIÓN DE POBLACIONES COMPLEJAS



**Ignacio Varela Egocheaga**

DEPARTAMENTO DE BIOLOGÍA MOLECULAR

Este material se publica bajo la siguiente licencia:

[Creative Commons BY-NC-SA 4.0](https://creativecommons.org/licenses/by-nc-sa/4.0/)

## **Tema 8: Caracterización de poblaciones complejas**

8.1 El problema de las poblaciones complejas en salud.

8.2 Estrategias dirigidas. Metataxonomía.

8.3 Estrategias no dirigidas. Ensamblaje *de novo*.

8.4 Estructuración de secuencias. *Binning*

8.5 Identificación de elementos funcionales.

8.6 Inferencia de funciones.

# **Tema 8: Caracterización de poblaciones complejas**

**8.1 El problema de las poblaciones complejas en salud.**

8.2 Estrategias dirigidas. Metataxonomía.

8.3 Estrategias no dirigidas. Ensamblaje *de novo*.

8.4 Estructuración de secuencias. *Binning*

8.5 Identificación de elementos funcionales.

8.6 Inferencia de funciones.

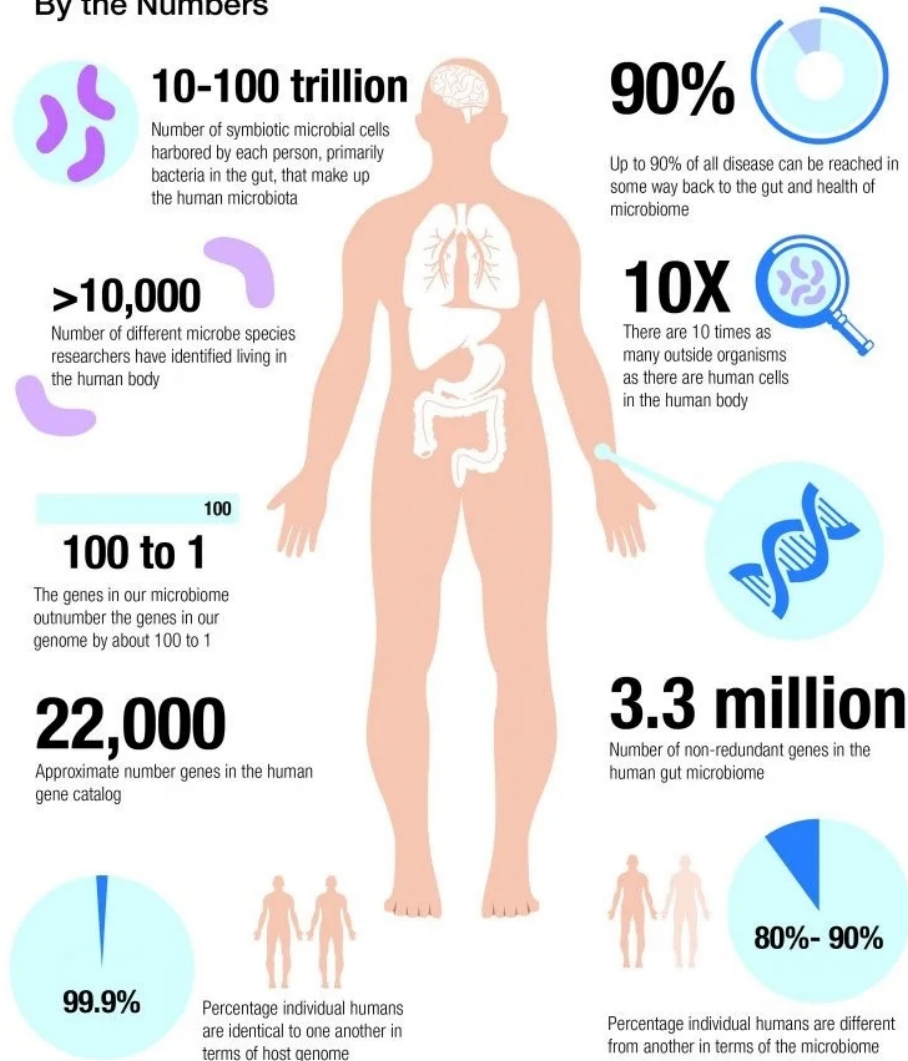
# Metagenoma humano

Un porcentaje muy importante del DNA de nuestro organismo proviene de la simbiosis con otros organismos. Un balance adecuado de esta microbiota es esencial para el correcto funcionamiento de nuestro cuerpo.

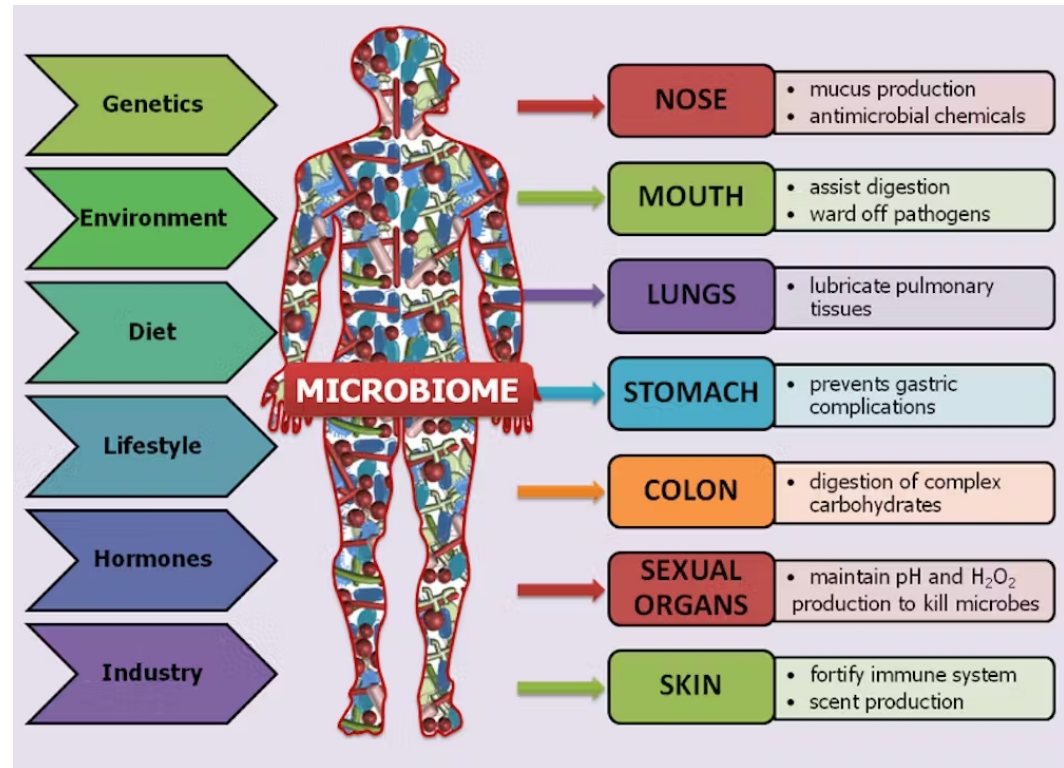
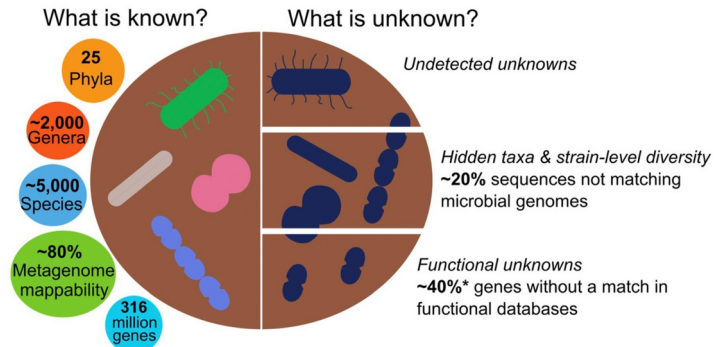
The Importance of the

## MICROBIOME

By the Numbers

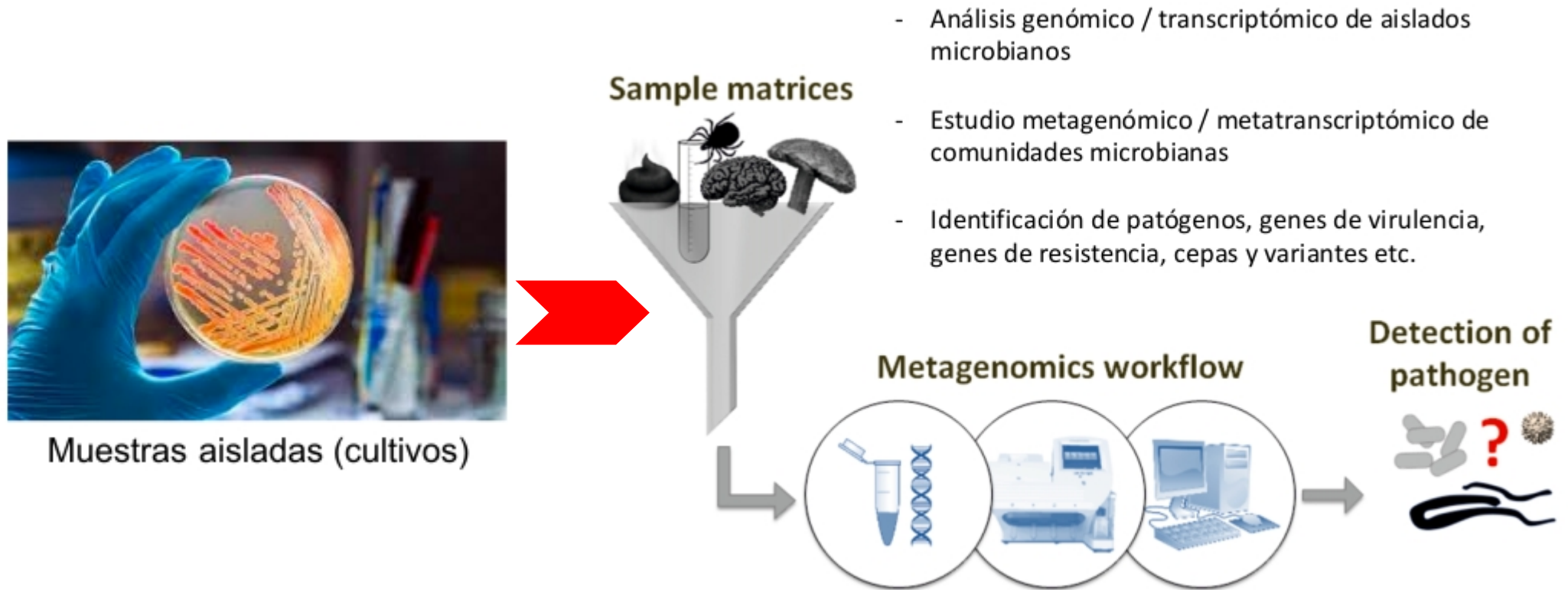


### The human microbiome



# Metagenoma humano

Inicialmente solo se podían estudiar los microorganismos que éramos capaces de aislar in vitro (< 2%). Actualmente, con las técnicas de secuenciación masiva somos capaces de caracterizar el DNA presente en una muestra sin necesidad de cultivar el organismo.



## **Tema 8: Caracterización de poblaciones complejas**

8.1 El problema de las poblaciones complejas en salud.

**8.2 Estrategias dirigidas. Metataxonomía.**

8.3 Estrategias no dirigidas. Ensamblaje *de novo*.

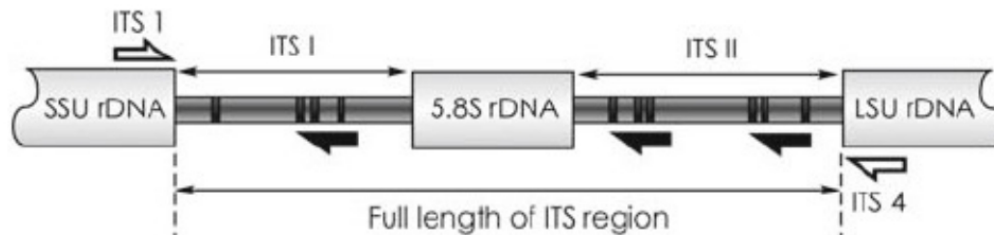
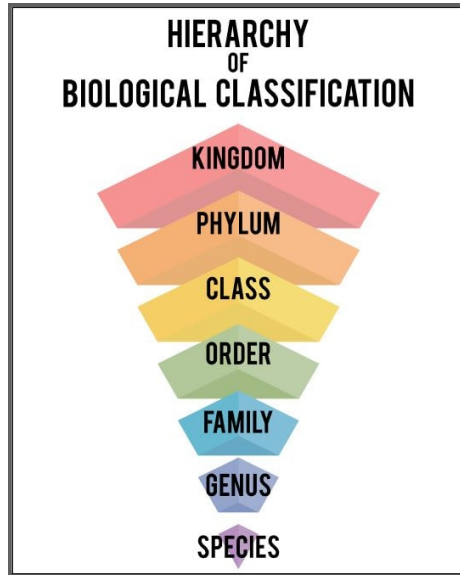
8.4 Estructuración de secuencias. *Binning*

8.5 Identificación de elementos funcionales.

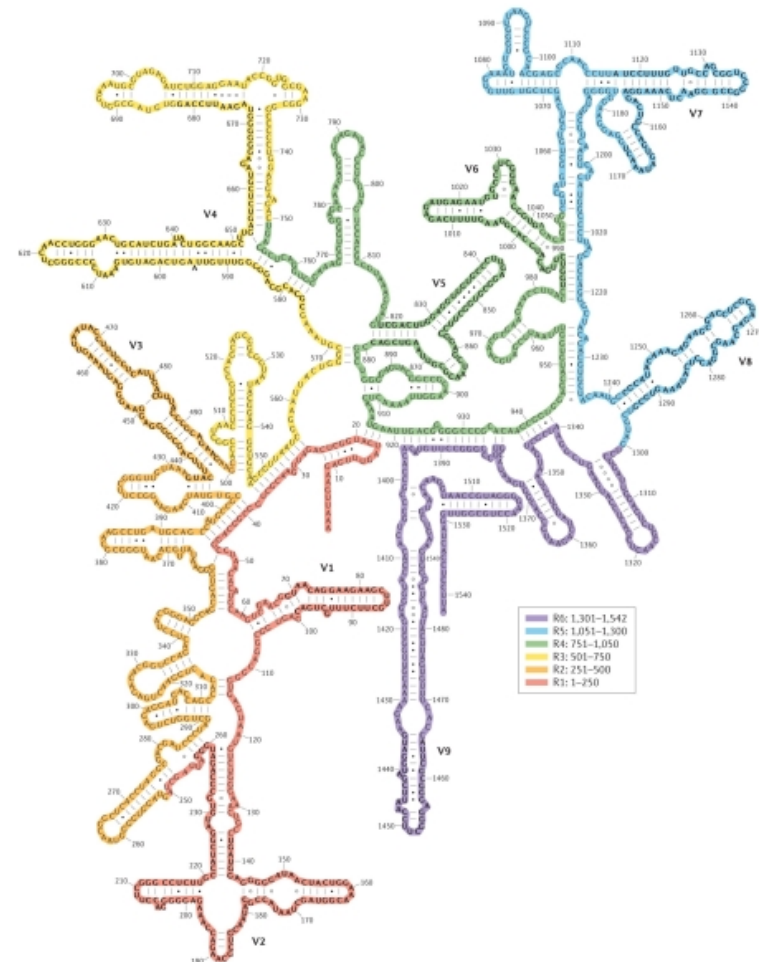
8.6 Inferencia de funciones.

# Metataxonomía

El análisis dirigido de las zonas variables de los genes de RNAs ribosómicos permite el estudio de la abundancia de taxones en una mezcla compleja

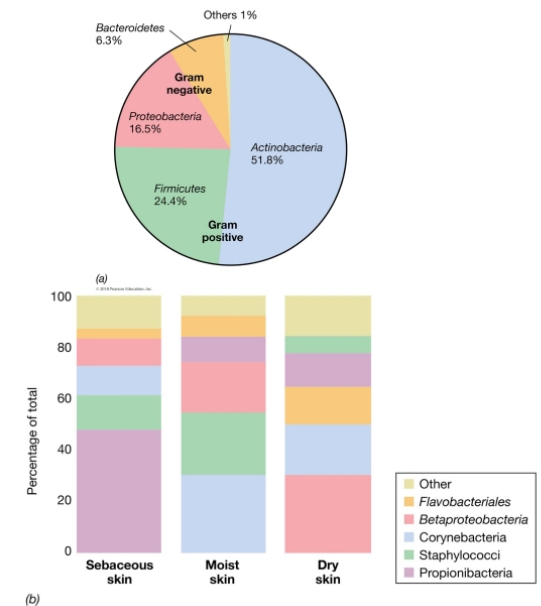
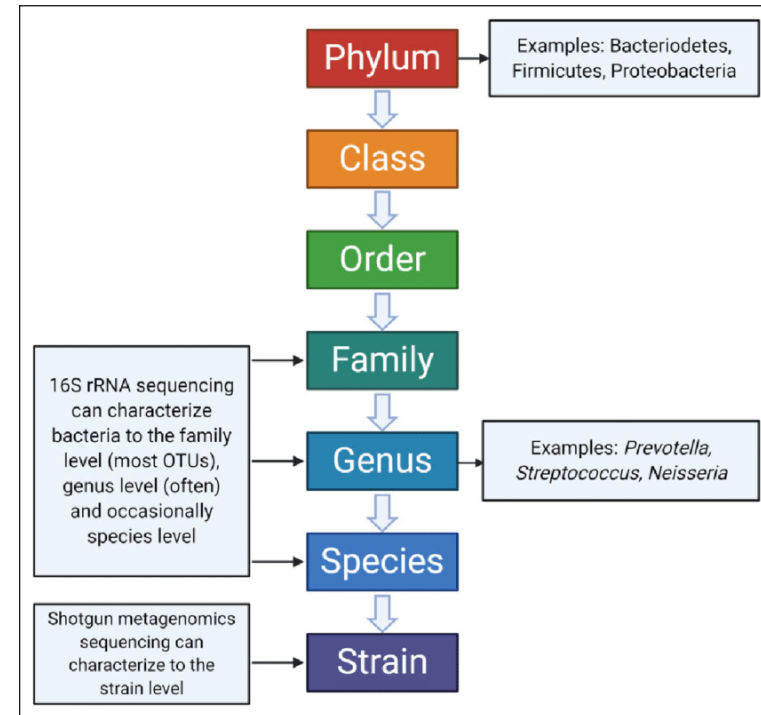
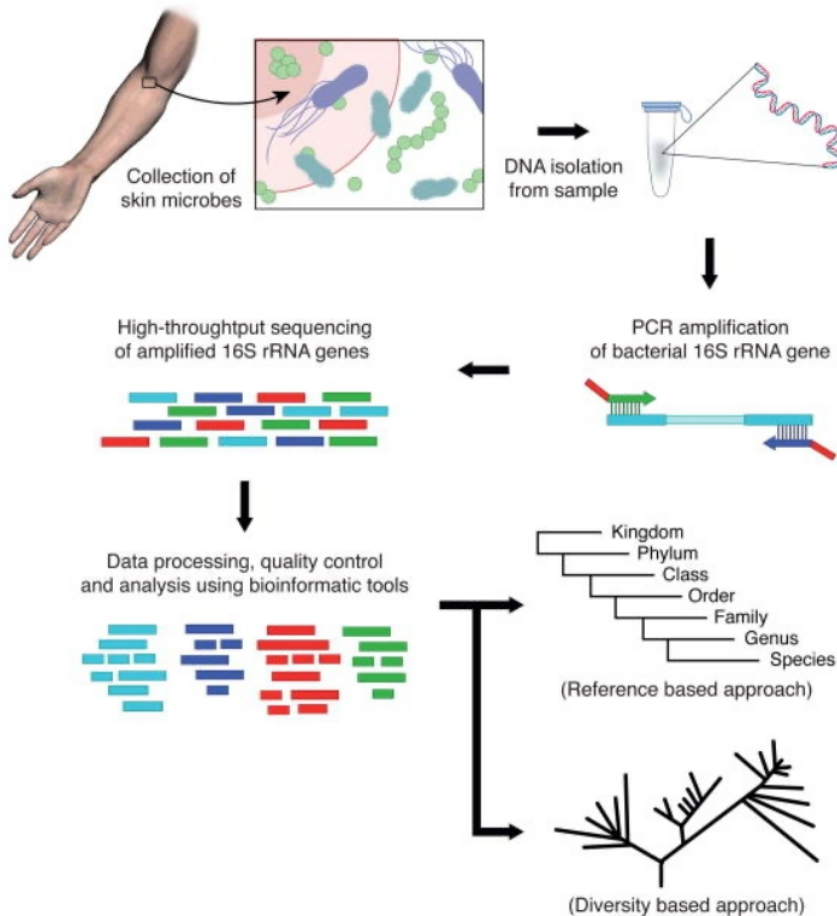


**CONSERVED REGIONS:** unspecific applications  
**VARIABLE REGIONS:** group or species-specific applications



# Metataxonomía

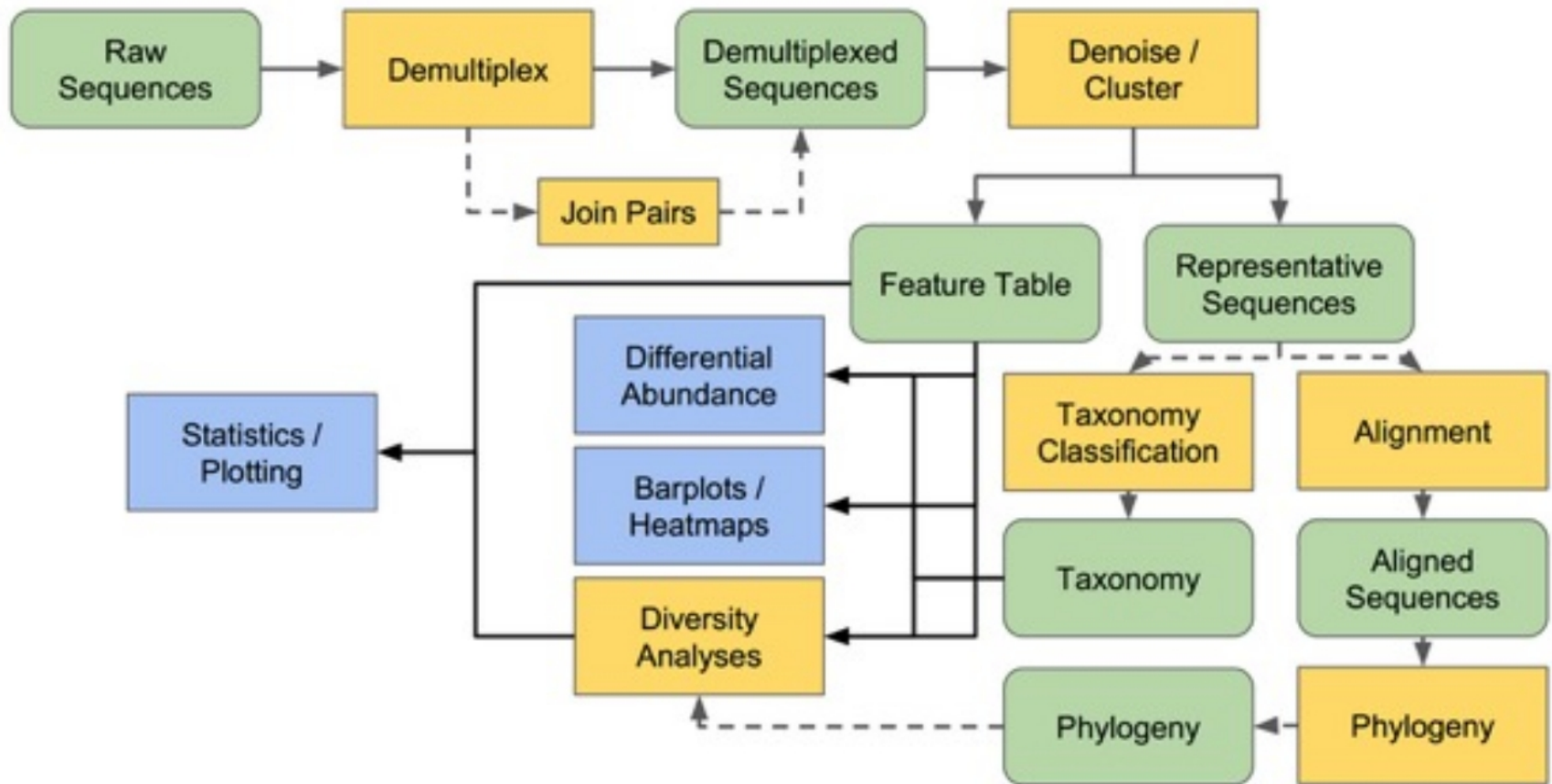
El estudio metataxonómico permite identificar en el mejor de los casos la abundancia de las distintas especies presentes en la muestra pero no da información sobre los genes/plásmidos o funcionalidades especiales de esa cepa en concreto.



# Metataxonomía



## QIIME2 Pipeline



## Tema 8: Caracterización de poblaciones complejas

8.1 El problema de las poblaciones complejas en salud.

8.2 Estrategias dirigidas. Metataxonomía.

**8.3 Estrategias no dirigidas. Ensamblaje *de novo*.**

8.4 Estructuración de secuencias. *Binning*

8.5 Identificación de elementos funcionales.

8.6 Inferencia de funciones.

# Problemas con el genoma de referencia

En las muestras complejas no disponemos de la información necesaria para saber las especies presentes en la muestra pero aún así la gran variabilidad entre las cepas no permite tener un genoma de referencia.

99.9%



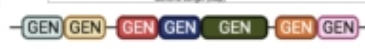
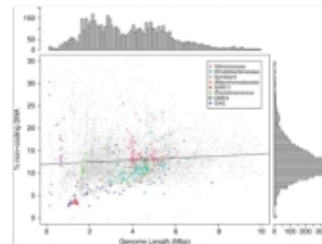
Alto nivel de identidad genética



Conservación numérica y estructural



Extensas regiones no codificantes



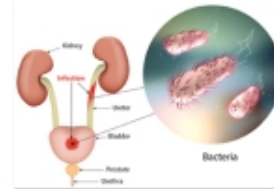
Bajo porcentaje DNA no codificante



*Escherichia coli*  
(comensal )  
4.6 Mb



*Escherichia coli*  
(enterohemorrágica)  
5.5 Mb



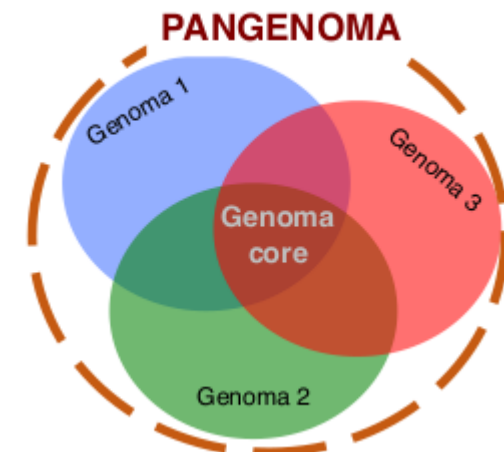
*Escherichia coli*  
(uropatogénica)  
5.2 Mb



Identidad genética variable

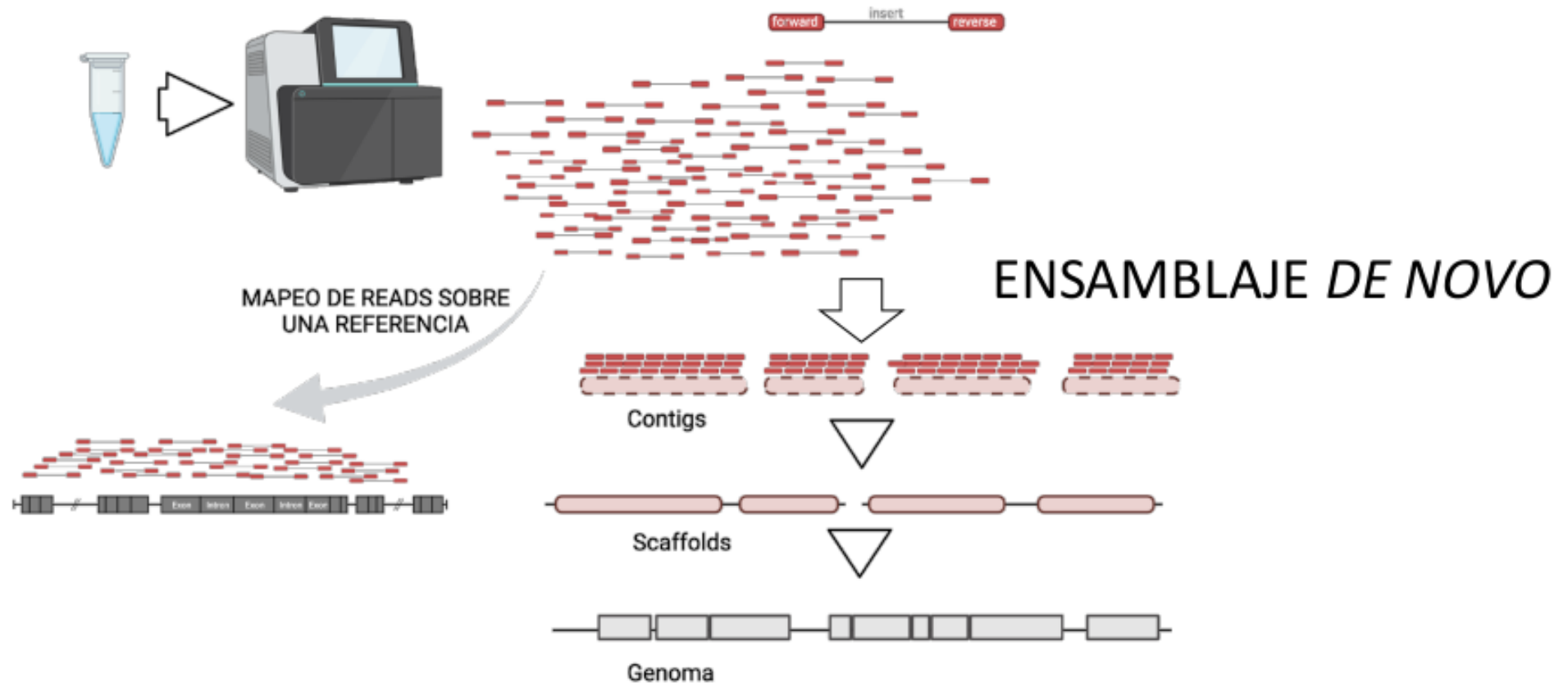


Variación estructural y numérica (plásmidos)



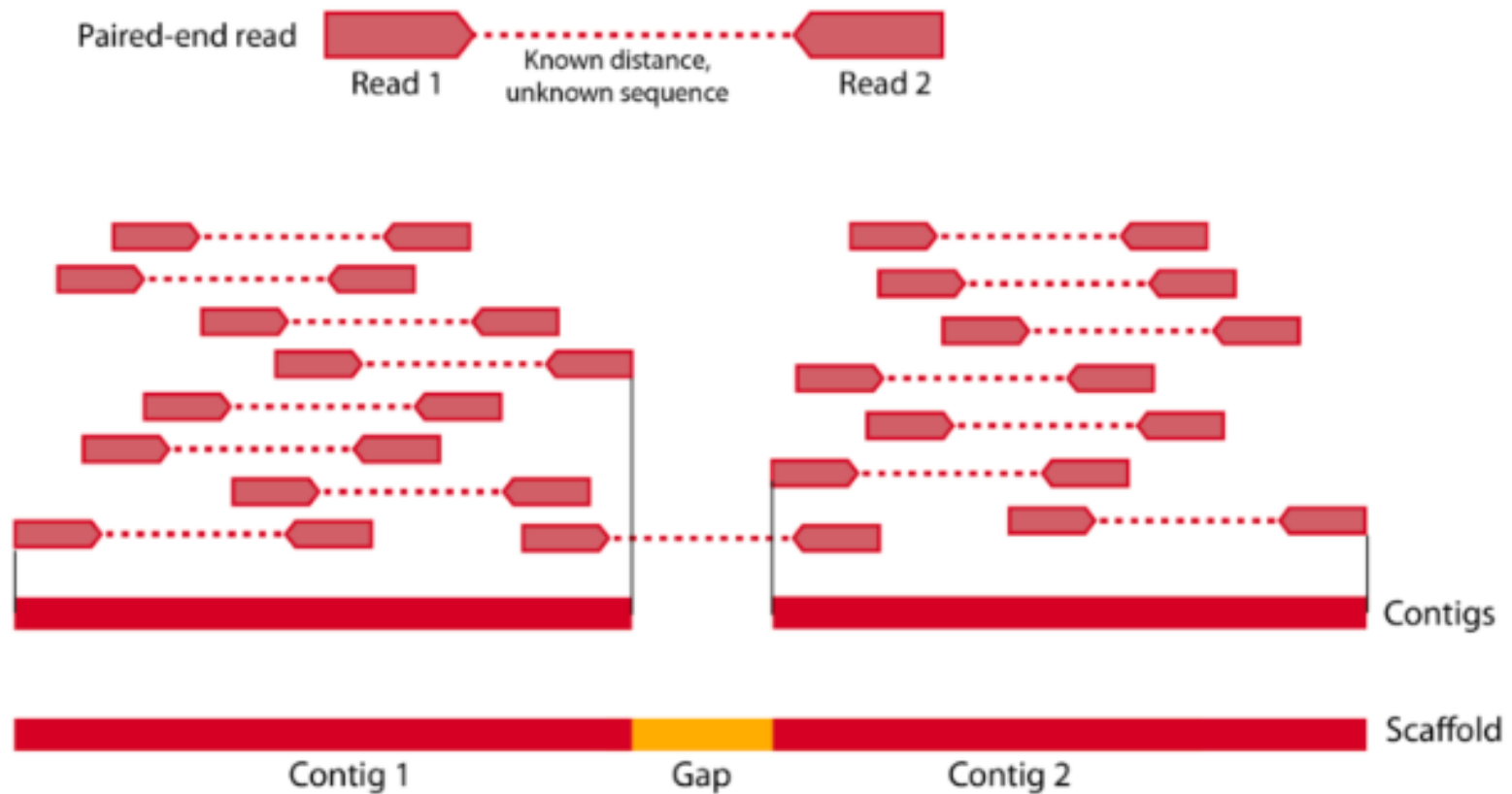
# Ensamblaje *de novo*

El ensamblaje consiste en la reconstrucción del genoma o genomas de partida a partir de las secuencias cortas generadas en la reacción de secuenciación masiva mediante la identificación de secuencias solapadas entre las lecturas.



# Ensamblaje *de novo*

La reconstrucción a partir de las lecturas generan secuencias consenso de mayor tamaño (**Contigs**) que pueden agruparse u ordenarse en estructuras más grandes (**Scaffolds**) en donde puede haber regiones no secuenciadas. En la situación ideal podría llegar a reconstruirse el genoma completo de las especies pero esto rara vez se consigue.



# Ensamblaje *de novo*

## 1.- Algoritmos avariciosos (Greedy)

ACGACACATTTGCA

TGCACACATCACCTGTACCA

CTGTACCACGTAGCTTGTGCACCA

---

ACGACACATTTGCACACATCACCTGTACCACGTAGCTTGTGCACCA

Se basa en buscar sobre una lectura cuales solapan y así extender la secuencia. Esto solo es posible para ensamblar unas pocas lecturas .

Problemas:

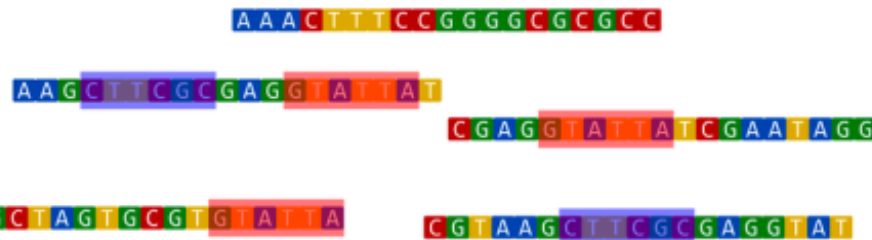
- **Longitud de los reads**
- **Redundancia en el DNA**

Los métodos avariciosos de ensamblaje solo funcionan correctamente en genomas pequeños y muy sencillos. Fallan mucho en genomas con regiones repetitivas o con alta homología y con la presencia de errores en la secuenciación.

# Ensamblaje *de novo*

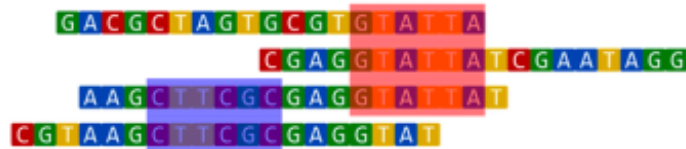
## 2.- Algoritmos OLC (Overlay, layout and consensus)

### Overlap



a.- **Overlap**: alineamiento de todos contra todos

### Layout



b.- **Layout** : estructura más sencilla que explica el overlap

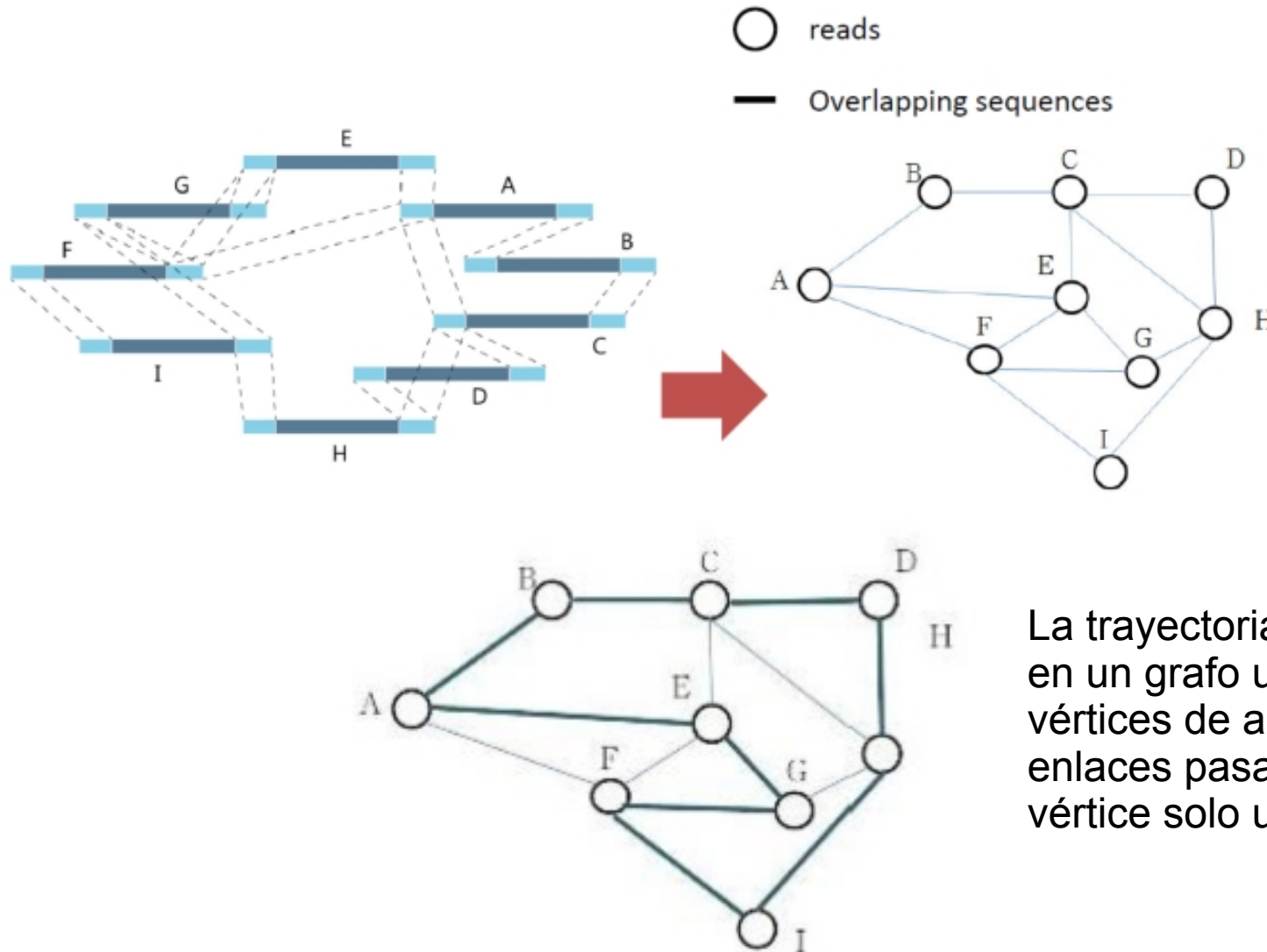
### Consensus



c.- **Consensus** : se corrigen los errores de secuenciación

# Ensamblaje *de novo*

## 2.- Algoritmos OLC (Overlay, layout and consensus)



La trayectoria **Hamiltoniana** en un grafo une todos los vértices de a partir de los enlaces pasando por cada vértice solo una vez.

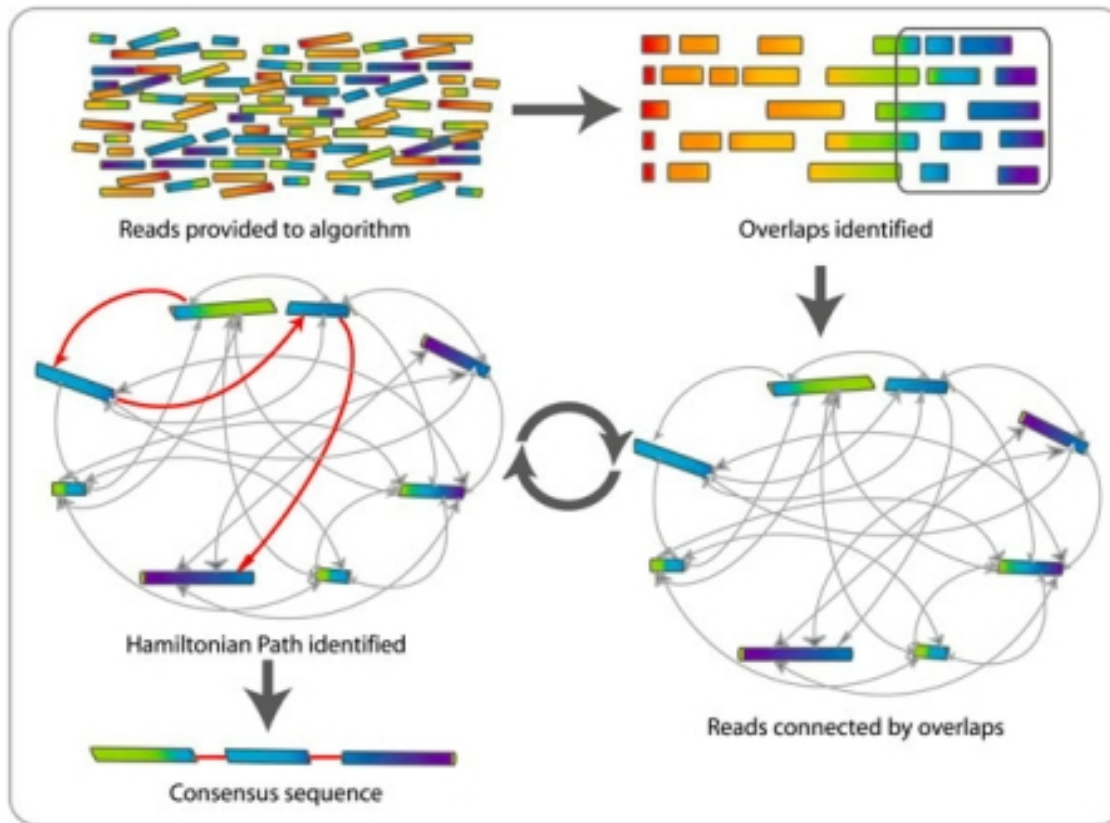
# Ensamblaje *de novo*

## 2.- Algoritmos OLC (Overlay, layout and **consensus**)



# Ensamblaje *de novo*

## 2.- Algoritmos OLC (Overlay, layout and consensus)



- La reconstrucción con algoritmos OLC es la estrategia más precisa para el ensamblaje de la secuencia. Sin embargo tiene limitaciones:

- 1.- Los recursos informáticos necesarios crecen de manera exponencial al número de nodos.
- 2.- La presencia de pequeños errores en las lecturas puede confundir el ensamblaje y hace necesario poner a punto el cálculo de solapamientos “aproximados” lo que computacionalmente es pesado y complicado.

# Ensamblaje *de novo*

## 3.- Gráficos de Bruijn

Los gráficos de Bruijn dividen las lecturas en kmeros para el ensamblaje. Los kmeros son las distintas combinaciones ordenadas de nucleótidos de tamaño k que pueden generarse a partir de una determinada secuencia.

ACCACGCACCATGTGCCATGTGCACCATGGCGCAC

ACC  
CCA  
CAC

} Kmero de 3 nt

El uso de kmeros en sustitución de las secuencias enteras se basan en varias premisas

# Ensamblaje *de novo*

1.- Una secuencia puede representarse de manera única por su combinación de kmeros y se puede reconstruir mediante un análisis de grafos a partir de los kmeros.

ACCACGCACCAT

ACC

CCA

CAC

ACG

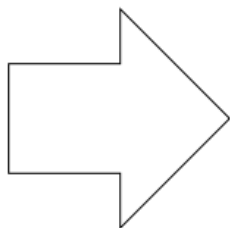
CGC

GCA

CAC

ACC

CAT



ACC.....2

CCA.....1

CAC.....2

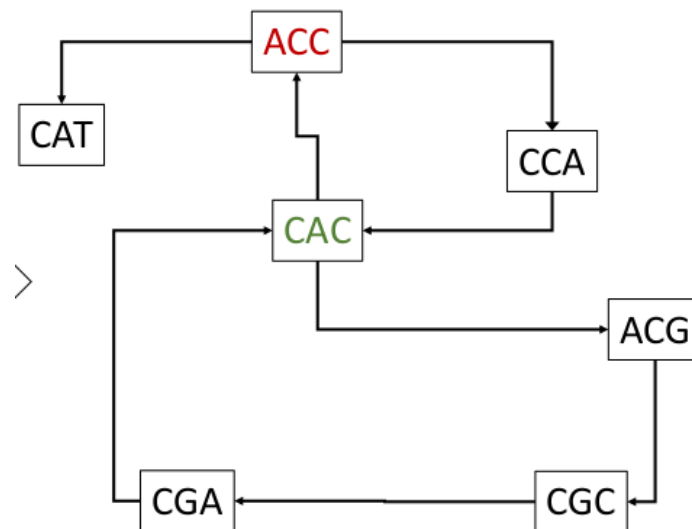
ACG.....1

CGC.....1

GCA.....1

CAT.....1

HASH TABLE

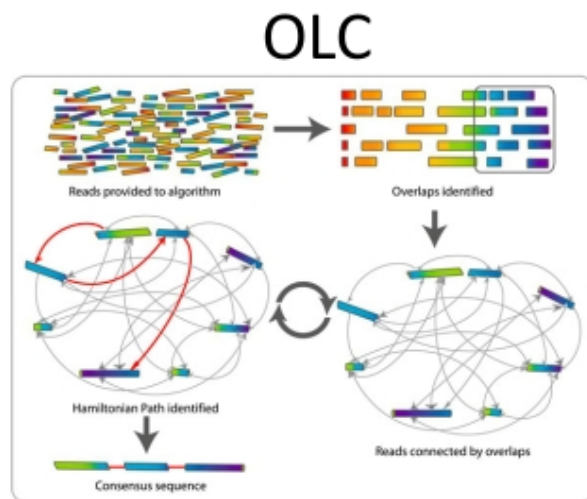


2.- La combinación de kmeros puede generarse tanto desde la secuencia completa como desde los fragmentos/lecturas generados desde esa secuencia completa

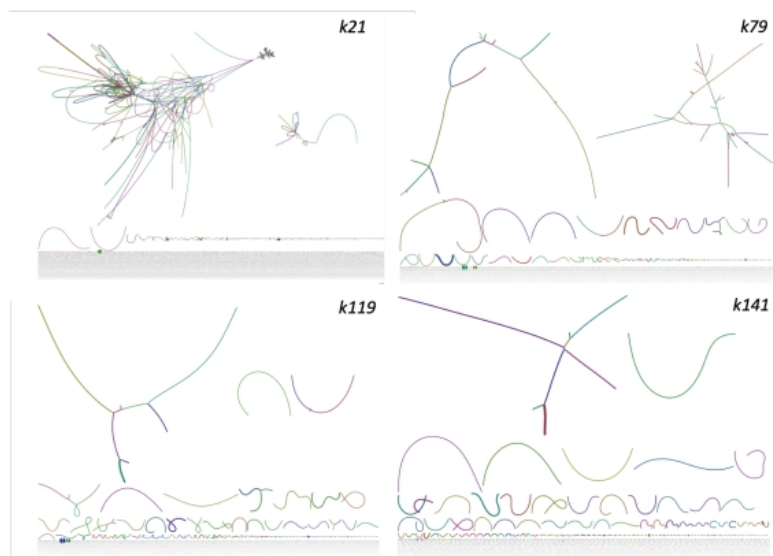
3.- El grado de homología entre dos lecturas es equivalente a comparar el porcentaje de kmeros de equivalencia exacta entre las dos secuencias.

4.- El número de combinaciones posibles de kmeros (vértices del grafo) es limitado a  $4^k$  siendo k el tamaño del kmero por lo que los recursos informáticos dependen exclusivamente del tamaño del kmero y no del número de lecturas.

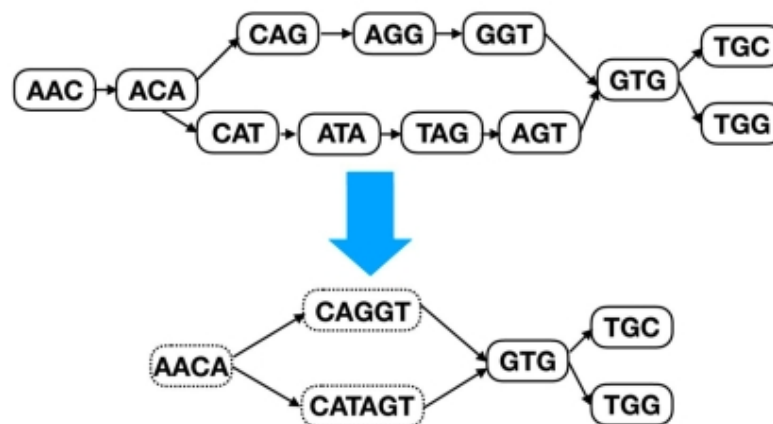
# Ensamblaje *de novo*



Un algoritmo OLC es preciso, pero lento y computacionalmente muy intensivo



## De Bruijn

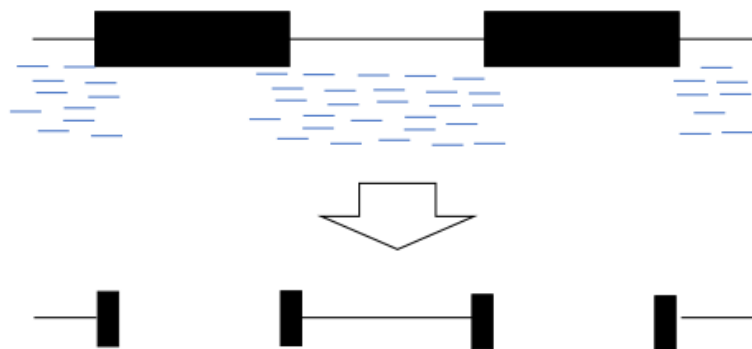


En lugar de usar reads, ensamblamos k-meros y a partir del grafo, reconstituimos la secuencia

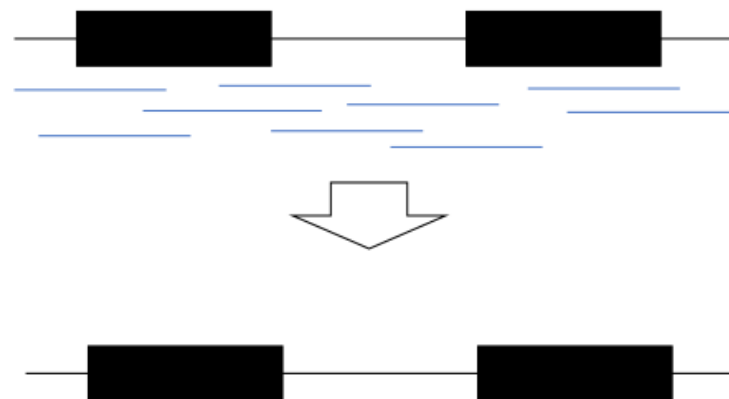
En la práctica, la mayoría de los algoritmos como **Megahit** prueban con distintas longitudes de kmeros hasta que son capaces de reconstruir trayectorias coherentes con las que forman contigs.

# Ensamblaje *de novo*

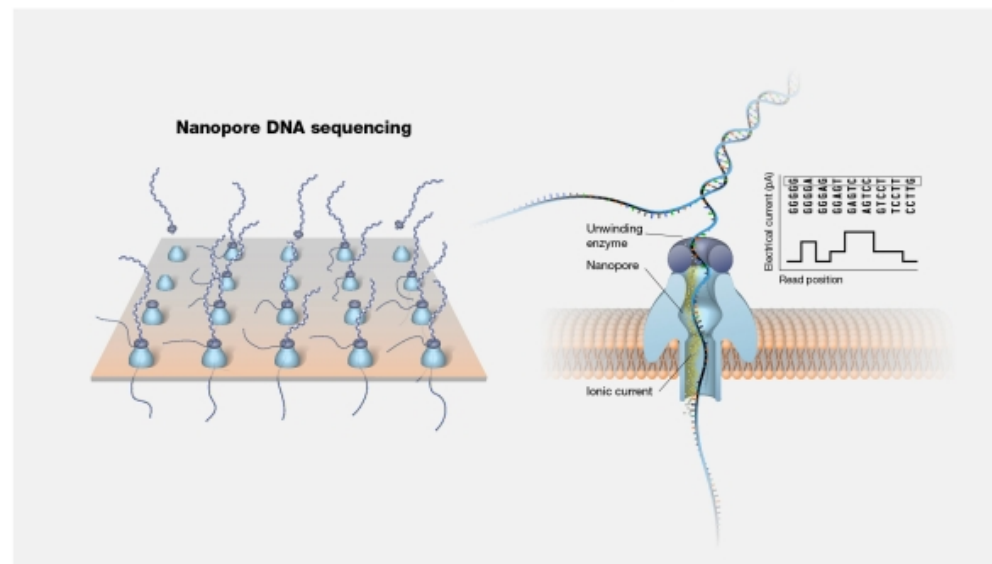
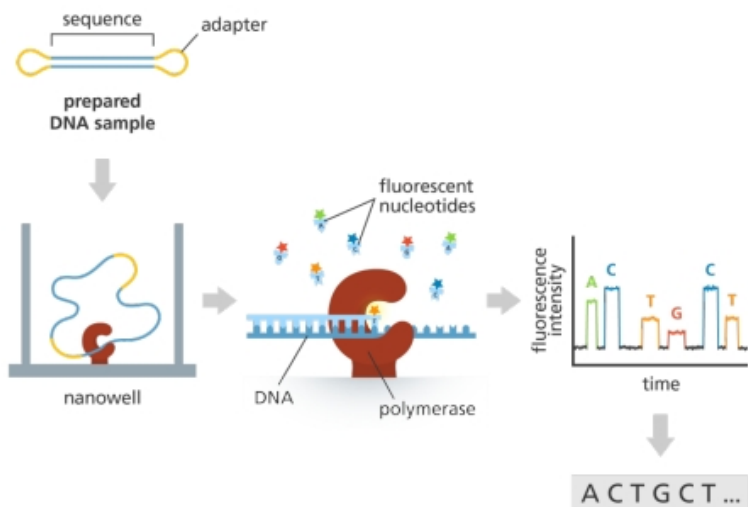
Cuanto más largas las secuencias, más fácil es el ensamblaje. En este sentido, las nuevas plataformas que generan secuencias más largas son de gran ayuda en metagenómica.



**PacBIO**



**Oxford Nanopore**



## **Tema 8: Caracterización de poblaciones complejas**

8.1 El problema de las poblaciones complejas en salud.

8.2 Estrategias dirigidas. Metataxonomía.

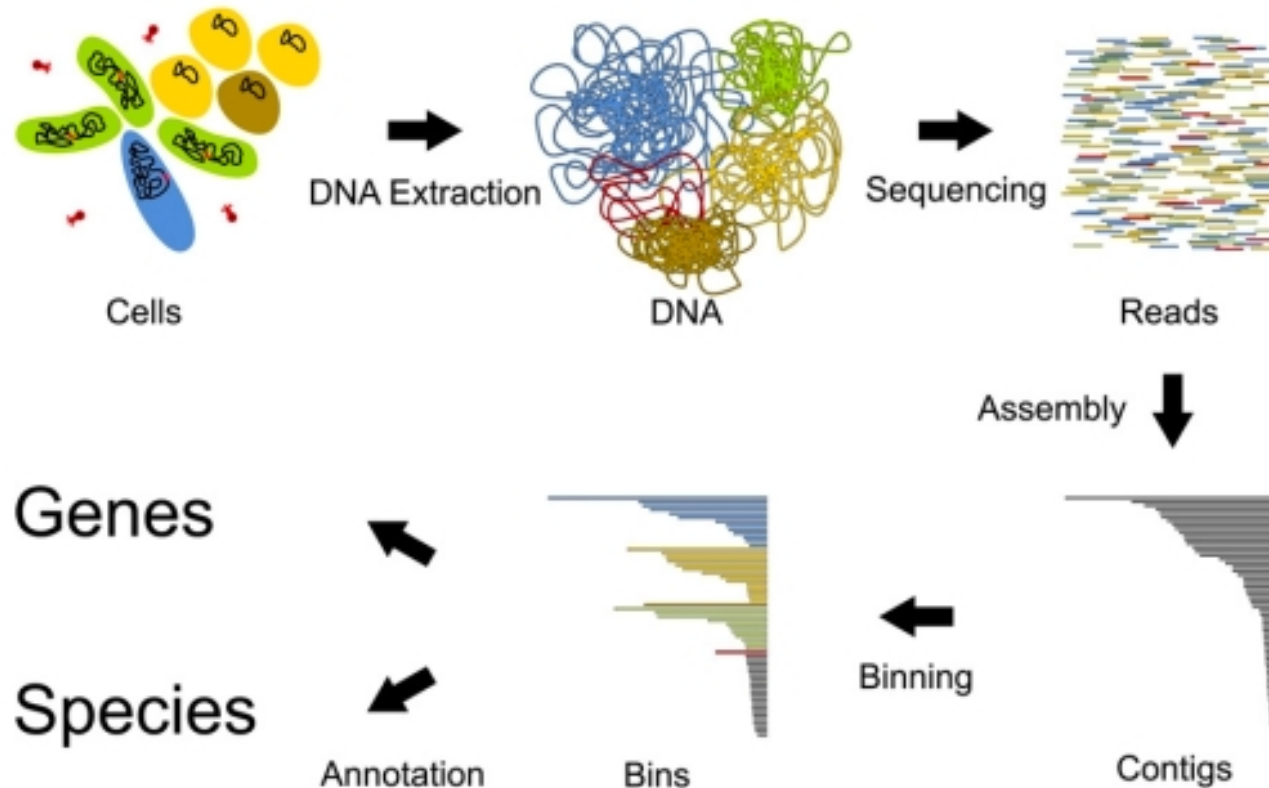
8.3 Estrategias no dirigidas. Ensamblaje *de novo*.

**8.4 Estructuración de secuencias. *Binning***

8.5 Identificación de elementos funcionales.

8.6 Inferencia de funciones.

# Estructuración de las secuencias. *Binning*

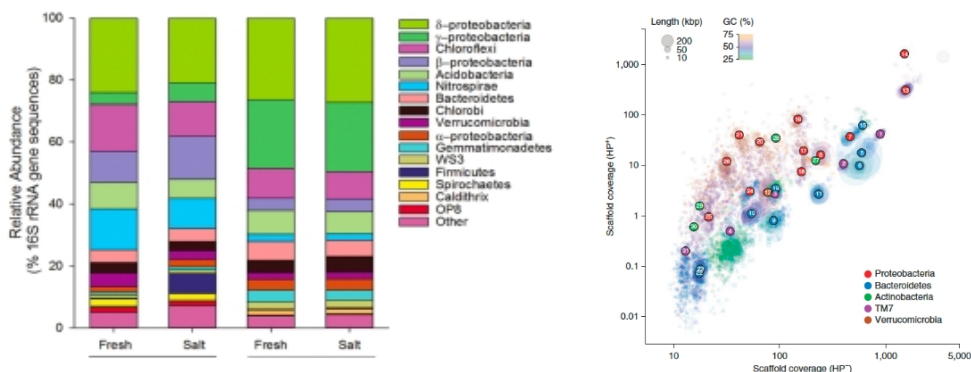


Una vez reconstruidas las secuencias en contigs/scaffolds necesitamos separarlas por especie. A este proceso en el campo se le denomina *Binning* porque separa las secuencias en *bins* o grupos

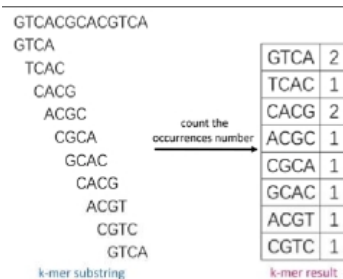
# Estructuración de las secuencias. *Binning*

La mayoría de los algoritmos tienen distintas características para separar las lecturas en grupos.

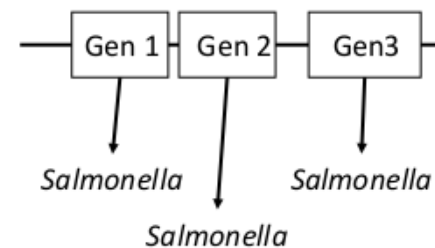
1.- **Profundidad de secuencia**. Los contigs/scaffolds de la misma especie tendrán más o menos la misma profundidad. Para comprobar esto será necesario **alineal** las reads frente a los contigs ensamblados



2.- **Abundancia de kmeros**. Los contigs de la misma especie tendrán una frecuencia de kmeros similar



3.- **Homología**. Los genes contenidos en los contigs de la misma especie tendrán una mayor homología a los genes en las bases de datos de una especie



# Estructuración de las secuencias. *Binning*

| Genome binner   | Parameters                                                                          | Model                                                                                                                                                       | Version to validate | Publication | Last update | Resources                                                                                                                     |
|-----------------|-------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------|-------------|-------------|-------------------------------------------------------------------------------------------------------------------------------|
| MaxBin          | k-mer frequencies, coverage, single-copy genes                                      | Expectation-maximization, bin number estimated from single-copy marker gene analysis                                                                        | 2.2.6               | 2014        | 2019        | <a href="https://sourceforge.net/projects/maxbin">https://sourceforge.net/projects/maxbin</a>                                 |
| MetaBat         | 4-mer frequencies, coverage                                                         | Modified K-medoids algorithm                                                                                                                                | 1&2.13              | 2015        | 2020        | <a href="https://bitbucket.org/berkeleylab/metabat/src/master">https://bitbucket.org/berkeleylab/metabat/src/master</a>       |
| Groopm          | coverage, contig's length, tetranucleotide frequency                                | Two way clustering, Hough partitioning, self-organizing map                                                                                                 | 2                   | 2014        | 2017        | <a href="https://github.com/timbalam/GroopM">https://github.com/timbalam/GroopM</a>                                           |
| CONCOCT         | k-mer frequencies, coverage                                                         | Gaussian mixture models, bin number determined by variable Bayesian                                                                                         | 1.0.0               | 2014        | 2019        | <a href="https://github.com/BinPro/CONCOCT">https://github.com/BinPro/CONCOCT</a>                                             |
| MyCC            | k-mer frequencies, coverage (optional), universal single-copy genes                 | Affinity propagation                                                                                                                                        | 1                   | 2016        | 2017        | <a href="https://sourceforge.net/projects/sb2nhri">https://sourceforge.net/projects/sb2nhri</a>                               |
| MetaWatt        | tetranucleotide frequency, coverage                                                 | Firstly clustering by empirical relationship of the average standard deviation at tetranucleotide frequency mean, then employing interpolated Markov models | 3.5.3               | 2012        | 2016        | <a href="https://sourceforge.net/projects/metawatt">https://sourceforge.net/projects/metawatt</a>                             |
| BMC3C           | frequency variation of oligonucleotides, coverage, codon usage                      | Ensemble k-means, construct a weigh graph and partition it by Normalized cuts [49, 50]                                                                      | \                   | 2018        | 2018        | <a href="http://mlda.swu.edu.cn/codes.php?name=BMC3C">http://mlda.swu.edu.cn/codes.php?name=BMC3C</a>                         |
| Binsanity       | coverage, tetranucleotide frequency, percent GC content                             | Affinity propagation                                                                                                                                        | 0.2.8               | 2017        | 2020        | <a href="https://github.com/edgraham/BinSanity">https://github.com/edgraham/BinSanity</a>                                     |
| Autometa        | sequence homology, single-copy genes, 5-mer frequency, coverage, single-copy genes  | Lowest common ancestor analysis, DBSCAN algorithm, supervised decision tree classifier recruit unclustered contigs                                          | \                   | 2019        | 2020        | <a href="https://bitbucket.org/jasong_c_kwan/autometa/src/master">https://bitbucket.org/jasong_c_kwan/autometa/src/master</a> |
| COCACOLA        | k-mer frequency, coverage, co-alignment, paired-end read linkage                    | K-means based on L1 distance, non-negative matrix factorization with sparse regularization, hierarchical clustering                                         | \                   | 2017        | 2017        | <a href="https://github.com/younglululu/COCACOLA">https://github.com/younglululu/COCACOLA</a>                                 |
| SolidBin-naive  | single-copy mark genes, tetranucleotide frequencies, coverage, pairwise constraints | Semi-supervised spectral Normalized cut                                                                                                                     | 1.1                 | 2019        | 2020        | <a href="https://github.com/sufforest/SolidBin">https://github.com/sufforest/SolidBin</a>                                     |
| Vamb            | tetranucleotide frequencies, coverage                                               | Variational autoencoders, iterative medoid clustering algorithm                                                                                             | 2.0.1               | 2018        | 2020        | <a href="https://github.com/RasmussenLab/vamb">https://github.com/RasmussenLab/vamb</a>                                       |
| DAS Tool        | original binner output bin sets                                                     | Refine bins according shared contigs between two original binner results                                                                                    | 1.1.1               | 2018        | 2019        | <a href="https://github.com/cmks/DAS_Tool">https://github.com/cmks/DAS_Tool</a>                                               |
| MetaWrap        | original binner output bin sets                                                     | Separating every pair of contigs in different bins, selecting the best bin sets according completion and contamination                                      | 1.2.2               | 2018        | 2019        | <a href="https://github.com/bxlab/metaWRAP">https://github.com/bxlab/metaWRAP</a>                                             |
| Binning_refiner | original binner output bin sets, single-copy genes                                  | Scoring bins based on single-copy genes and picking up high-score bins iteratively                                                                          | 1.4.0               | 2017        | 2019        | <a href="https://github.com/songweizhi/Binning_refiner">https://github.com/songweizhi/Binning_refiner</a>                     |

## **Tema 8: Caracterización de poblaciones complejas**

8.1 El problema de las poblaciones complejas en salud.

8.2 Estrategias dirigidas. Metataxonomía.

8.3 Estrategias no dirigidas. Ensamblaje *de novo*.

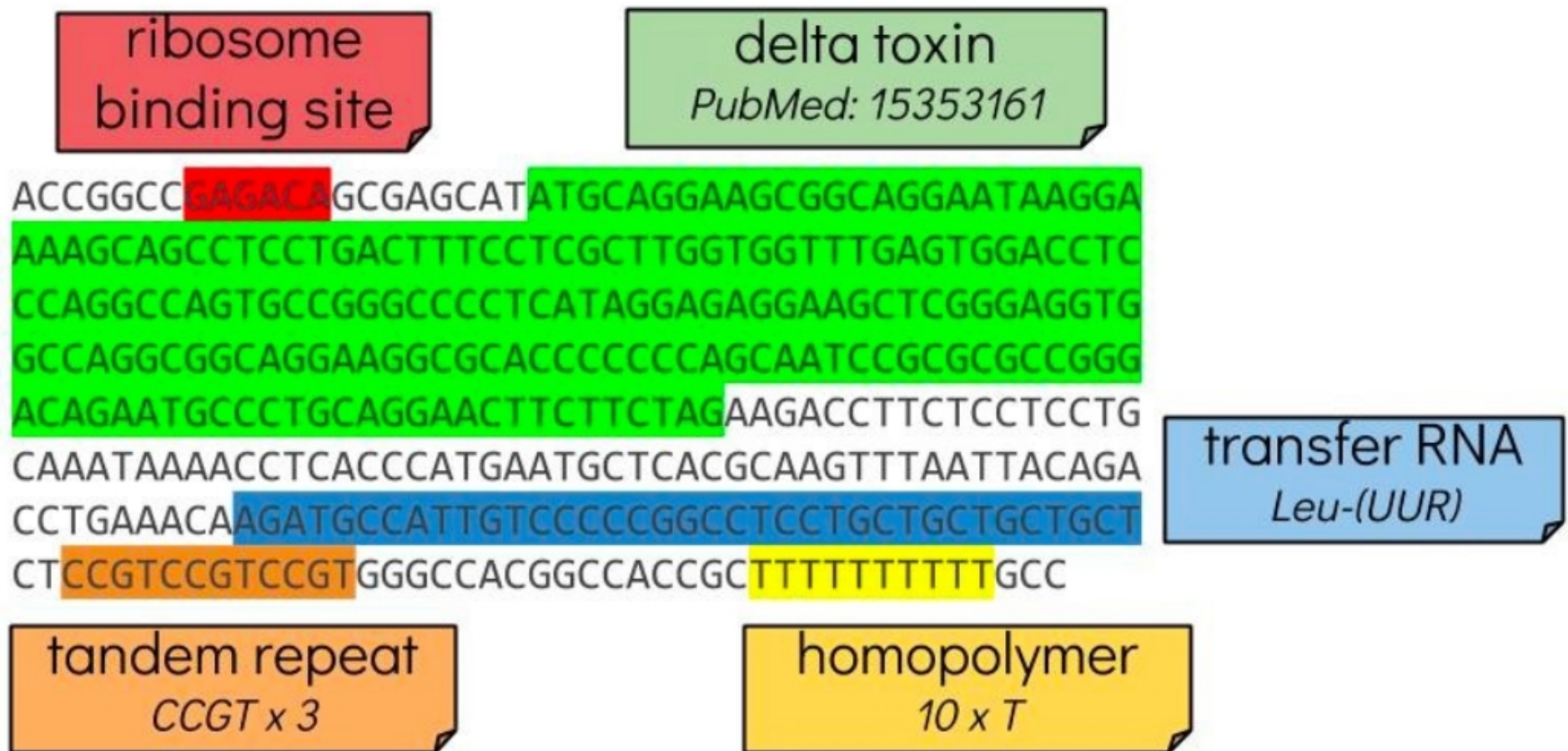
8.4 Estructuración de secuencias. *Binning*

**8.5 Identificación de elementos funcionales.**

8.6 Inferencia de funciones.

# Identificación de elementos funcionales

Una vez ensambladas y separadas las secuencias, el siguiente paso consiste en la identificación de los genes y los elementos funcionales de una secuencia.

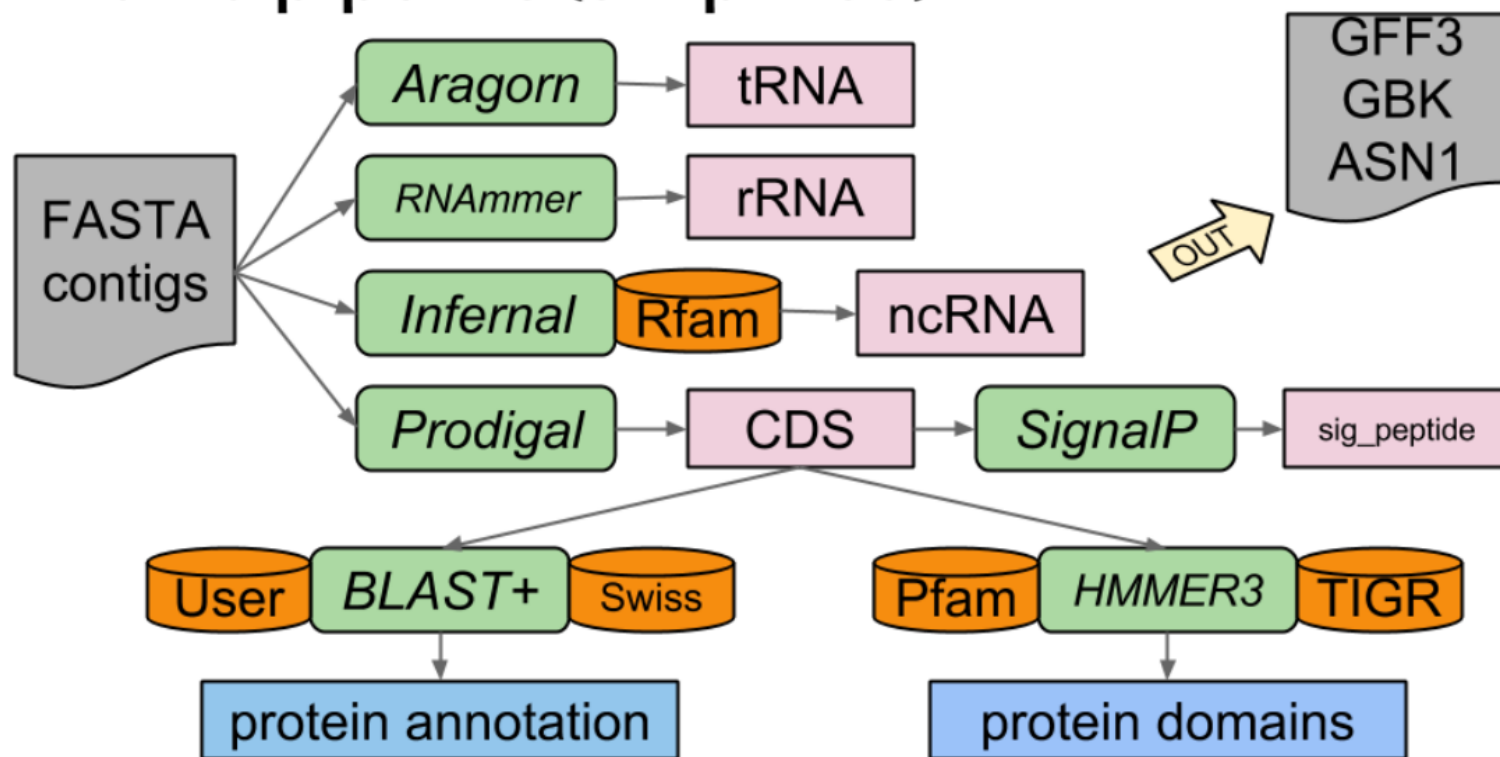


# Identificación de elementos funcionales

La identificación de todos los elementos funcionales de una secuencia típicamente es un proceso complejo que implica el análisis con diversas herramientas informáticas así como la integración supervisada de todos los resultados.



## Prokka pipeline (simplified)

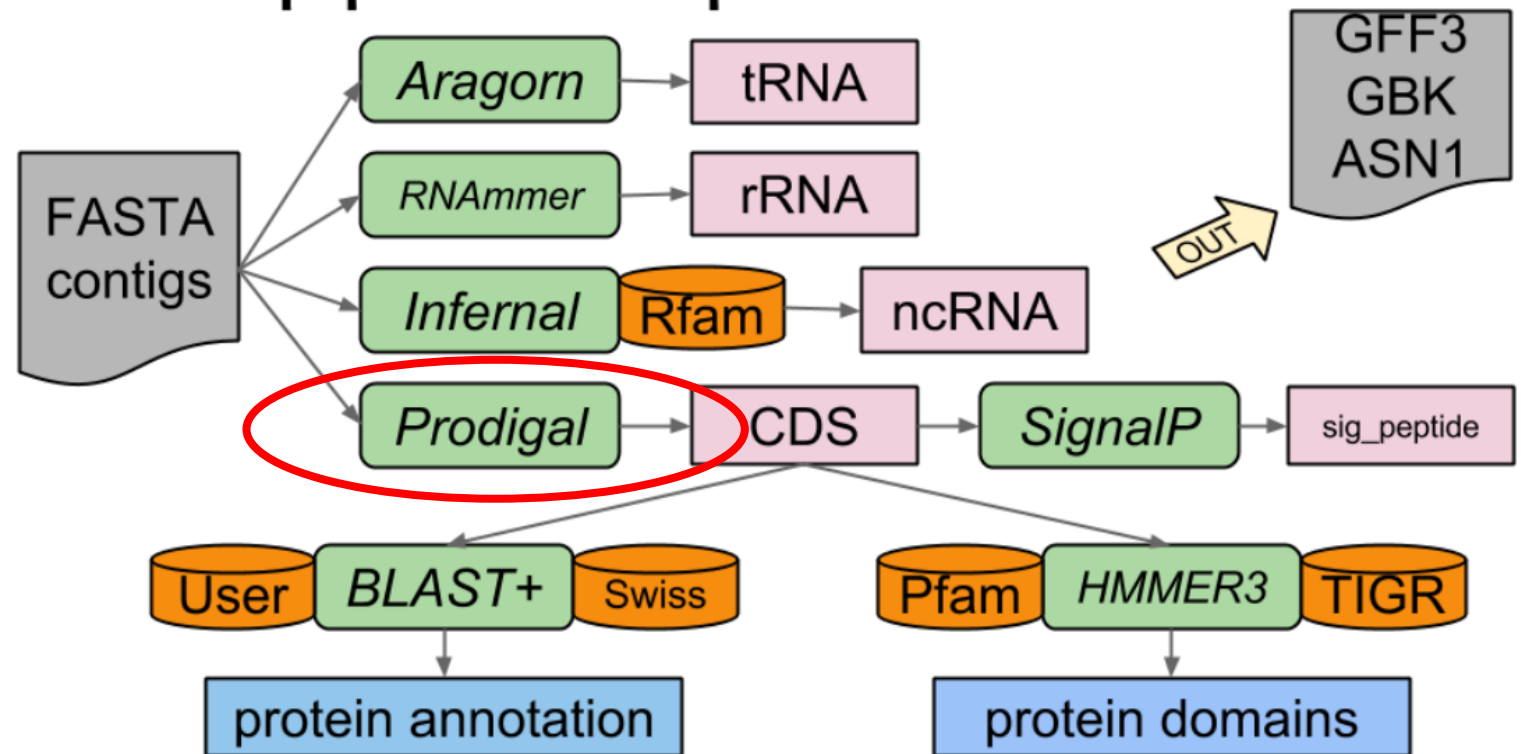


# Identificación de elementos funcionales

La identificación de todos los elementos funcionales de una secuencia típicamente es un proceso complejo que implica el análisis con diversas herramientas informáticas así como la integración supervisada de todos los resultados.



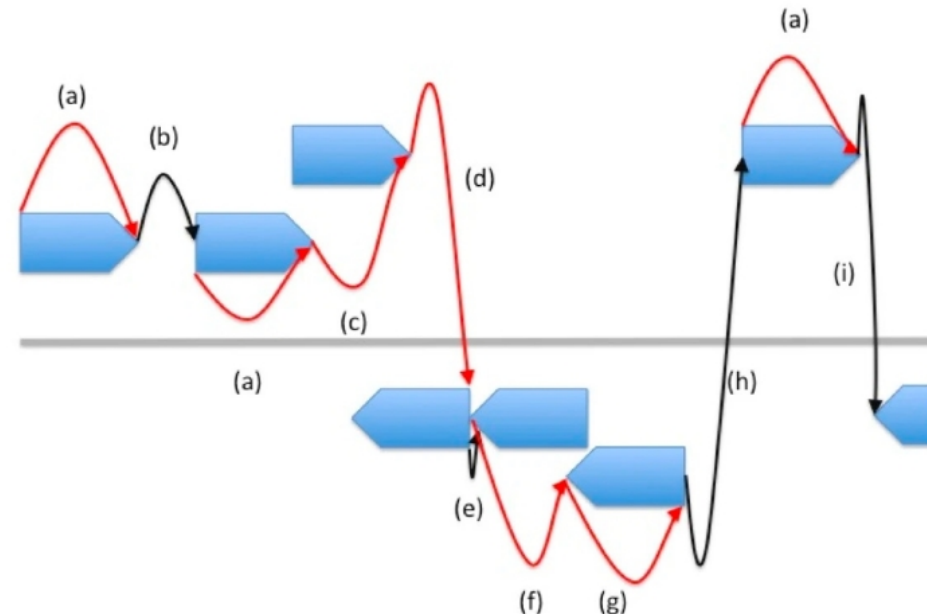
## Prokka pipeline (simplified)



# Identificación de elementos funcionales

**Prodigal** es una herramienta diseñada para identificar pautas abiertas de lectura (ORFs) mediante diversos pasos de identificación de sitios de inicio y final de la traducción así como posibles traducciones de estas secuencias en las distintas pautas de lectura para identificar puntos prematuros de parada

1. Read in the sequence
2. Locate all starts and stops in the genome
3. Scan all open reading frames and record numbers of G's and C's in each codon position
4. Build a frame bias model based on ORF length and G/C codon position within each ORF
5. Record the highest scoring start nodes in each frame that overlap a stop codon by  $\leq 60$  bp
6. Do the first pass dynamic programming, connecting nodes based on frame bias scores
7. Create a hexamer background of all 6-mers in the entire sequence
8. FOR each gene model in the dynamic programming output:
  1. Gather all hexamer statistics
  2. Create log table of hexamer coding scores
10. FOR each gene model in the dynamic programming output:
  1. Calculate a coding score based on hexamer statistics
  2. Penalize the score if there is a higher scoring start upstream in the same ORF
  3. IF the gene is very long but has a negative score, THEN give it a barely positive score
11. FOR 10 iterations
  1. Build a ribosomal binding site and ATG/GTG/TTG background for all nodes
  2. FOR each gene with a score of  $> 35.0$ :
    1. Gather its Shine-Dalgarno RBS motif data and ATG/GTG/TTG data
  3. Modify RBS and ATG/GTG/TTG weights by the observations
12. IF organism is not determined to use Shine-Dalgarno THEN run the non-SD finder
13. FOR each gene model:
  1. Assign a final score of start score + coding score
  2. Penalize the final score of genes  $< 250$ bp
14. Do the second pass dynamic programming, connecting nodes based on hexamer coding
15. FOR each gene model in the final dynamic programming:
  1. Eliminate negative scoring models
  2. Resolve very close start pairs ( $\leq 15$  bp from each other)
16. Print final output



## Tema 8: Caracterización de poblaciones complejas

8.1 El problema de las poblaciones complejas en salud.

8.2 Estrategias dirigidas. Metataxonomía.

8.3 Estrategias no dirigidas. Ensamblaje *de novo*.

8.4 Estructuración de secuencias. *Binning*

8.5 Identificación de elementos funcionales.

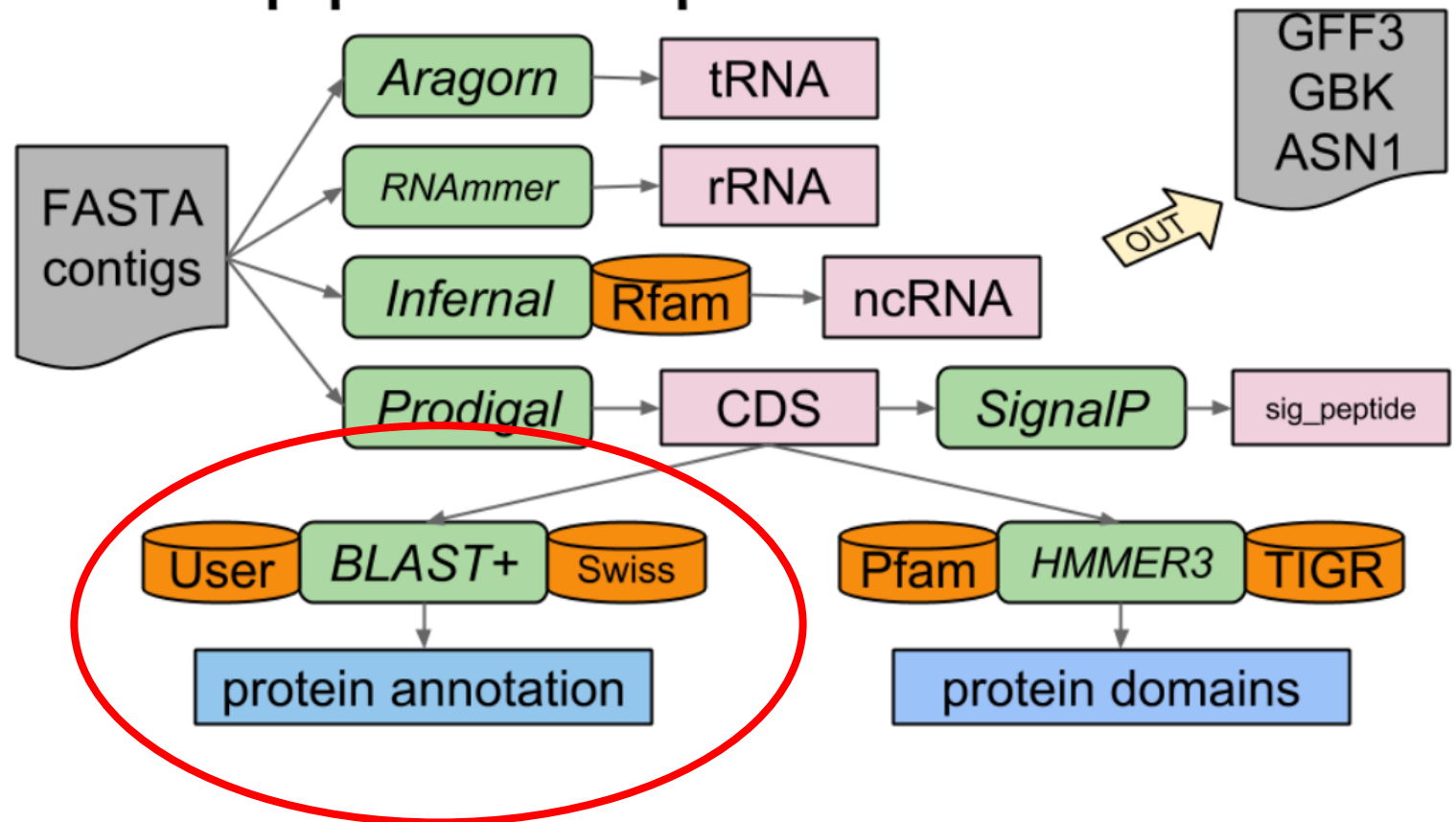
**8.6 Inferencia de funciones.**

# Identificación de elementos funcionales

La identificación de todos los elementos funcionales de una secuencia típicamente es un proceso complejo que implica el análisis con diversas herramientas informáticas así como la integración supervisada de todos los resultados.



## Prokka pipeline (simplified)



# Inferencia funcional de los genes

La herramienta **BLAST** busca secuencias parecidas en una base de datos de secuencias identificadas y conocidas en todas las especies descritas.

# BLAST

## Basic Local Alignment Search Tool



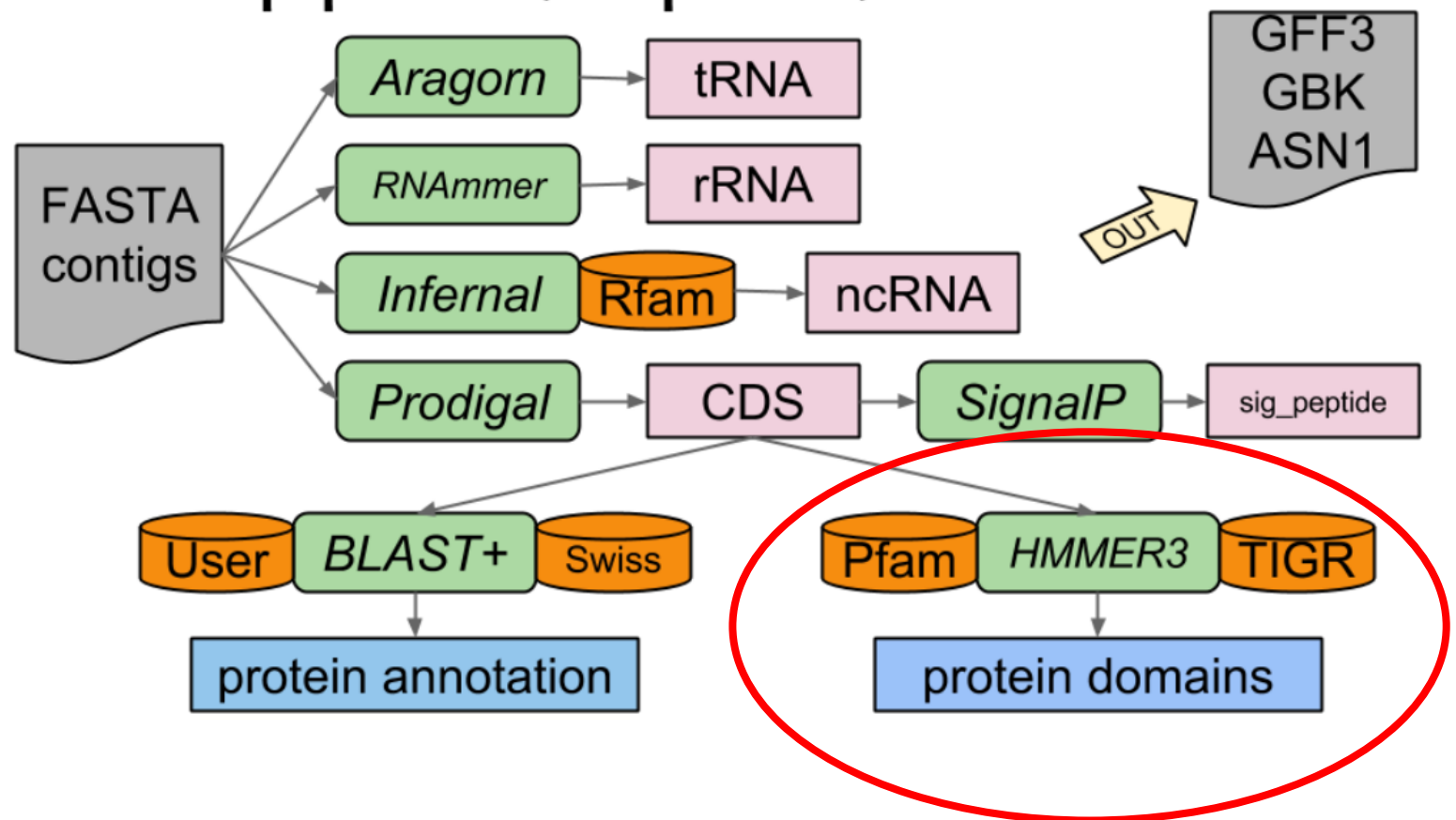
| Nature           | Program | Query                 | Database              | Program | Query Type                                                                              | Subject Type                                                                                                                                                                  | Computation |
|------------------|---------|-----------------------|-----------------------|---------|-----------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|
| Nucleotide BLAST | blastn  | Nucleotide (DNA, RNA) | Nucleotide (DNA, RNA) | blastn  | N  |   N | ~ 1X        |
| Protein BLAST    | blastp  | Protein               | Protein               | blastp  | P  |   P | ~ 1X        |
| Mixed BLAST      | blastx  | Translated nucleotide | Protein               | blastx  | N  |   P | ~ 6X        |
|                  | tblastn | Protein               | Translated nucleotide | tblastn | P  |   N | ~ 6X        |
|                  | tblastx | Translated nucleotide | Translated nucleotide | tblastx | N  |   N | ~36X        |

# Identificación de elementos funcionales

La identificación de todos los elementos funcionales de una secuencia típicamente es un proceso complejo que implica el análisis con diversas herramientas informáticas así como la integración supervisada de todos los resultados.



## Prokka pipeline (simplified)

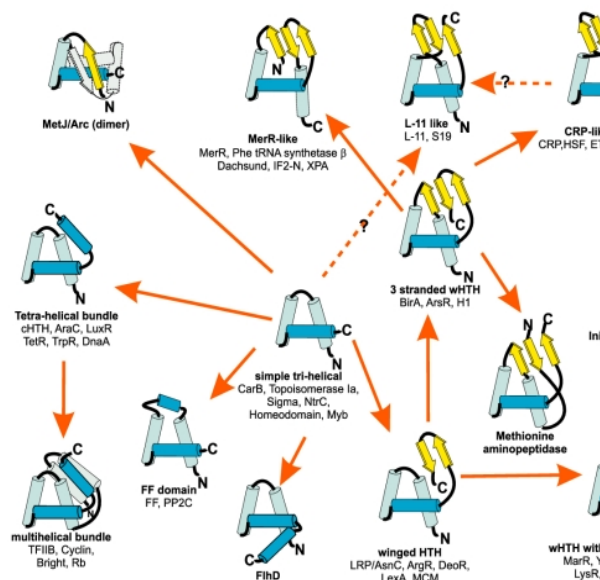
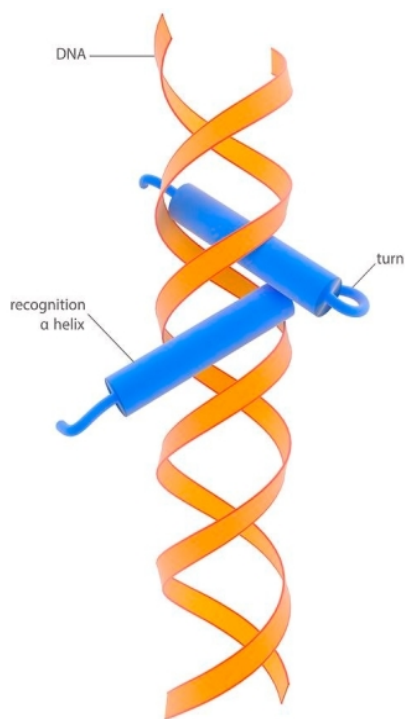


# Inferencia funcional de los genes

# Pfam

Bases de datos  
de HMM

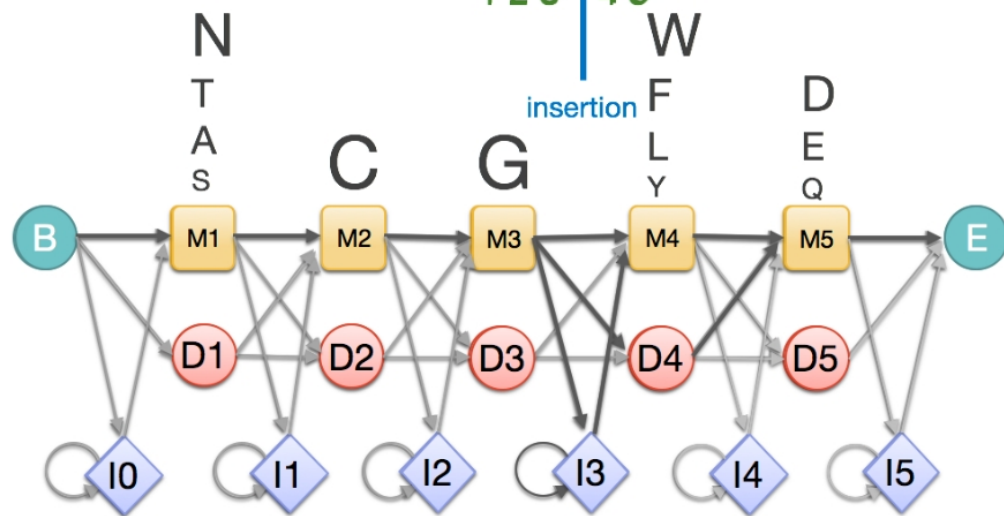
Existen distintos algoritmos que buscan patrones de aminoácidos similares a los ya encontrados en proteínas conocidas. Proteínas con funciones parecidas deben tener estructuras tridimensionales similares.



seq1 A C G - L D  
seq2 S C G - - E  
seq3 N C G g F D  
seq4 T C G - W Q

1 2 3 4 5

deletion



# Bioinformática y análisis de datos ómicos

