

# **Introducción a la Bioinformática**

**Sequencing Methods**

**Sequence Assembly**

**Short Read Mapping**

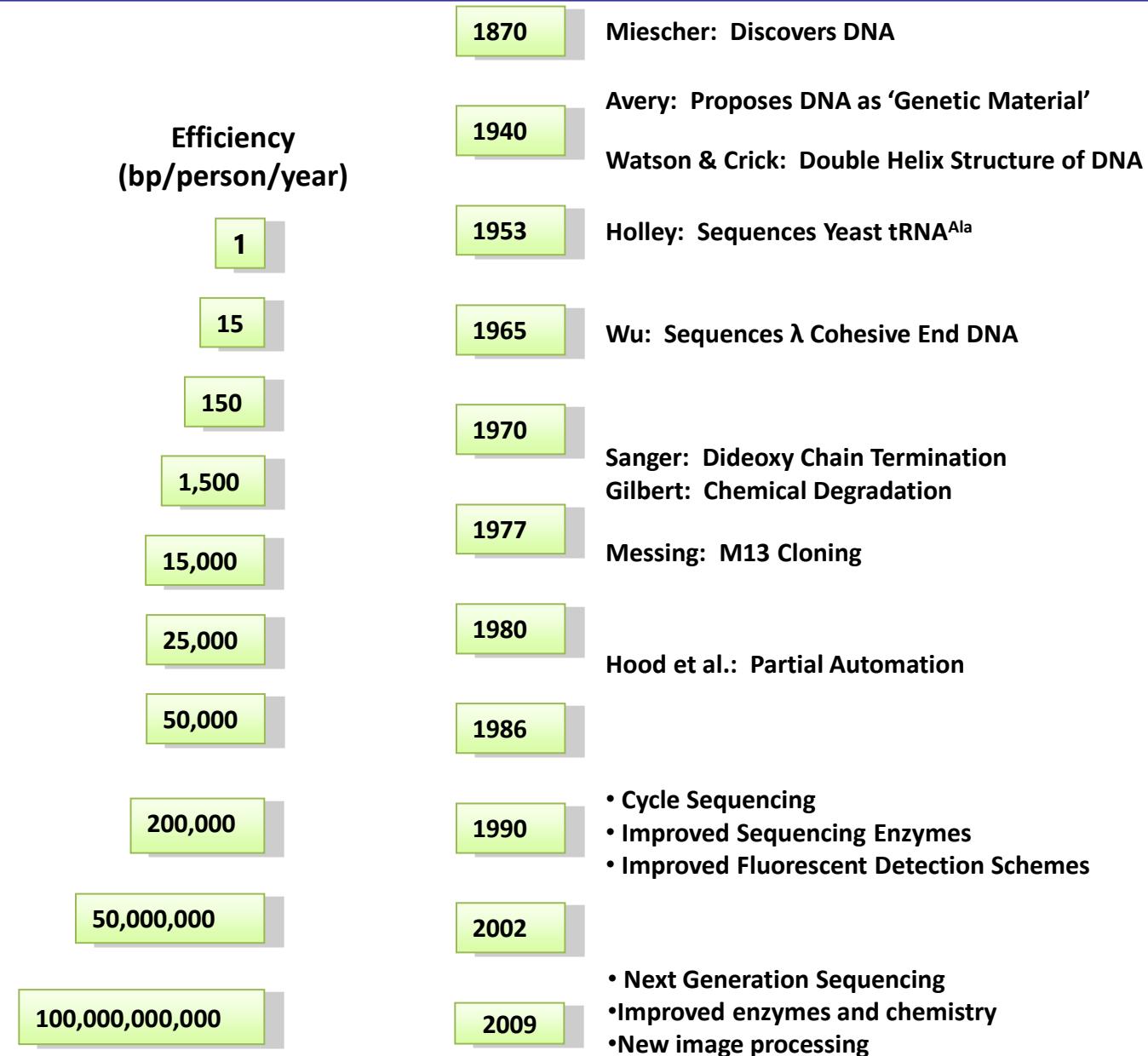
**Fernán Agüero**

Instituto de Investigaciones Biotecnológicas

Universidad Nacional de General San Martín

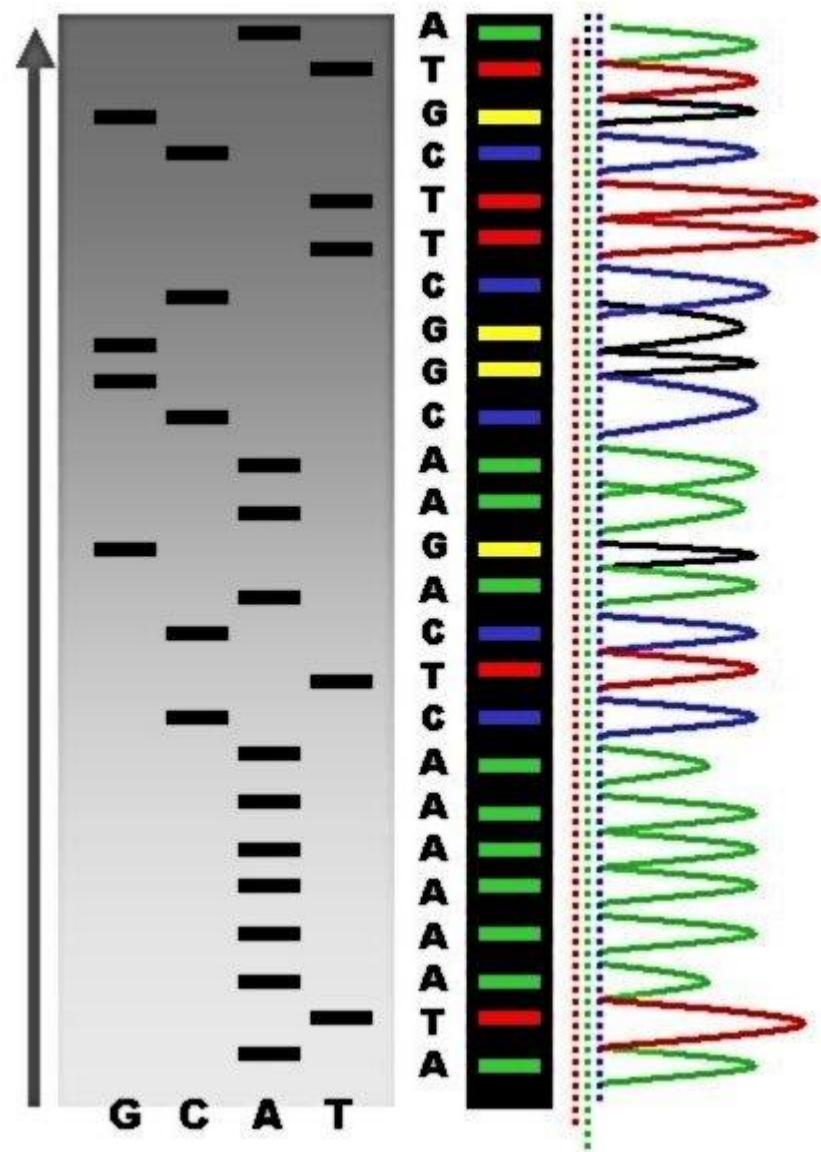
[fernан@iib.unsam.edu.ar](mailto:fernан@iib.unsam.edu.ar)

# Next-gen sequencing



# *Sanger sequencing (old technology)*

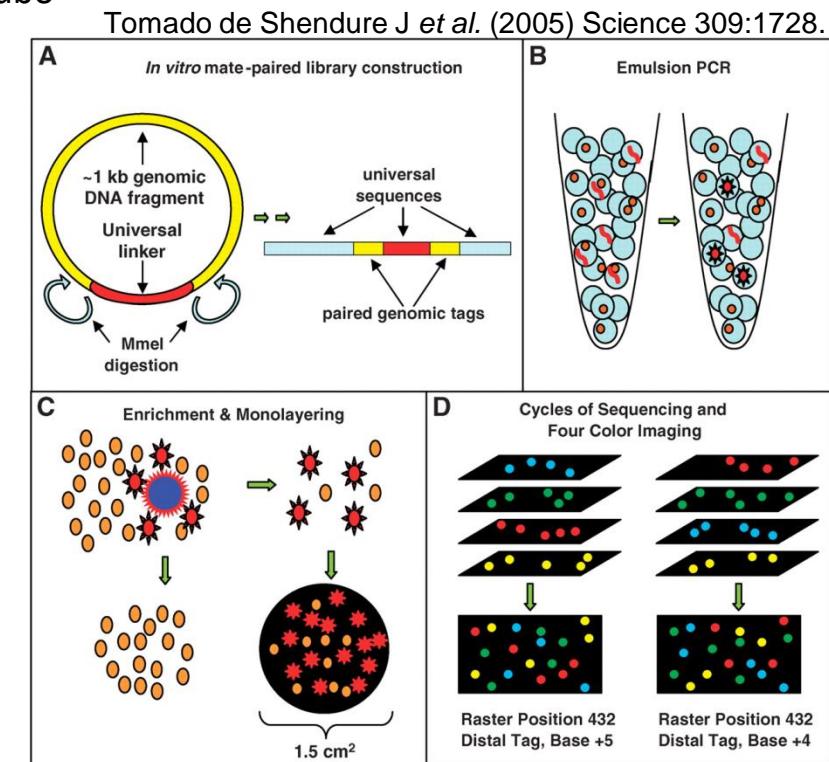
- Clonar el DNA.
- Generar una escalera de moléculas **etiquetadas** (con fluoróforos) o marcadas radioactivamente
- Cada fragmento difiere en 1 nucleótido del proximo
- Separar la mezcla en alguna matriz (electroforesis).
- Detectar cada fragmento
- Interpretar los picos de emisión como una cadena de bases (DNA).
- Se generan cadenas de 500 a 1,000 bases de longitud
- 1 secuenciador genera ~ 57,000 nucleotidos/corrida
- Ensamblar las cadenas en un **todo**



# New sequencing technologies

- Breakthrough

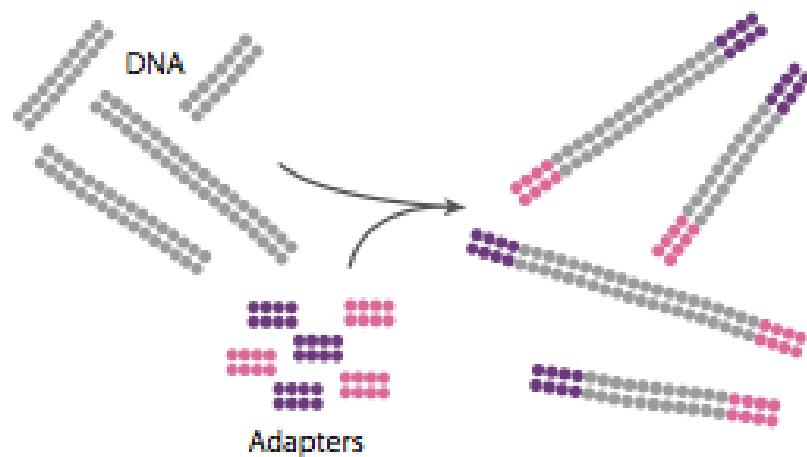
- Polymerase colonies – *polony / polonies*
  - In situ localized amplification and contact replication of many individual DNA molecules. Mitra RD, Church GM. (1999) Nucleic Acids Res. 27: e34.
- Se elimina la necesidad de clonar moléculas en *E. coli*
- Multiplex-amplification, manteniendo agrupamiento físico de amplicones idénticos
  - Se *amplifican* y *clonian* moléculas en el tubo
  - Emulsion-PCR (beads)
  - *In situ* polonies (matrix)



# New sequencing technologies: solexa

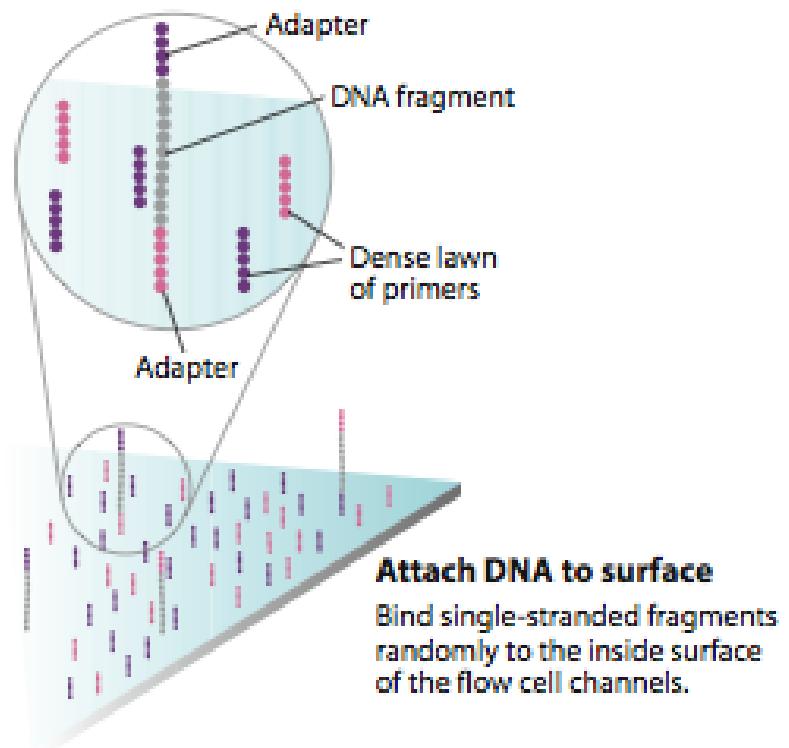
- Construcción de bibliotecas
- Attachment al soporte

a



## Prepare genomic DNA sample

Randomly fragment genomic DNA and ligate adapters to both ends of the fragments.

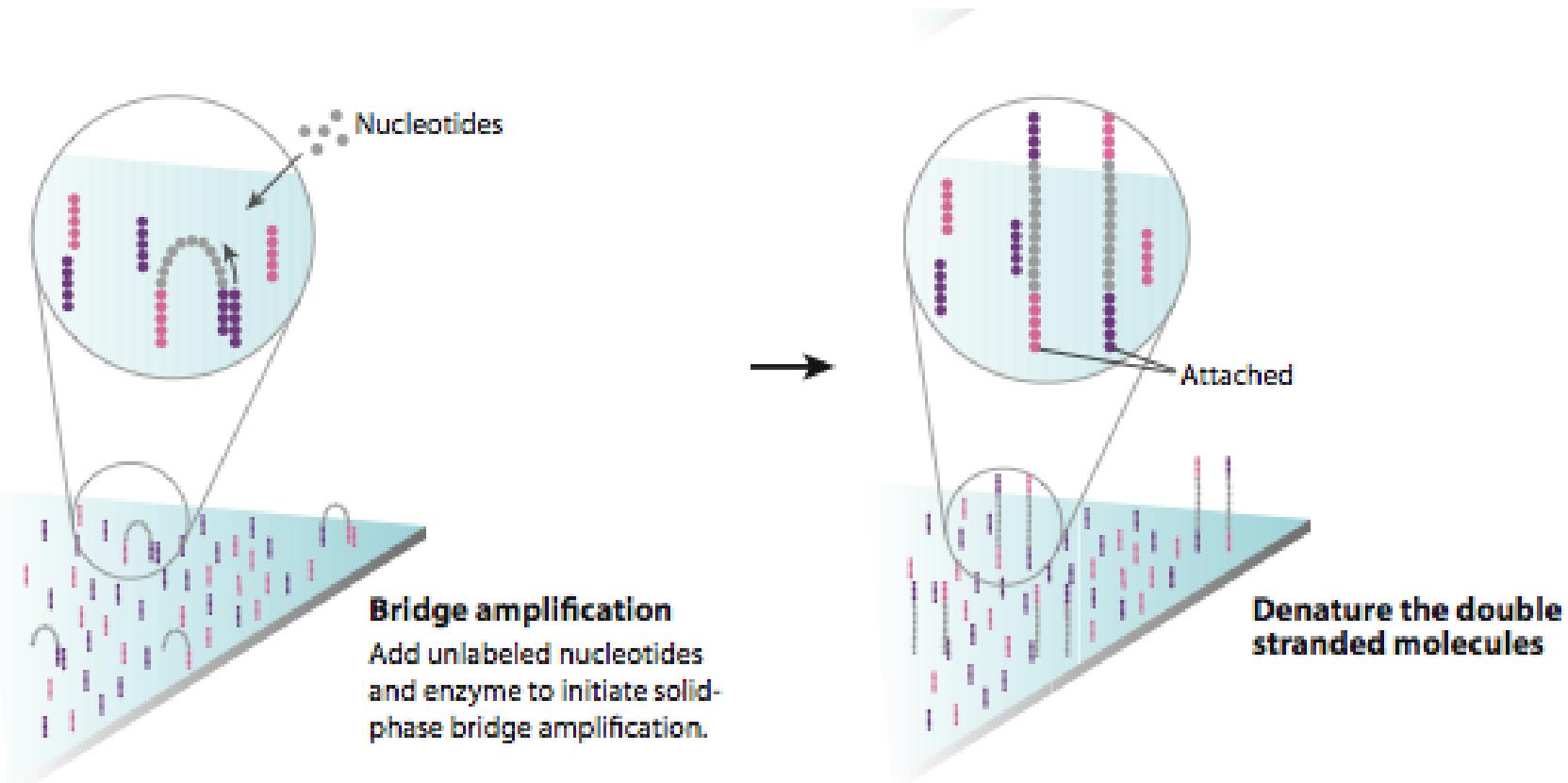


## Attach DNA to surface

Bind single-stranded fragments randomly to the inside surface of the flow cell channels.

# New sequencing technologies: solexa

- Amplificación de las colonias



# New sequencing technologies: solexa

- Reacciones de extensión + lectura del slide usando laser

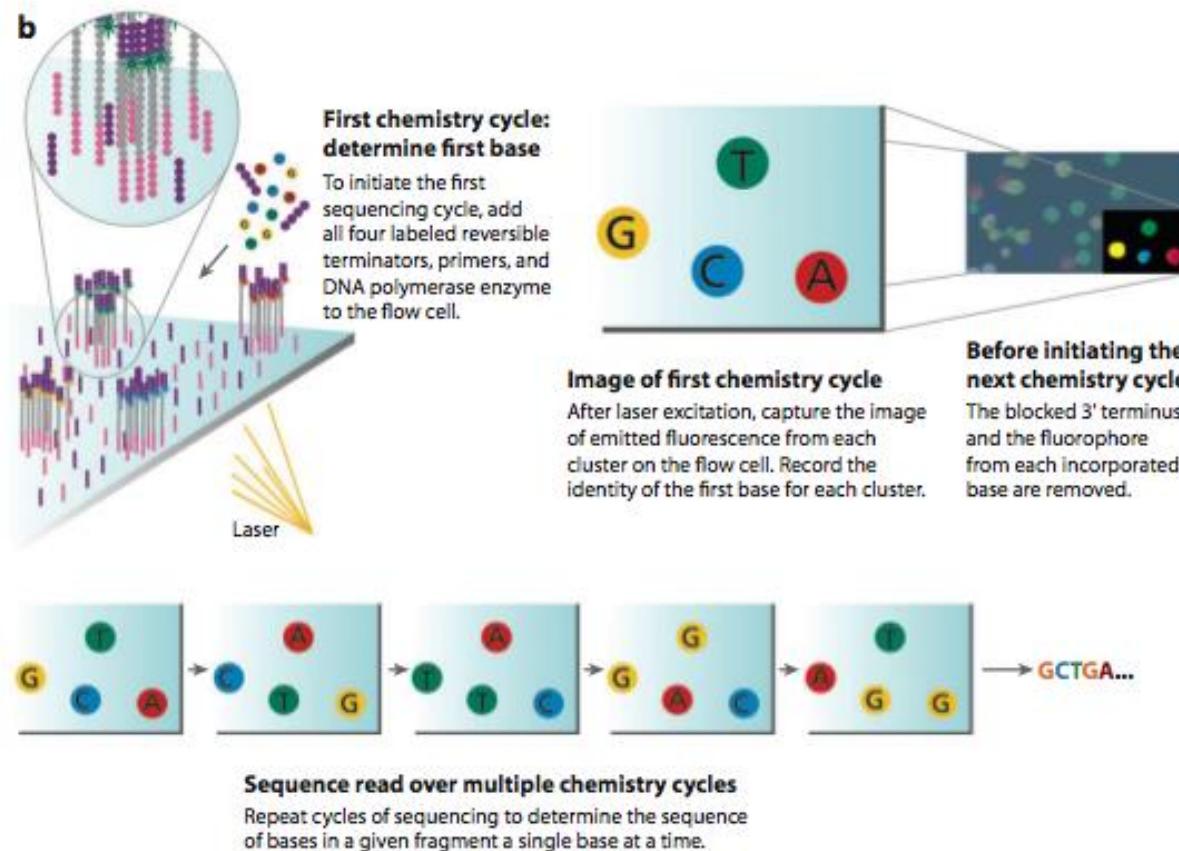
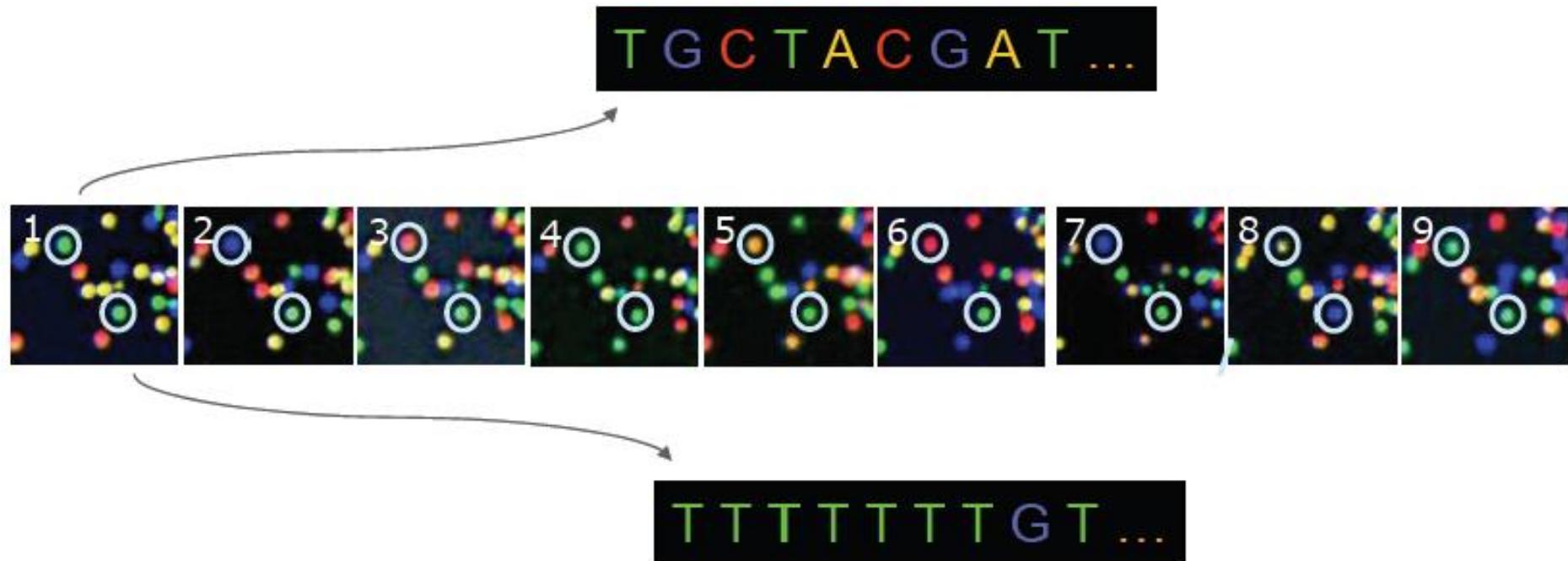


Figure 2  
(Continued)

# New sequencing technologies: solexa

- Base calling – Asignación de bases en la secuencia



# New sequencing technologies: SOLiD

- ABI SOLiD
  - Sequencing by Oligonucleotide Ligation and Detection



# New sequencing technologies: 454

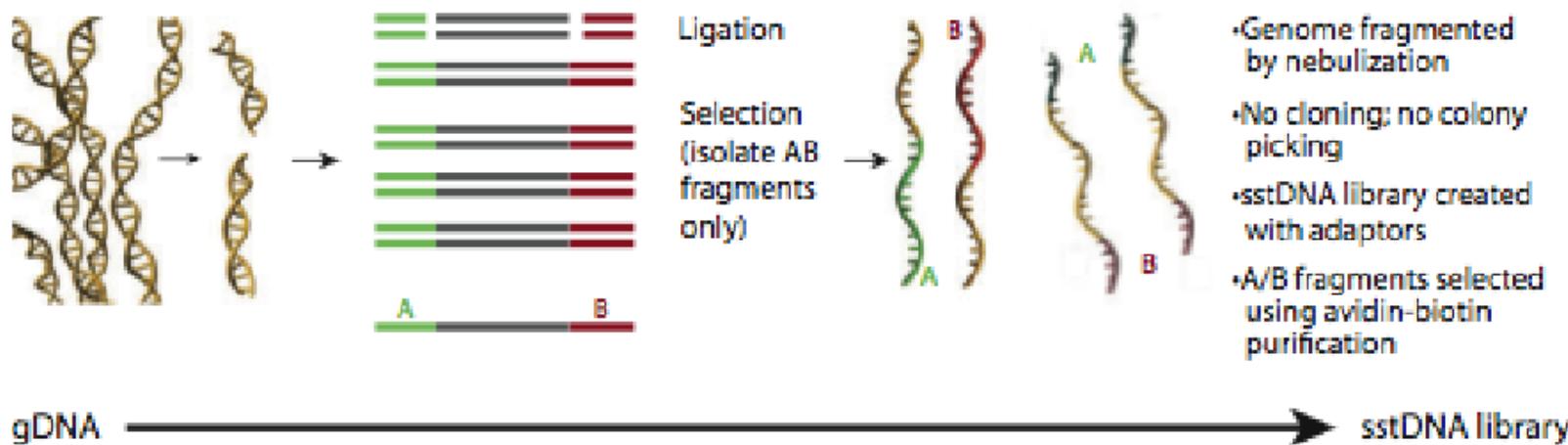
- Roche 454
  - Also known as *pyrosequencing*
  - 500 million bp/run
  - 10 hr/run
  - 400-500 bp/read & > 1 M reads



a

## DNA library preparation

4.5 hours



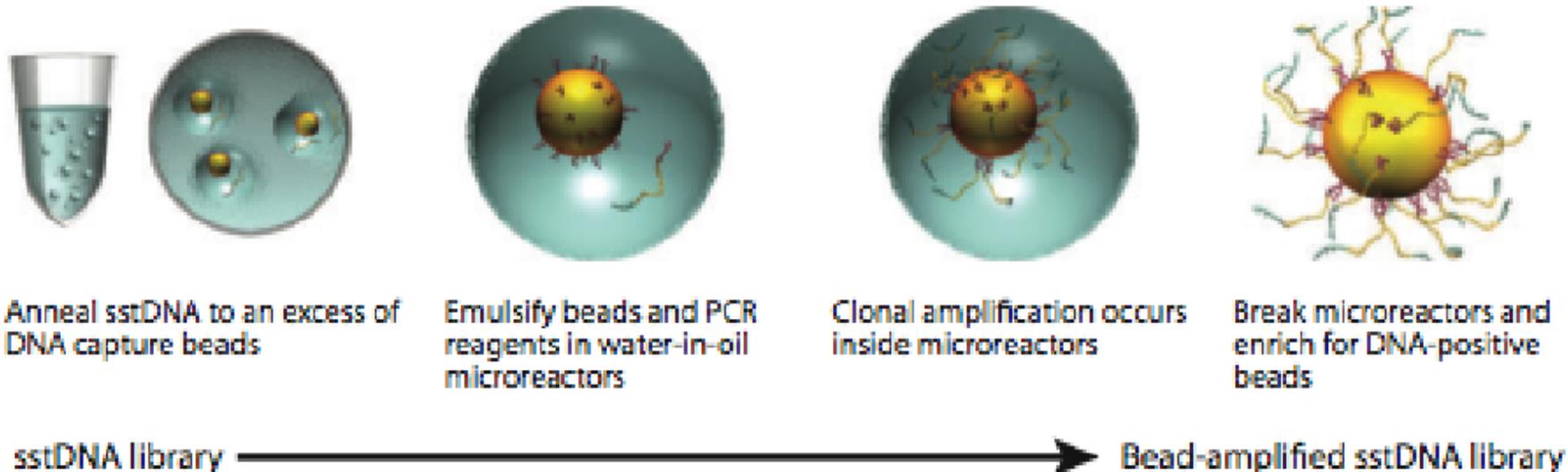
# New sequencing technologies: 454

- PCR en emulsión

b

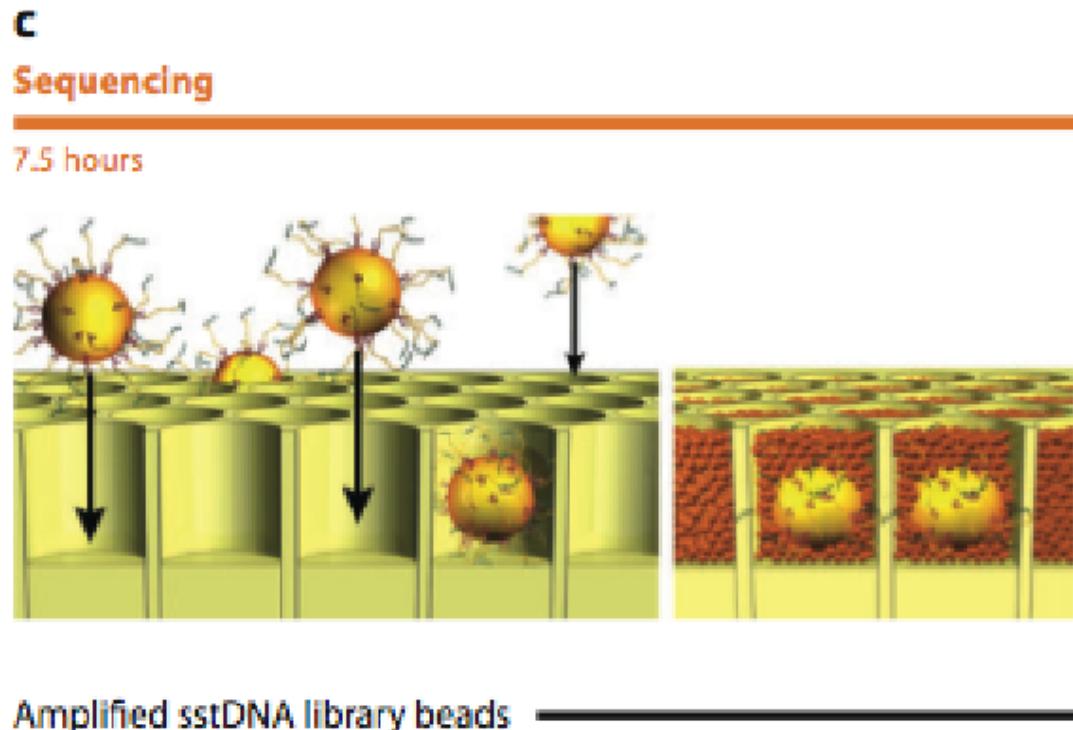
## Emulsion PCR

8 hours



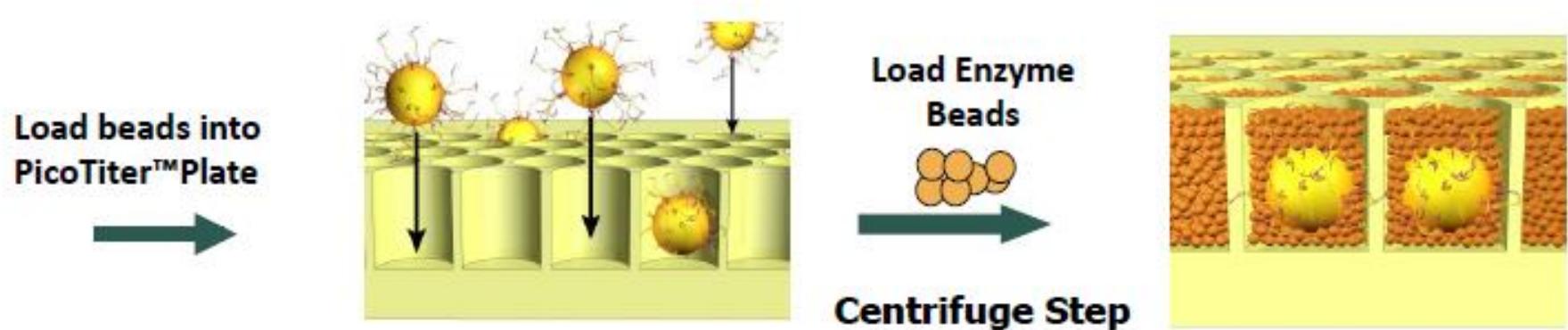
# New sequencing technologies: 454

- Secuenciación en nanowells

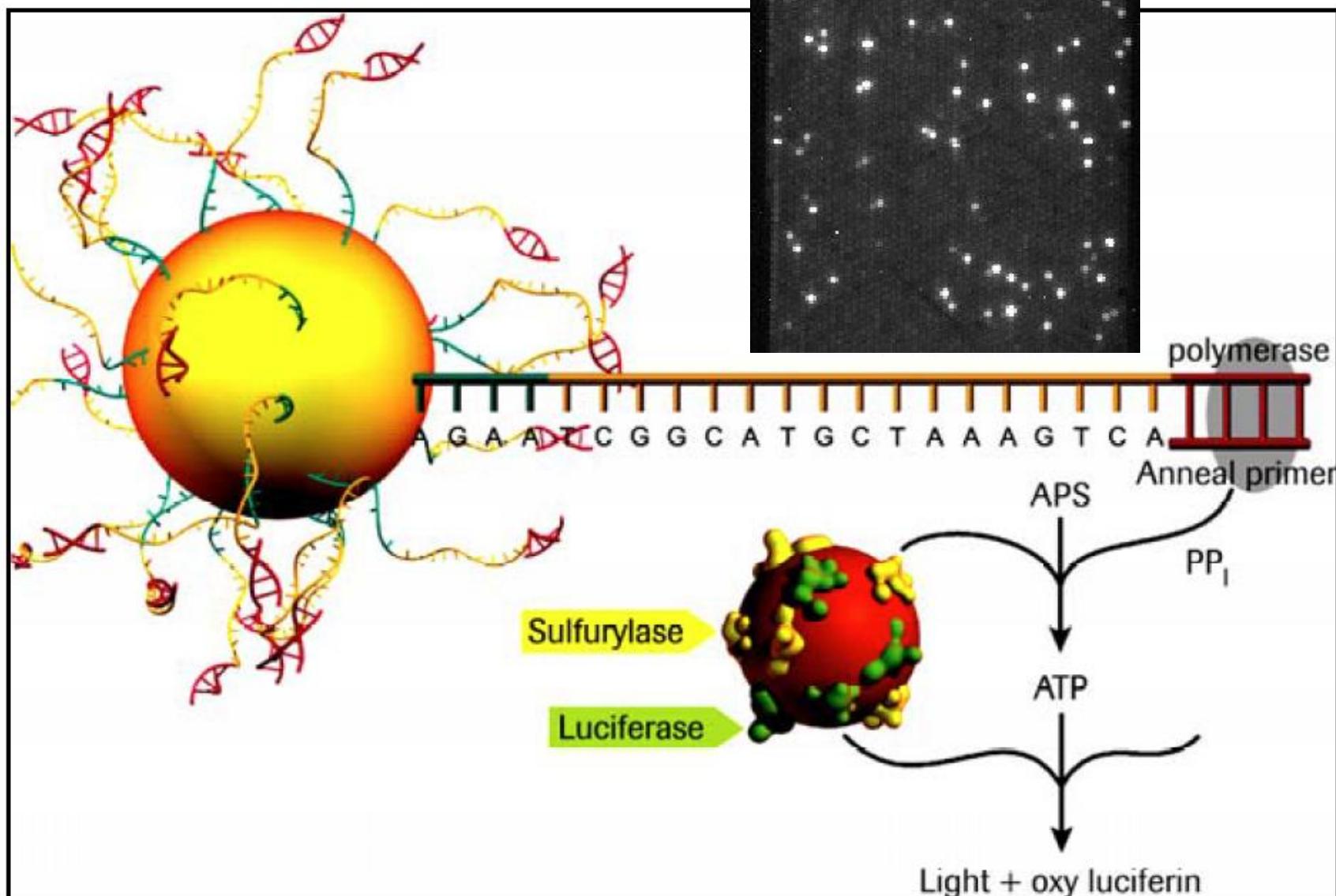


- Well diameter: average of 44  $\mu\text{m}$
- 400,000 reads obtained in parallel
- A single cloned amplified ssDNA bead is deposited per well

# 454 sequencing explained



# 454 sequencing explained

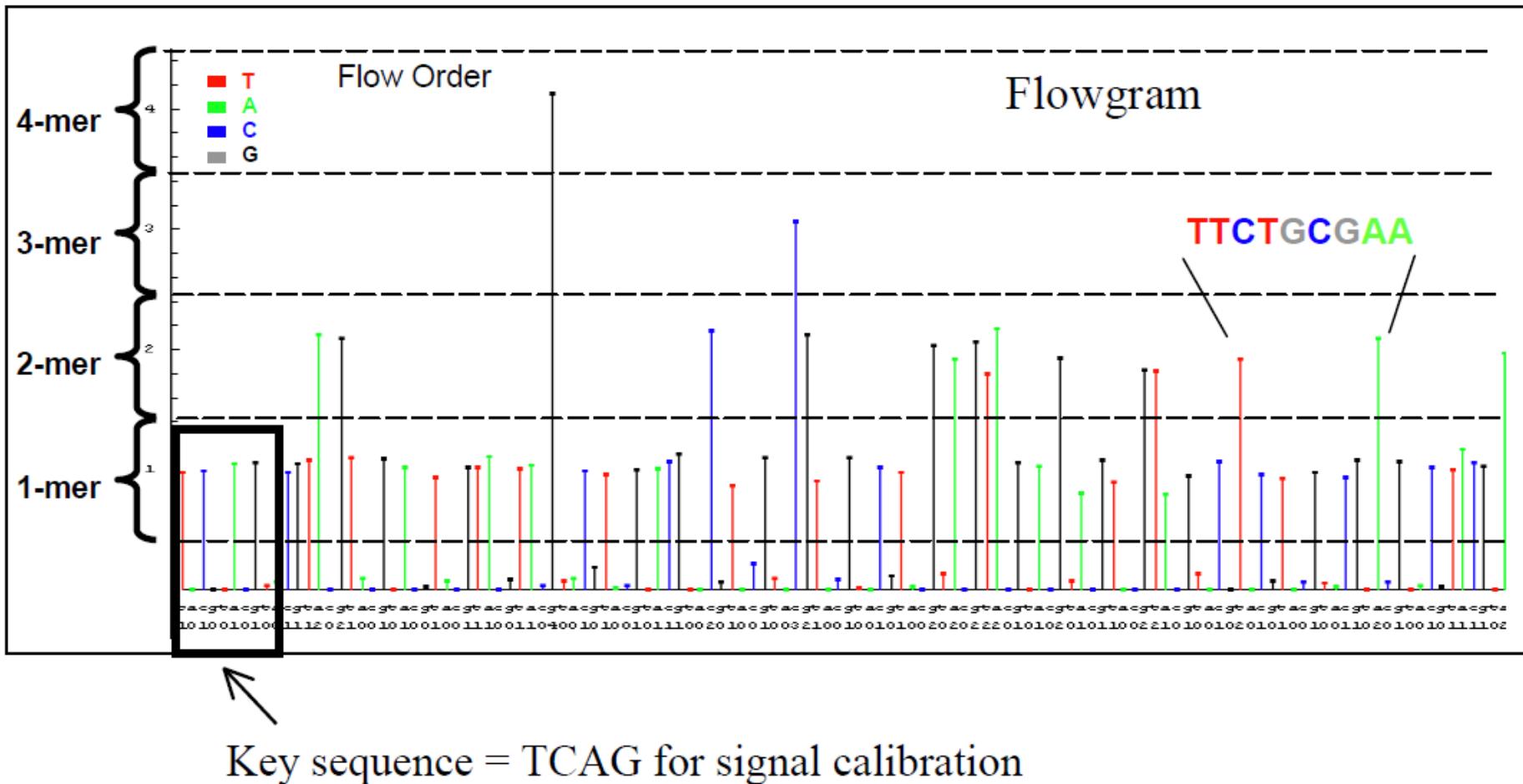


# 454 sequencing explained

- Cada base se inyecta en forma secuencial en la platina de reacción (PicoTiter Plate), de a una por vez
  - Por ejemplo, 100 veces para un secuenciador 454-FLX
- Si el nucleótido es complementario al molde, se polimeriza en la cadena naciente. La reacción genera pirofosfato, que es transformado en una señal luminosa
- La señal es leída por una cámara
- La intensidad de la señal es proporcional al número de nucleótidos incorporados
  - Si hay 3 'T' en el molde, la luz emitida va a ser ~ 3 veces la de una sola 'T'
- La secuencia se lee a partir de un '*flowgram*'

*Margulies M, et al (2005) Genome sequencing in microfabricated high-density picolitre reactors. Nature DOI: [10.1038/nature03959](https://doi.org/10.1038/nature03959)*

# A 454 flowgram

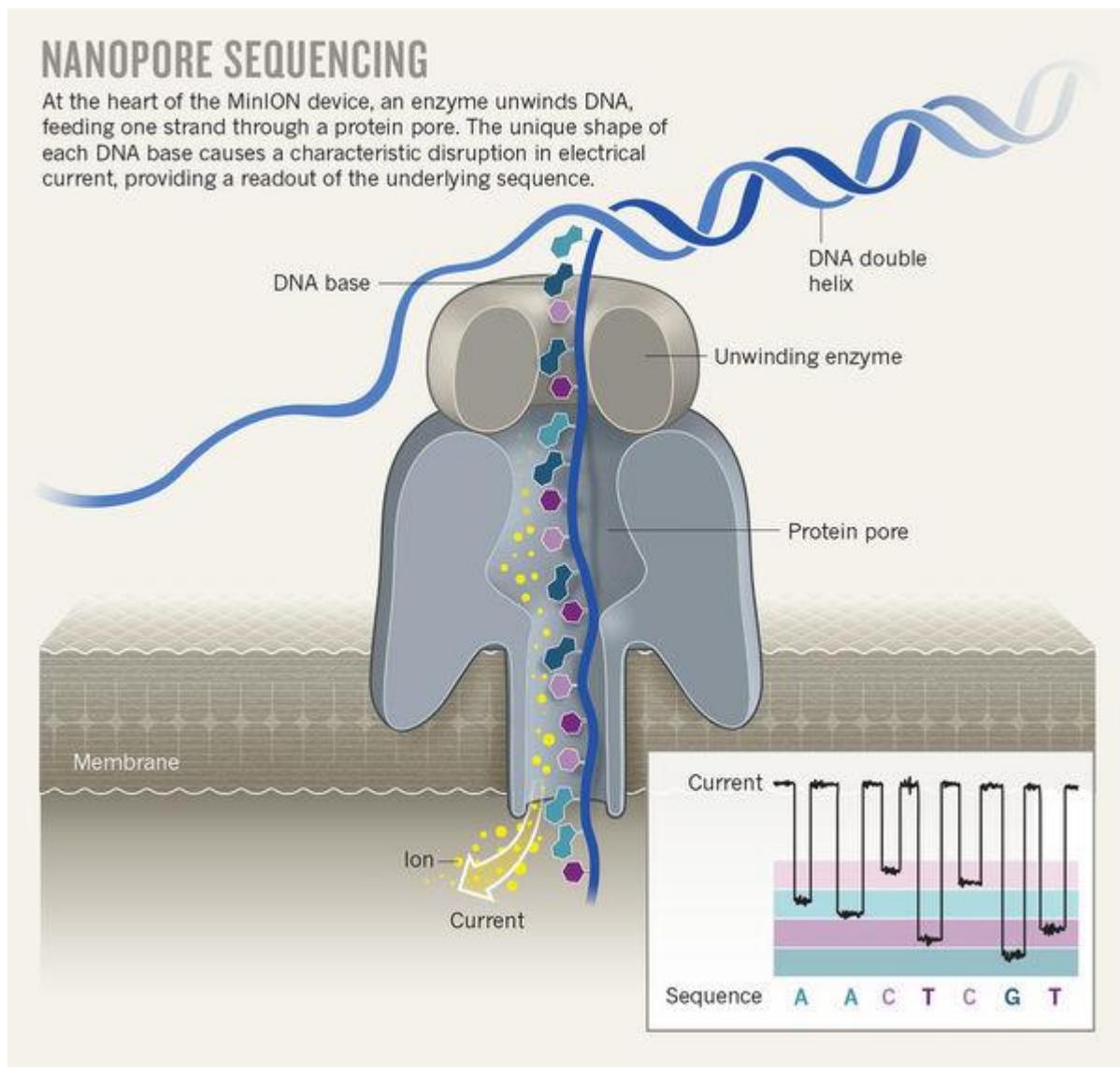


# Oxford Nanopore

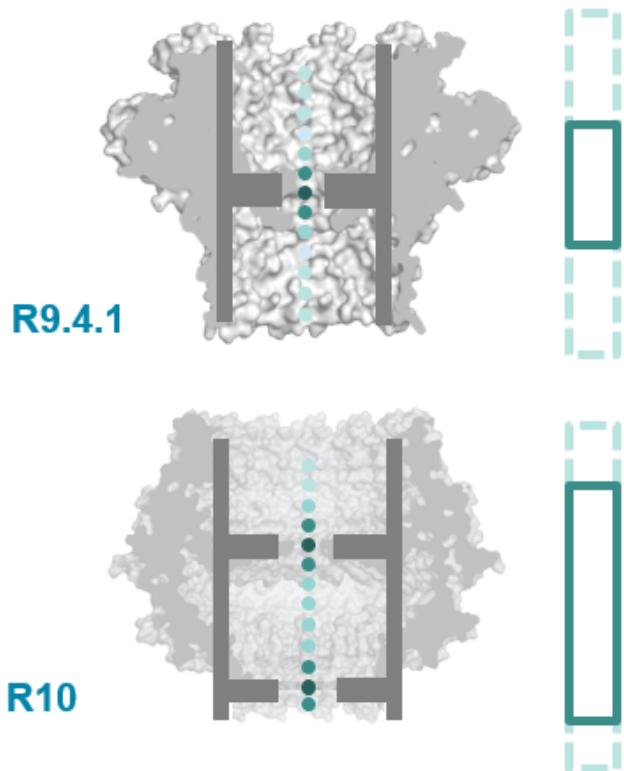
A **nanopore** is a nano-scale hole. In its devices, Oxford

**Nanopore** passes an ionic current

through **nanopores** and measures the changes in current as biological molecules pass through the **nanopore** or near it. The information about the change in current can be used to identify that molecule.



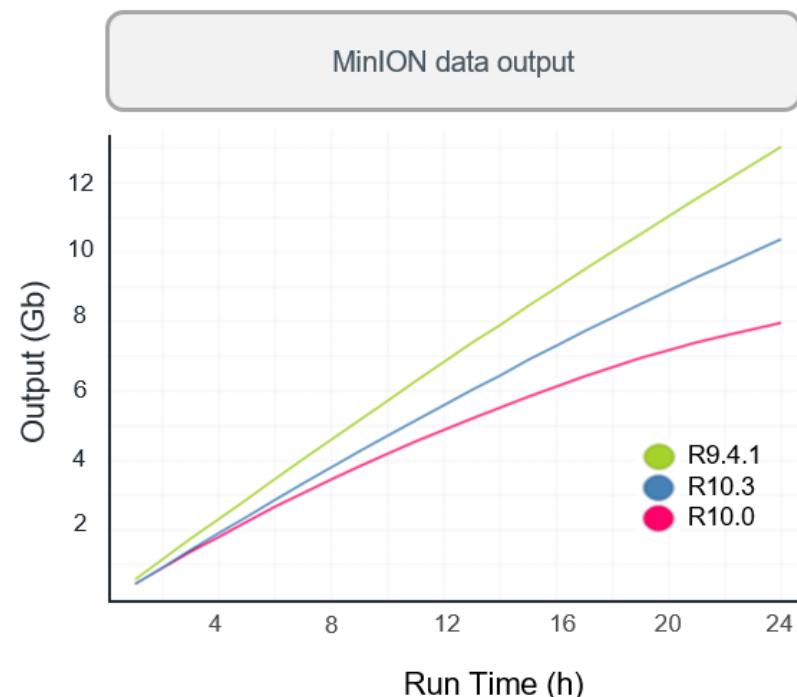
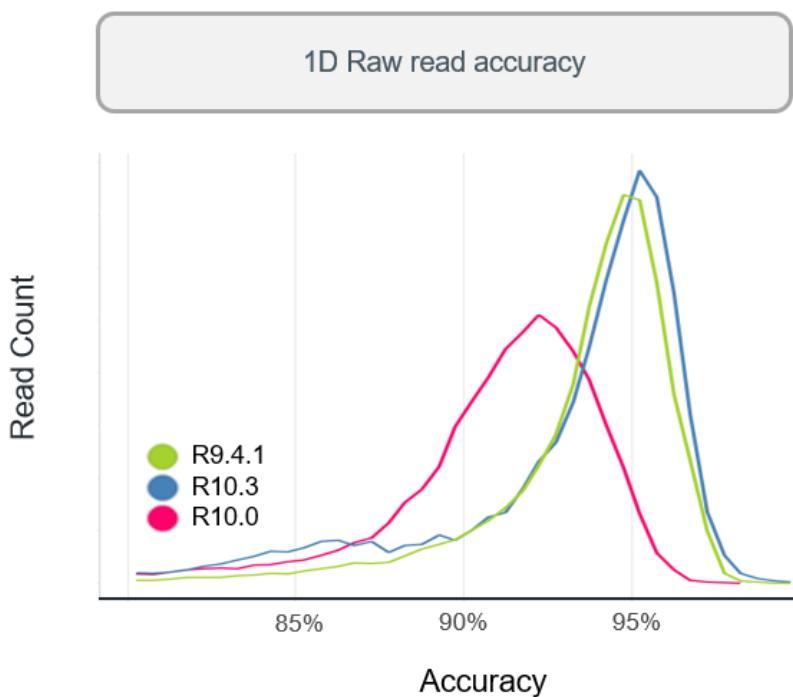
La evolución de la química de secuenciación incluye el uso de distintos tipos de nanoporos que van mejorando precisión de lectura, velocidad, throughput, etc.



R10 = Marzo, 2019,

<https://nanoporetech.com/about-us/news/r103-newest-nanopore-high-accuracy-nanopore-sequencing-now-available-store>

La evolución de la química de secuenciación incluye el uso de distintos tipos de nanoporos que van mejorando precisión de lectura, velocidad, throughput, etc.



Input Requirements:	R9.4.1	R10.3	R10.0
5 – 50 fmol	25 – 75 fmol	50 – 100 fmol	

# Pacific Biosciences (PacBio)

- The zero-mode waveguide (ZMW) is a nanophotonic confinement structure
- ZMW holes are ~70 nm in diameter and ~100 nm in depth.
- Due to the behavior of light when it travels through a small aperture, the optical field decays exponentially inside the chamber.
- The volume in a ZMW is ~20 zeptoliters ( $20 \times 10^{-21}$  liters)
- Within this volume, the activity of DNA polymerase incorporating a single nucleotide can be readily detected

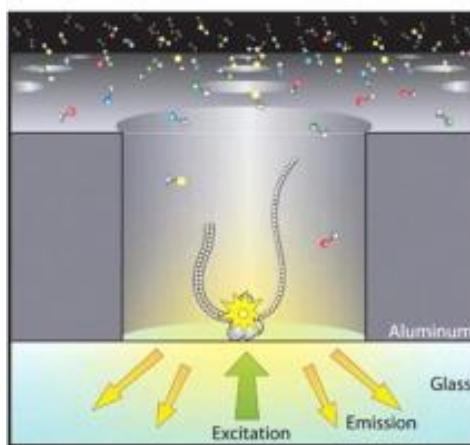
Circular DNA



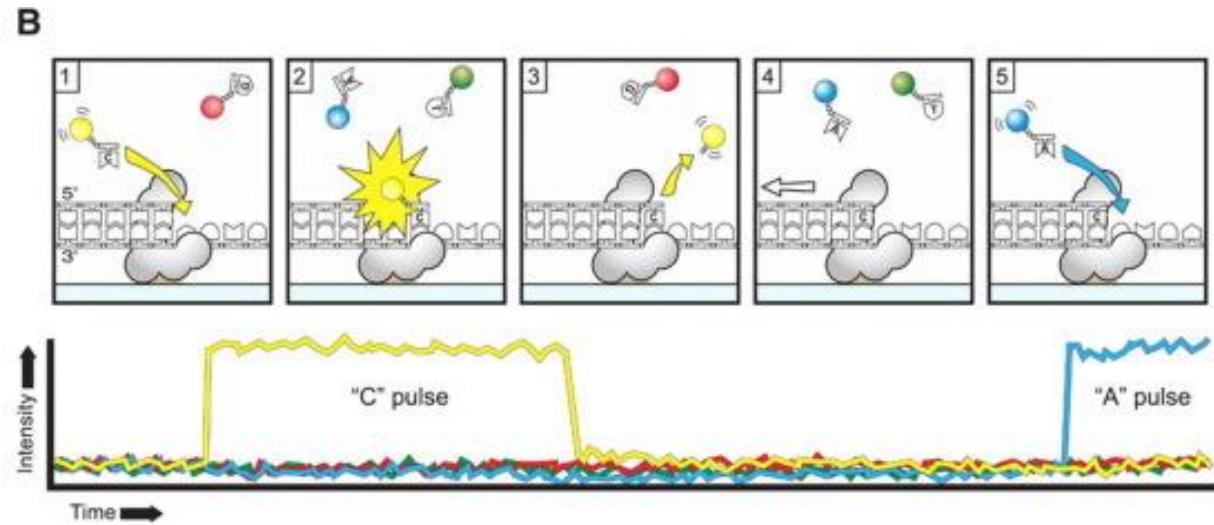
Zero mode  
waveguide unit  
(ZMW)



A

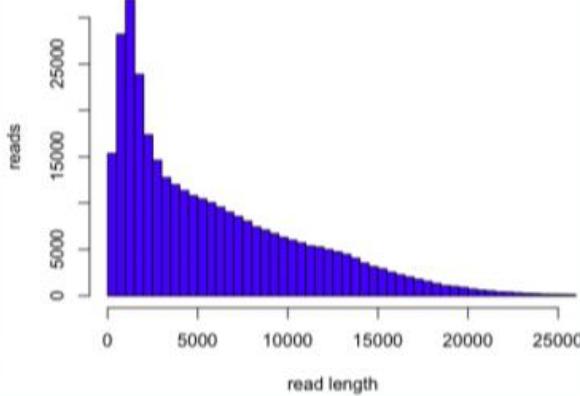


B

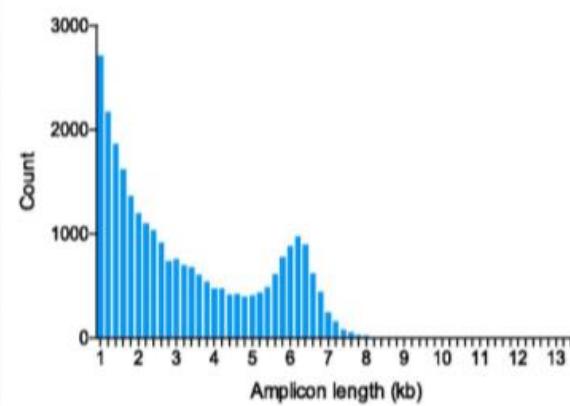


# Single Molecule Sequencing Technologies

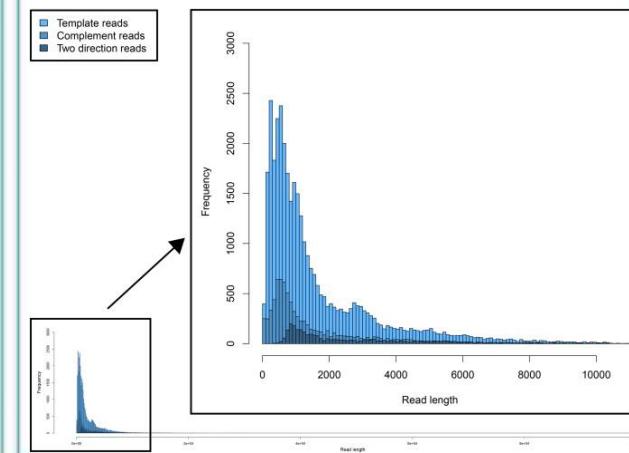
## PacBio RS II



## Moleculo



## Oxford Nanopore



El uso de valores de calidad para la asignación de bases a partir de picos en un cromatograma comenzó con el paquete phred/phrap/consed. [www.phrap.org](http://www.phrap.org)

Phred/Phrap/Consed es un paquete de software utilizado para:

- Leer cromatogramas (trace files)
- Asignar valores de calidad a las bases individuales de una secuencia
- Identificar y enmascarar secuencias correspondientes a vector (plásmido) o secuencias repetitivas
- Ensamblar secuencias individuales en contigs
- Visualizar assemblies (contigs)
- Hacer ‘sequence finishing’ auto dirigido (automatic finishing)

# Phred: a basecaller

- **Genome Res 8 (1998): 175**
- **Genome Res 8 (1998): 186**

RESEARCH

## Base-Calling of Automated Sequencer Traces Using *Phred*. I. Accuracy Assessment

Brent Ewing,<sup>1</sup> LaDeana Hillier,<sup>2</sup> Michael C. Wendl,<sup>2</sup> and Phil Green<sup>1,3</sup>

<sup>1</sup>Department of Molecular Biotechnology, University of Washington, Seattle, Washington 98195-7730 USA;

63108 USA

RESEARCH

EARCH 175

## Base-Calling of Automated Sequencer Traces Using *Phred*. II. Error Probabilities

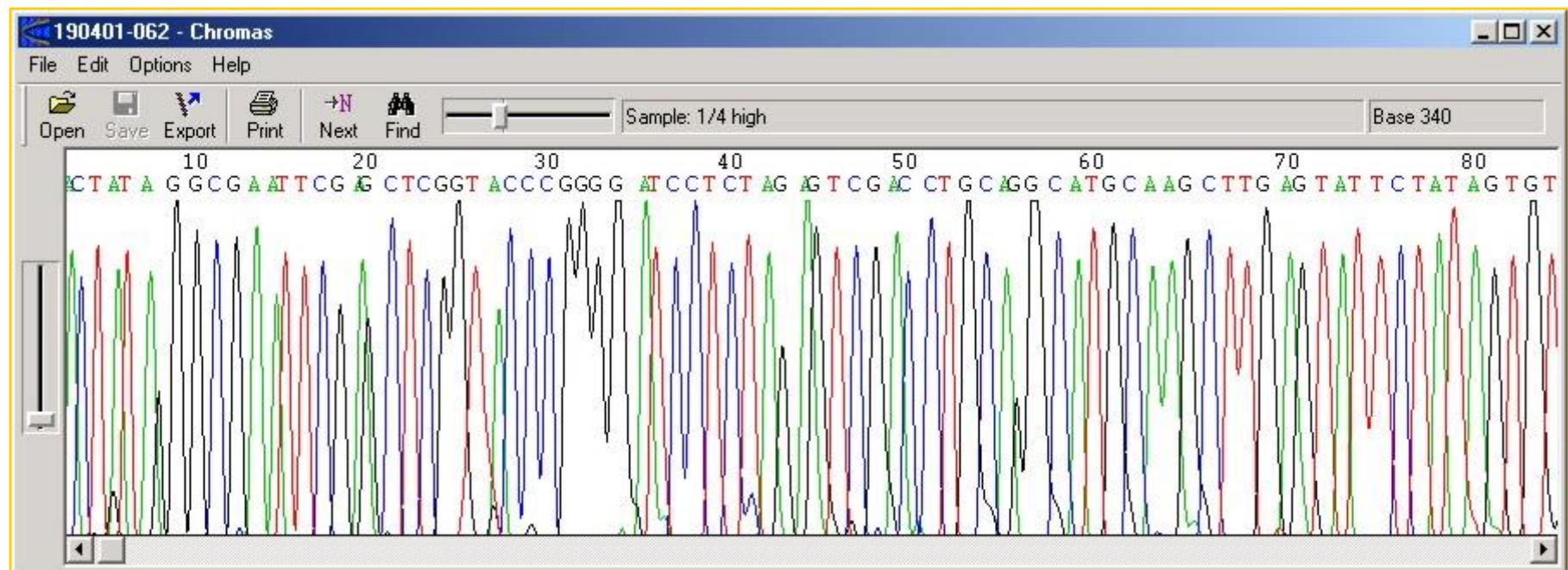
Brent Ewing and Phil Green<sup>1</sup>

Department of Molecular Biotechnology, University of Washington, Seattle, Washington 98195-7730 USA

- **Phred is a program that performs several tasks:**
  - Reads trace files – compatible with most file formats: SCF (standard chromatogram format), ABI (373/377/3700), ESD (MegaBACE) and LI-COR.
  - Calls bases – attributes a base for each identified peak with a lower error rate than the standard base calling programs.
  - Assigns quality values to the bases – a “Phred value” based on an error rate estimation calculated for each individual base.
  - Creates output files – base calls and quality values are written to output files.

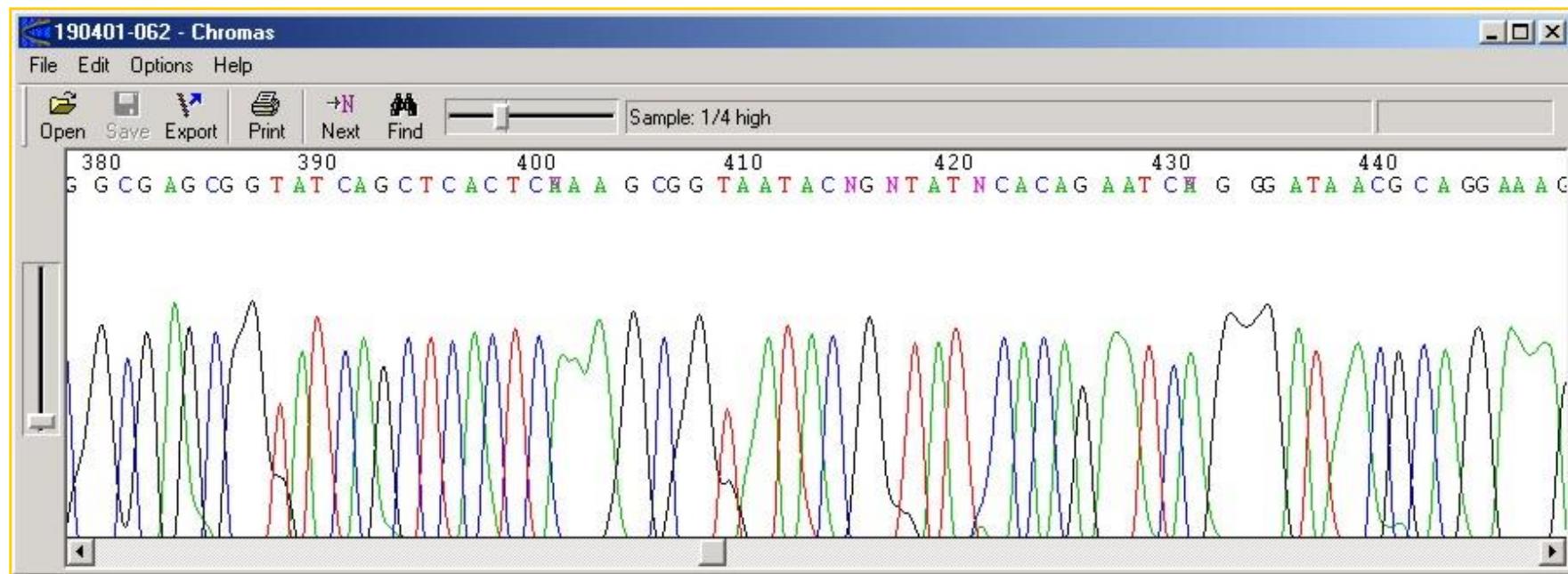
# Trace files

- Alta calidad, sin ambigüedad



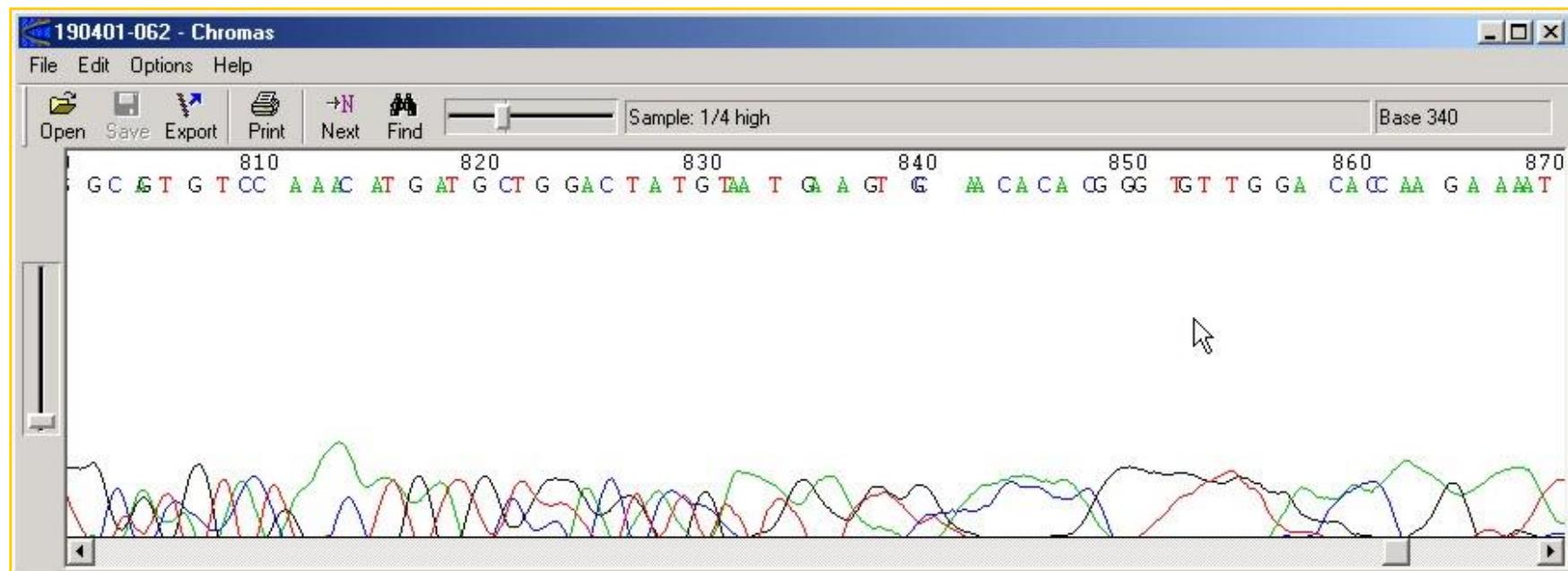
# Trace files

- Calidad media, algunas ambigüedades



# Trace files

- Baja calidad
  - la confianza en la asignación de bases es menor



# Phred qualities

$$q = -10 \times \log_{10}(p)$$

Donde:

- **q = quality value**
- **p = estimated probability error for a base call**

Phred quality scores are logarithmically linked to error probabilities

Phred Quality Score	Probability of incorrect base call	Base call accuracy
10	1 in 10	90 %
20	1 in 100	99 %
30	1 in 1000	99.9 %
40	1 in 10000	99.99 %
50	1 in 100000	99.999 %

[http://en.wikipedia.org/wiki/Phred\\_quality\\_score](http://en.wikipedia.org/wiki/Phred_quality_score)

$$p = 10^{\frac{-q}{10}}$$

Donde:

- **q = quality value**
- **p = estimated probability error for a base call**

Q = quality value	P = estimated probability of error
0	1
1	0.794
2	0.631
3	0.501
4	0.398
...	...
10	0.1
20	0.01
30	0.001

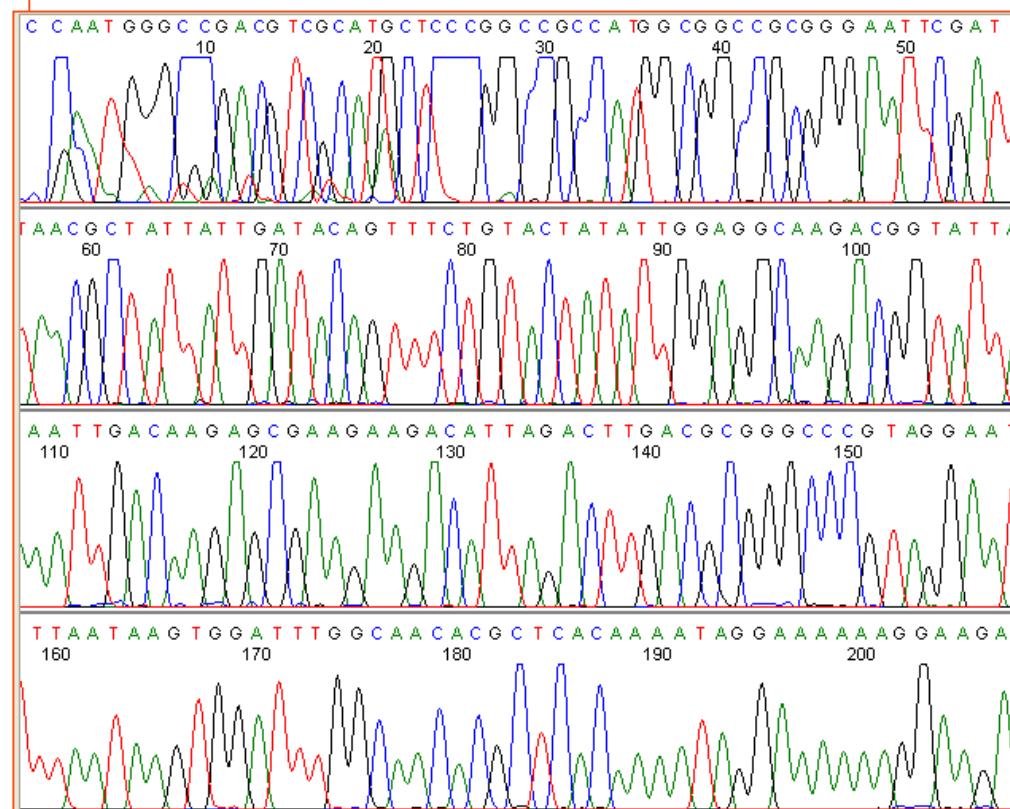
# Phred: PHD files

```
BEGIN_SEQUENCE 01EBV10201A02.g
BEGIN_COMMENT
CHROMAT_FILE: EBV10201A02.g
ABI_THUMPRINT:
PHRED_VERSION: 0.990722.g
CALL_METHOD: phred
QUALITY_LEVELS:99
TIME: Thu May 24 00:18:58 2001
TRACE_ARRAY_MIN_INDEX: 0
TRACE_ARRAY_MAX_INDEX: 12153
TRIM:
CHEM: term
DYE: big
END_COMMENT
BEGIN_DNA
t 8 5
c 13 17
a 19 26
c 19 32
t 24 2221
a 24 2232
a 22 2245
a 27 2261
g 25 2272
c 19 2286
c 12 2302
t 19 2314
g 12 2324
g 15 2331
g 19 2346
g 23 2363
t 33 2378
g 36 2390
c 44 2404
c 44 2419
t 39 2433
a 39 2446
a 34 2460
t 35 2470
g 34 2482
t 16 8191
g 19 8200
t 13 8211
c 13 8229
g 4 8241
n 4 8253
c 4 8263
t 10 8276
t 9 8286
c 12 8301
t 16 8313
c 12 8329
c 12 8336
c 15 8343
t 19 8356
c 9 8371
g 13 8386
g 14 8397
a 7 8417
g 9 8427
g 4 8445
t 6 11908
a 6 11921
g 6 11927
t 6 11947
c 6 11953
a 6 11964
g 6 11981
c 4 11994
n 4 12015
c 4 12037
n 4 12044
n 4 12058
n 4 12071
n 4 12085
n 4 12098
n 4 12111
n 4 12124
c 4 12144
n 4 12151
END_DNA
END_SEQUENCE
```

# Phred: QUAL files

- Quality values in FASTA format

```
>106 542 0 542 ABI trimmed
15 15 16 16 16 13 14 16 16 17 16 12 14 15 19 13 15
18 19 18 13 22 29 20 10 13 11 13 13 19 23 25 26 22
23 25 25 29 33 29 19 12 12 16 25 27 48 48 44 40 40
40 40 40 35 35 35 35 40 51 51 45 45 45 45
45 45 51 45 45 45 45 45 51 51 56 56 56 51 51
45 45 45 51 51 51 45 45 45 45 45 45 45 45 45
51 51 51 51 45 45 45 51 51 51 56 56 56 56 56
56 56 56 56 56 51 51 51 51 51 51 51 51 51 51
51 51 51 56 51 39 39 35 35 40 40 56 51 56 56 56
56 56 56 56 56 56 56 56 56 51 51 51 51 51 51 51
51 56 56 56 56 56 56 56 56 56 45 45 45 45 45
45 56 56 45 45 45 45 45 56 56 56 56 56 51 51 51
56 56 56 56 56 56 56 56 51 51 51 51 51 56 56 56
56 56 56 56 56 56 56 51 51 51 51 51 51 45 45 41
45 51 56 56 56 56 56 56 56 56 56 56 56 56 51 51
51 51 51 56 56 56 51 51 51 51 56 56 56 56 56 56
56 56 56 56 56 56 51 51 51 51 51 56 56 56 56 56
56 56 56 56 51 51 45 45 37 37 37 40 45 45 45 45 51
51 51 51 51 56 56 45 45 45 45 45 45 45 45 45 51 40
40 40 40 40 51 51 51 56 56 56 56 56 56 56 56 56
56 56 56 51 51 51 51 40 40 45 45 40 40 40 40 45 45
56 45 45 45 45 51 56 56 56 51 39 39 35 35 35 37
46 51 51 51 51 51 56 56 56 51 51 51 51 51 51 51 40
40 40 40 40 40 40 40 40 34 34 34 32 40 40 32
32 32 32 32 32 32 29 29 31 40 56 56 56 40 51 51
51 43 43 56 56 56 56 45 40 40 40 39 40 40 40 40
40 51 44 44 40 40 40 39 32 29 29 27 29 31 34 34
32 25 25 18 13 13 19 32 40 40 34 29 29 29 40 40 24
17 8 8 9 19 24 40 29 29 25 27 29 29 27 20 14 12 9
9 12 9 10 15 18 24 25 21 23 24 24 27 29 32 33 33 27
23 18 18 23 21 25 29 29 29 29 32 40 23 19 9 9 9
15 24 29 29 29 29 29 40 40 32 32 24
```

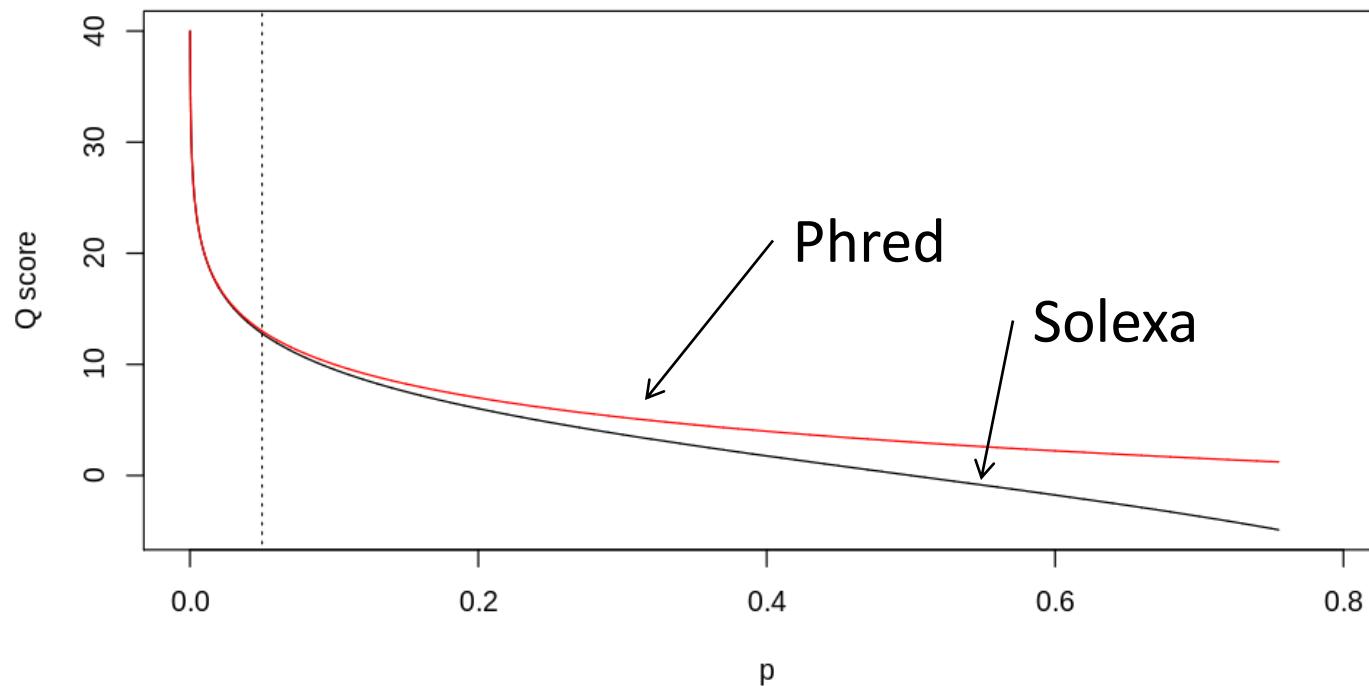


# Quality values: Solexa (Illumina)

## Solexa (Illumina) qualities for their version 1.3 pipeline

$$q = -10 \times \log_{10}\left(\frac{p}{1-p}\right)$$

Odds



Relationship between  $Q$  and  $p$  using the Sanger (red) and Solexa (black) equations (described above). The vertical dotted line indicates  $p = 0.05$ , or equivalently,  $Q \approx 13$ .

# FASTQ Format

El formato **FASTQ** guarda información de secuencia y de calidad en el mismo archivo.

```
@SEQ_ID
GATTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTGTTCAACTCACAGTT
+
! ' ' * ( ( ( (***+) ) % % % ++ ) ( % % % ) . 1 *** - + * ' ' ) ) ** 55CCF>>>>CCCCCCCC65
```

**@** = linea de texto que contiene al identificador

**+** = separador (arriba la secuencia, abajo la calidad)

Los valores de calidad están **codificados**.

Los caracteres “@” y “+” pueden aparecer en esta cadena de caracteres!

Sanger format = **Phred Q (0 – 93)** se codifica utilizando los códigos ASCII 33 al 126

Solexa 1.0 = **Phred Q (-5 – 62)** se codifica utilizando ASCII 59 al 126

Solexa 1.3 = **Phred Q (0 – 62)** se codifica utilizando ASCII 64 al 126

Solexa 1.8 = Sanger format (**Phred Q + 33**)

# ASCII Table

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0 000	000	<b>NUL</b> (null)	32	20 040	000	&#32;	<b>Space</b>	64	40 100	000	&#64;	<b>Ø</b>	96	60 140	000	&#96;	<b>`</b>
1	1 001	001	<b>SOH</b> (start of heading)	33	21 041	001	&#33;	<b>!</b>	65	41 101	001	&#65;	<b>A</b>	97	61 141	001	&#97;	<b>a</b>
2	2 002	002	<b>STX</b> (start of text)	34	22 042	002	&#34;	<b>"</b>	66	42 102	002	&#66;	<b>B</b>	98	62 142	002	&#98;	<b>b</b>
3	3 003	003	<b>ETX</b> (end of text)	35	23 043	003	&#35;	<b>#</b>	67	43 103	003	&#67;	<b>C</b>	99	63 143	003	&#99;	<b>c</b>
4	4 004	004	<b>EOT</b> (end of transmission)	36	24 044	004	&#36;	<b>\$</b>	68	44 104	004	&#68;	<b>D</b>	100	64 144	004	&#100;	<b>d</b>
5	5 005	005	<b>ENQ</b> (enquiry)	37	25 045	005	&#37;	<b>%</b>	69	45 105	005	&#69;	<b>E</b>	101	65 145	005	&#101;	<b>e</b>
6	6 006	006	<b>ACK</b> (acknowledge)	38	26 046	006	&#38;	<b>&amp;</b>	70	46 106	006	&#70;	<b>F</b>	102	66 146	006	&#102;	<b>f</b>
7	7 007	007	<b>BEL</b> (bell)	39	27 047	007	&#39;	<b>'</b>	71	47 107	007	&#71;	<b>G</b>	103	67 147	007	&#103;	<b>g</b>
8	8 010	010	<b>BS</b> (backspace)	40	28 050	010	&#40;	<b>(</b>	72	48 110	010	&#72;	<b>H</b>	104	68 150	010	&#104;	<b>h</b>
9	9 011	011	<b>TAB</b> (horizontal tab)	41	29 051	011	&#41;	<b>)</b>	73	49 111	011	&#73;	<b>I</b>	105	69 151	011	&#105;	<b>i</b>
10	A 012	012	<b>LF</b> (NL line feed, new line)	42	2A 052	012	&#42;	<b>*</b>	74	4A 112	012	&#74;	<b>J</b>	106	6A 152	012	&#106;	<b>j</b>
11	B 013	013	<b>VT</b> (vertical tab)	43	2B 053	013	&#43;	<b>+</b>	75	4B 113	013	&#75;	<b>K</b>	107	6B 153	013	&#107;	<b>k</b>
12	C 014	014	<b>FF</b> (NP form feed, new page)	44	2C 054	014	&#44;	<b>,</b>	76	4C 114	014	&#76;	<b>L</b>	108	6C 154	014	&#108;	<b>l</b>
13	D 015	015	<b>CR</b> (carriage return)	45	2D 055	015	&#45;	<b>-</b>	77	4D 115	015	&#77;	<b>M</b>	109	6D 155	015	&#109;	<b>m</b>
14	E 016	016	<b>SO</b> (shift out)	46	2E 056	016	&#46;	<b>.</b>	78	4E 116	016	&#78;	<b>N</b>	110	6E 156	016	&#110;	<b>n</b>
15	F 017	017	<b>SI</b> (shift in)	47	2F 057	017	&#47;	<b>/</b>	79	4F 117	017	&#79;	<b>O</b>	111	6F 157	017	&#111;	<b>o</b>
16	10 020	020	<b>DLE</b> (data link escape)	48	30 060	020	&#48;	<b>Ø</b>	80	50 120	020	&#80;	<b>P</b>	112	70 160	020	&#112;	<b>p</b>
17	11 021	021	<b>DC1</b> (device control 1)	49	31 061	021	&#49;	<b>1</b>	81	51 121	021	&#81;	<b>Q</b>	113	71 161	021	&#113;	<b>q</b>
18	12 022	022	<b>DC2</b> (device control 2)	50	32 062	022	&#50;	<b>2</b>	82	52 122	022	&#82;	<b>R</b>	114	72 162	022	&#114;	<b>r</b>
19	13 023	023	<b>DC3</b> (device control 3)	51	33 063	023	&#51;	<b>3</b>	83	53 123	023	&#83;	<b>S</b>	115	73 163	023	&#115;	<b>s</b>
20	14 024	024	<b>DC4</b> (device control 4)	52	34 064	024	&#52;	<b>4</b>	84	54 124	024	&#84;	<b>T</b>	116	74 164	024	&#116;	<b>t</b>
21	15 025	025	<b>NAK</b> (negative acknowledge)	53	35 065	025	&#53;	<b>5</b>	85	55 125	025	&#85;	<b>U</b>	117	75 165	025	&#117;	<b>u</b>
22	16 026	026	<b>SYN</b> (synchronous idle)	54	36 066	026	&#54;	<b>6</b>	86	56 126	026	&#86;	<b>V</b>	118	76 166	026	&#118;	<b>v</b>
23	17 027	027	<b>ETB</b> (end of trans. block)	55	37 067	027	&#55;	<b>7</b>	87	57 127	027	&#87;	<b>W</b>	119	77 167	027	&#119;	<b>w</b>
24	18 030	030	<b>CAN</b> (cancel)	56	38 070	030	&#56;	<b>8</b>	88	58 130	030	&#88;	<b>X</b>	120	78 170	030	&#120;	<b>x</b>
25	19 031	031	<b>EM</b> (end of medium)	57	39 071	031	&#57;	<b>9</b>	89	59 131	031	&#89;	<b>Y</b>	121	79 171	031	&#121;	<b>y</b>
26	1A 032	032	<b>SUB</b> (substitute)	58	3A 072	032	&#58;	<b>:</b>	90	5A 132	032	&#90;	<b>Z</b>	122	7A 172	032	&#122;	<b>z</b>
27	1B 033	033	<b>ESC</b> (escape)	59	3B 073	033	&#59;	<b>:</b>	91	5B 133	033	&#91;	<b>[</b>	123	7B 173	033	&#123;	<b>{</b>
28	1C 034	034	<b>FS</b> (file separator)	60	3C 074	034	&#60;	<b>&lt;</b>	92	5C 134	034	&#92;	<b>\</b>	124	7C 174	034	&#124;	<b> </b>
29	1D 035	035	<b>GS</b> (group separator)	61	3D 075	035	&#61;	<b>=</b>	93	5D 135	035	&#93;	<b>]</b>	125	7D 175	035	&#125;	<b>}</b>
30	1E 036	036	<b>RS</b> (record separator)	62	3E 076	036	&#62;	<b>&gt;</b>	94	5E 136	036	&#94;	<b>^</b>	126	7E 176	036	&#126;	<b>~</b>
31	1F 037	037	<b>US</b> (unit separator)	63	3F 077	037	&#63;	<b>?</b>	95	5F 137	037	&#95;	<b>_</b>	127	7F 177	037	&#127;	<b>DEL</b>

```

@HWUSI-EAS582_157:6:1:1:1501/1 ←
NCACAGACACACACGAACACACACAAAGACATGCCATATGAAGAT ←
+
% .7786867:778556858746575058873/347777476035 ←
@HWUSI-EAS582_157:6:1:1:1606/1
NCTGGCACCTTGATTGGACTTCCCAGCCTCCAGAACTGTGAG
+
%19489888879898836689888648998788898888588
@HWUSI-EAS582_157:6:1:1:453/1
NCTGCTTGCACCCCTGAAGTCACTGATCACATTCAAGGGTCACC
+
% /86899898888867668888986644788988413488885
@HWUSI-EAS582_157:6:1:1:1844/1
NGATTGACATTGGCAAAGAGGACAAC TGATTGCAAAC TTCA CAC
+
%-7;:::::;86499;75574586::635:62687666887879
@HWUSI-EAS582_157:6:1:1:1707/1
NAGGCTCAGGCGCACGGCCTACATCGTCGCTGTCGGCCAAGGGG
+

```

“Read” (sequence)

Quality scores (phred-33)

#### Illumina sequence identifiers

Sequences from the Illumina software use a systematic identifier:

`@HWUSI-EAS100R:6:73:941:1973#0/1`

<b>HWUSI-EAS100R</b>	the unique instrument name
<b>6</b>	flowcell lane
<b>73</b>	tile number within the flowcell lane
<b>941</b>	'x'-coordinate of the cluster within the tile
<b>1973</b>	'y'-coordinate of the cluster within the tile
<b>#0</b>	index number for a multiplexed sample (0 for no indexing)
<b>/1</b>	the member of a pair, /1 or /2 (paired-end or mate-pair reads only)

[http://en.wikipedia.org/wiki/FASTQ\\_format](http://en.wikipedia.org/wiki/FASTQ_format)

# Qué hacer con las secuencias?

## Obtuvimos nuestros datos: y ahora qué?

- **Analizar la calidad**
- **Pre-procesar (filtrar, recortar)**

Esto permite identificar contaminaciones, y problemas en la construcción de las bibliotecas, y mejorar los datos para los pasos subsiguientes.

- **Ensamblar**
- **Mapear contra referencia**

# Analizar calidad global

Se analiza la calidad de toda la corrida!

Una herramienta muy útil es **FASTQC**

The screenshot shows the Babraham Bioinformatics website. The header features the institute's logo (a stylized 'B' inside a square) and the text "Babraham Bioinformatics". Below the header is a navigation bar with links: "About", "People", "Services", "Projects", "Training", and "Publications". The main content area is titled "FastQC" in large blue text. Below this is a table with the following data:

<b>Function</b>	A quality control tool for high throughput sequence data.
<b>Language</b>	Java
<b>Requirements</b>	A suitable Java Runtime Environment
<b>Code Maturity</b>	Stable. Mature code, but feedback is appreciated.
<b>Code Released</b>	Yes, under <a href="#">GPL v3 or later</a> .
<b>Initial Contact</b>	<a href="#">Simon Andrews</a>

At the bottom of the FastQC section is a button labeled "Download Now".

**Hay otros:**

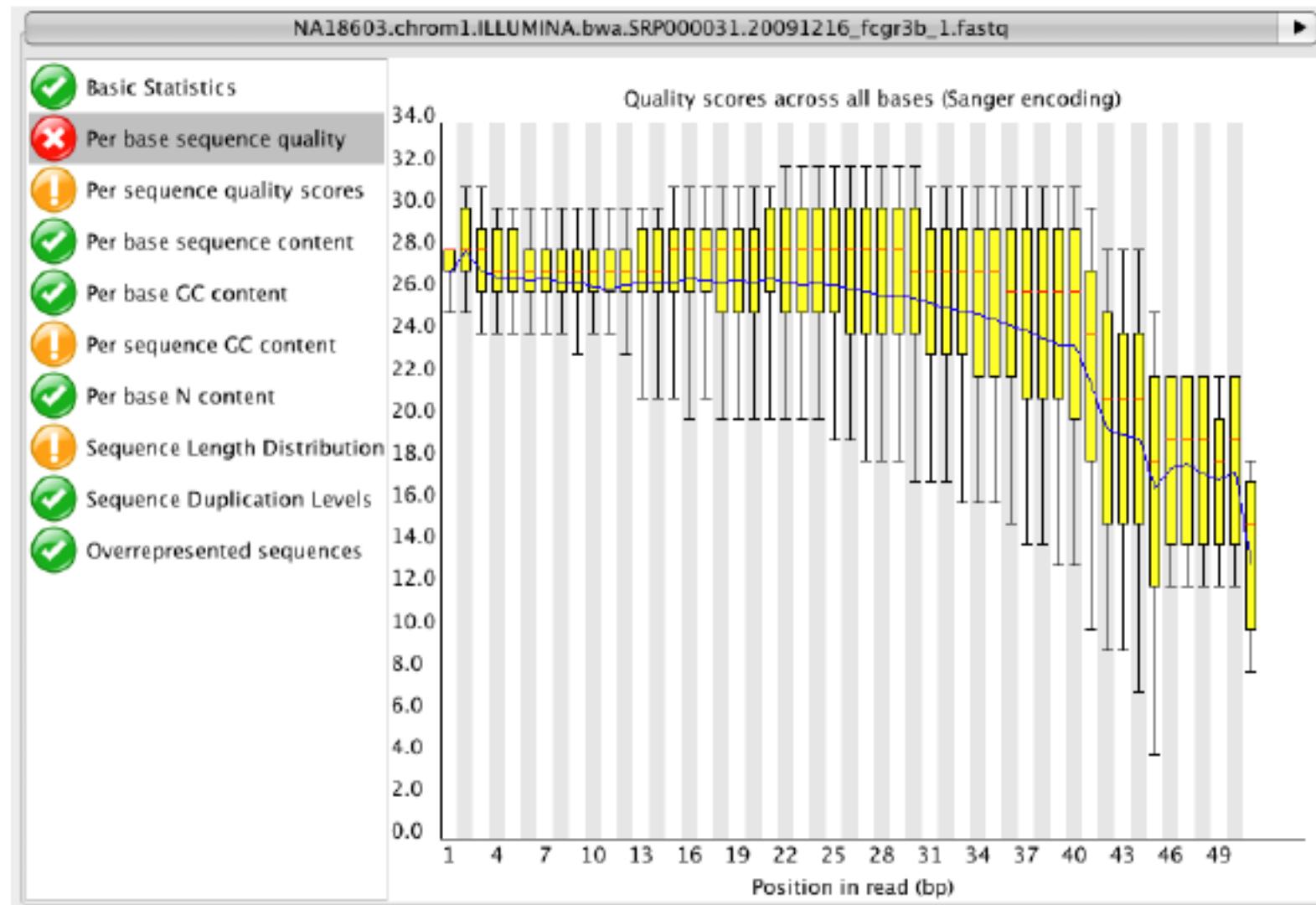
- FASTX**
- PRINSEQ**
- TagCleaner**

<http://www.bioinformatics.bbsrc.ac.uk/projects/fastqc/>

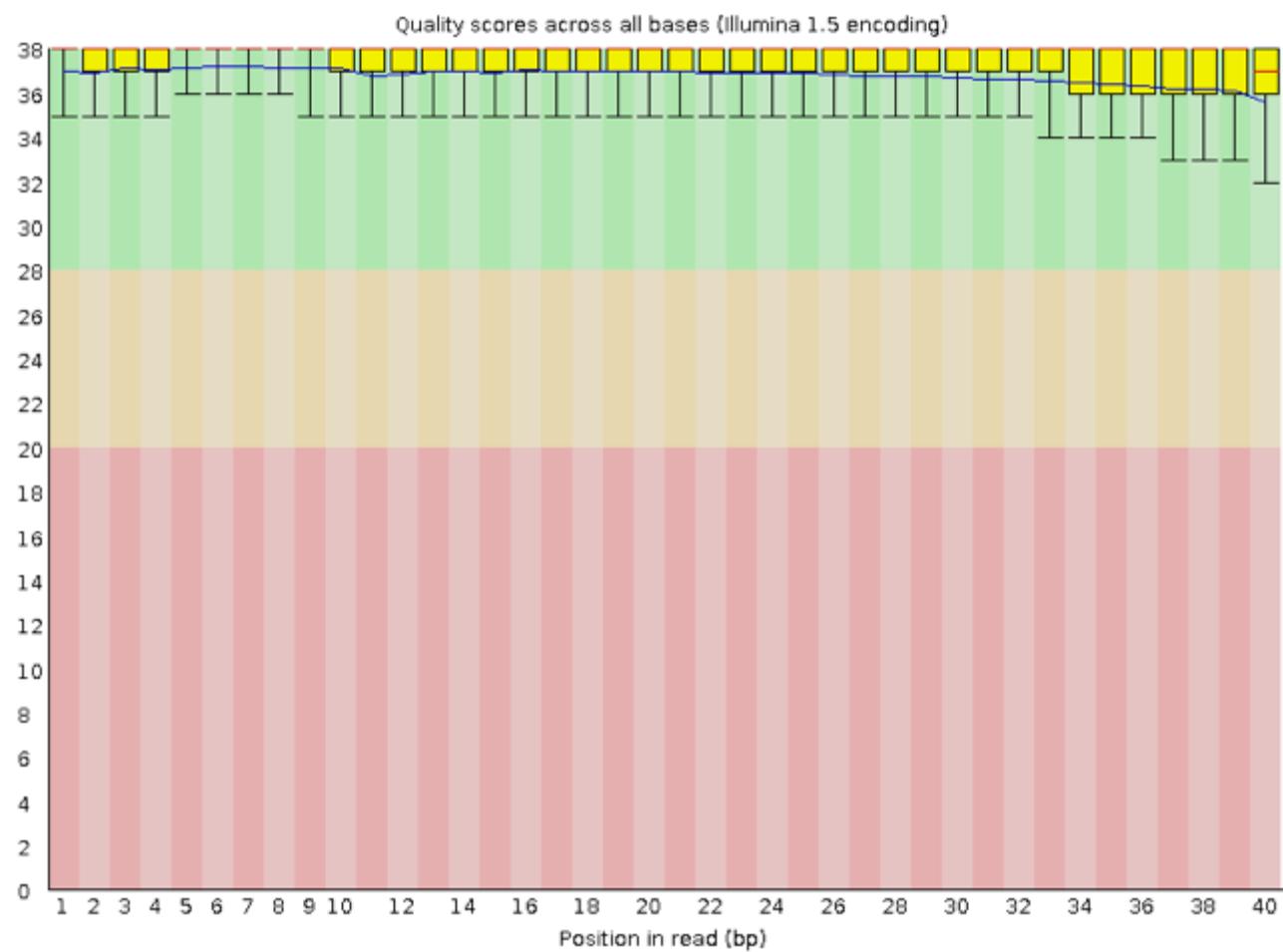
## Qué cosas se chequean:

- **Calidad**
  - Calidad por base
  - Calidad por lectura
- **Composición**
  - Por base
  - Perfil de composición de GC
- **Identificación de contaminantes**
  - Secuencias sobre-representadas (k-mers)
  - Niveles de duplicación

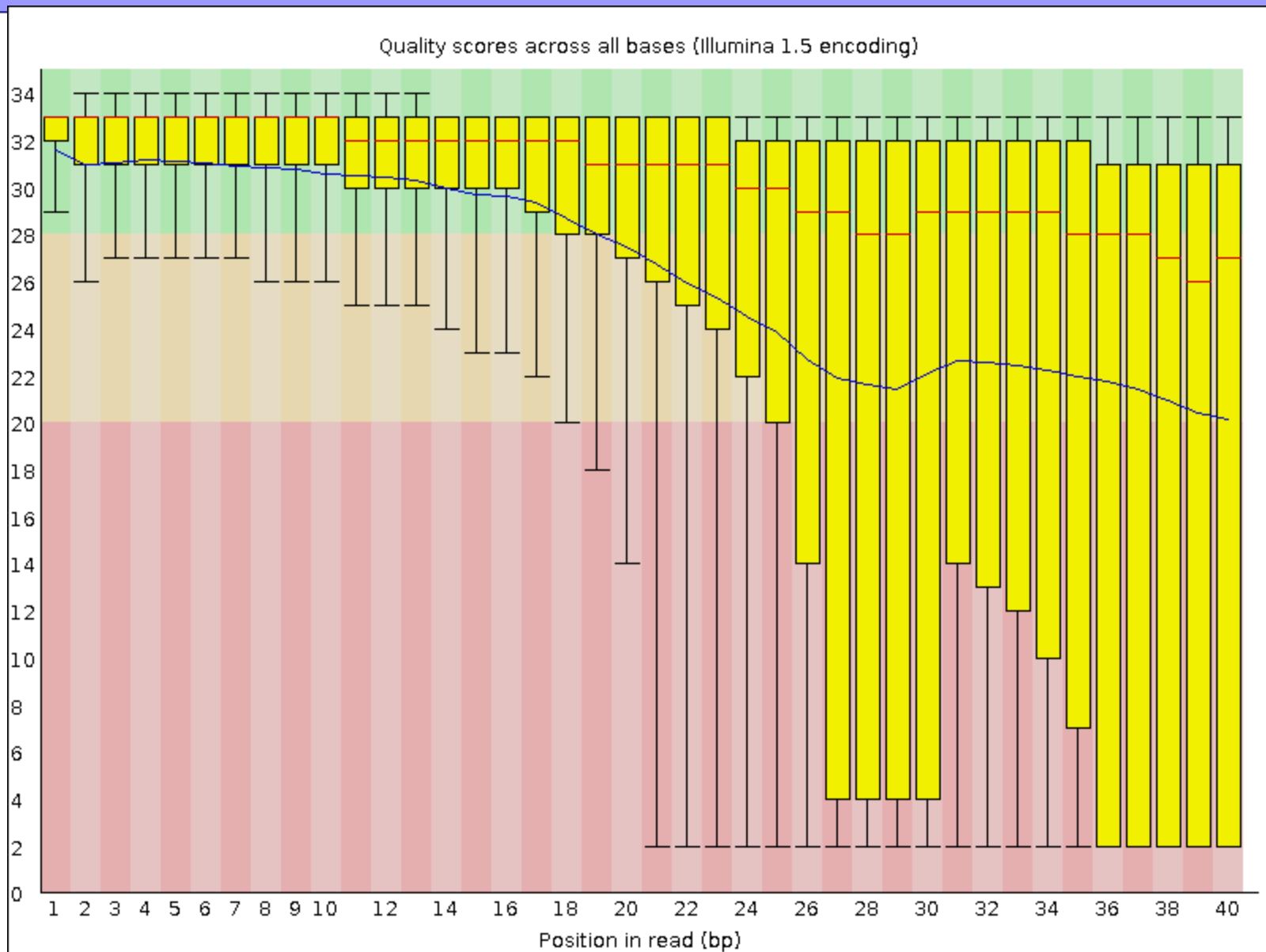
# FASTQC

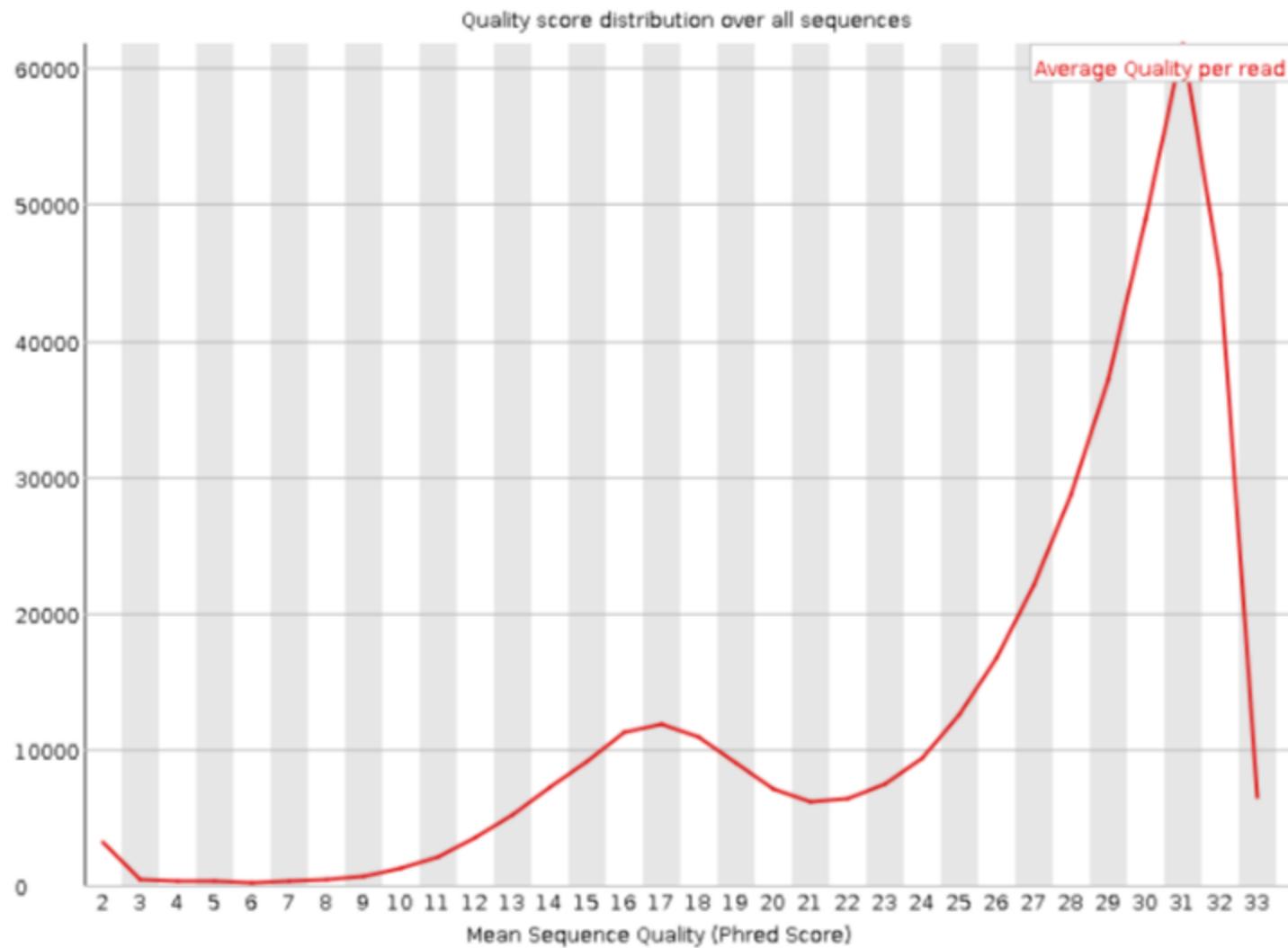


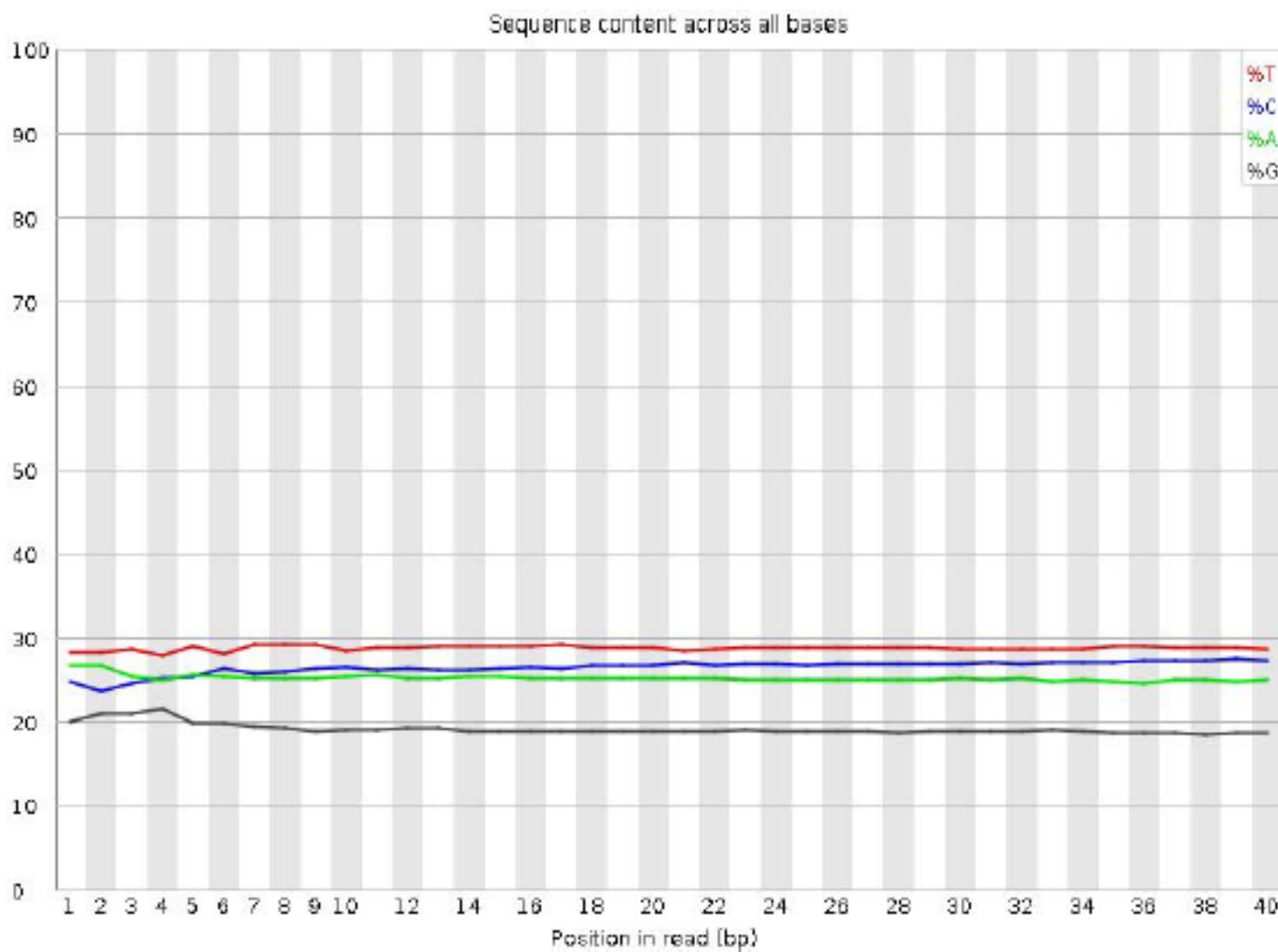
# FASTQC

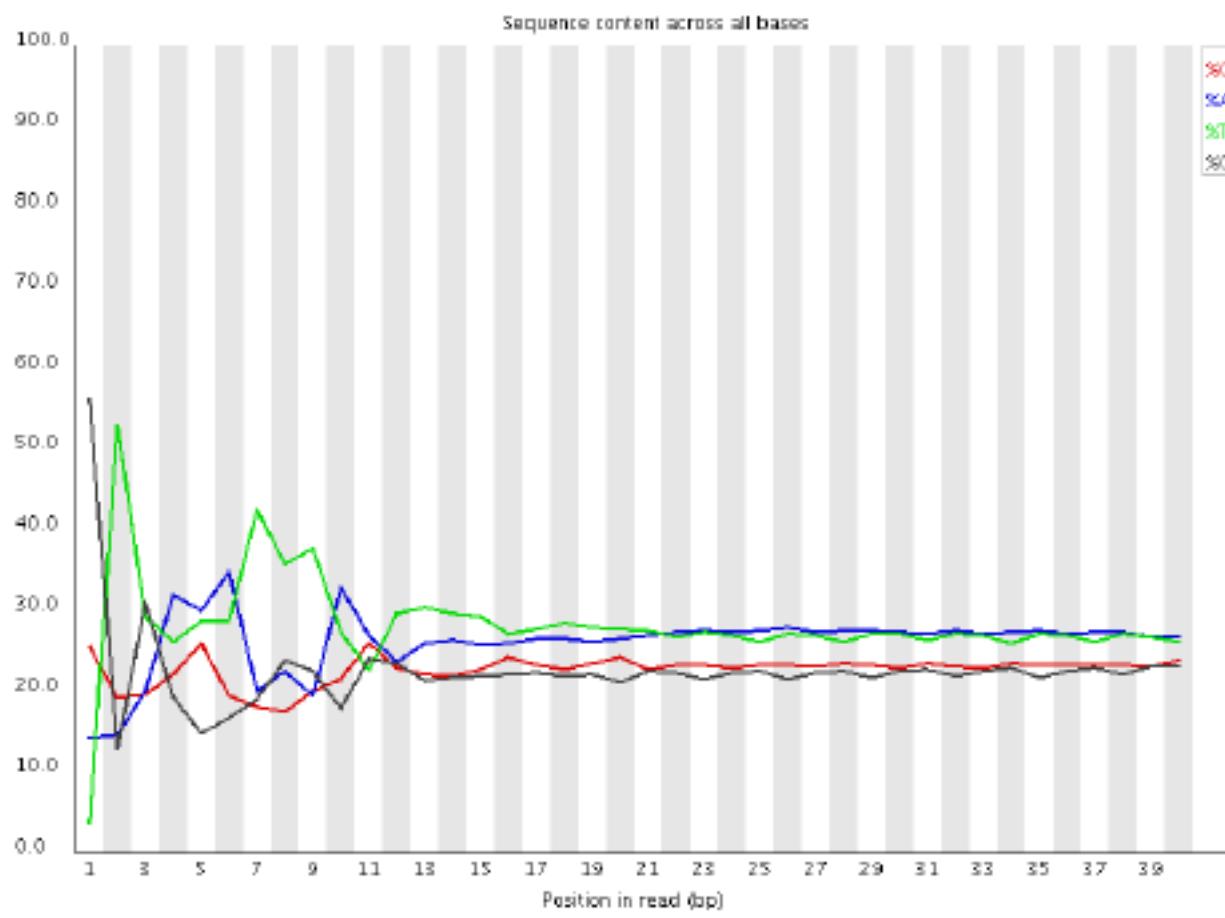


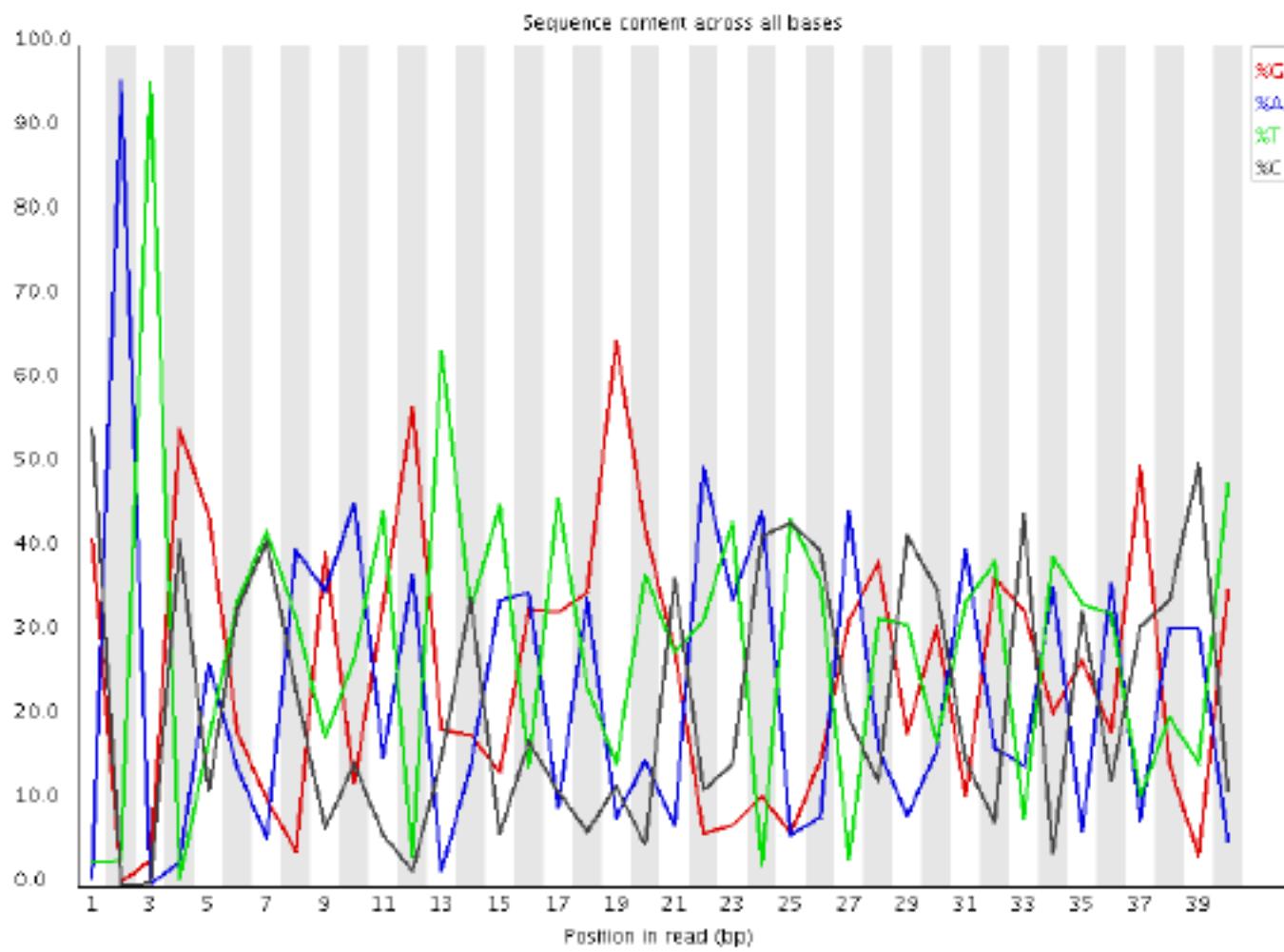
# FASTQC





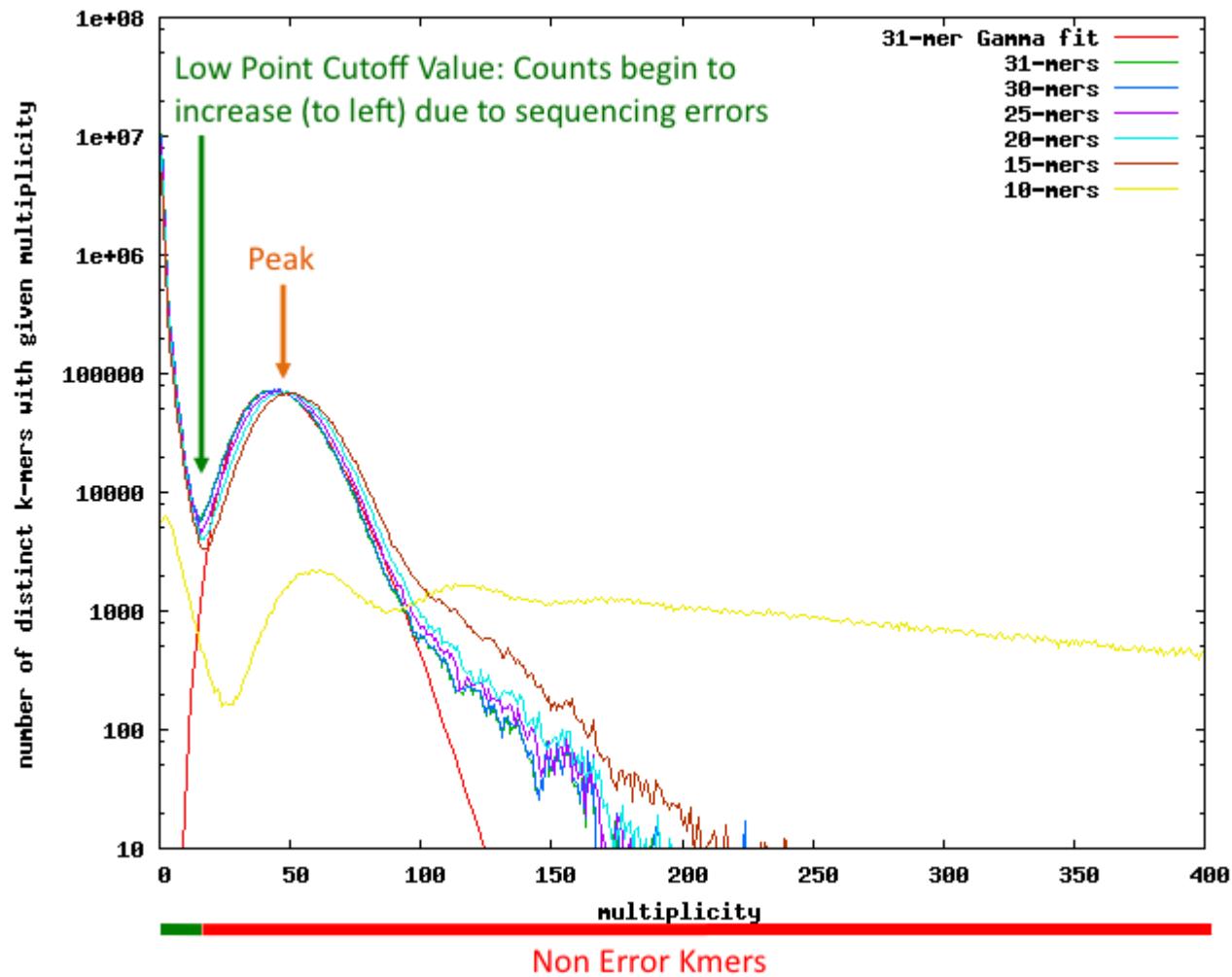






# Análisis de abundancia de kmers

kmers = secuencias de longitud  $k$



Metodos, algoritmos, técnicas para ensamblar genomas

# **SEQUENCE ASSEMBLY**

# Qué es un assembly?

Dada una colección de lecturas (“*reads*”) con secuencia de DNA conocida, y una lista de ***datos adicionales*** sobre sus posicionamientos, encontrar la secuencia de ADN de la molécula original.

***Datos adicionales*** = datos opcionales, auxiliares que pueden ayudar a posicionar las secuencias

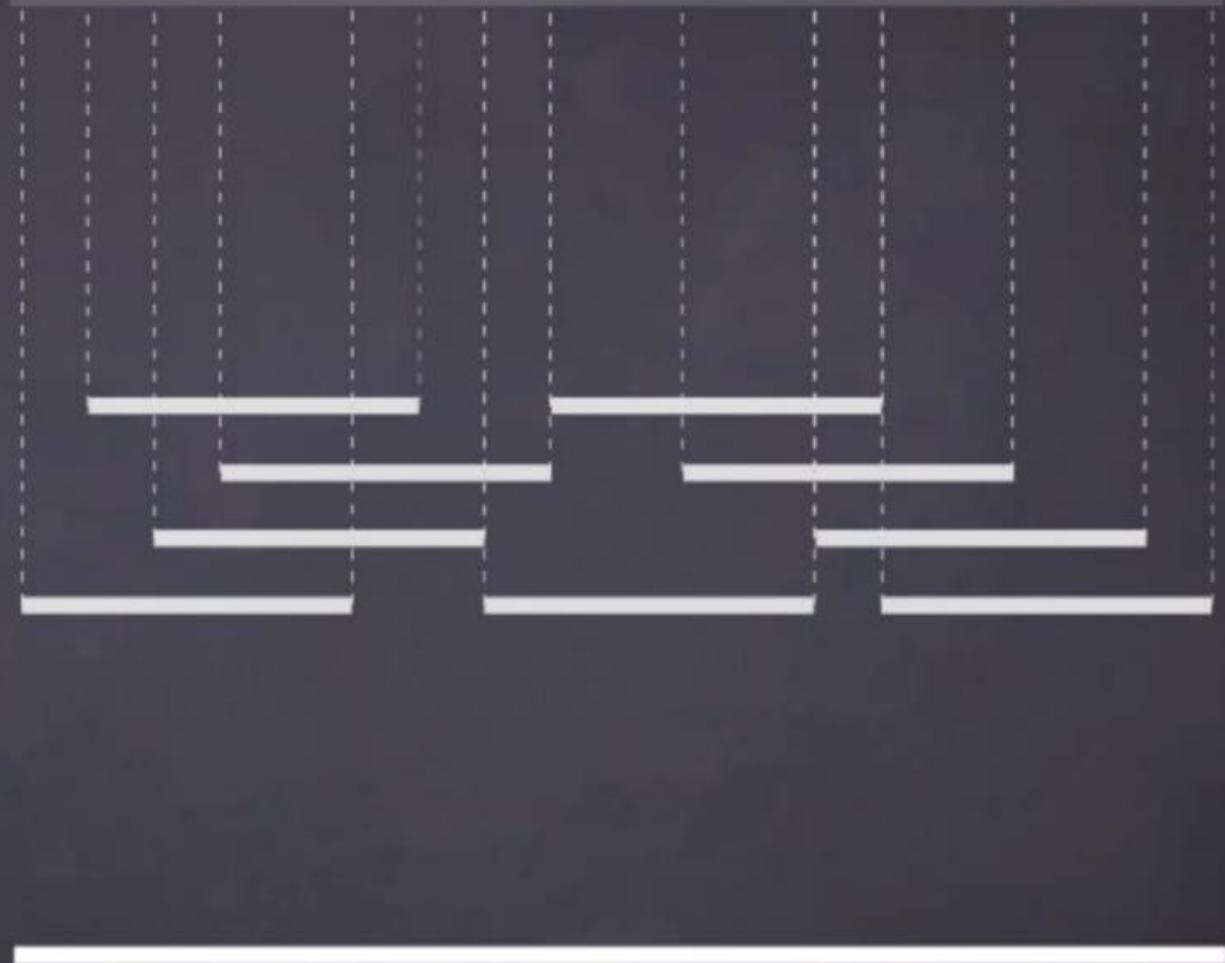
# Ensamblar secuencias

**genome**  
*not known*

**reads**  
*overlapping  
substrings  
that cover  
the genome  
redundantly*



**assembly**  
*what we think  
the genome is*



# Terminología / Jerga

## Assembly

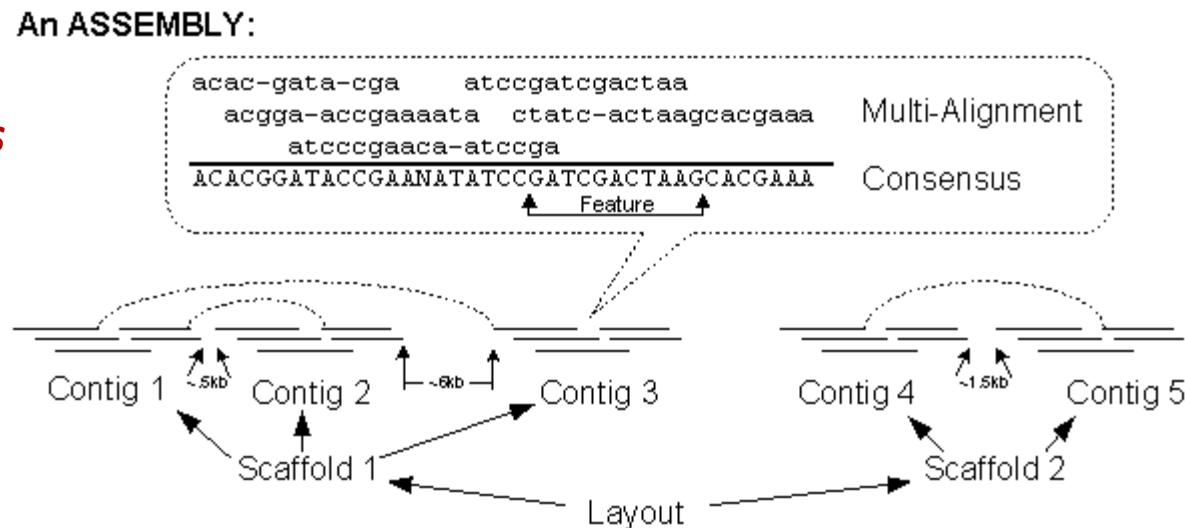
- Un conjunto de *scaffolds*

## Scaffold

- Un conjunto de *contigs* ordenados y orientados

## Contig

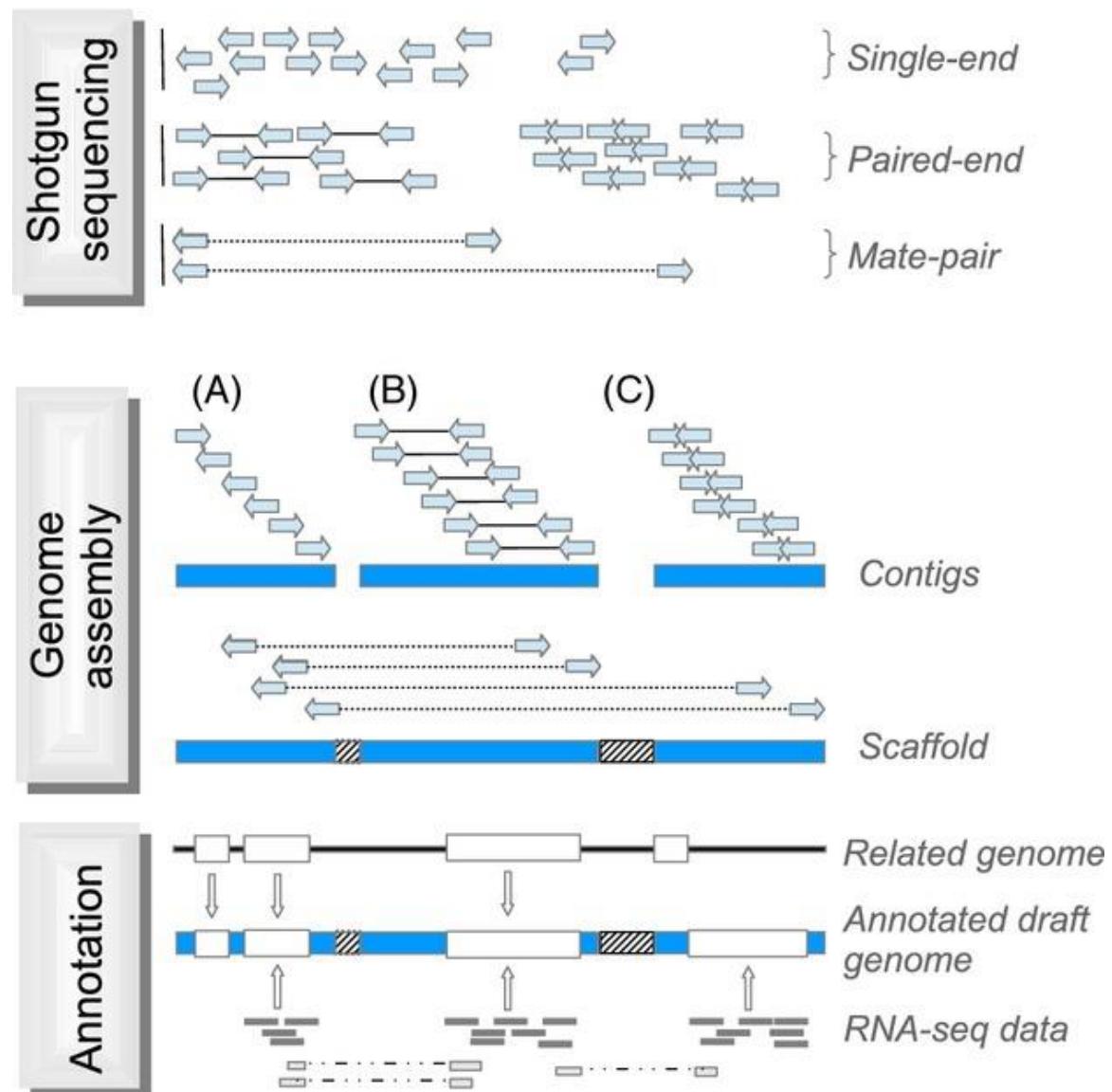
- Un conjunto de *reads*
- Un *layout* que posiciona y ordena todos los *reads* sin dejar gaps
- Un alineamiento múltiple de los reads
- Una secuencia consenso



[http://wgs-assembler.sourceforge.net/wiki/index.php/Celera\\_Assembler\\_Terminology](http://wgs-assembler.sourceforge.net/wiki/index.php/Celera_Assembler_Terminology)

# Genome sequencing, assembly and annotation: overview

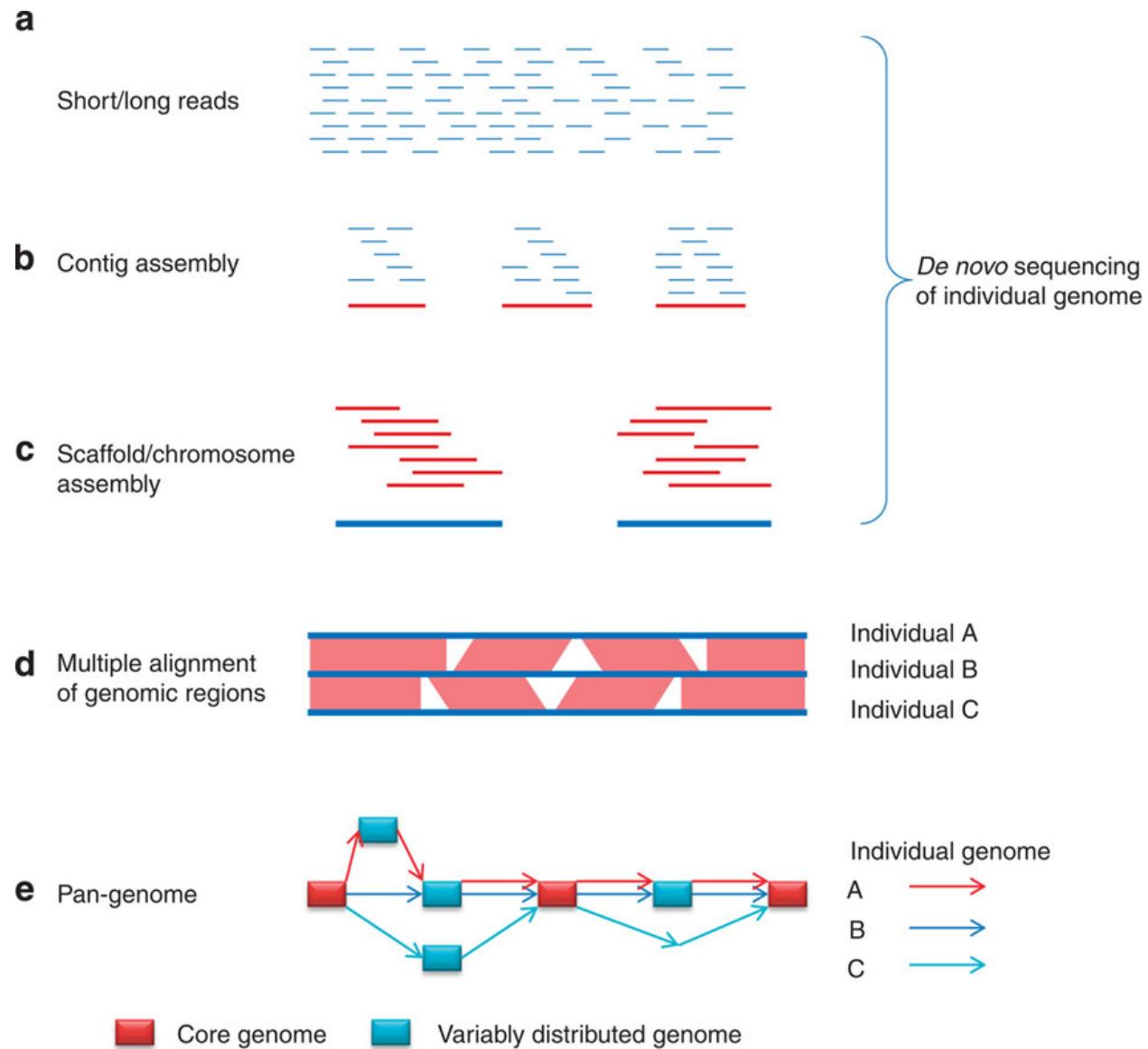
- Distintos tipos de estrategias *shotgun*
- Uso de la información del shotgun para guiar el assembly



# From genomes to pan-genomes

## Hacia la genómica de poblaciones

- **Core genome**
- **Variable genome**



# Sequence assembly problem

Mapear “palabras” en cadenas de texto más largas es un problema conocido: “Exact string matching”

## Naïve algorithm

ATAGGAGCACGTTAAGGTT  
| |  
AGGAGC

# Sequence assembly problem

“Exact string matching”

Naïve algorithm

The diagram shows two lines of text. The top line is "ATAGGACGCACGTTAAGGTT" in black font. The bottom line is "AGGAGC" in red font. Two vertical green lines connect the 'A' in "ATAGGACGCACGTTAAGGTT" to the 'A' in "AGGAGC".

ATAGGACGCACGTTAAGGTT

AGGAGC

# Sequence assembly problem

“Exact string matching”

Naïve algorithm

ATAGGACGCACGTTAAGGTT  
| | | |  
AGGACG

# Sequence assembly problem

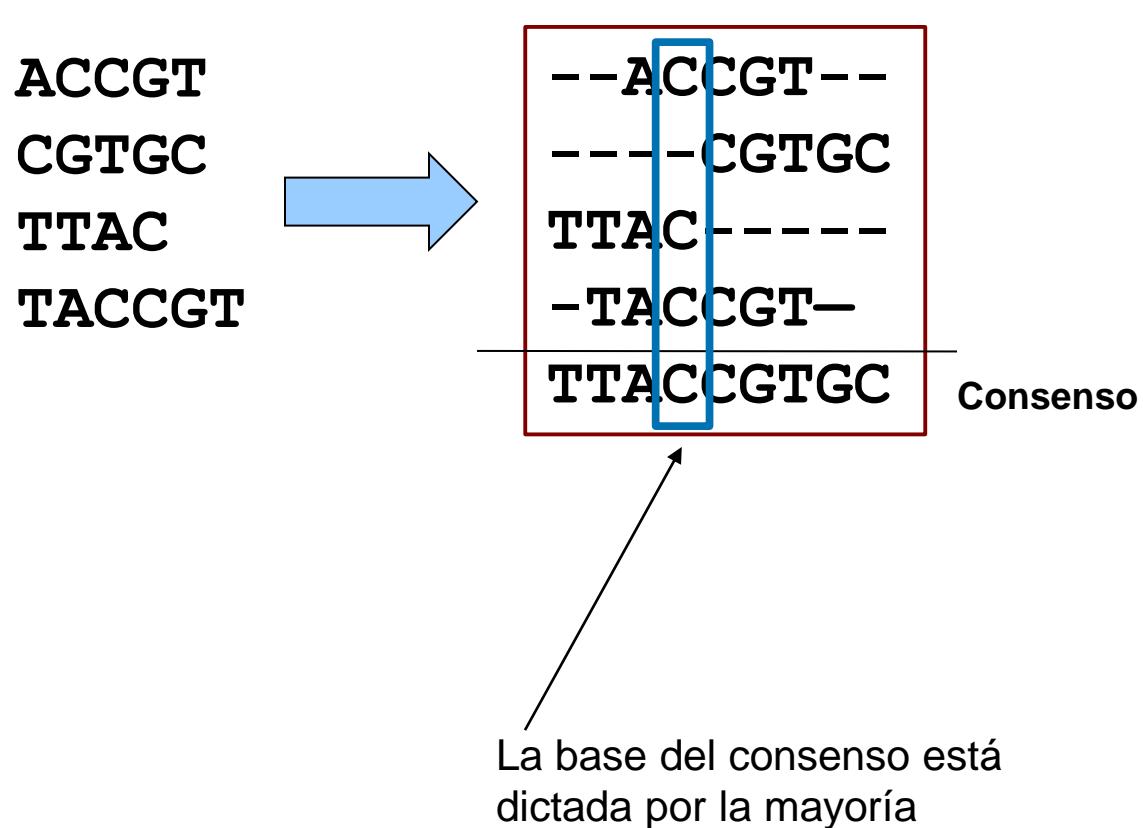
“Exact string matching”

Naïve algorithm

ATAGGACGCACGTTAAGGTT  
| | | |  
AGGACG  
GGACGC  
GACGCA  
ACGCAC

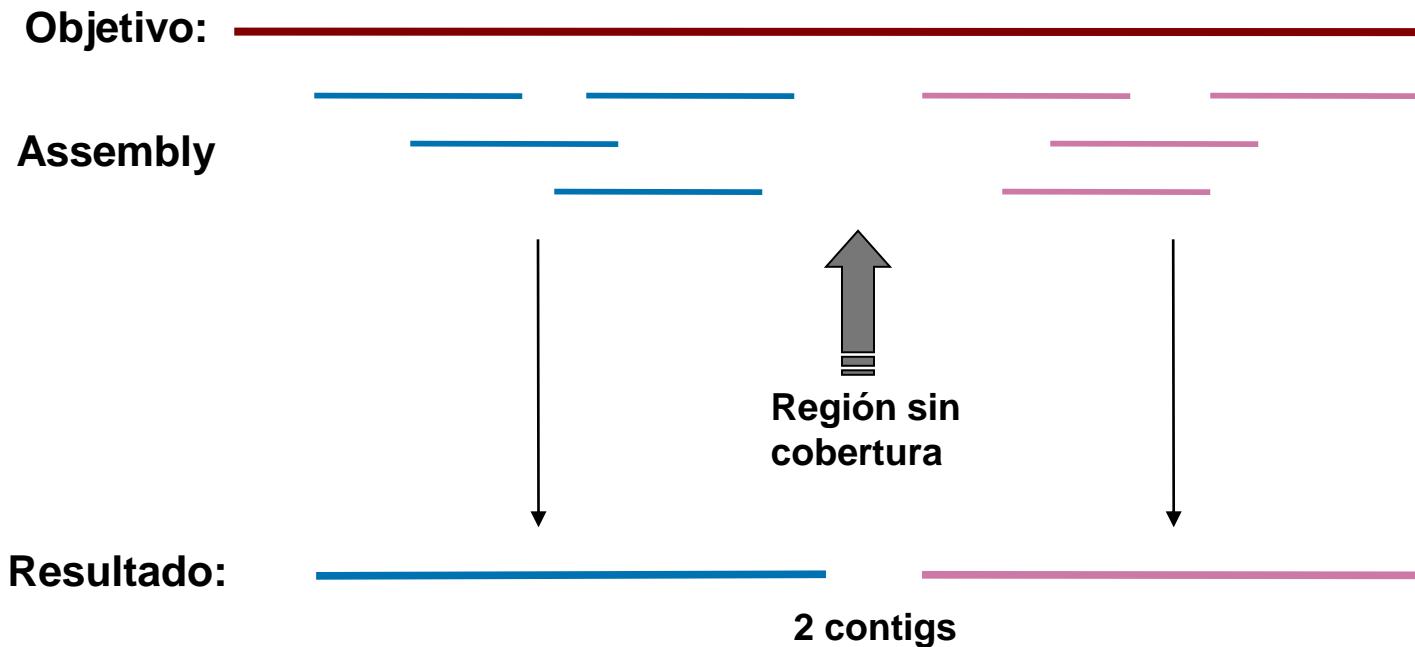
# El problema de ensamblar secuencias

- Fragment assembly problem
- El caso *ideal*



# Control de calidad del ensamblado

- Quality metrics



# Control de calidad del ensamblado

- **Linkage** –
  - grado de solapamiento (*overlap*) de los fragmentos

Objetivo:



- Alta cobertura (coverage)
- Solapamiento promedio **pobre**
- Solapamiento *mínimo* también **pobre**

# Cobertura de secuenciación

## • Cálculos de cobertura

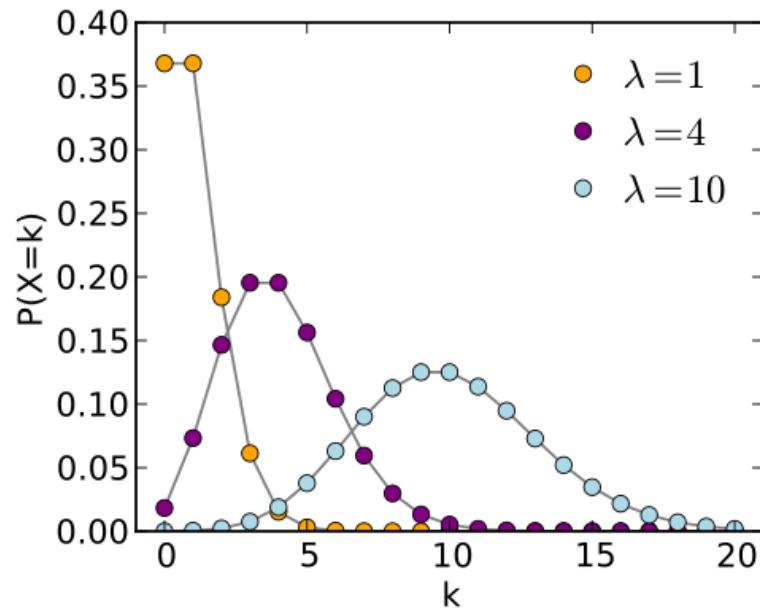
- Queremos secuenciar cada base 5x para tener un nivel de error aceptable
- Qué cobertura promedio necesitamos para asegurar que el 95% de un genoma se secuencie *al menos* 5 veces?

## • Se usa la distribución de Poisson

- Si el número esperado de eventos (ocurrencias) es  $\lambda$  entonces, la probabilidad de observar exactamente  $k$  eventos es

$$f(k; \lambda) = \frac{\lambda^k e^{-\lambda}}{k!}$$

Distribución de Poisson



# Ejemplo

- Cobertura promedio = 5X
  - Número de veces *esperado* que va a ser leída una base ( $\lambda$ )
- La probabilidad de una base de haber sido secuenciada 10 veces es
  - Número de veces *observado* ( $\kappa$ )

$$f(10; 5) = \frac{5^{10} e^{-5}}{10!} = 0.018$$

- 0.018 (1.8%) del genoma va a ser leido 10 veces
  - Es decir: 1.8 % del genoma va a tener una cobertura de 10x

# Por qué es importante la cobertura?

Respuesta: Base quality values

$$q = -10 \times \log_{10}(p)$$

Donde:

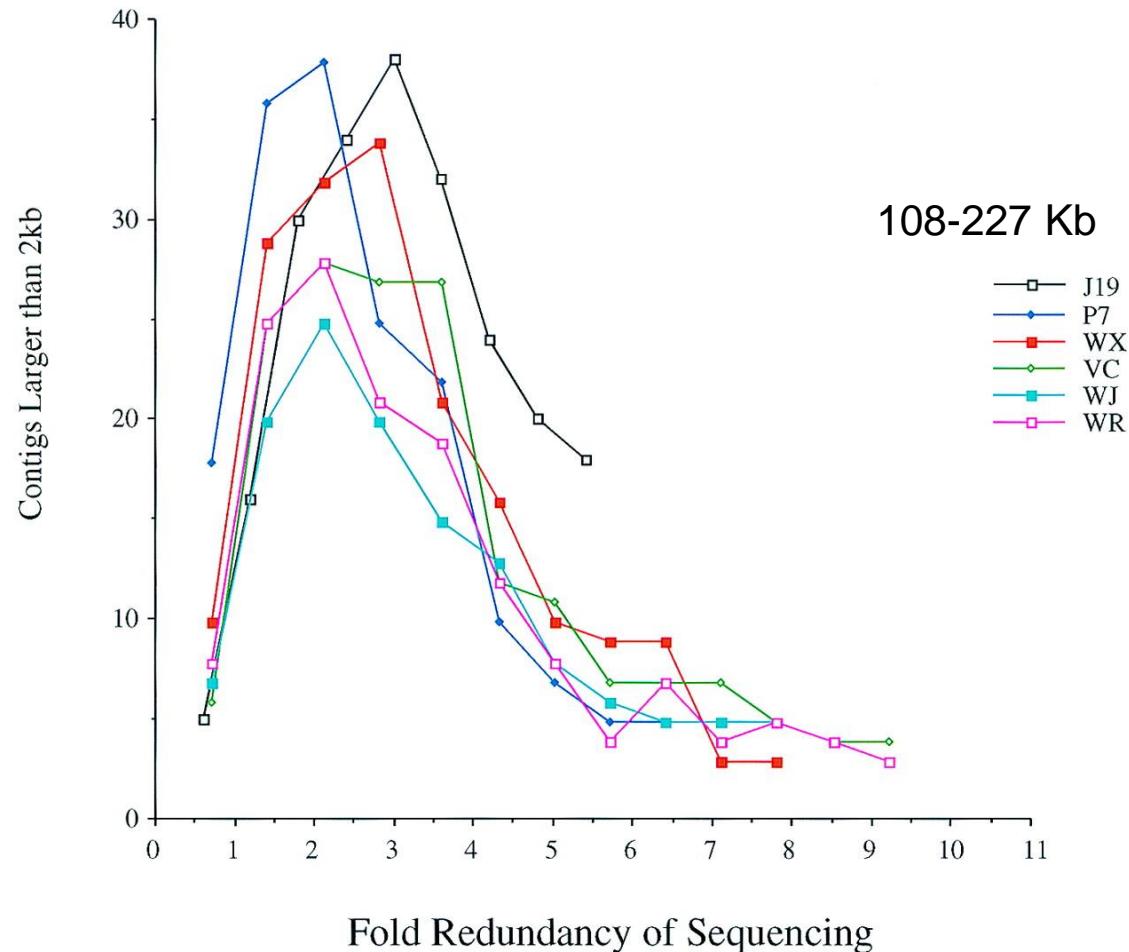
- $q$  = quality value
- $p$  = estimated probability error for a base call

Ejemplos:

- $q = 20$  significa  $p = 10^{-2}$  (1 error cada 100 bases)
- $q = 30$  significa  $p = 10^{-3}$  (1 error cada 1000 bases)
- $q = 40$  significa  $p = 10^{-4}$  (1 error cada 10000 bases)

# Modelos para estimar gaps

- **Shotgun sequencing**
  - Los clones que serán secuenciados se seleccionan al azar
  - Genera redundancia
  - La cobertura aumenta con el número de secuencias (pero no en forma lineal)

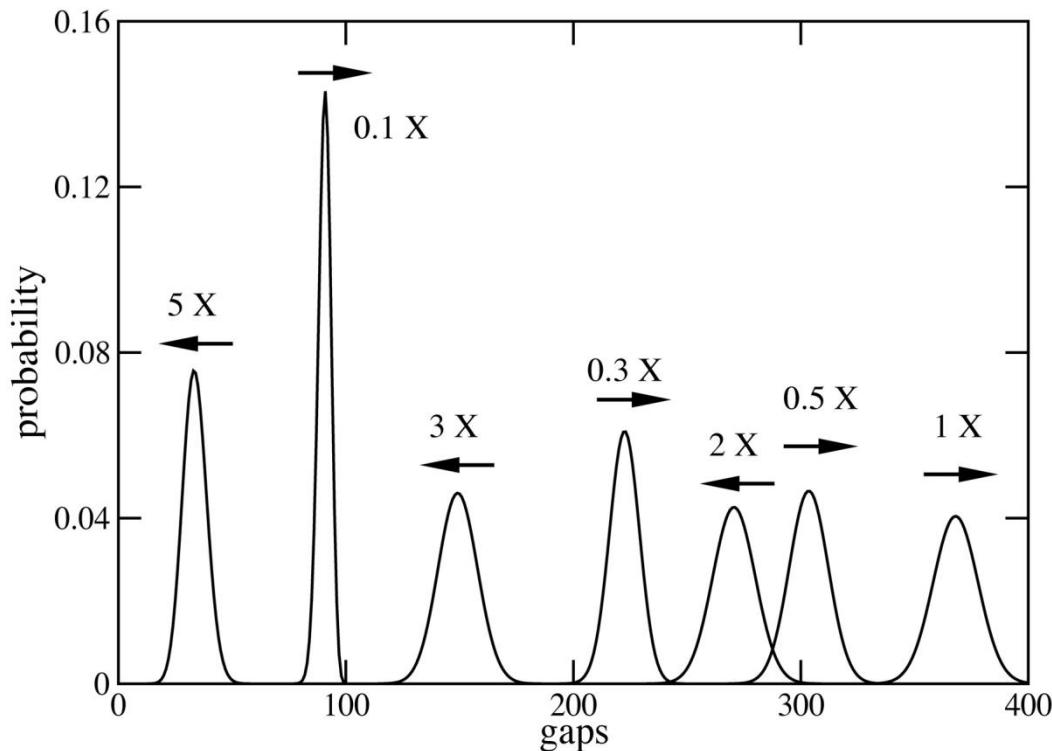


Contig formation at lower redundancy of sequencing. The number of contigs that were larger than 2 kb was calculated for each low redundancy simulation. The fold redundancy of each clone was calculated based on the number of bases that had a Phred value >20. The projects that were examined are listed at right. Tomado de Bouck *et al.* (1998) Genome Res 8: 1074.

# Modelos para estimar gaps

- Wendl MC and Waterston RH. (2002). Generalized Gap Model for Bacterial Artificial Chromosome Clone Fingerprint Mapping and Shotgun Sequencing. *Genome Res* 12: 1943.
  - Función de densidad de probabilidades para  $i$  gaps en  $N$  clones

Evolution of probability density function for a hypothetical project ( $L/G = 0.001$ ,  $T/L = 0$ ) up to 5 $\times$  coverage as evaluated by equation 4. Arrows indicate whether the average number of gaps is increasing ( $\rightarrow$ ) or decreasing ( $\leftarrow$ ) for each distribution.  
L = clone length  
G = project length

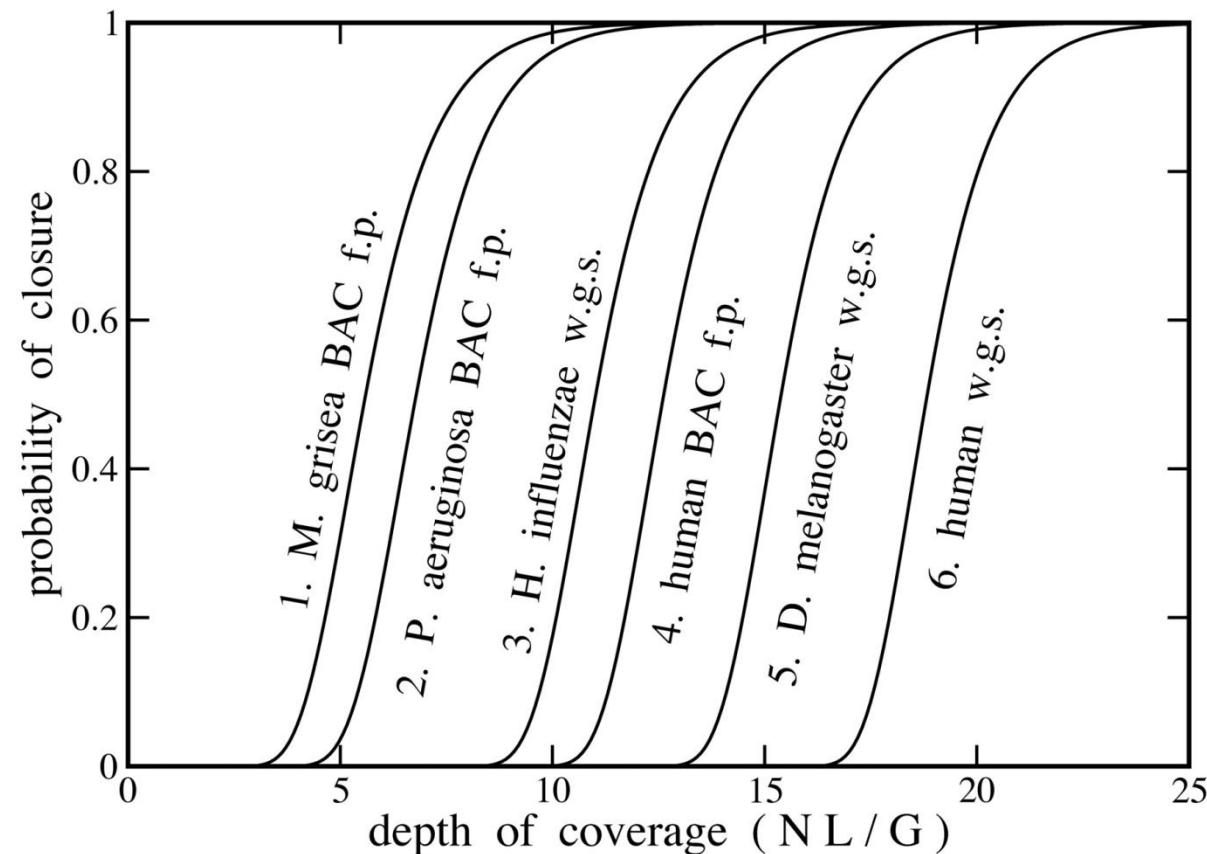


# Modelos para estimar gaps: closure

- **Closure**

- En proyectos de secuenciación por shotgun ***closure*** se refiere al momento donde un aumento de cobertura ya no produce cambios en la disminución del número de gaps (aumento del No. de contigs)
- Probability of closure,  $p(0,N)$

Probability of closure as a function of depth of coverage for various projects: 1. Zhu et al. (1999); 2. Dewar et al. (1998); 3. Fleischmann et al. (1995); 4. McPherson et al. (2001); 5. Adams et al. (2000); 6. Venter et al. (2001). Abbreviations "f.p." and "w.g.s." represent fingerprint mapping and whole genome shotgun sequencing projects, respectively. Cases 1 and 2 were evaluated using equation 4, whereas the remaining cases were determined using equation 9. Tomado de: Wendl MC and Waterston RH (2002). Genome Res 12: 1943



# Errores usuales

- **Artefactos de clonado**
  - Quimeras (dos insertos ligados en el mismo vector)
- **Errores en la asignación de las bases**

--ACCGT--  
-----CGTGC  
TTAC-----  
-T**G**CCGT-  

---

TTACCGTGC

Base Call Error

--ACC|GT--  
-----CA|GTGC  
TTAC-----  
-TACC|GT-  

---

TTACCGTGC

Insertion Error

--AC|CGT--  
-----CGTGC  
TTAC-----  
-TAC|GT-  

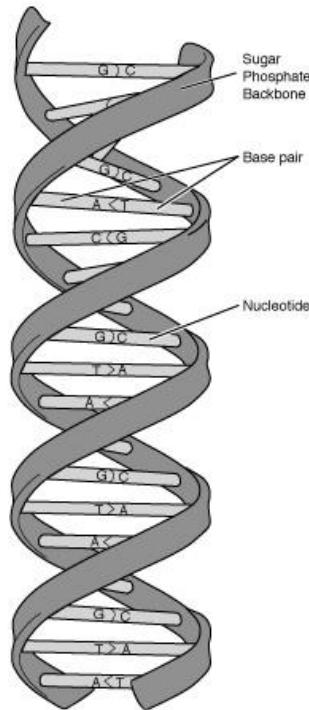
---

TTACCGTGC

Deletion Error

# Cosas que hay que resolver

- Los fragmentos secuenciados pueden provenir de cualquiera de las 2 hebras del ADN originario

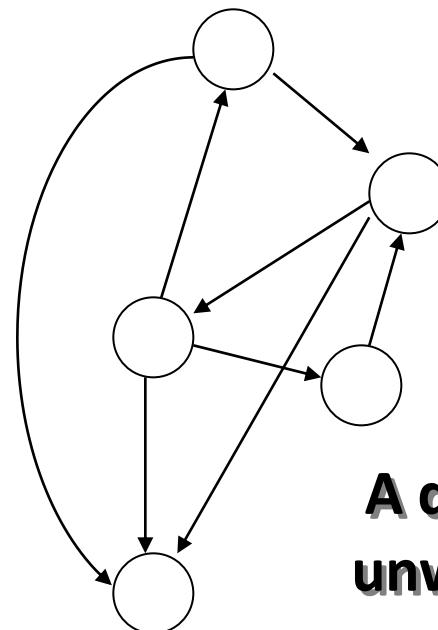


CACGT	→	CACGT
ACGT	→	-ACGT
ACTACG	←	--CGTAGT
GTACT	←	-----AGTAC
ACTGA	→	-----ACTGA
CTGA	→	-----CTGA

- **Shortest common superstring (SCS)**
  - Input: una colección  $\mathcal{F}$  de cadenas de caracteres (fragmentos)
  - Output: la cadena más corta posible  $S$  en la cual se cumpla que
    - Por cada  $f \in \mathcal{F}$ ,  $S$  es una supercadena de  $f$
- **Ejemplo 1**
  - $\mathcal{F} = \{ \text{ACT}, \text{CTA}, \text{AGT} \}$
  - $S = \text{ACTAGT}$
- **Ejemplo 2**
  - Alfabeto = {0,1}
  - Todos los 3-mers posibles para este alfabeto
  - $\mathcal{F} = \{ 000, 001, 010, 011, 100, 101, 110, 111 \}$
  - $S = 0001110100$

# Grafos

- La mayoria de las soluciones de reconstruccion de contigs a partir de fragmentos se resuelven modelando el problema como un **grafo**
- Un **grafo** es una colección de **nodos** y **aristas** (o **vertices**) que conectan los nodos
  - Dirigidos vs no dirigidos
  - Pesados (weighted) vs unweighted
- Vamos a ver mas sobre grafos ...

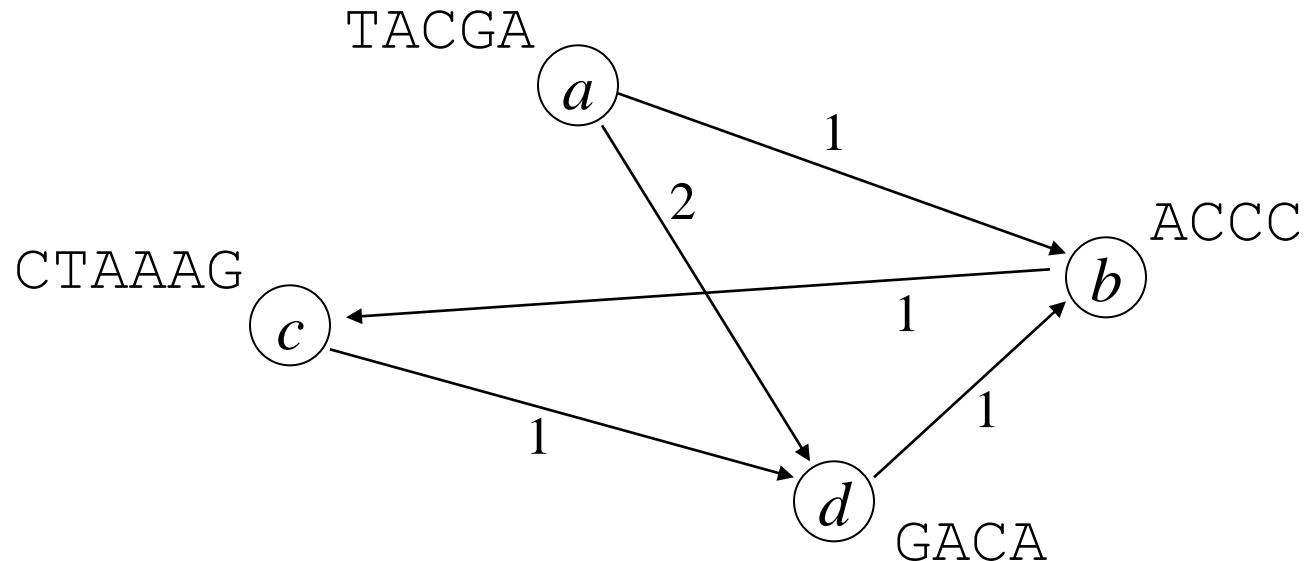


**A directed,  
unweighted  
graph**

# Grafo de maximo solapamiento

## Maximum overlap graph

- El **peso** de cada vertice  $(u,v)$  corresponde a la longitud maxima de solapamiento entre un **prefijo** de  $u$  y un **sufijo** de  $v$



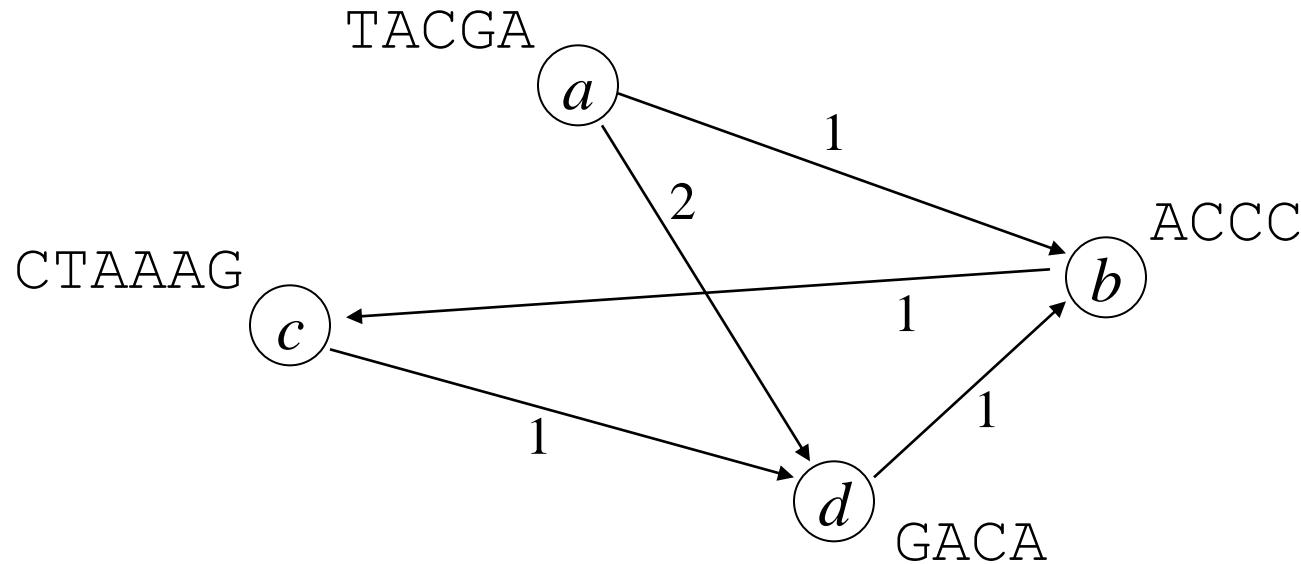
# Assembly graphs

El camino *dbc* corresponde al alineamiento:

GACA-----

---ACCC-----

-----CTAAAG



# Assembly graphs

- Cada *camino* (path) dentro de un grafo, que recorra todos los nodos es un *superstring*

- Los vertices con peso = 0 corresponden a alineamientos del tipo

GACA-----

-----GCC-----

-----TTAAAG

- Vertices con pesos mas altos, producen alineamientos con mayor overlap (y por lo tanto cadenas mas cortas)
  - El superstring comun mas corto (SCS) es el camino con mayor peso que cubre todos los nodos

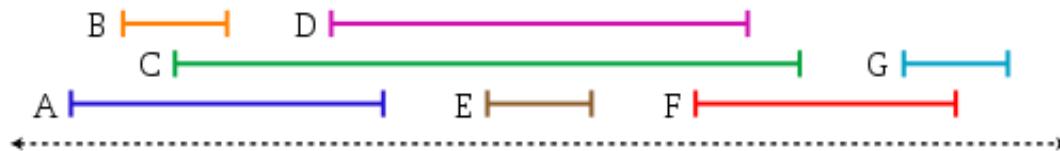
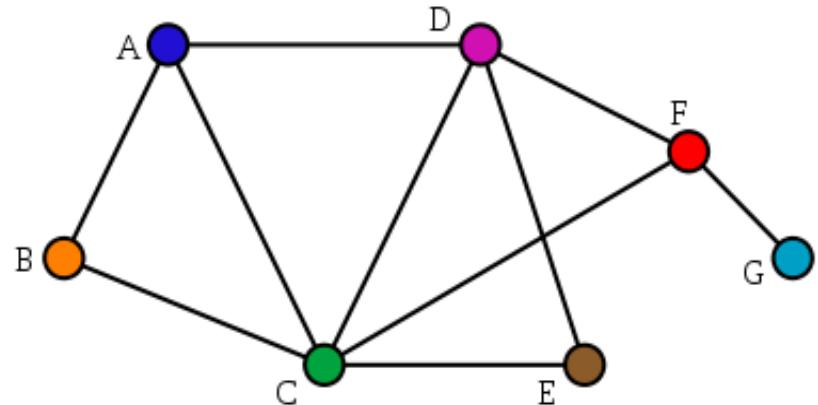
- **Problema:**

- Input: un grafo dirigido, con pesos
  - Output: el camino con mayor peso (score) que recorre todos los nodos
  - *Suena familiar?*

# Interval graphs

- **Grafos de intervalos**

- Resultan de representar intervalos en forma de *grafo*
- Los intervalos son la proyección 1D del grafo
  - 1 nodo por cada fragmento o intervalo
  - 1 arista entre cada par de intervalos que se *solapan*



Tomado de [http://en.wikipedia.org/wiki/Interval\\_graph](http://en.wikipedia.org/wiki/Interval_graph)

# Ensamblando un genoma con grafos

Reconstruir (ensamblar) un genoma circular:

ATGGCGTGCA

A partir de una serie de reads:

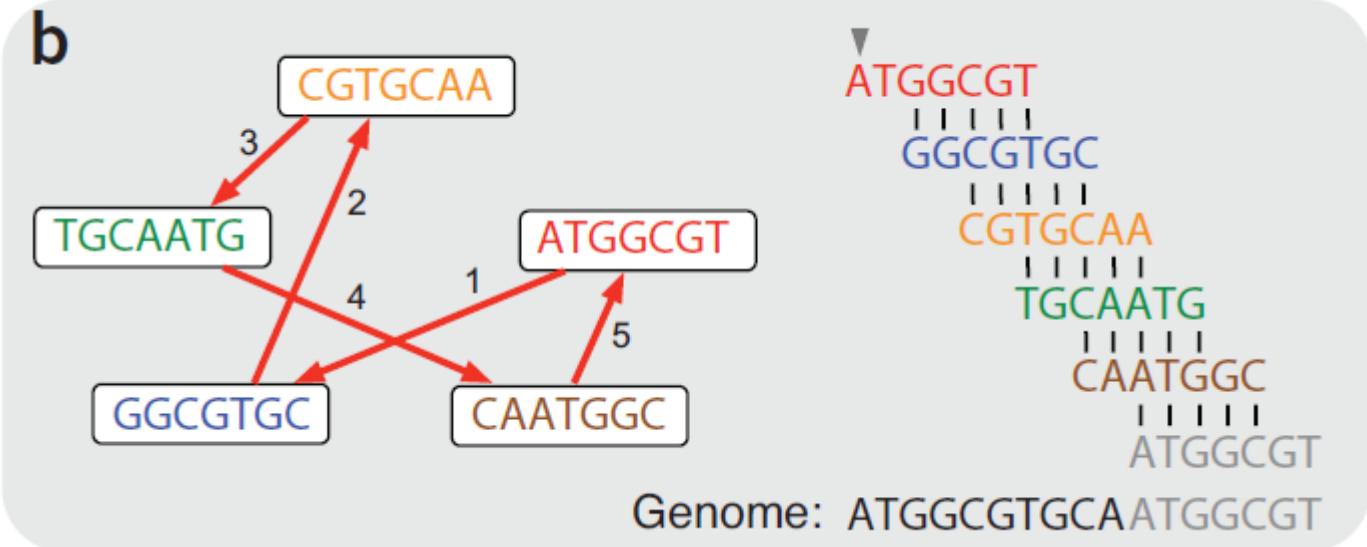
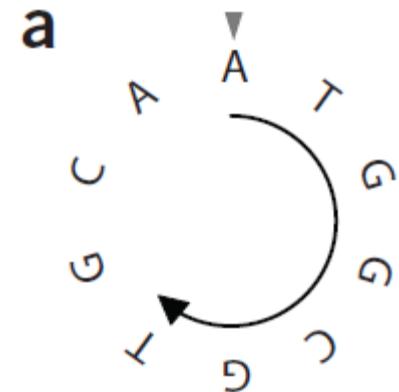
CGTGCAA

TGCAATG

ATGGCGT

GGCGTGC

CAATGGC



# Solución 1: Hamiltonian cycles

Una posible solución es representar las secuencias como un grafo de *k-mers*, donde los *edges* indiquen sufijos compartidos entre nodos. Ensamblar es buscar un camino en el grafo que pase por **todos** los *k-mers* (nodos).

Reads:

CGTGCAA

TGCAATG

ATGGCGT

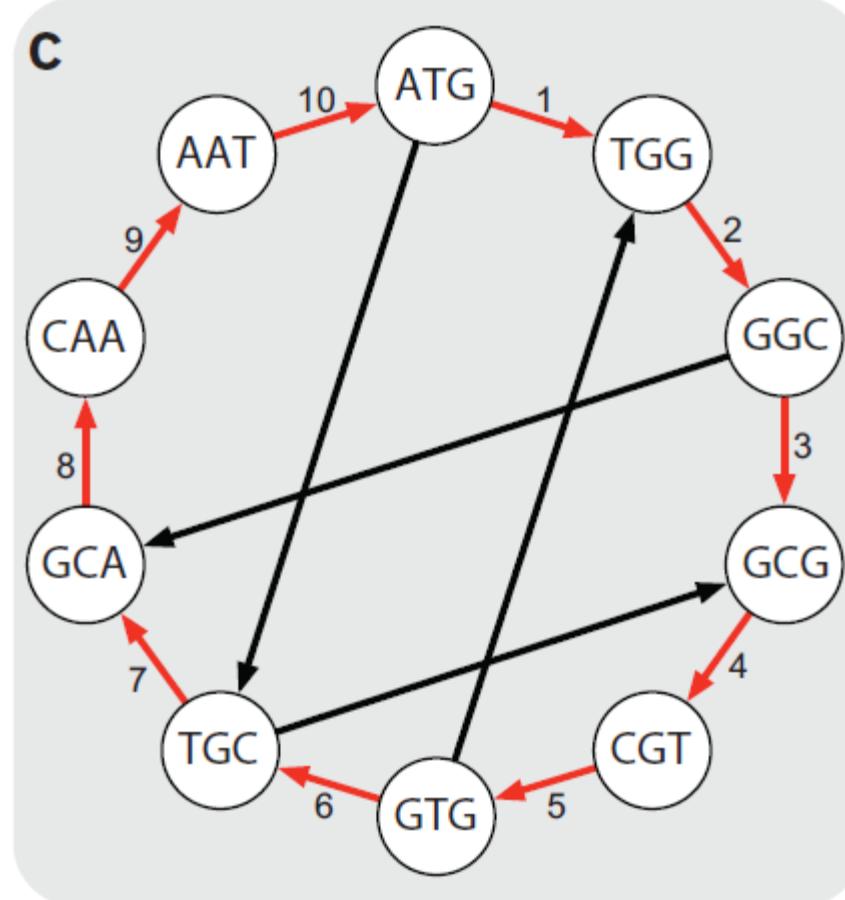
GGCGTGC

CAATGGC

Para  $k=3$ , los *k-mers* son:

CGT, GTG, TGC, GCA, CAA,

AAT, ATG, TGG, GGC, GCG



**Hamiltonian cycle**  
Visit each vertex once

## Solución 2: Eulerian cycles

Otra solución posible: representar las secuencias como un grafo de *k-mers*, donde cada *edge* es un *k-mer*, y donde los nodos son prefijos y sufijos de cada *k-mer*. En este caso hay que buscar un camino que pase por **todos** los *k-mers* (ejes).

Reads:

CGTGCAA

TGCAATG

ATGGCGT

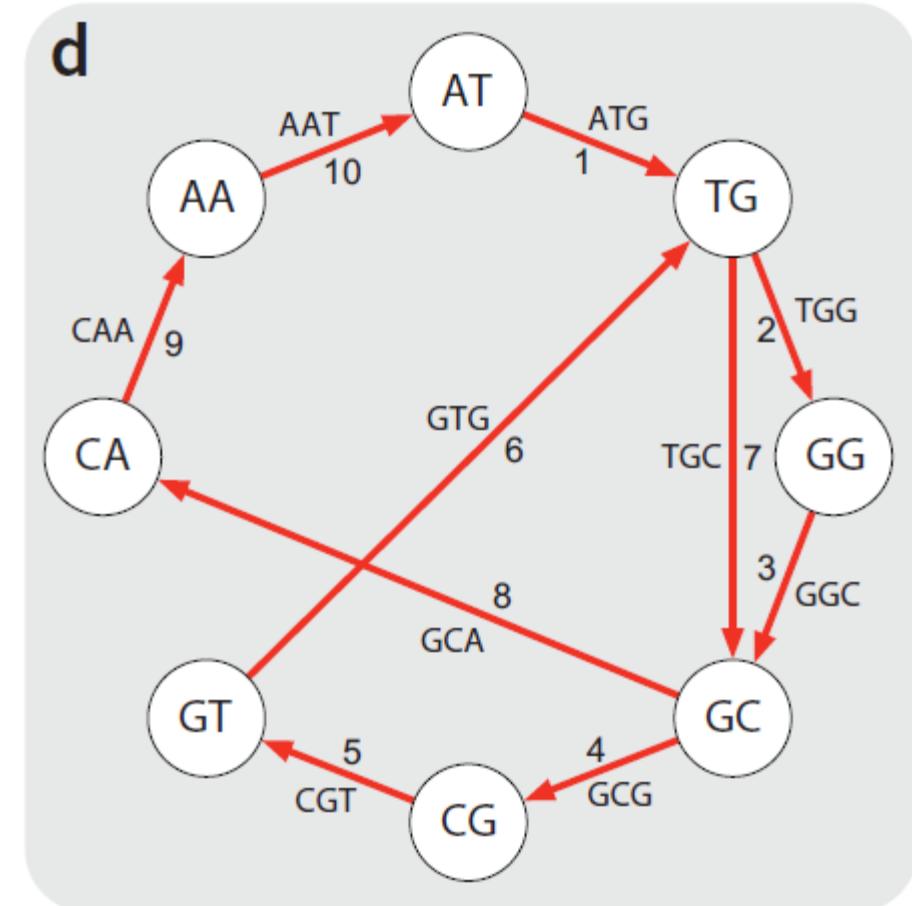
GGCGTGC

CAATGGC

Para  $k=3$ , los *k-mers* son:

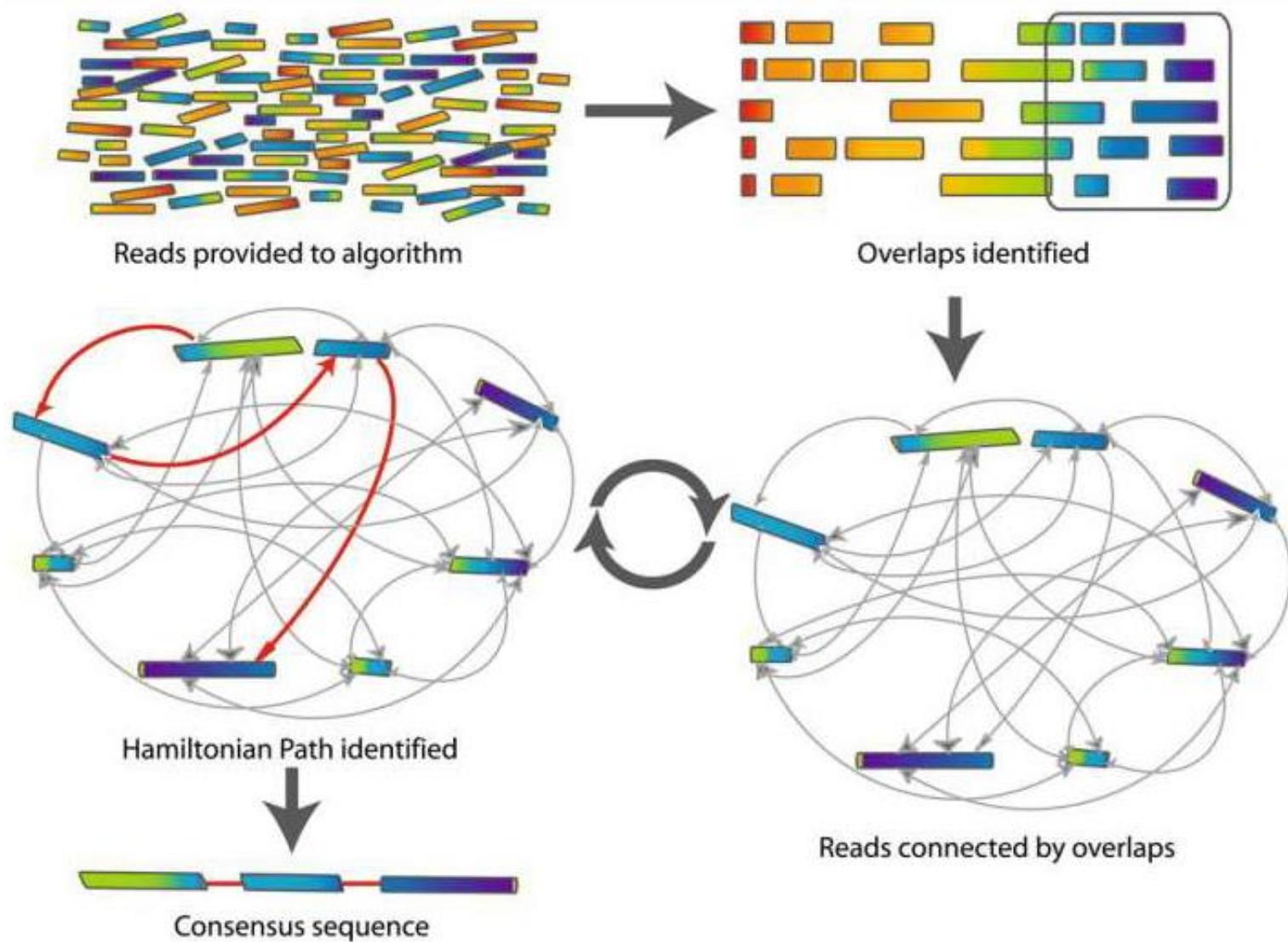
CGT, GTG, TGC, GCA, CAA,

AAT, ATG, TGG, GGC, GCG



Eulerian cycle  
Visit each edge once

# Genome assembly using graphs: overview



# Assembly of contigs using graphs

- Zhang *et al.* (1994). An algorithm based on graph theory for the assembly of contigs in physical mapping of DNA. *Bioinformatics* 10: 309–317
  - “An algorithm is described for mapping DNA contigs based on an interval graph (IG) representation ... CPU time is essentially linear with respect to the number of cosmids analyzed”

## Velvet

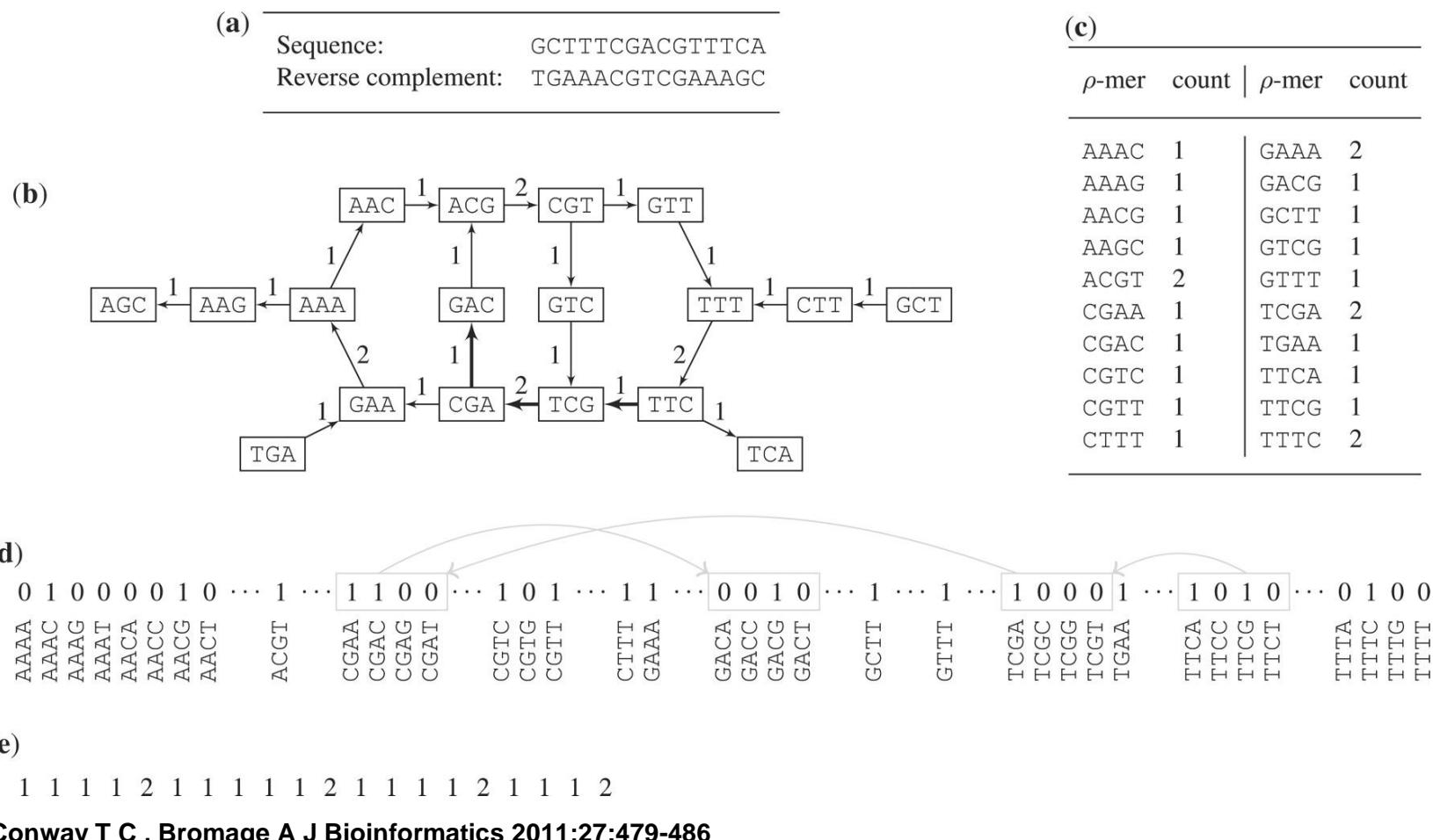
Produce un genoma borrador (*draft*) usando millones de secuencias cortas (35-100 bp, Illumina)

Basados en grafos de de Bruijn (buscan Eulerian cycles)

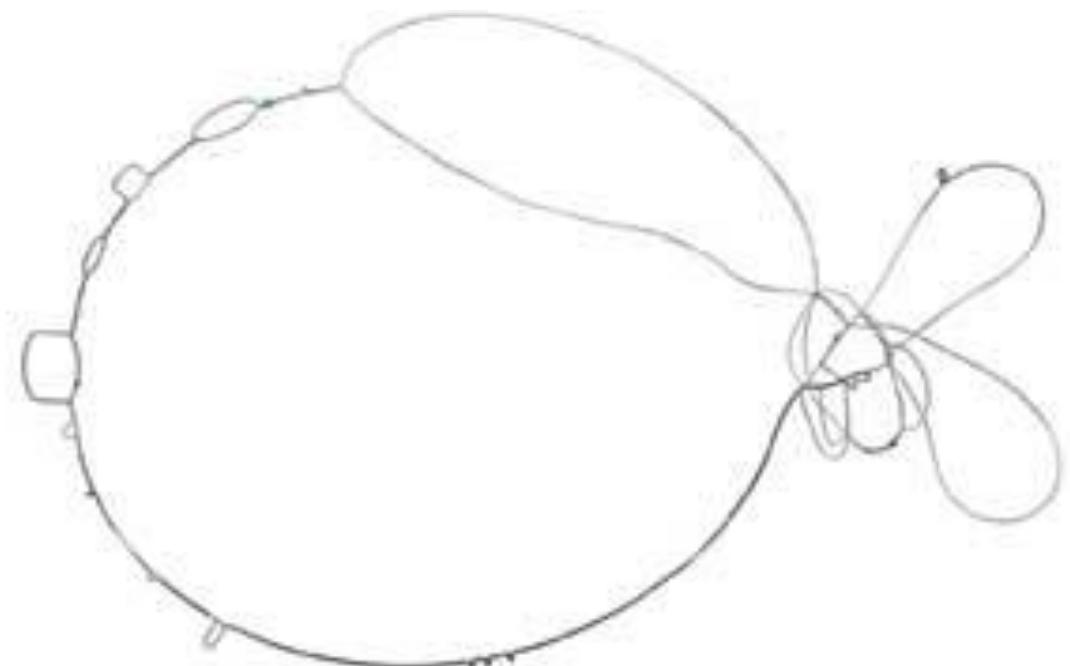
Usan k-mers de tamaño menor al tamaño de secuencia (por qué?)

Usan estrategias (heurísticas) para resolver problemas (bubbles), etc.

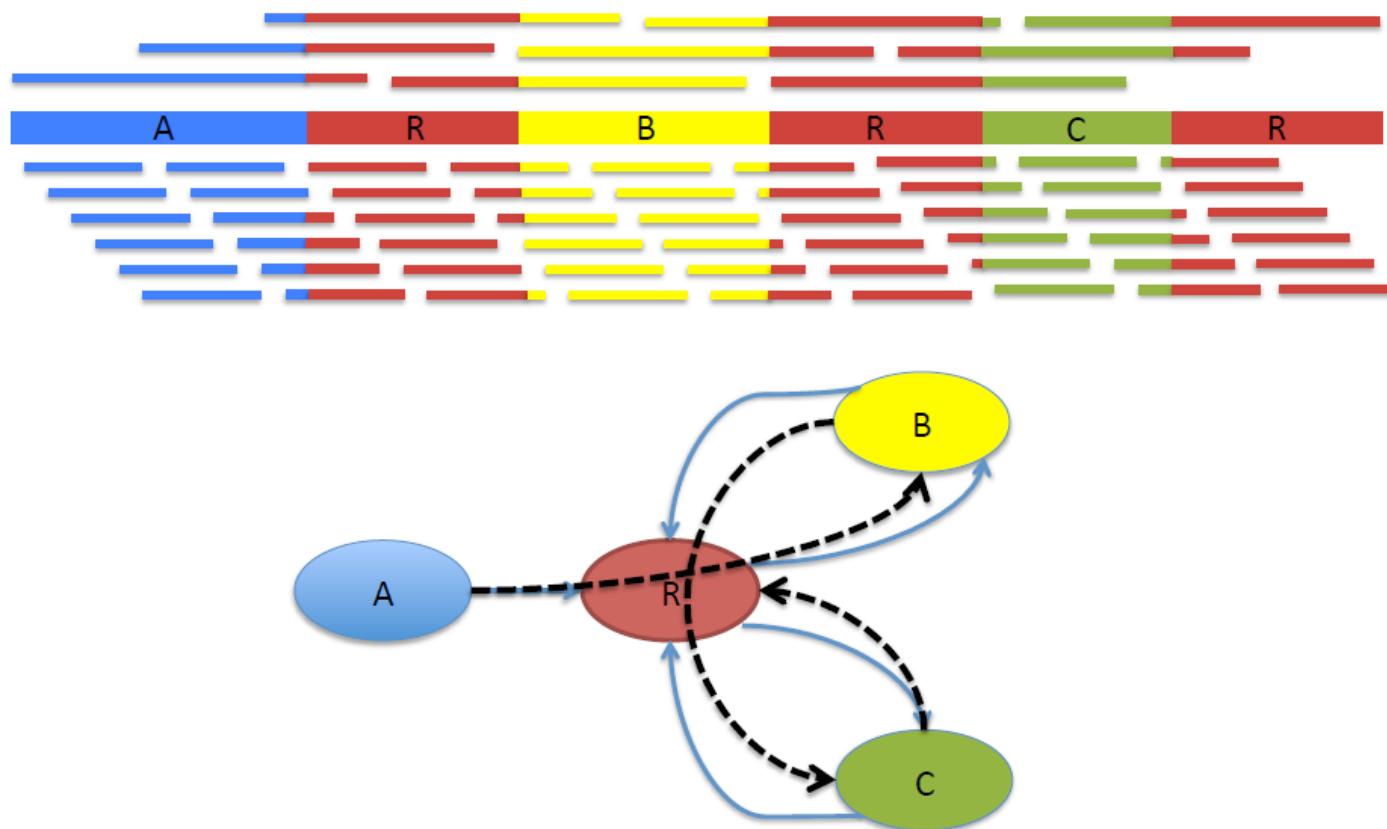
# A de Bruijn assembly graph and its representation.



## Bacterial genome assembled using a de Bruijn graph



# Assembly complexity

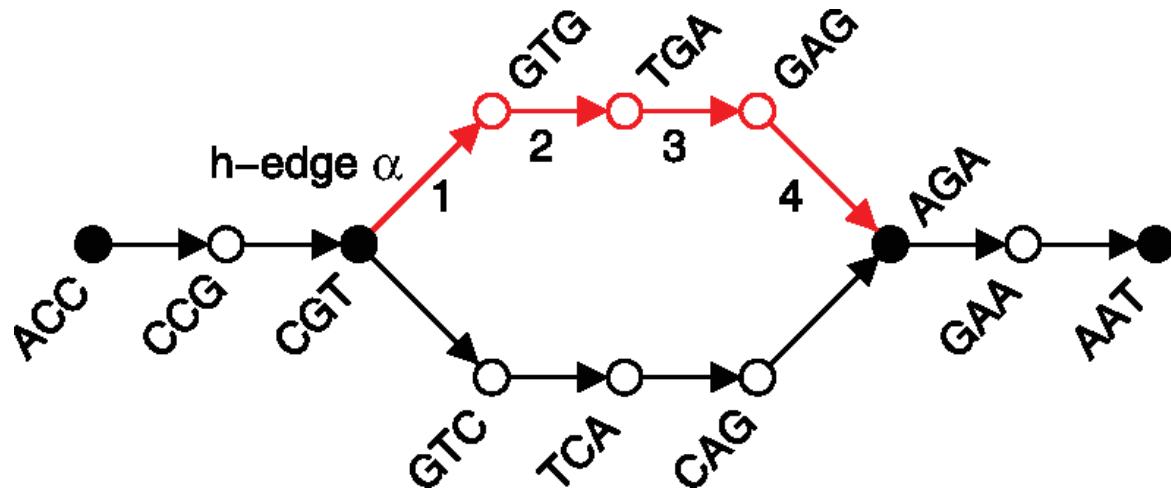
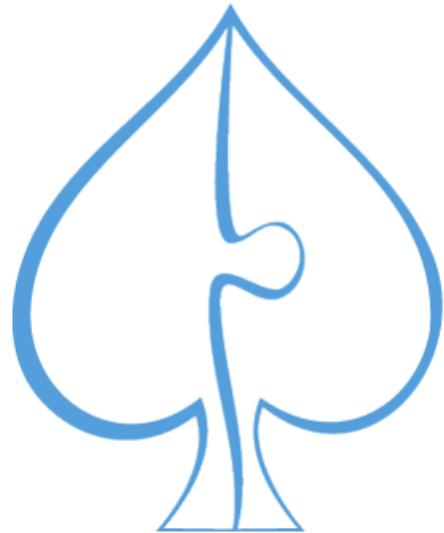


## De Bruijn graphs: selection of k

- The choice of k is important to the construction of a de Bruijn graph
- smaller k results in more tangled graphs (more repeats will be glued)
  - Smaller k works better with low coverage regions
- larger k may not adequately detect overlaps, leading to fragmented graphs.
  - Larger k works better with high coverage regions

# Multi-size de Bruijn graphs

Spades uses several values for  $k$  (manually set or inferred automatically) to create a ***multisized*** graph that minimized tangledness and fragmentation by combining various  $k$ -mers



**SPAdes: A New Genome Assembly Algorithm and Its Applications to Single-Cell Sequencing.** Anton Bankevich et al Journal of Computational Biology 19, 2012 <https://doi.org/10.1089/cmb.2012.0021>

**Assembly of long error-prone reads using de Bruijn graphs**

Yu Lin, Jeffrey Yuan, Mikhail Kolmogorov, Max W. Shen, Mark Chaisson, and Pavel A. Pevzner  
PNAS 2016 <https://doi.org/10.1073/pnas.1604560113>

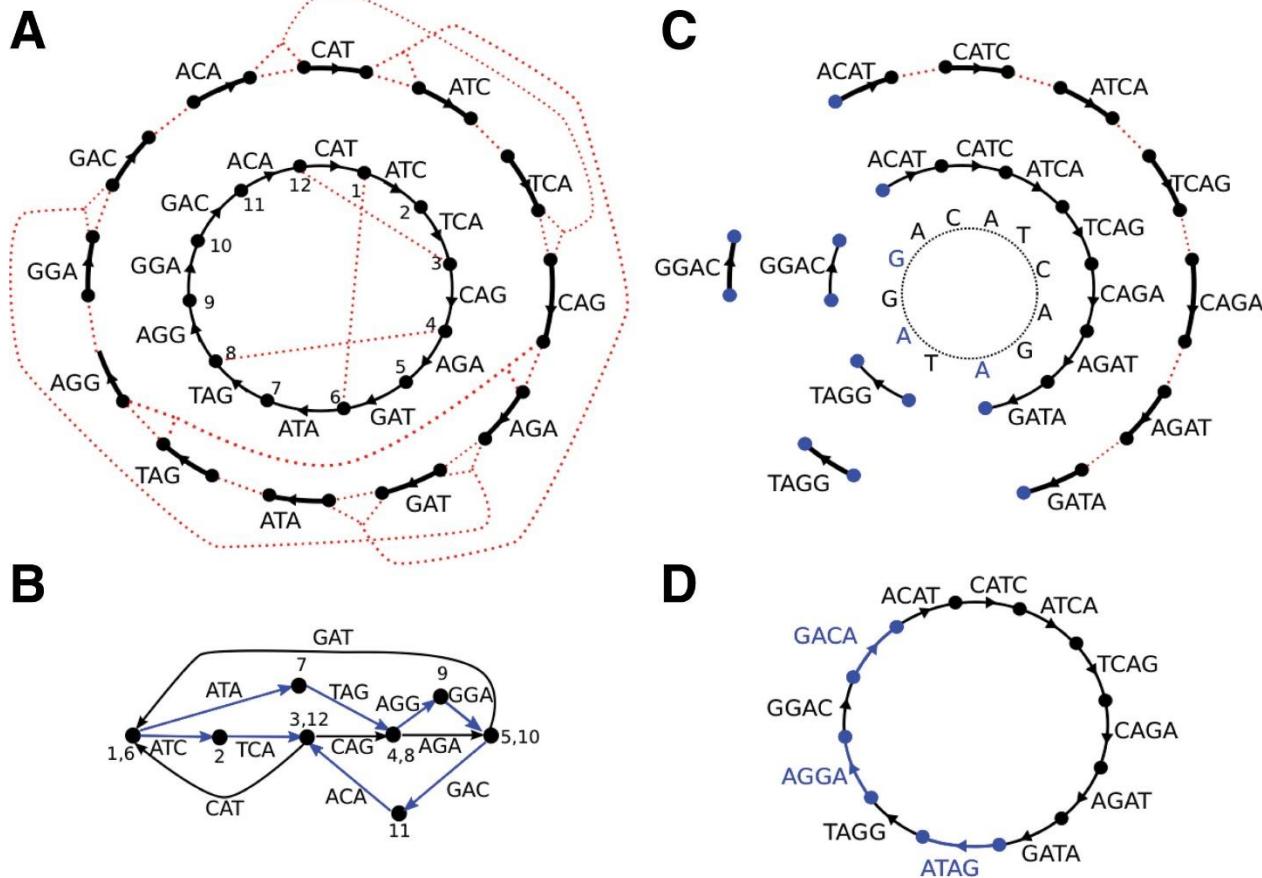
# SPAdes: multi-sized de Bruijn graphs

Genome = CATCAGATAGGA

Reads (4-mers) =  
 {ACAT, CATC, ATCA, TCAG,  
 CAGA, AGAT, GATA, TAGG,  
 GGAC}

Missing = {ATAG, AGGA,  
 GACA}

Theoretical (3-mers) = {all}



Multisized de Bruijn graph. A circular Genome CATCAGATAGGA is covered by a set of Reads consisting of nine 4-mers, {ACAT, CATC, ATCA, TCAG, CAGA, AGAT, GATA, TAGG, GGAC}. Three out of 12 possible 4-mers from Genome are missing from Reads (namely {ATAG, AGGA, GACA}), but all 3-mers from the Genome are present in the Reads. (A) The outside circle shows a separate black edge for each 3-mer from Reads. Dotted red lines indicate vertices that will be glued. The inner circle shows the result of applying some of the glues. (B) The graph DB(Reads, 3) resulting from all the glues is tangled. The three h-paths of length 2 in this graph (shown in blue) correspond to h-reads ATAG, AGGA, and GACA. Thus Reads<sub>3,4</sub> contains all 4-mers from Genome. (C) The outside circle shows a separate edge for each of the nine 4-mer reads. The next inner circle shows the graph DB(Reads, 4), and the innermost circle represents the Genome. The graph DB(Reads, 4) is fragmented into 3 connected components. (D) The multisized de Bruijn graph DB(Reads, 3, 4). Figure and text from [Bankevich et al. 2012](#).

# Short read mapping using Suffix Trees/Arrays

Qué es un Suffix Array?

Consideremos una cadena a indexar: “**AGGAGC\$**” (**\$ = ultima posición**)

i	0	1	2	3	4	5	6
S[i]	A	G	G	A	G	C	\$

Cadena indexada

List of suffixes

Prefix	I
AGGAGC\$	0
GGAGC\$	1
GAGC\$	2
AGC\$	3
GC\$	4
C\$	5
\$	6

Ordered list of suffixes

Prefix	I
\$	6
AGC\$	3
AGGAGC\$	0
C\$	5
GAGC\$	2
GC\$	4
GGAGC\$	1

Suffix Array

i	A[i]
0	6
1	3
2	0
3	5
4	2
5	4
6	1

Cuáles son todos los sufijos que empiezan con AG?

# Applications of suffix trees

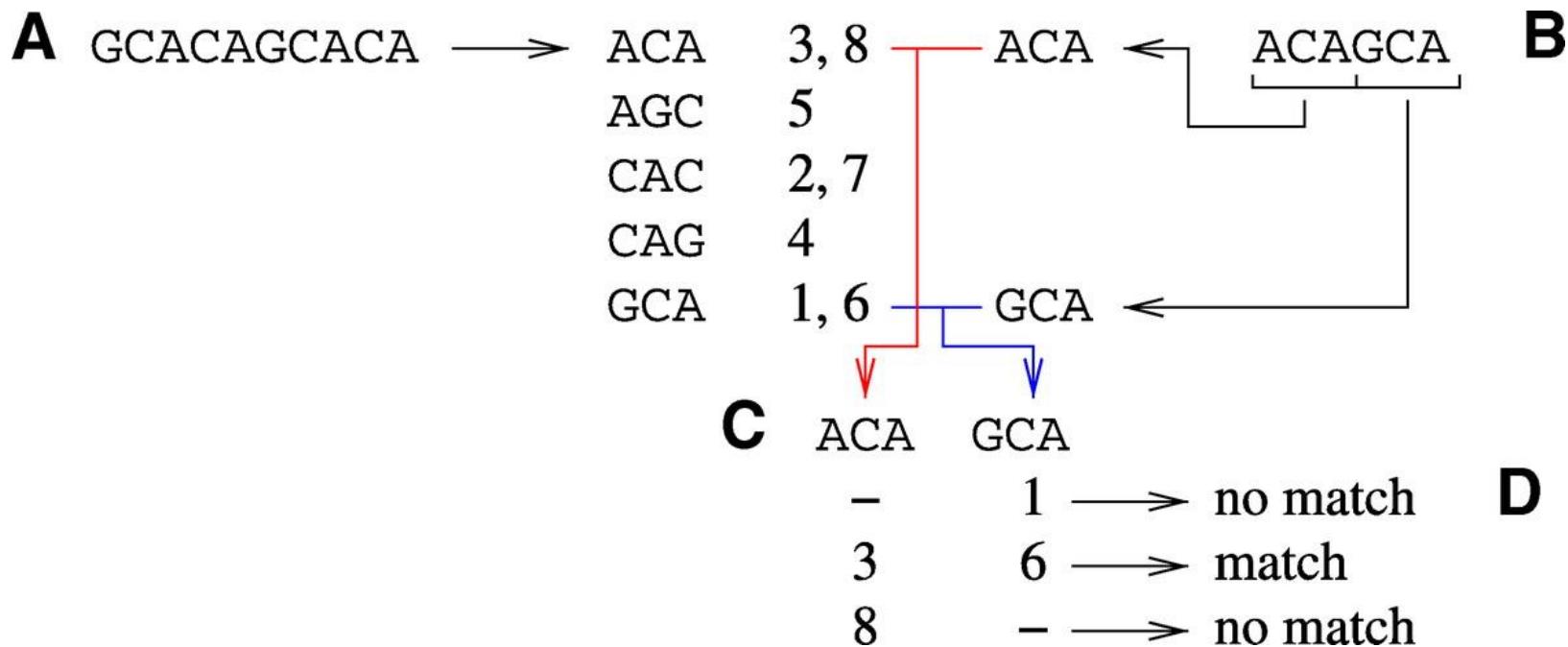
Se acuerdan del “Exact string matching” (Naïve algorithm)?



Cómo aplicarían Suffix Arrays a este problema?

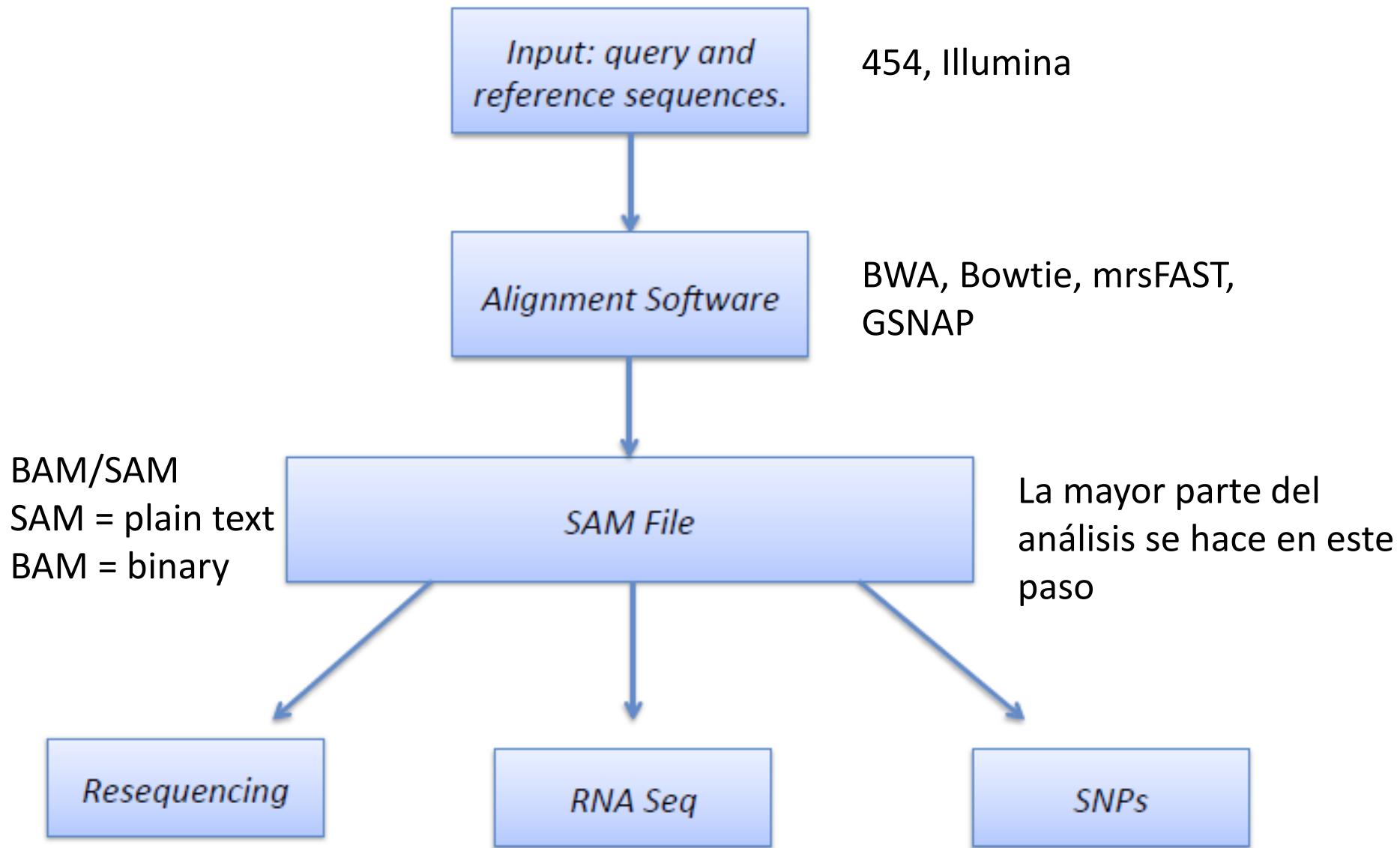
Suffix Trees/Arrays son la base algorítmica del programa  
BWA (Burrows-Wheeler Aligner, short read alignment)

# Read mapping using hashing algorithm



**FIG. 1.** The hashing algorithm. **(A)** The genome is cut into overlapping 3-mers, and their respective positions in the genome are stored. **(B)** The read is cut into 3-mers. The 3-mers from the reads are compared to 3-mers from the genome using a hashing procedure. **(C)** Positions for each seed are sorted and compared to the other seeds. **(D)** Compatible positions are kept.

# Short read mapping/alignment



# SAM format

Contiene un header (opcional) y una sección de alineamientos

@HD	VN:1.0	SO:coordinate							
@SQ	SN:1	LN:249250621 AS:NCBI37	UR:/data/local/ref/GATK/human_g1k_v37.fasta						
	M5:1b22b98cdeb4a9304cb5d48026a85128								
@SQ	SN:2	LN:243199373 AS:NCBI37	UR:/data/local/ref/GATK/human_g1k_v37.fasta						
	M5:a0d9851da00400dec1098a9255ac712e								
@SQ	SN:3	LN:198022430 AS:NCBI37	UR:/data/local/ref/GATK/human_g1k_v37.fasta						
	M5:fd7d811849cc2fadefbc929bb925902e5								
@RG	ID:UM0098:1	PL:ILLUMINA PU:HWUSI-EAS1707-615LHAAXX-L001		LB:80		DT:2010-05-05T20:00:00-0400		SM:SD37	

1:497:R:-272+13M17D24M	113	1	497	37	37M	15	100338662	0	
	CGGGTCTGACCTGAGGAGAACTGTGCTCCGCCTTCAG			0;==-=9;>>>->>>>>>>->>>>>>>>			XT:A:U		NM:i:0
SM:i:37	AM:i:0	X0:i:1	X1:i:0	XM:i:0	XO:i:0	XG:i:0	MD:Z:37		
19:20389:F:275+18M2D19M	99	1	17644	0	37M	=	17919	314	
	TATGACTGCTAATAATACCTACACATGTTAGAACCAT			>>>>>>>>>>>>>>>>>><>><>><>>4:>><9			RG:Z:UM0098:1		
XT:A:R	NM:i:0	SM:i:0	AM:i:0	X0:i:4	X1:i:0	XM:i:0	XO:i:0	XG:i:0	
MD:Z:37									
19:20389:F:275+18M2D19M	147	1	17919	0	18M2D19M	=	17644	-314	
	GTAGTACCAACTGTAAGTCCTTATCTTCATACTTTGT			;44999;499<8<8<<8<><<<><7;<<<><<			XT:A:R		NM:i:2
SM:i:0	AM:i:0	X0:i:4	X1:i:0	XM:i:0	XO:i:1	XG:i:2	MD:Z:18^CA19		
9:21597+10M2I25M:R:-209	83	1	21678	0	8M2I27M	=	21469	-244	
	CACCACATCACATACCAAGCCTGGCTGTGCTTCT			<9<<5><<<><>><9>><>>9>>>>			XT:A:R		NM:i:2
SM:i:0	AM:i:0	X0:i:5	X1:i:0	XM:i:0	XO:i:1	XG:i:2	MD:Z:35		

**Table 2.**

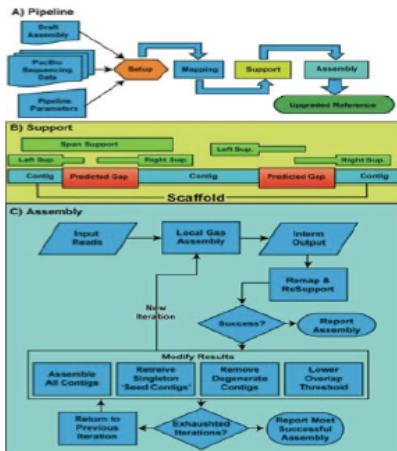
Global characteristics of the mapping tools

<i>Tool</i>	<i>Format</i>	<i>Algorithm</i>	<i>Threads</i>	<i>Gaps</i>	<i>Mismatches</i>
BWA	SAM	BWT	yes	yes	yes
Novoalign	SAM	hash the ref.	yes	yes	yes
Bowtie	SAM	BWT	yes	no	yes
SOAP2	perso	BWT	yes	no	at most 2
BFAST	SAM	hash the ref.	yes	yes	yes
SSAHA2	SAM	hash the ref.	no	no	yes
MPscan	perso	suffix tree	no	no	no
GASSST	SAM	hash the ref.	yes	yes	yes
PerM	SAM	hash the ref.	no	no	yes

SAM, Sequence Alignments Map.

# PacBio Assembly Algorithms

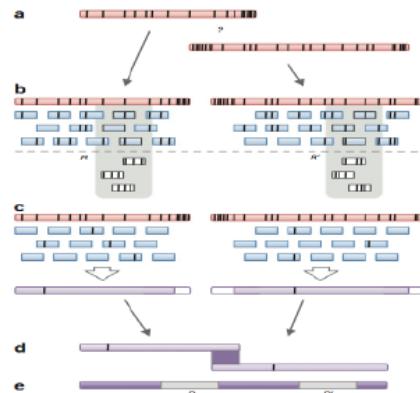
## PBJelly



### Gap Filling and Assembly Upgrade

English et al (2012)  
PLOS One. 7(11):e47768

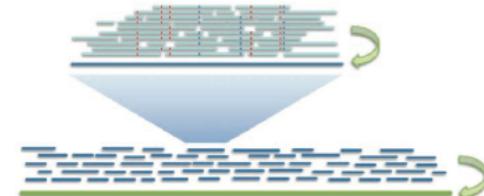
## PacBioToCA & ECTools



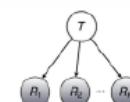
### Hybrid/PB-only Error Correction

Koren, Schatz, et al (2012)  
Nature Biotechnology. 30:693–700

## HGAP & Quiver



$$\Pr(R | T)$$
$$\Pr(R | T) = \prod_k \Pr(R_k | T)$$



Quiver Performance Results Comparison to Reference Genome ( <i>M. ruber</i> ; 3.1 MB; SMRT® Cells)		
	Initial Assembly	Quiver Consensus
QV	43.4	54.5
Accuracy	99.99540%	99.99964%
Differences	141	11

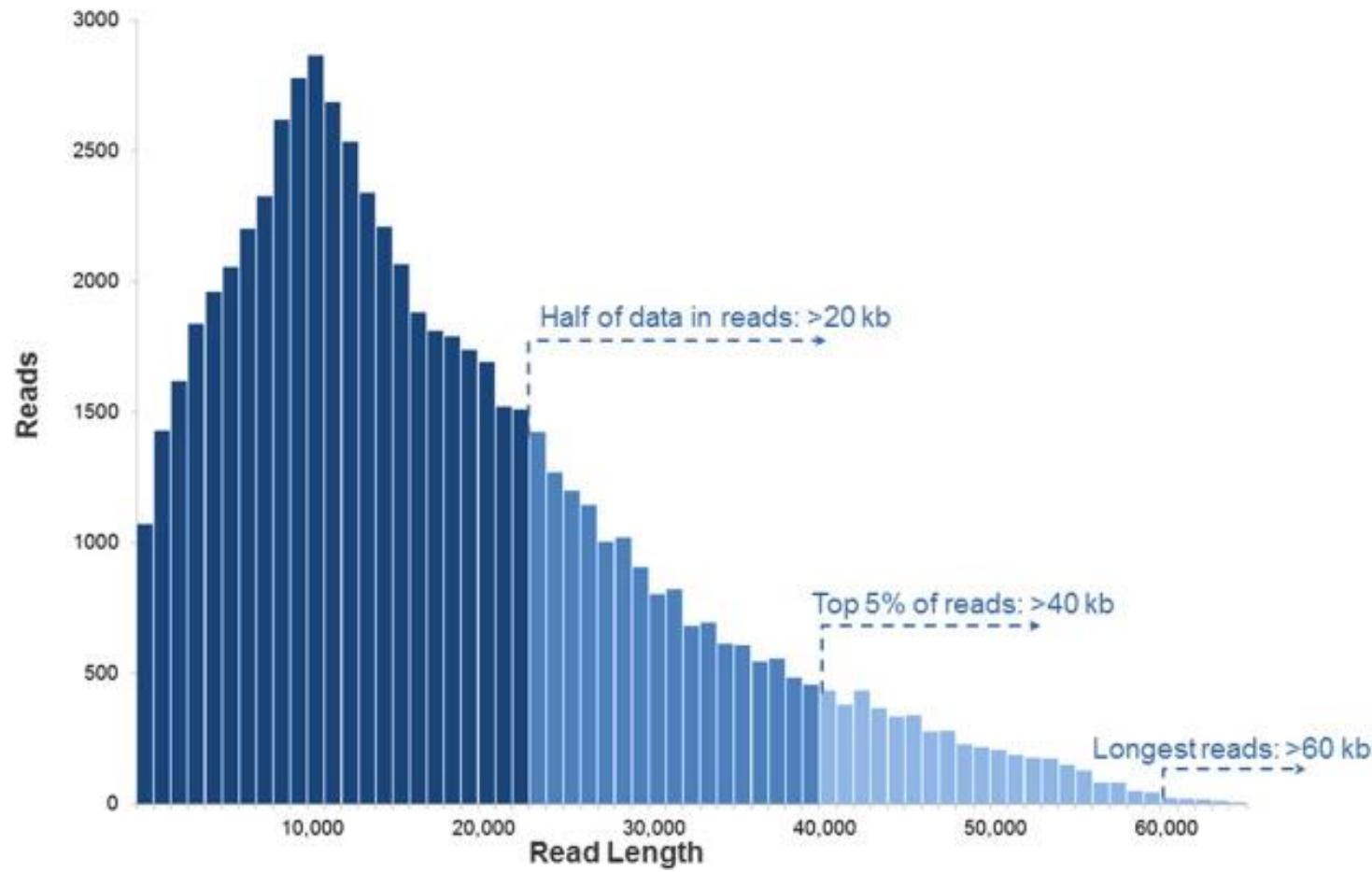
### PB-only Correction & Polishing

Chin et al (2013)  
Nature Methods. 10:563–569

< 5x

PacBio Coverage

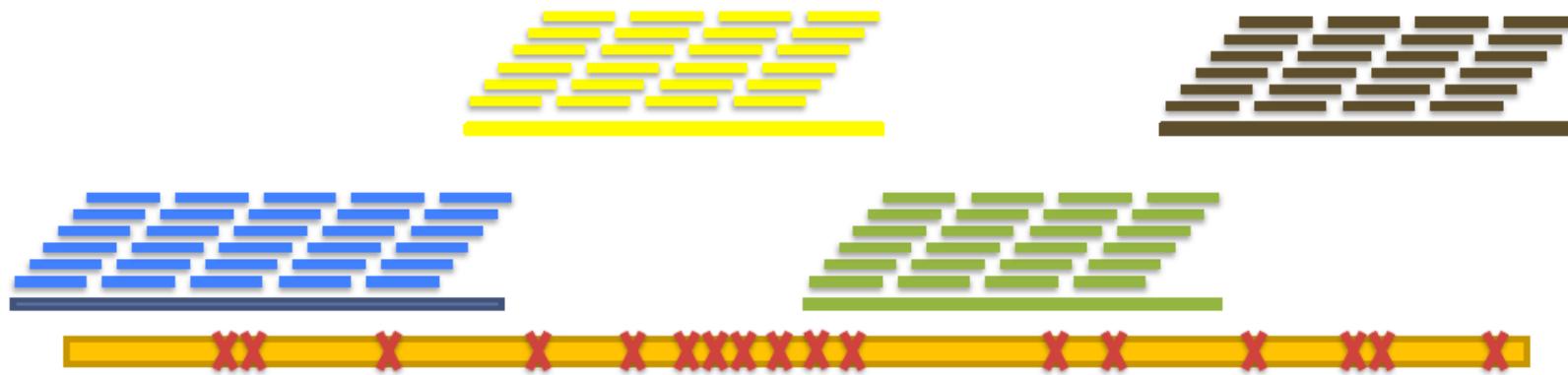
> 50x



# Correct errors in long reads using short reads

## ECTools: Error Correction with pre-assembled reads

<https://github.com/jgurtowski/ectools>



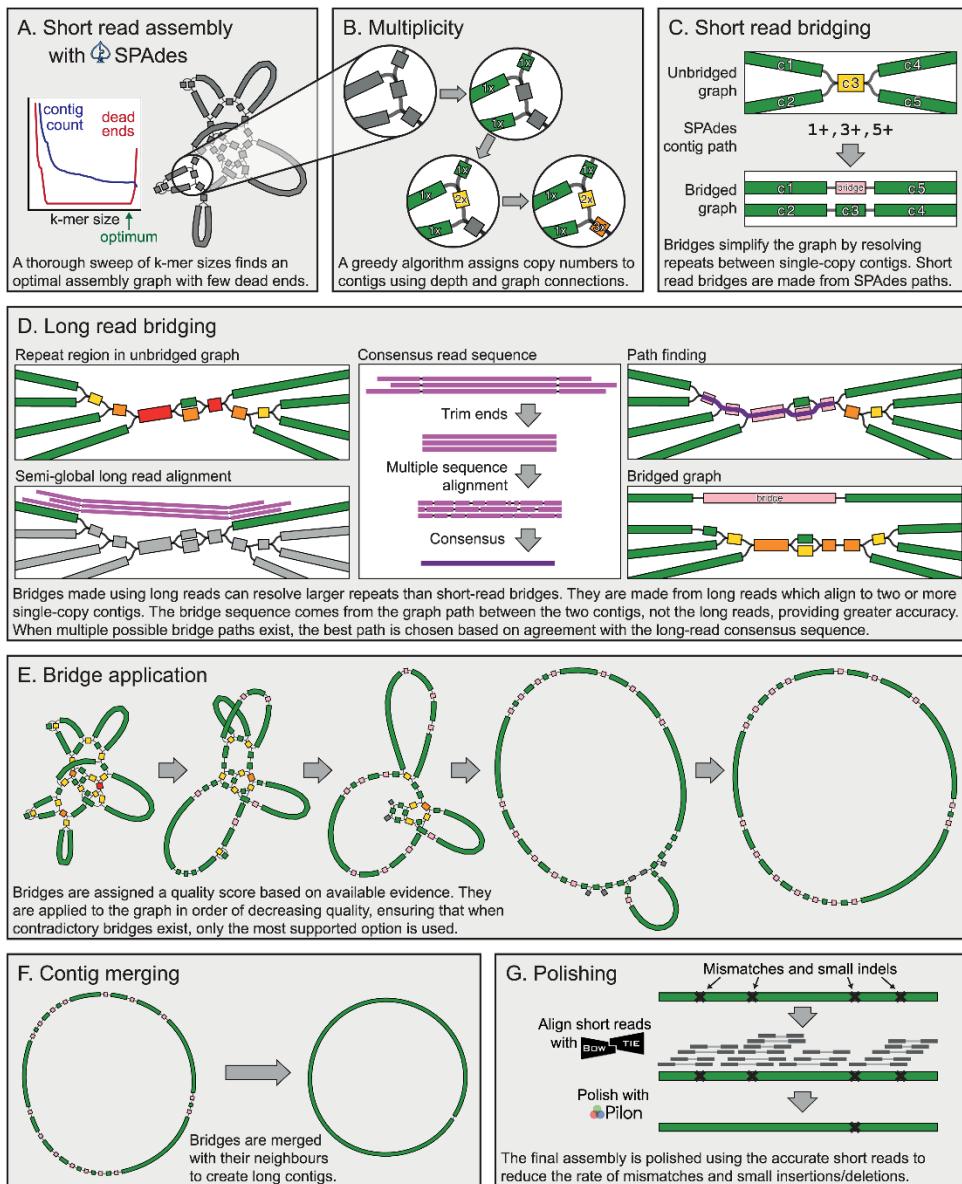
**Short Reads -> Assemble Unitigs -> Align & Select -> Error Correct**

Can Help us overcome:

1. Error Dense Regions – Longer sequences have more seeds to match
2. Simple Repeats – Longer sequences easier to resolve

**However, cannot overcome Illumina coverage gaps & other biases**

# Hybrid assemblies: short + long reads

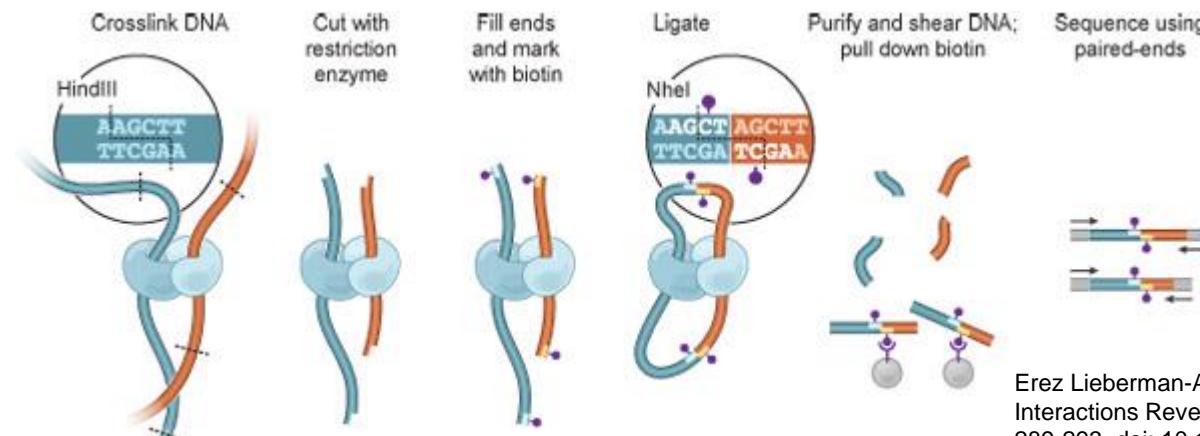
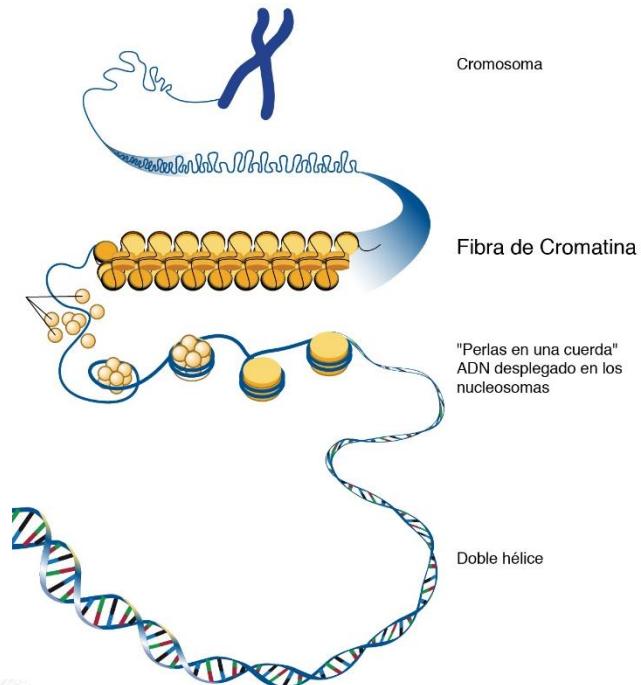


- Unicycler is designed specifically for **hybrid assembly** (that is, using both short- and long-read sequencing data) of small (e.g., bacterial, viral, organellar) genomes.
- Unicycler employs a multi-step process that utilizes a number of software tools

# Último hito: estudio de estructura cromosómica

## Hi-C: proximity ligation + secuenciación

- Estudio **no sesgado** a escala genómica de interacciones a nivel de la cromatina
- Revela arquitectura cromosómica a diferentes niveles
  - Territorios cromosómicos
  - Regiones donde la cromatina es abierta vs cerrada
  - Estructura de la cromatina a escala de megabases (millones de bases)

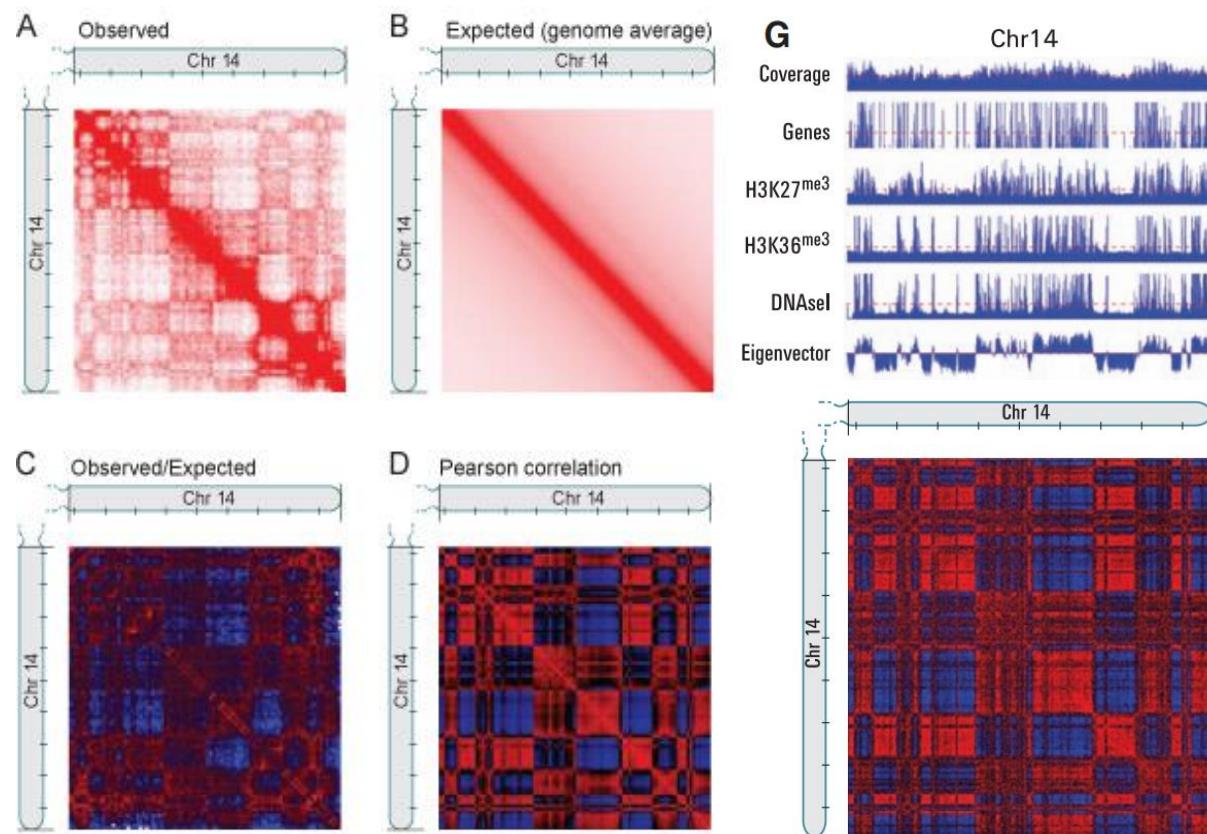


Erez Lieberman-Aiden E, et al. Comprehensive Mapping of Long-Range Interactions Reveals Folding Principles of the Human Genome. *Science* (326), 289-293, doi: 10.1126/science.1181369 (2009).  
van Berkum, NL, et al. Hi-C: A Method to Study the Three-dimensional Architecture of Genomes. *J. Vis. Exp.* (39), e1869, doi:10.3791/1869 (2010).

# Hi-C: proximity ligation + sequencing

Hi-C revela interacciones  
*intercromosómicas* +  
*intracromosómicas*

Ayudan a mejorar ensambles:  
corrijen mala asignación de  
contigs a scaffolds o  
cromosomas), además de  
identificar inversions y  
translocaciones, en estudios  
comparativos.



Erez Lieberman-Aiden E, et al. Comprehensive Mapping of Long-Range Interactions Reveals Folding Principles of the Human Genome. *Science* (326), 289-293, doi: 10.1126/science.1181369 (2009).

*Assessment / Validation*

# **ASSEMBLY VALIDATION**

## N50

- Calidad del ensamble en términos de *contiguidad*
- **N50** es similar a una mediana o media de longitudes de contigs
- *Es la longitud del contig más corto* a partir del cual el 50% de las bases se encuentran repartidas entre él mismo y otros contigs más cortos
- Ej si tenemos 7 contigs con longitudes
  - 1, 1, 3, 5, 8, 12, 20
- El N50 es **12** porque:
  - $1 + 1 + 3 + 5 + 8 + 12 + 20 = 50$  (la longitud acumulada de todo el ensamble)
  - $50/2 = 25$  (la mitad de la longitud sumada de todo el ensamble)
  - Y si empezamos desde el contig más corto y vamos sumando hasta conseguir llegar a una longitud acumulada  $\geq 25$  ...
  - $1 + 1 + 3 + 5 + 8 + 12 = 30$
  - O sea, el sexto contig (de longitud **12**) es el primero en el que alcanzamos o pasamos la mitad de la longitud del ensamble

# Métricas para evaluar assemblies

## L50

- El *número mínimo de contigs* cuya longitud suma 50% del tamaño del ensamble
- Ej si tenemos 7 contigs con longitudes
  - 1, 1, 3, 5, 8, 12, 20
  - El L50 es 6 porque seis es el número de contigs con los que alcanzamos o pasamos la mitad de la longitud del ensamble

## N90

- Similar al N50 (pero pide 90% de las bases)

## NG50

- Similar al N50 pero en lugar de referirse a la longitud total del ensamble, se refiere a la longitud total del *genoma*
- *Util porque el N50 no permite comparar entre ensambles de diferentes tamaños (pero NG50 si)*

Y hay más métricas .... Ver [https://en.wikipedia.org/wiki/N50,\\_L50,\\_and\\_related\\_statistics](https://en.wikipedia.org/wiki/N50,_L50,_and_related_statistics)

Una métrica es solamente eso. Una herramienta.

**Usarla con cuidado!**

## PROBLEMAS con el N50!

Si intentamos optimizar el N50 podemos forzar  
(recompensar) malos ensambles

- Un assembler agresivo puede excederse al unir contigs simplemente buscando incrementar el N50
  - Ej 1, 1, 3, 5, **8, 12**, 20 (contigs del ejemplo anterior,  $N50 = 12$ )
  - 1, 1, 3, 5, 8, **20**, 20 (aggressive join de los contigs de longitudes 8 y 12)
  - Ahora el N50 es **20**

# Validación de los ensambles

## Auto-consistencia

- Mapear de nuevo reads contra contigs
- Chequear errores o inconsistencias

## Segunda opinión / validación externa

- Usar dos métodos de secuenciación complementarios
  - Illumina + PacBio
  - Illumina + Nanopore
- Validar regiones por PCR
  - Util para validar o para resolver regiones difíciles
- Hi-C (chromatin contact maps)
  - Hi-C, 3-C Seq, Capture-C
  - Familia de métodos para caracterizar interacciones a nivel de cromatina
  - Mapas de regiones del genoma que están cercanas entre si
- Mapa óptico global del genoma
  - [https://en.wikipedia.org/wiki/Optical\\_mapping](https://en.wikipedia.org/wiki/Optical_mapping)

# Material de lectura

- J. Setubal and J. Meidanis, **Introduction to Computational Molecular Biology**, PWS Publishing Company, Boston, 1997
- D. Gusfield, **Algorithms on Strings, Trees and Sequences**, Cambridge University Press, 1997.
- Compeau PEC, Pevzner PA, Tesler G. How to apply de Bruijn graphs to genome assembly. *Nature Biotechnol* 29: 987, 2011.
- Li H, Holmer N. A survey of sequence alignment algorithms for next-generation sequencing. *Briefings in Bioinformatics* 11: 473, 2010.
- Riberiro FJ et al. Finished bacterial genomes from shotgun sequence data. *Genome Res* 22: 2270, 2012
- Nagarajan, N., & Pop, M. (2013). Sequence assembly demystified. *Nature Reviews Genetics*, 14(3), 157–167. doi:10.1038/nrg3367
- Rice, E. S., & Green, R. E. (2018). New Approaches for Genome Assembly and Scaffolding. *Annual Review of Animal Biosciences*, 7(1). doi:10.1146/annurev-animal-020518-115344