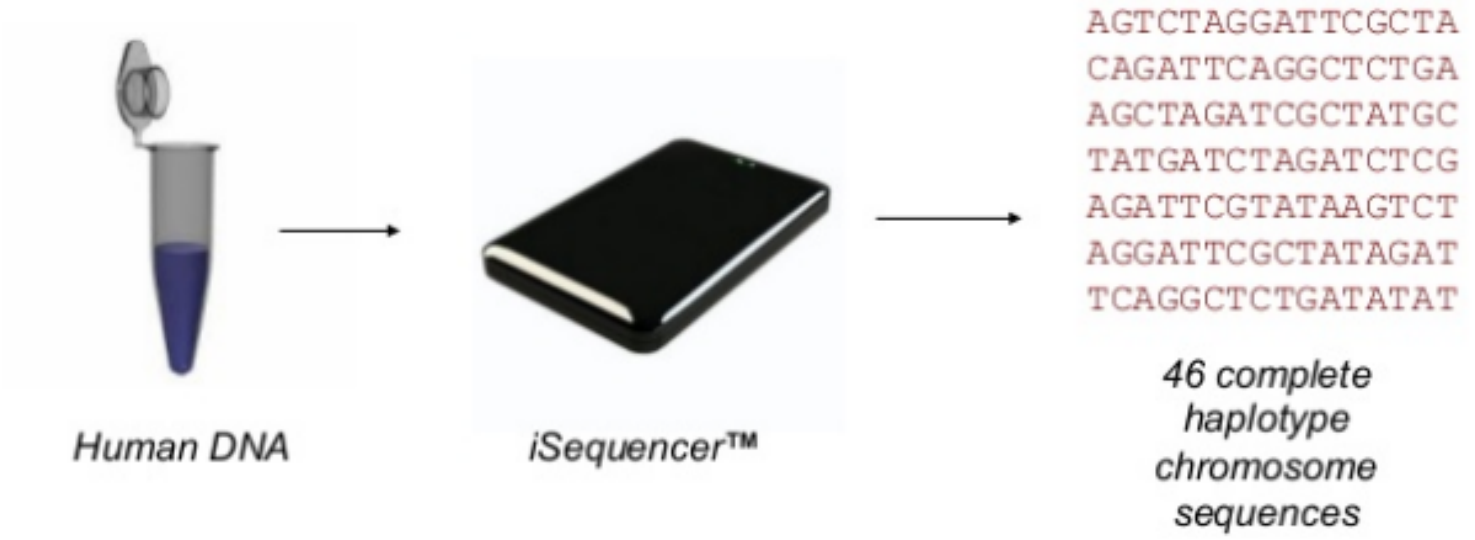
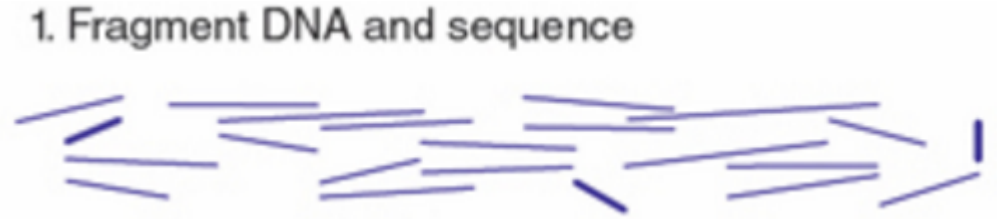
Lezione 6: Assemblaggio di un genoma (De-novo assembly)

Mondo ideale



Mondo reale



- Non è possibile sequenziare in un'unica reazione il genoma di molti organismi
  - Nessuno strumento disponibile (per ora)

- Si possono sequenziare frammenti corti da esso
  - 100 contemporaneamente (Sanger)
  - 100,000 cont. (Roche 454)
  - 1,000,000 cont. (PGM)
  - 10,000,000 cont. (Proton, MiSeq)
  - 100,000,000 cont. (HiSeq)

Cosa significa “assemblare” un genoma?

- Il processo mediante il quale si ricostruisce la sequenza della molecola di DNA originale utilizzando solamente le reads

- Paragonabile alla ricostruzione di un “puzzle”
  - Identificare quali pezzi stanno insieme (reads overlap)
  - Presenza di pezzi mancanti (sequencing bias)
  - Pezzi rovinati (sequencing errors)

Esempio

Friends,

Romans,

countrymen,

lend me your ears;

By William Shakespeare. (from Julius Caesar, spoken by Marc Antony)

Esempio

- ***Reads***

```
ds, Romans, count
ns, countrymen, le
Friends, Rom
send me your ears;
crymen, lend me
```

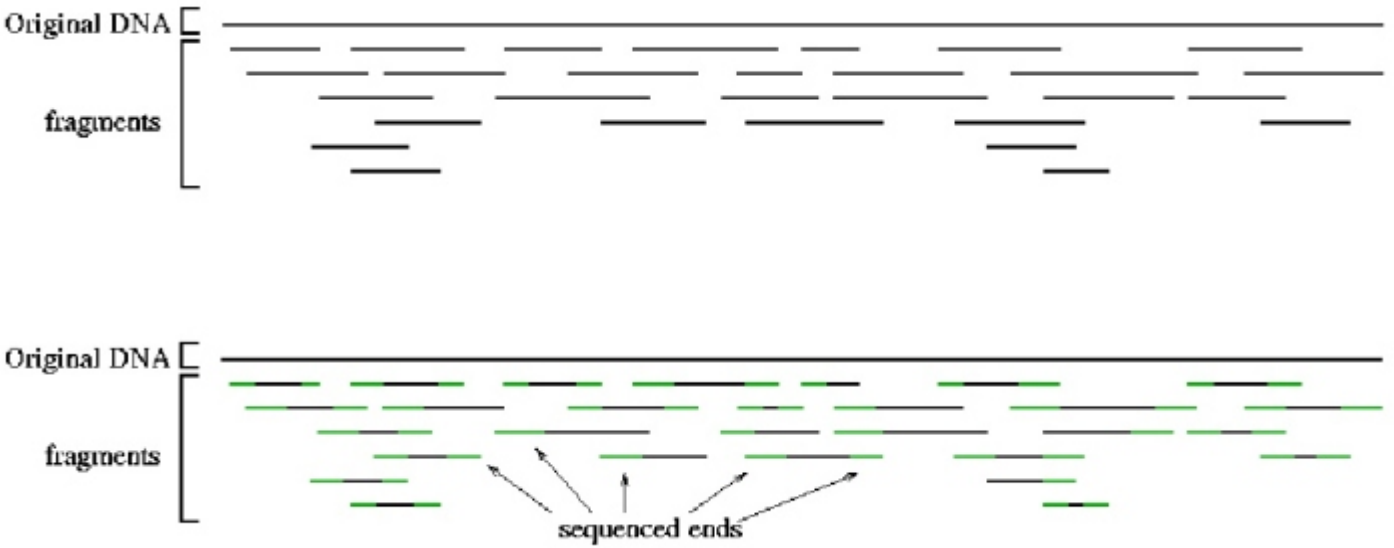
- ***Overlaps***

```
Friends, Rom
      ds, Romans, count
            ns, countrymen, le
                  crymen, lend me
                        send me your ears;
```

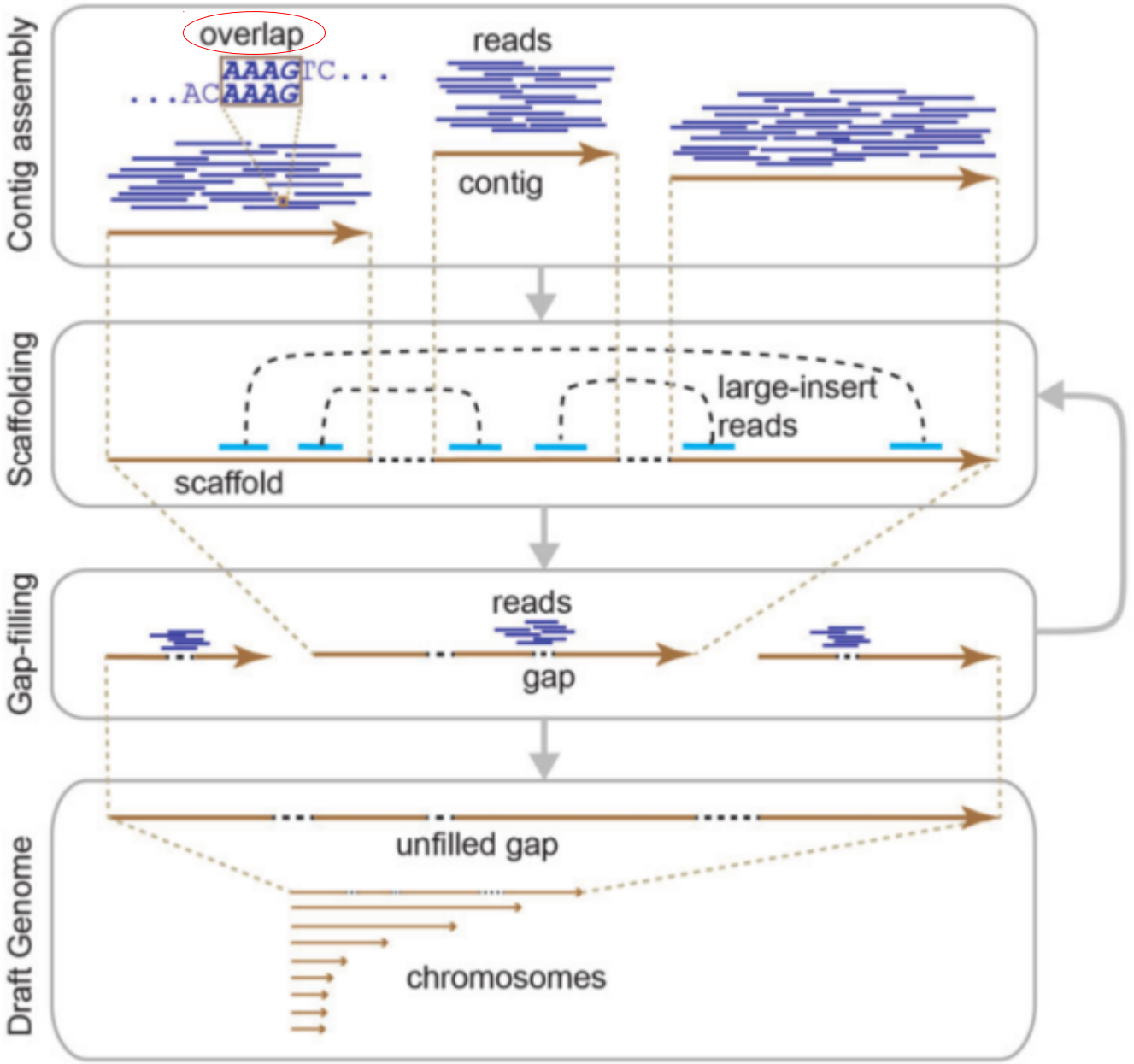
- ***Majority consensus***

```
Friends, Romans, countrymen, lend me your ears;
```

Esempio genomico



Schema generale



Contig

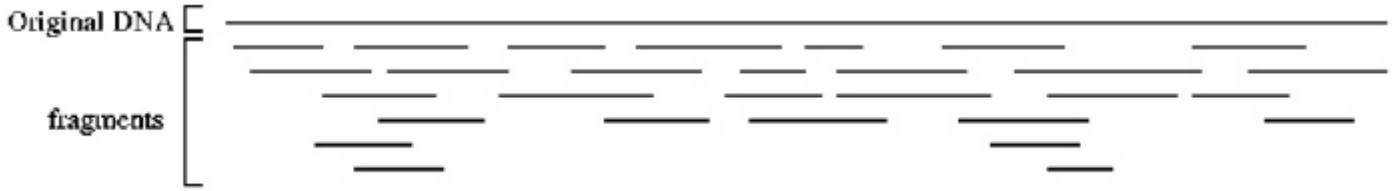
Scaffolds

Rifinitura

Bozza di genoma

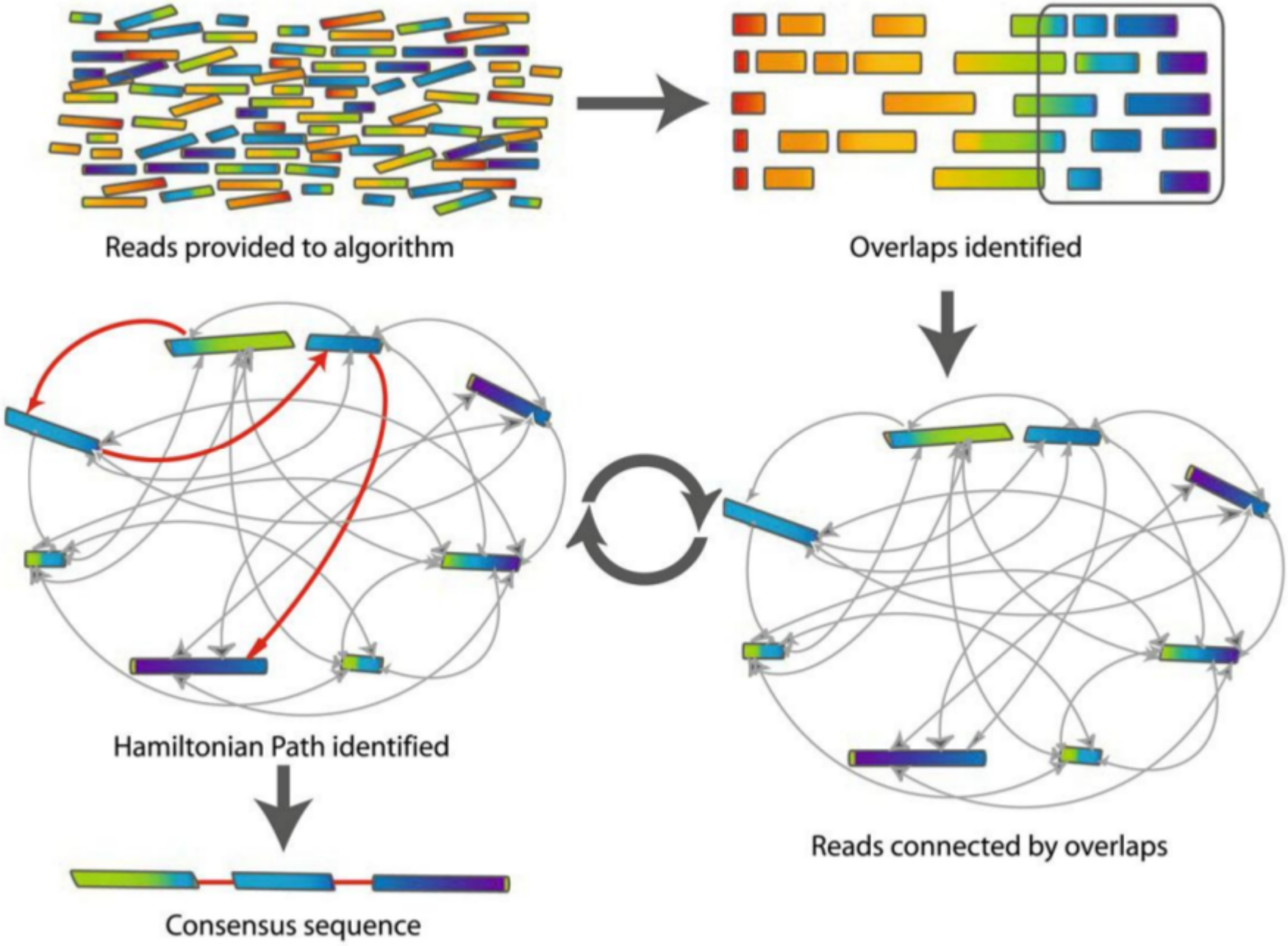
Approcci per l'assemblaggio

- Overlap-layout-consensus (OLC)
- de Bruijn graphs
- string graphs
- seed and extend
- Greedy assembly



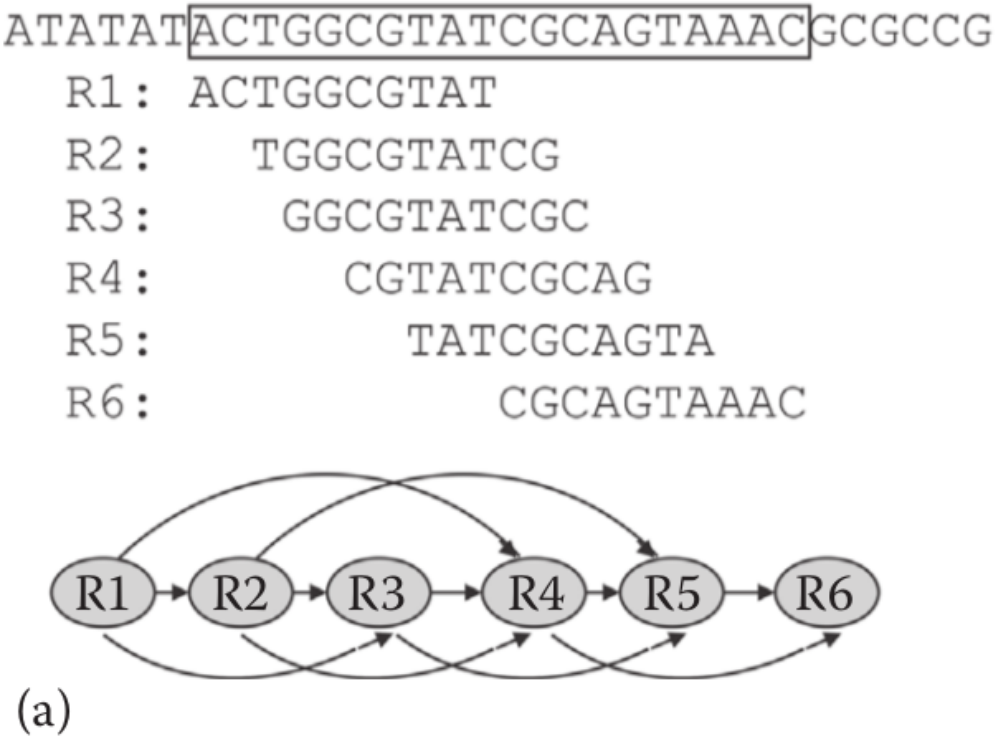
I diversi approcci cercano di **ricostruire l'intera sequenza di DNA** prima della frammentazione, attuando diverse strategie per diminuire il tempo di calcolo

Creazione dei contig



OLC

- Tre passaggi:
  - Identificazione della sovrapposizione tra reads
  - Rappresentazione delle sovrapposizioni in un grafo
  - Costruzione di un consenso:
    - Cammino Hamiltoniano: percorso che visita ogni nodo (reads) una sola volta



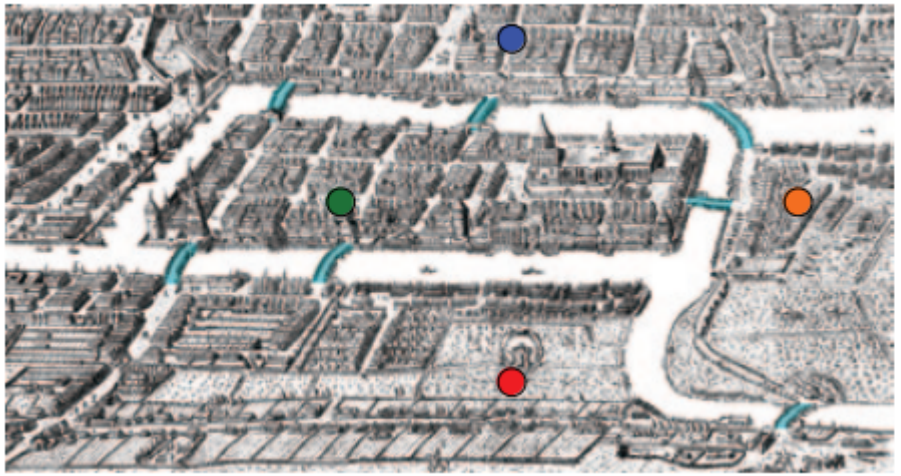
Svantaggi

- Se abbiamo N reads di lunghezza L
  - Dobbiamo eseguire 1/2 N(N-1) confronti → O(N2) problema quadratico
  - Ogni confronto è un allineamento di complessità O(L2)
  - Problematico con milioni di reads
- Quando abbiamo un overlap?
  - Lunghezza dell'overlap (es. 20pb)
  - Identità nella regione di overlap (es. 95%)
  - La scelta dipende da L e dal tasso di errore atteso (es. Illumina 2%)

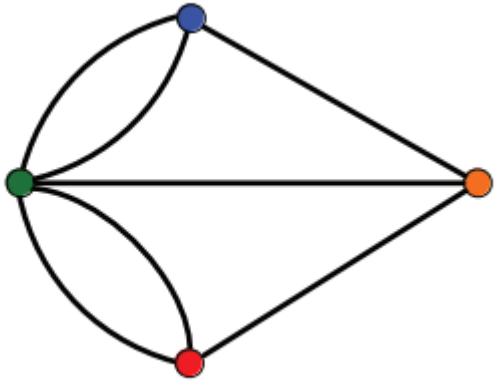
Grafi di “de Bruijn”

- **Problema dei ponti di Königsberg**: visitare ogni parte della città (4 distretti) attraversando ognuno dei 7 ponti una volta sola, e tornare al punto di partenza

a



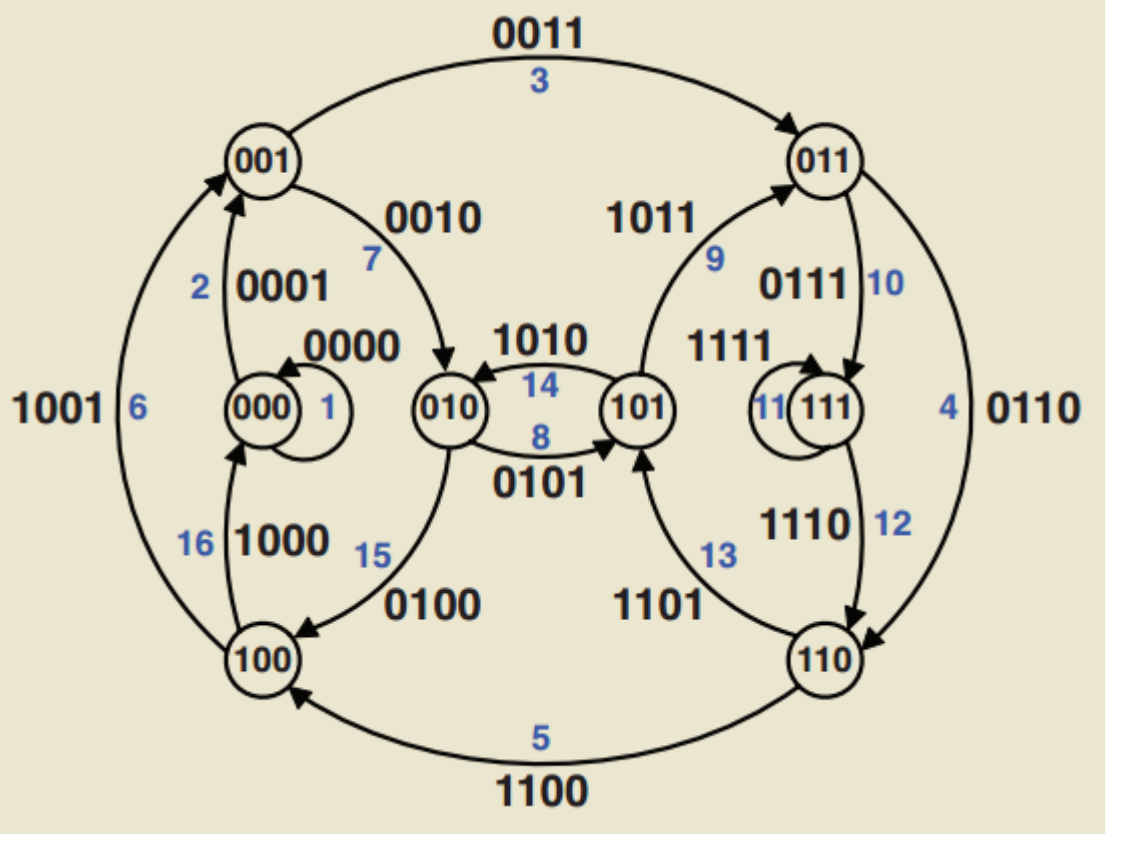
b



Risolto nel 1735 da Leonhard Euler.
Distretti → Nodi; Ponti → Collegamenti
Base teorica della teoria dei grafi

Grafi di “de Bruijn”

- Nicolaas de Bruijn (1946): dato un certo alfabeto, identificare una sequenza ciclica di lettere per cui ogni possibile parola di lunghezza *k* compaia come combinazione di caratteri una volta soltanto



Esempio di grafo di de Bruijn con k=4 e alfabeto {0,1}

Approccio tradizionale (OLC)

Piccolo genoma circolare di 10pb

5 reads:

CGTGCAA

ATGGCGT

CAATGGC

GGCGTGC

TGCAATG

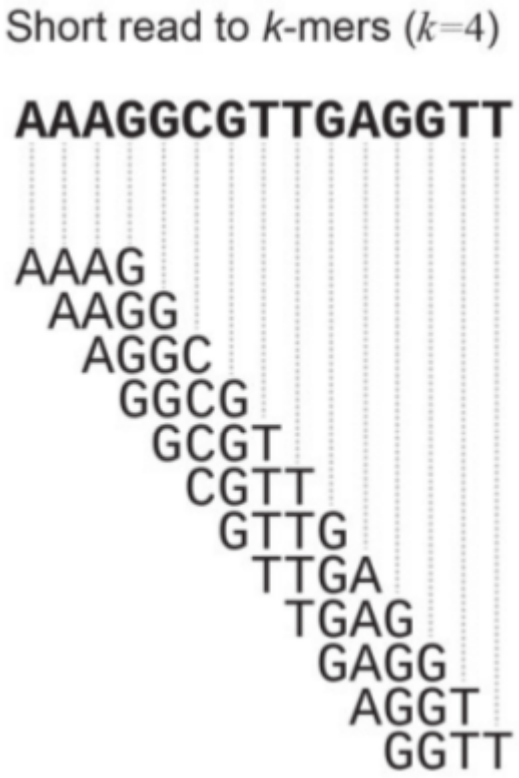
1 2 3 4 5 1

ATGGCGT → GGCGTGC → CGTGCAA → TGCAATG → CAATGGC → ATGGCGT

Ciclo Hamiltoniano

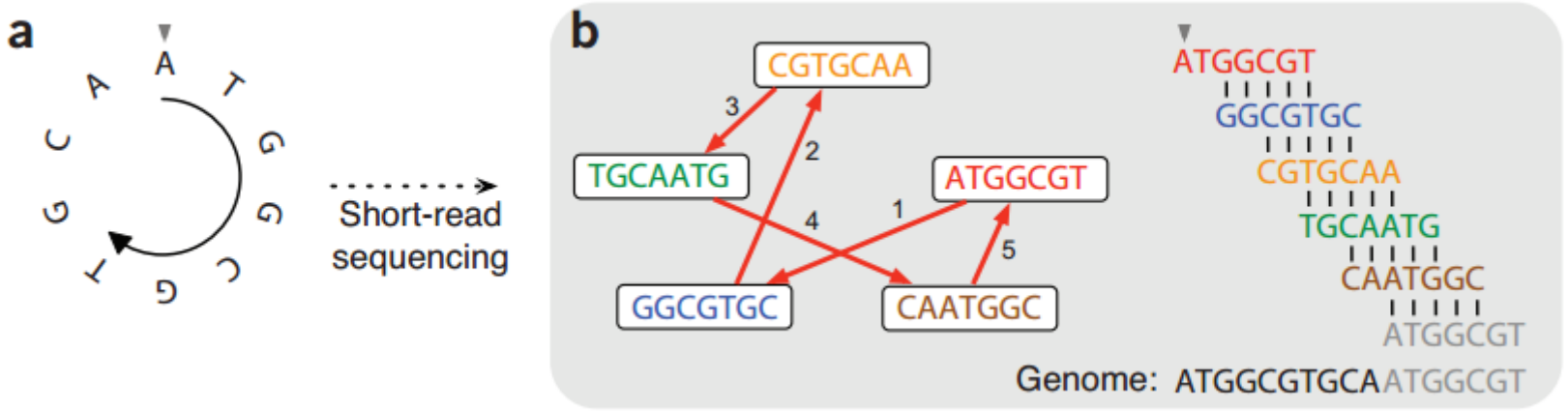
Approccio “de Bruijn”

- Utilizzo di k-mers invece che reads: divisione di ogni reads in sottosequenze di lunghezza *k*

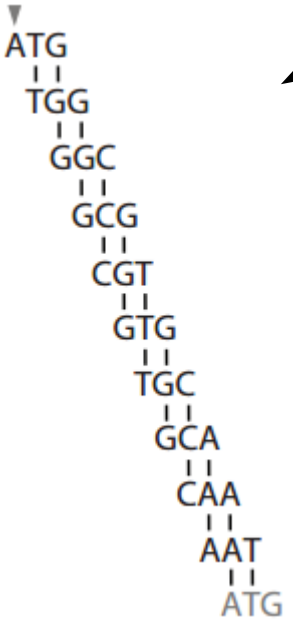


Numero di k-mers: L-k+1
L=15
k=4
K-mers = 15-4+1 = 12

Ciclo Hamiltoniano

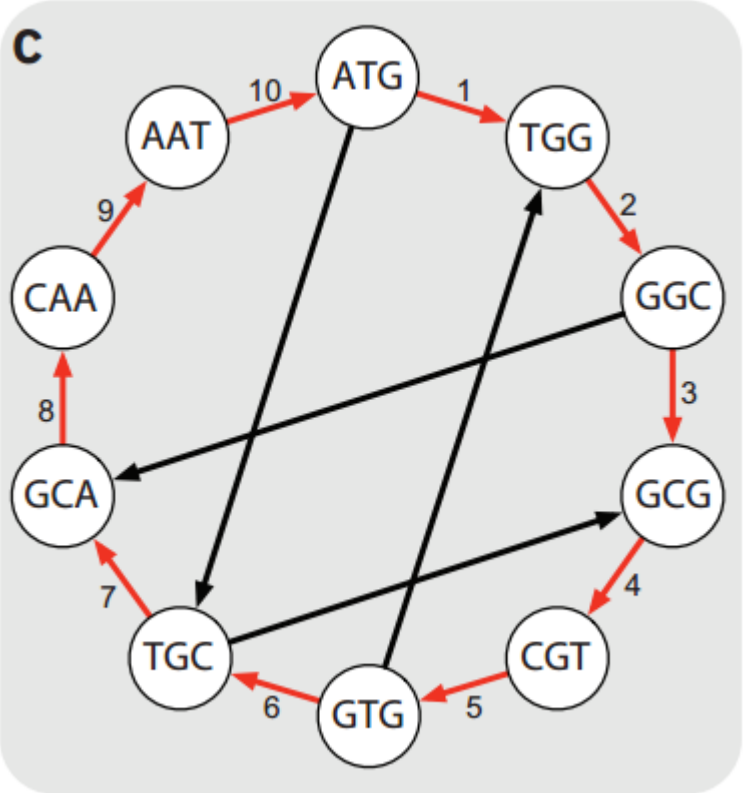


K-mers



1) Assegnare ad ogni k-mers un nodo
2)Per ogni k-mers definire il “suffisso” come la stringa formata da tutti i nucleotidi escluso il primo
3)Per ogni k-mers definire il “prefisso” come la stringa formata da tutti i nucleotidi escluso l'ultimo
4) Collega due k-mers se il suffisso del primo equivale al prefisso del secondo
5) Cercare un ciclo Hamiltoniano → identificazione del genoma

Ciclo Hamiltoniano



Visitare ogni nodo una volta sola

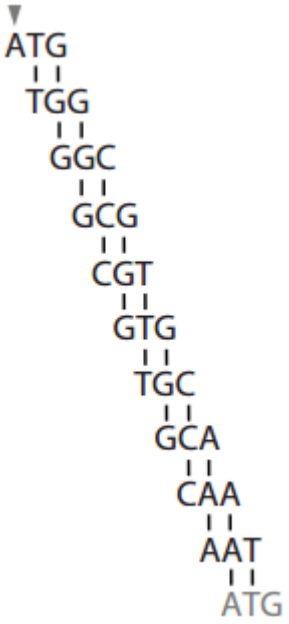
Ancora troppo inefficiente con dati NGS:
10^6 reads richedono 10^{12} allineamenti
10^9 reads richiedono 10^{18} allineamenti

Inoltre non esiste un metodo efficiente per identificare il ciclo Hamiltoniano

Ciclo Euleriano

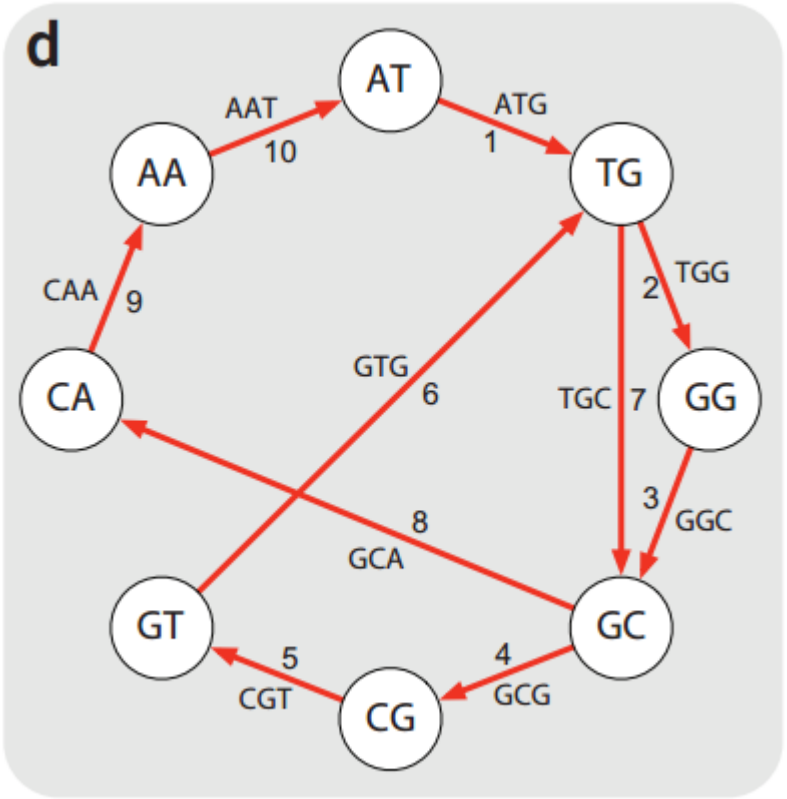
- Identificare un ciclo che visiti tutte le **connessioni** di un grafo esattamente una volta (molto più semplice)
  - Ogni k-mers viene assegnato ad una connessione
  - Prefissi e suffissi diventano invece i nodi

K-mers



1) Per ogni k-mers, assegnare un nodo ai prefissi e suffissi unici
2) Connettere i nodi x e y se un k-mer ha prefisso x e suffisso y
3) Assegnare alla connessione il k-mer corrispondente

Ciclo Euleriano



Visitare ogni collegamento una volta sola

Lo stesso approccio dei “ponti di Königsberg”

Si evita il “costoso” ciclo Hamiltoniano

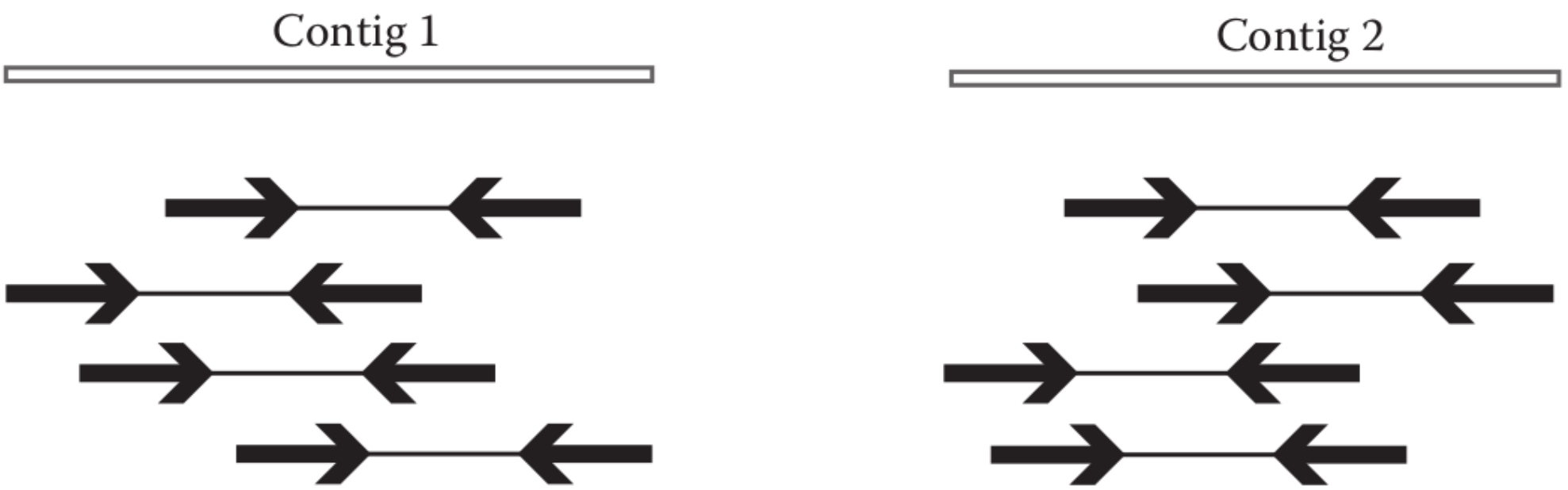
Il grafo di de Bruijn contiene un ciclo Euleriano **solo se:**
1) <u>stiamo considerando tutti i k-mer</u> presenti nel genoma da ricostruire.
2) Il grafo è <u>bilanciato</u>: stesso numero di collegamenti in entrata e in uscita da ogni nodo

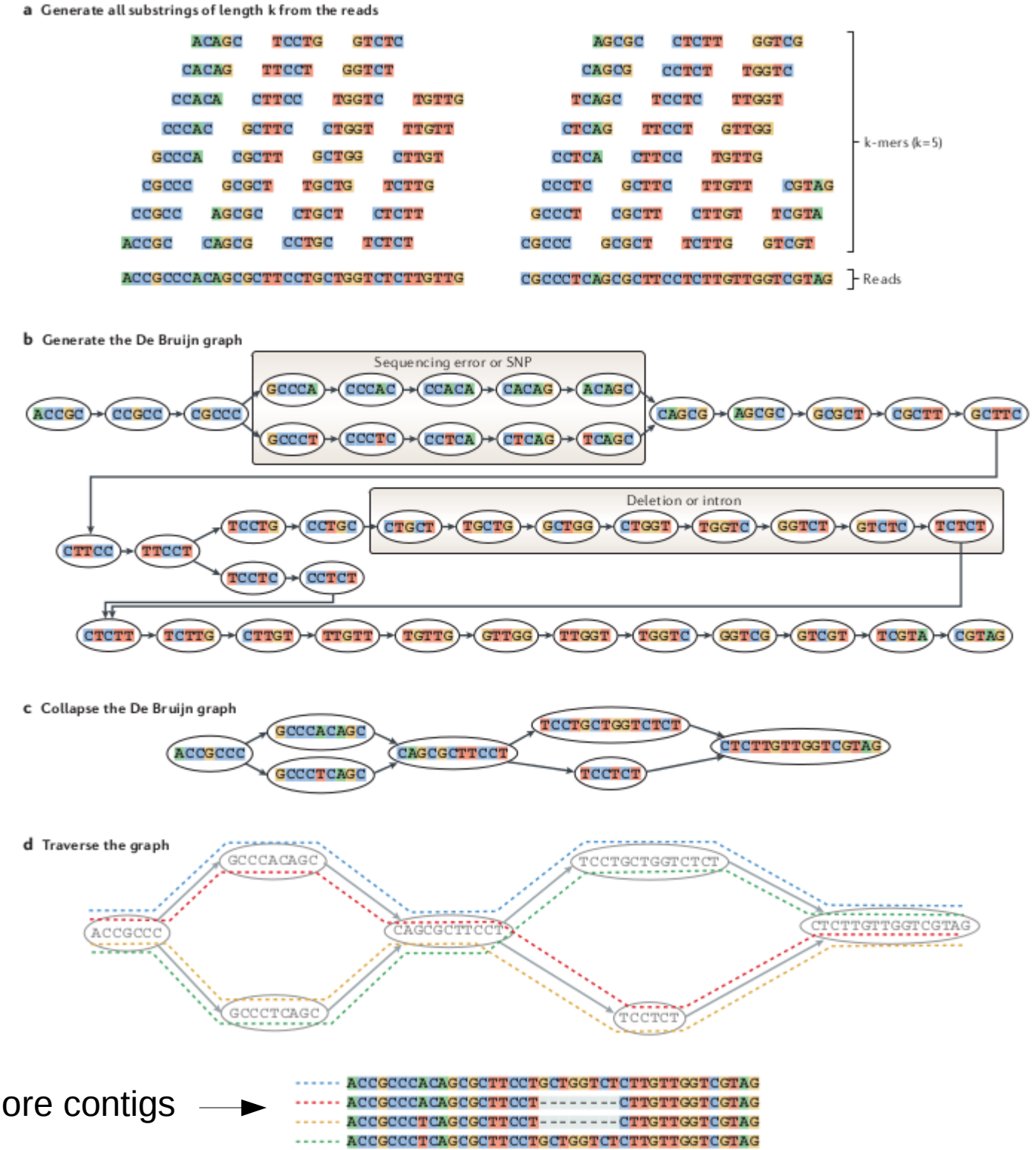
Perchè usare i k-mers e non le reads?

- Una read Illumina da 100pb può essere vista come un k-mer di lunghezza 100 ma:
  - Possiamo non avere tutti i k-mer di lunghezza 100
  - Errori di sequenziamento
  - Il coverage aiuta ma non risolve
  - Possibile violazione dell'assunzione Euleriana
- Se spezziamo le reads in k-mers più corti di 100pb
  - I k-mers ottenuti tendono a rappresentare meglio quelli di cui è composto un genoma
  - Identificazione del k ottimale (Generalmente k da 15 a 60)

Primo prodotto: “Contigs”

- Sequenza di nucleotidi assemblata in maniera non ambigua a partire dalle reads



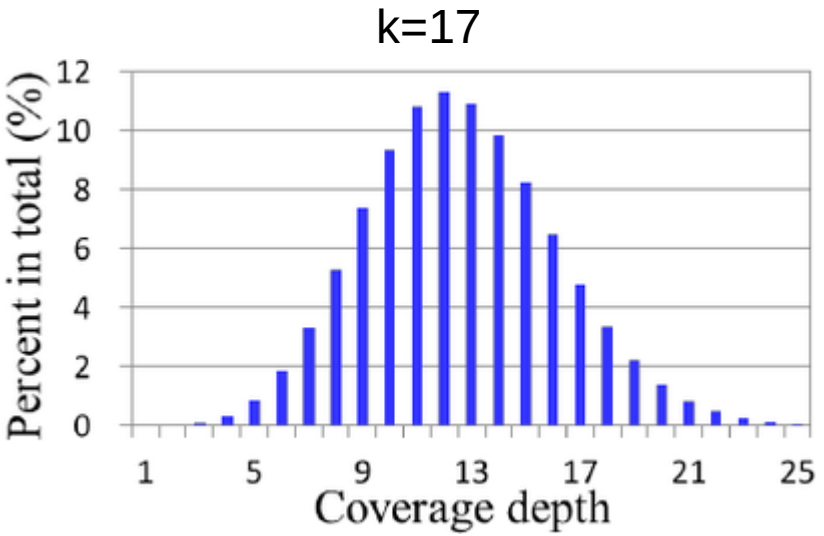

a Generate all substrings of length k from the reads
ACAGC TCCTG GTCTC
CACAG TTCCT GGTCT
CCACA CTTCC TGGTC TGTTG
CCCAC GCTTC CTGGT TTGTT
GCCCA CGCTT GCTGG CTTGT
CGCCC GCGCT TGCTG TCTTG
CCGCC AGCGC CTGCT CTCTT
ACCGC CAGCG CCTGC TCTCT
ACCGCCCACAGCGCTTCCTGCTGGTCTCTTGTTG
AGCGC CTCTT GGTCG
CAGCG CCTCT TGGTC
TCAGC TCCTC TTGGT
CTCAG TTCCT GTTGG
CCTCA CTTCC TGTTG
CCCTC GCTTC TTGTT CGTAG
GCCCT CGCTT CTTGT TCGTA
CGCCC GCGCT TCTTG GTCGT
CGCCCTCAGCGCTTCCTCTTGTTGGTCGTAG
k-mers (k=5)
Reads
b Generate the De Bruijn graph
Sequencing error or SNP
Deletion or intron
c Collapse the De Bruijn graph
d Traverse the graph
One o more contigs
ACCGCCCACAGCGCTTCCTGCTGGTCTCTTGTTGGTCGTAG
ACCGCCCACAGCGCTTCCT--------CTTGTTGGTCGTAG
ACCGCCCTCAGCGCTTCCT--------CTTGTTGGTCGTAG
ACCGCCCTCAGCGCTTCCTGCTGGTCTCTTGTTGGTCGTAG

Fattori da considerare

- Errori di sequenziamento
  - Impediscono l'estensione dei contig e scaffold
  - Influenzano il processo in modo differenziale (Illumina 1-2%, Pacbio/Nanopore 10-20%)
  - Devono essere corretti prima dell'assembly
- Coverage non omogeneo
  - Può causare interruzioni nella ricostruzione della seqeunza originale di DNA → GAPS
- Elementi ripetuti
  - Le reads sono generalmente più corte delle regioni ripetute
  - Meglio utilizzare reads lunghe
- Complessità algoritmica
  - Un assembly di un genoma di medio/grandi dimensioni spesso richiede > 100Gb di RAM e qualche settimana di tempo di calcolo
  - Genomi batterici, invece, alcuni minuti

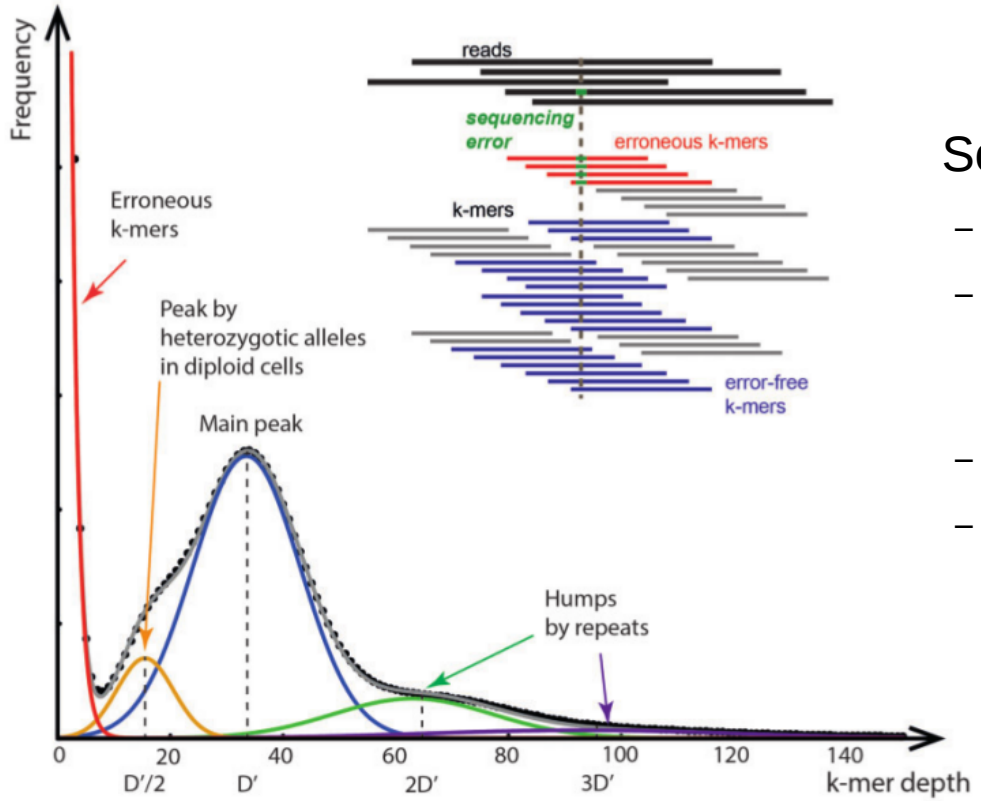
Errori di sequenziamento

- In condizioni ideali, le reads sono distribuite in maniera uniforme lungo il genoma:
  - L'istogramma della profondità di sequenziamento dei k-mer tende a formare una distribuzione normale
  - Si può sfruttare questa attesa per identificare gli errori



Errori di sequenziamento

- Se sono presenti errori di sequenziamento:
  - Creazione artificiale di k-mer a bassa frequenza e a bassa profondità di sequenziamento
  - Presenza di una curva esponenziale decrescente (in rosso)

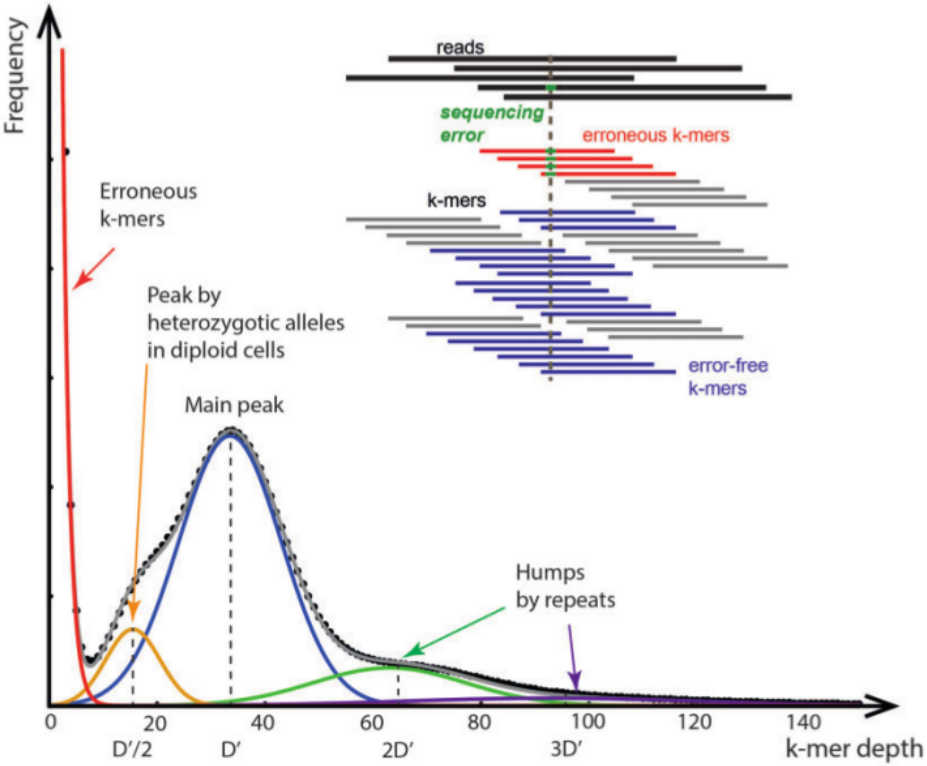


Soluzioni:

- Eliminare i k-mers al di sotto di una soglia di D'
- Correzione delle reads contenenti i k-mers con errori tramite allineamento multiplo con le altre reads
-
- Attenzione: la soglia è arbitraria e alcuni k-mers con errori di sequenziamento potrebbero rimanere!

Eterozigosi e regioni ripetute

- K-mers a bassa frequenza dovuti a:
  - Eterozigosi
    - Profilo di profondità di sequenziamento simile a quello di k-mers contenenti errori
    - Il loro picco è centrato su *D'/n*, dove *n* è la ploidia (*D'/2* per organismi diploidi)
  - Regioni ripetute
    - Alta frequenza di speficici k-mers
    - Difficile identificare errori in queste regioni



K-mers e dimensione del genoma

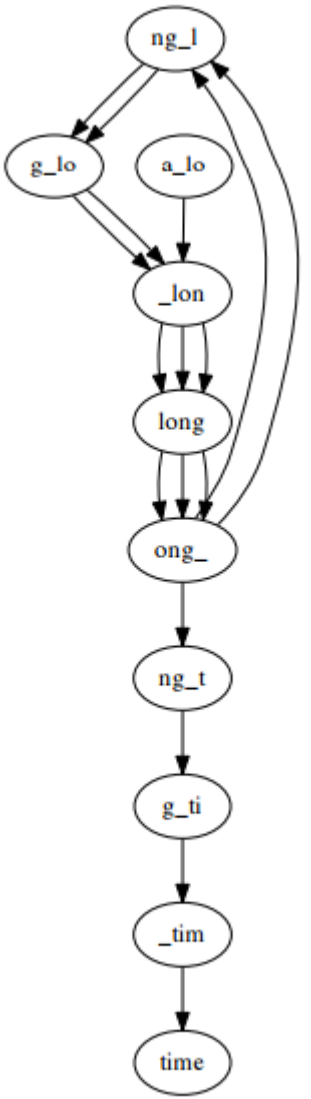
- La distribuzione di frequenza dei k-mer può essere usata per stimare la **dimensione del genoma** (G) da ricostruire

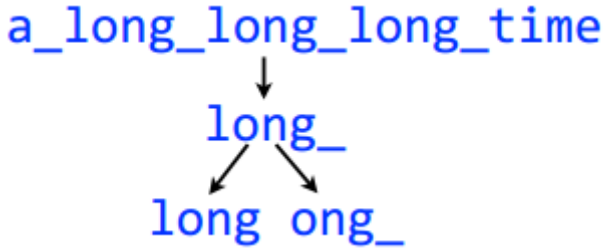
$$D = \frac{D' \times L}{L - k + 1} \qquad G = \frac{N_{base}}{D} = \frac{N_{read} \times (L - k + 1)}{D'}$$

- Dove:
  - D, profondità di sequenziamento del genoma
  - G, dimensione del genoma in paia di basi
  - L, lunghezza media delle reads
  - K, lunghezza del k-mer
  - D', profondità di sequenziamento corrispondente al picco della distribuzione di frequenza
  - (L-k+1), numero di k-mer in una read
  - Nbase, numero di basi sequenziate
  - Nread, numero di reads sequenziate

Regioni ripetute

- Identificare un ciclo Euleriano completo non è spesso possibile a causa della presenza di regioni ripetute:
  - STR, LINE, SINE, Centromeri, Telomeri, ecc..
  - Generazione di percorsi ambigui nel grafo

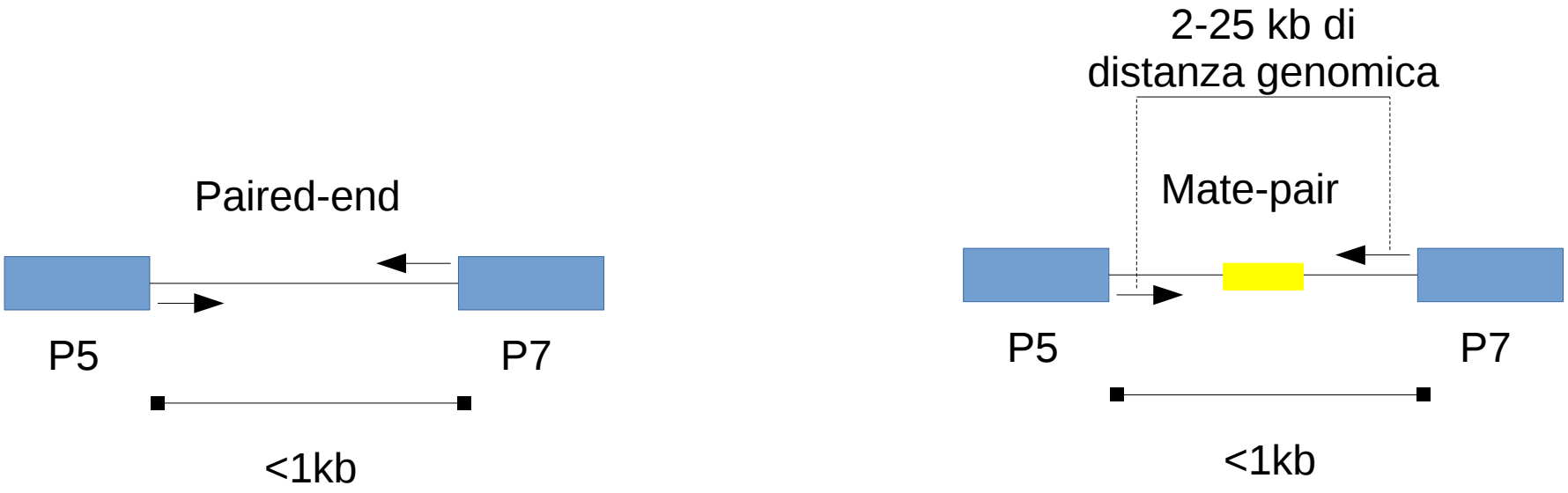




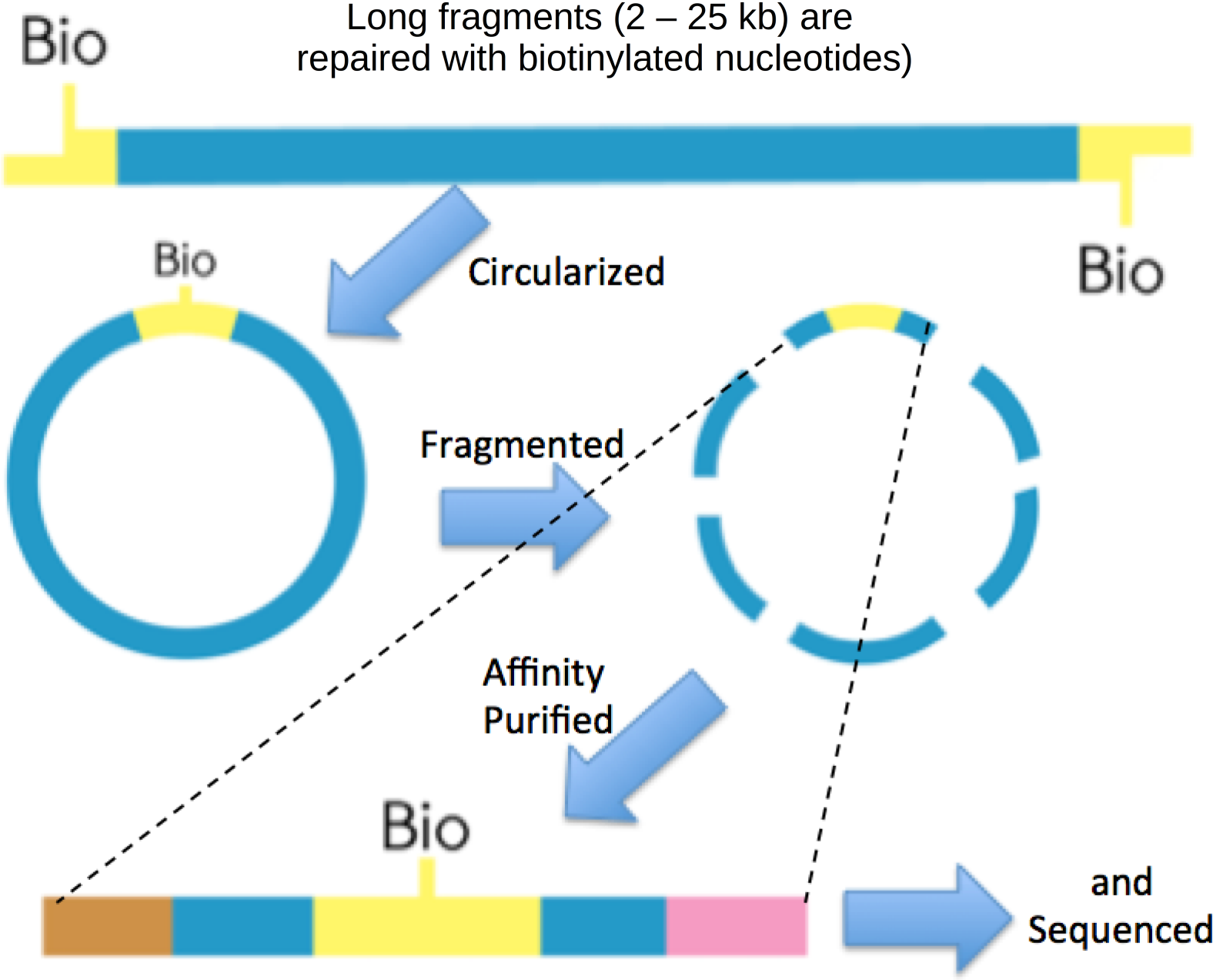
k=5

Risoluzione delle regioni ripetute

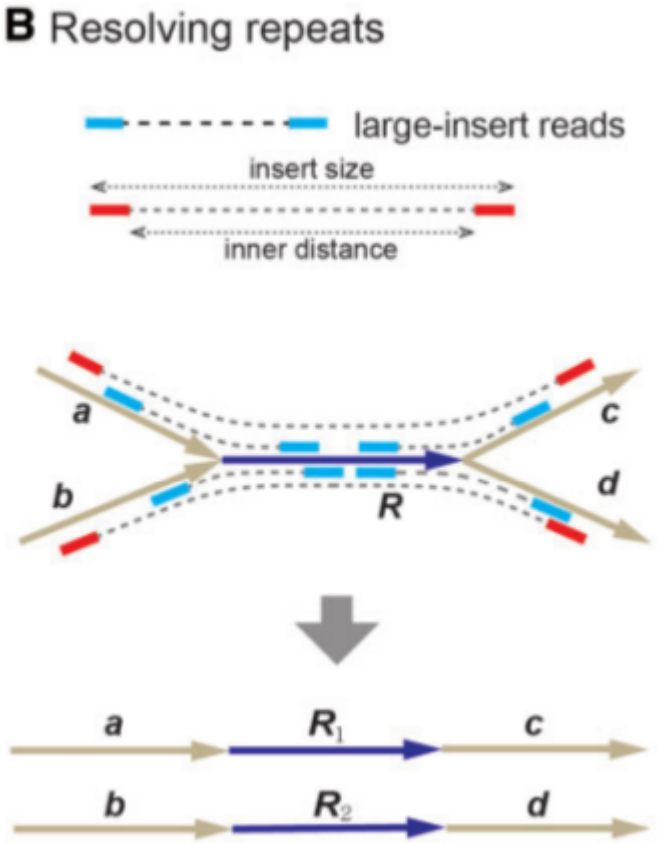
- Necessario l'utilizzo di paired-end libraries con diverse dimensioni dell'inserto o mate-pair:
  - Paired-end: dimensione inserto da 300 a 800 pb
  - Mate-pair: stessa dimensione dell'inserto ma creato per unire loci che distano dalle 2 alle 25 kb



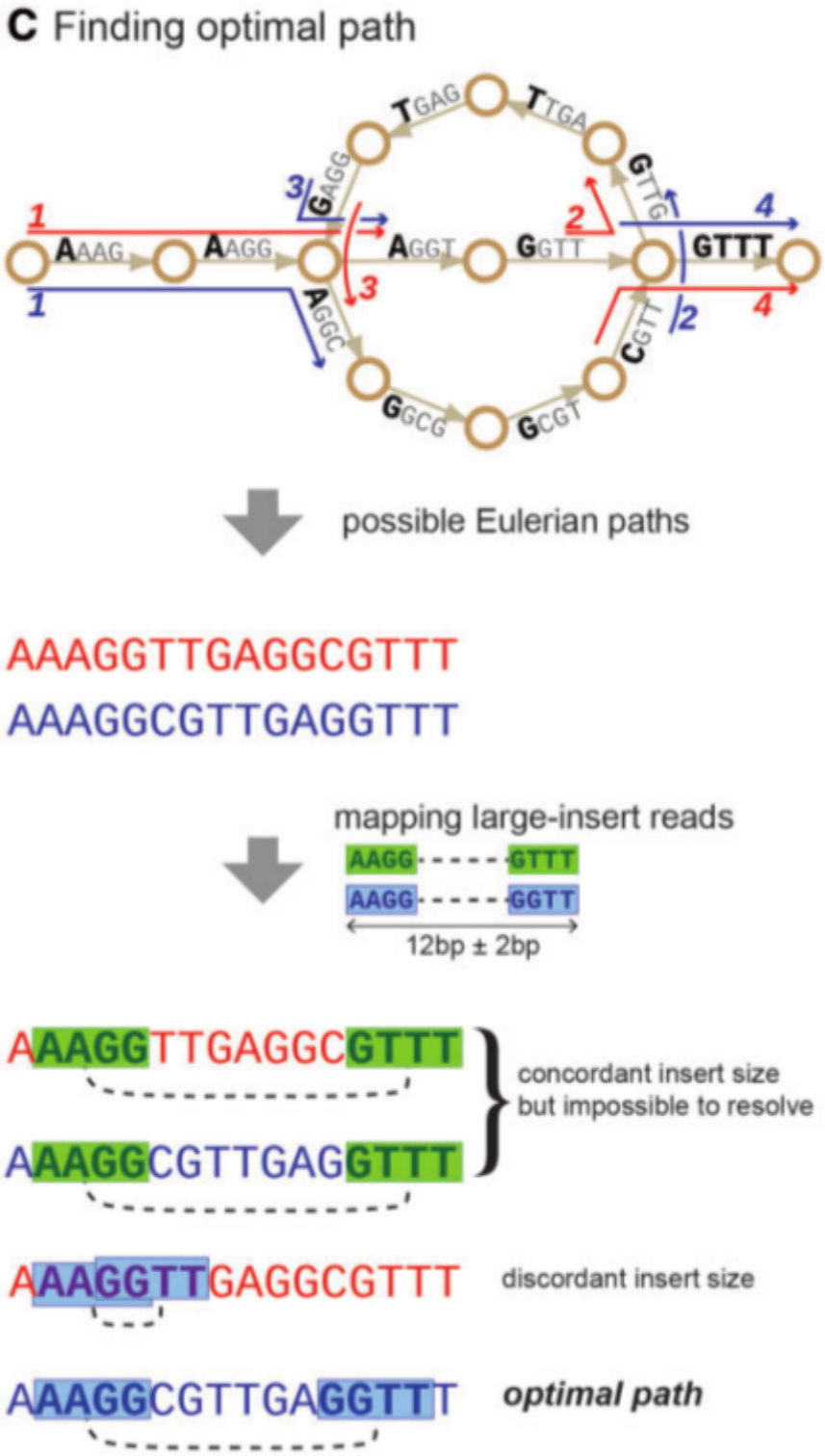
Library mate-pair



Risoluzione regioni ripetute

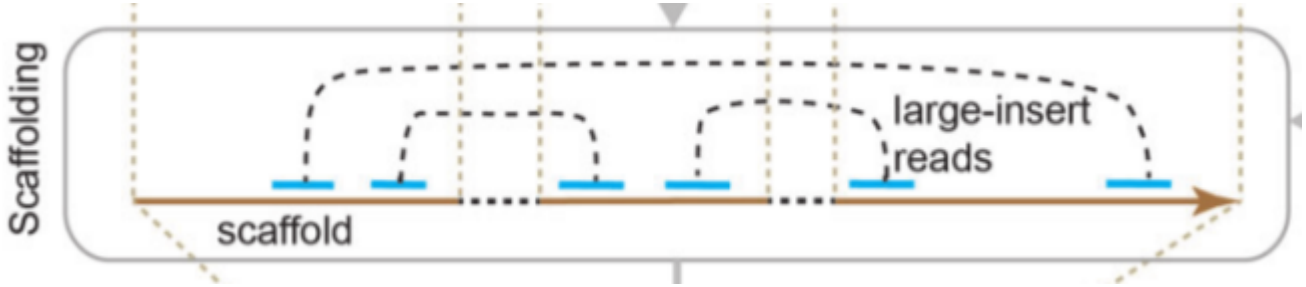


Consigliabile usare il “short-first approach”

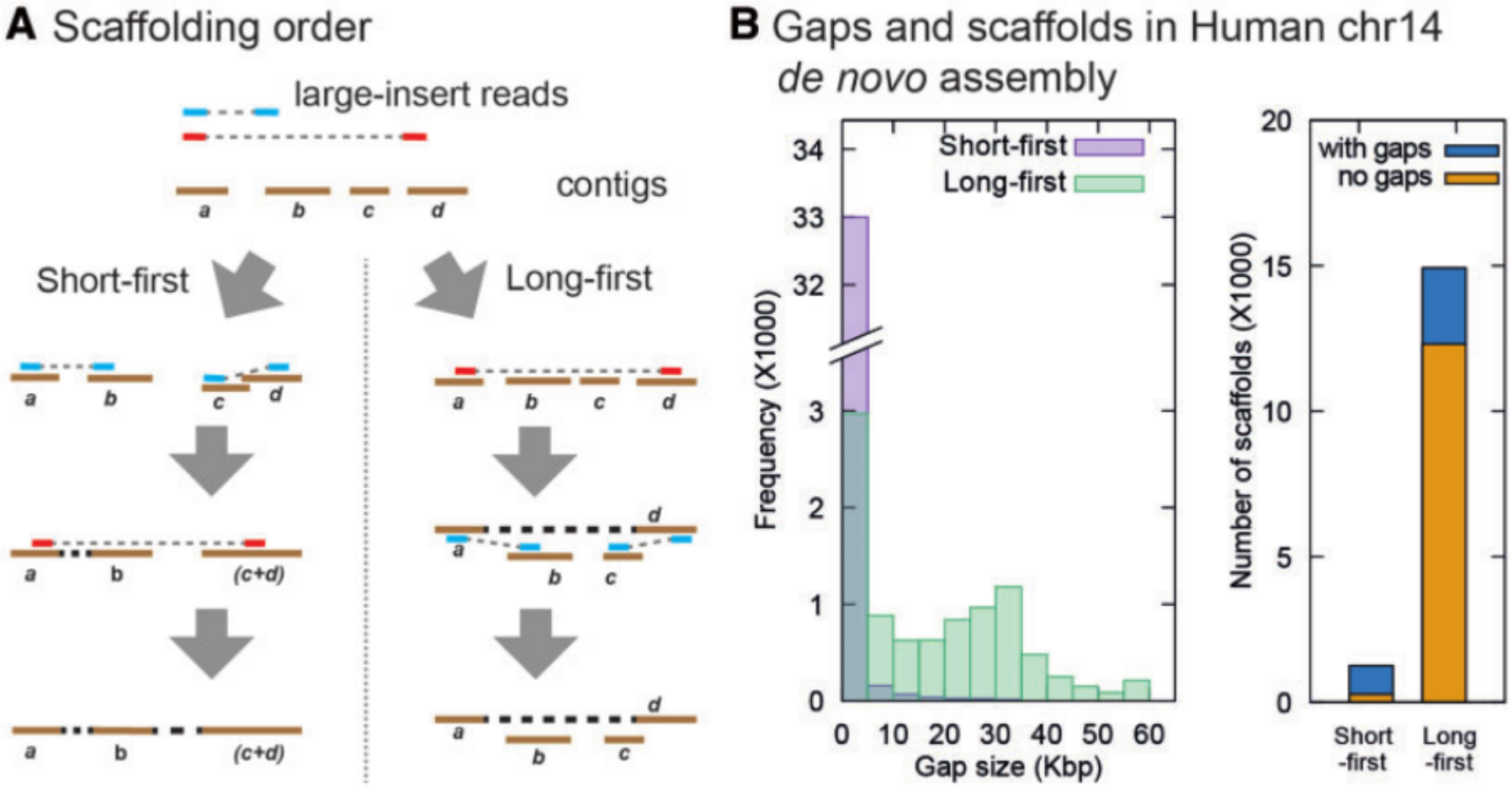


Scaffolding

- Utilizzo di paired-end o mate-pair library per orientare ed unire coppie di contig in sequenze di lunghezza maggiore (dette scaffold)
  - Due possibili approcci:
    - "Short first", si utilizzano prima le library con inserti più corti e successivamente quelle con inserti più lunghi (migliore)
    - "Long first", si utilizzano prima le library con inserti più lunghi e successivamente quelle con inserti più corti (peggiore)



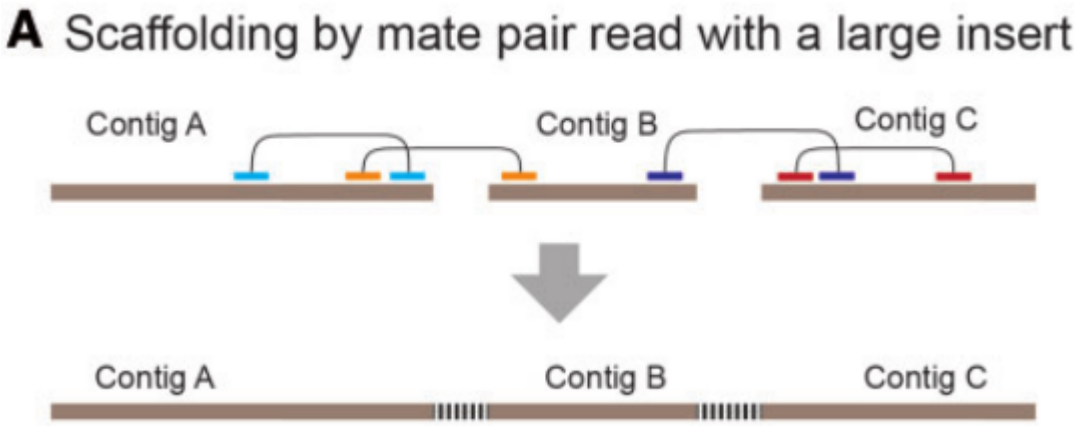
Scaffolding

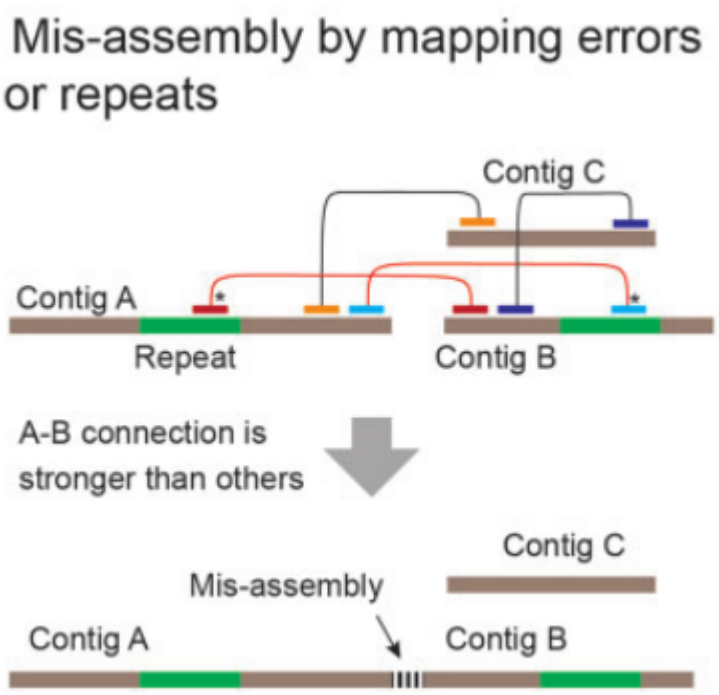


I'approccio “long first” può portare alla generazione di largi gaps e contig non ancorati a scaffold → Più efficiente l'approccio “short first”

Regioni ripetute

- Se due frammenti di un “pair” mappano in regioni ripetute possono generare un falso link tra due contig distanti

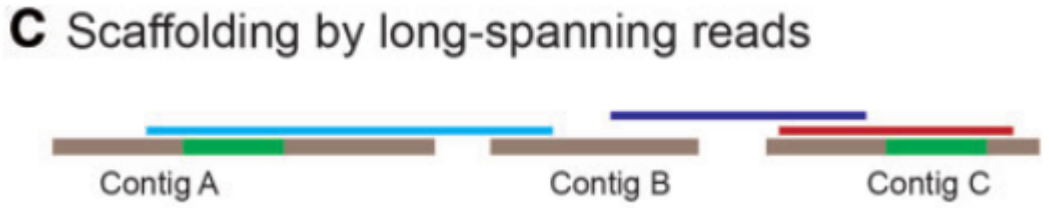




* errori di mapping

Regioni ripetute

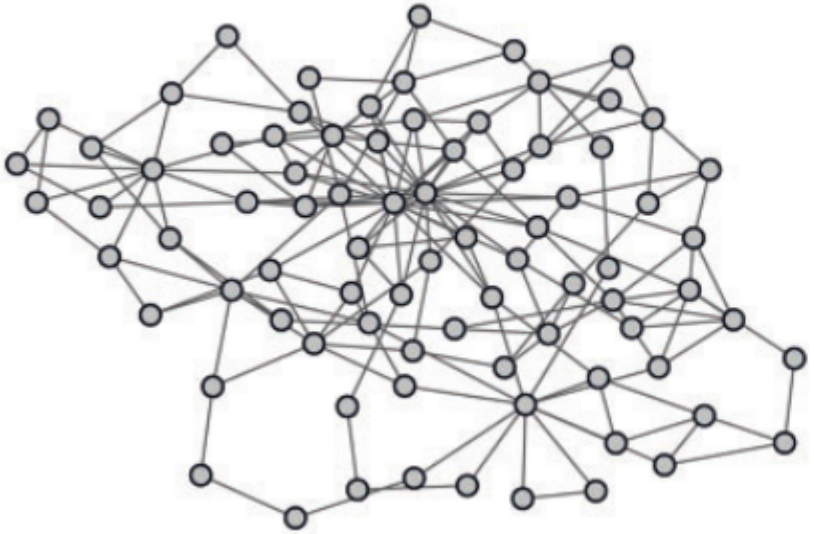
- Il problema può essere parzialmente risolto attraverso l'uso di reads lunghe (PacBio/Nanopore)



Regioni ripetute

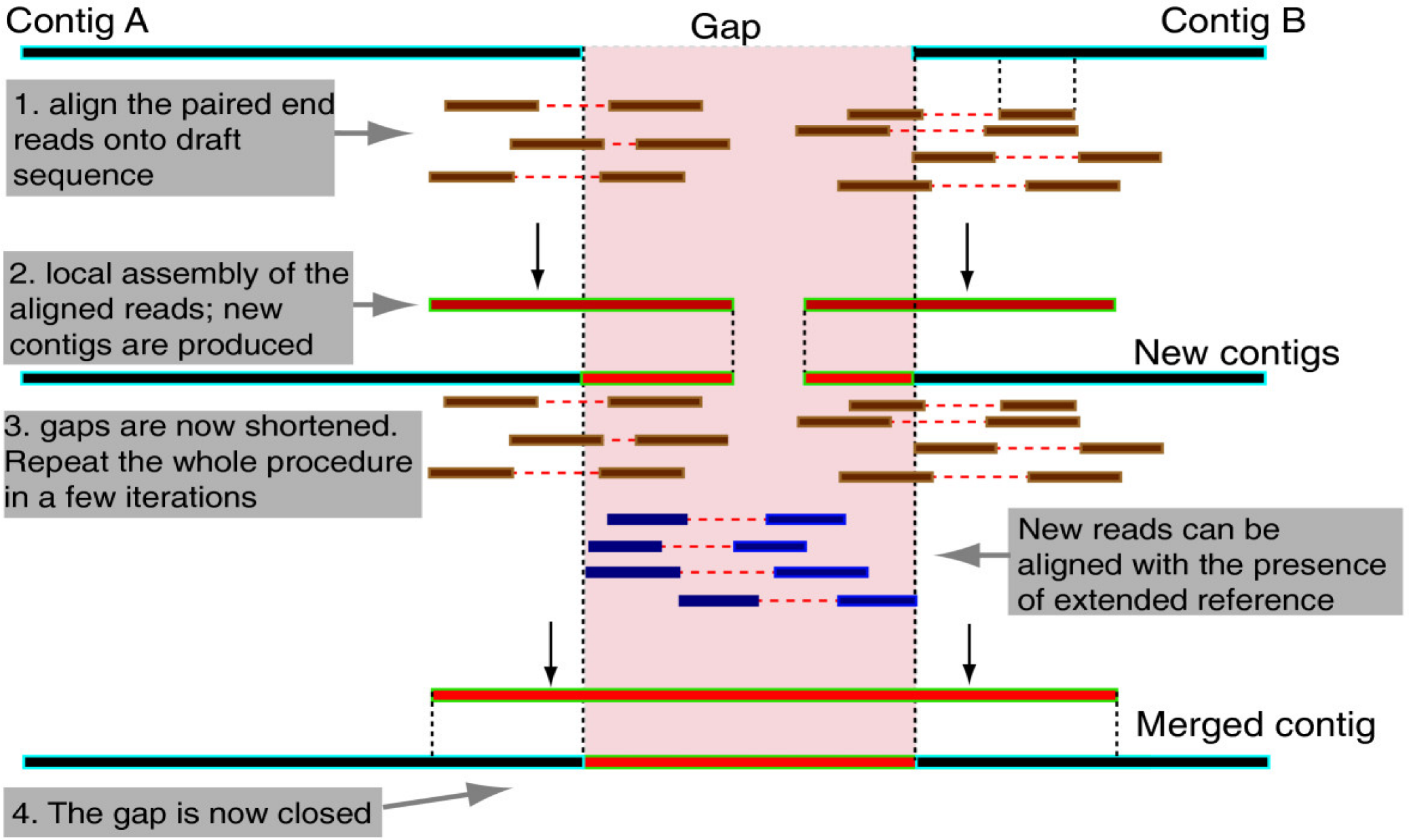
- Anche l'uso di reads lunghe risolve “parzialmente il problema delle regioni ripetute
  - Queste regioni tendono a formare dei motivi “a stella” nel grafo
  - Difficile identificare il percorso euleriano ottimale
  - Spesso queste regioni vengono escluse e sostituite da gap nell'assembly finale

D Assembly gaph by long-spanning reads



E Assembly graph by mate-pair reads

Rifinitura: Chiusura dei GAP

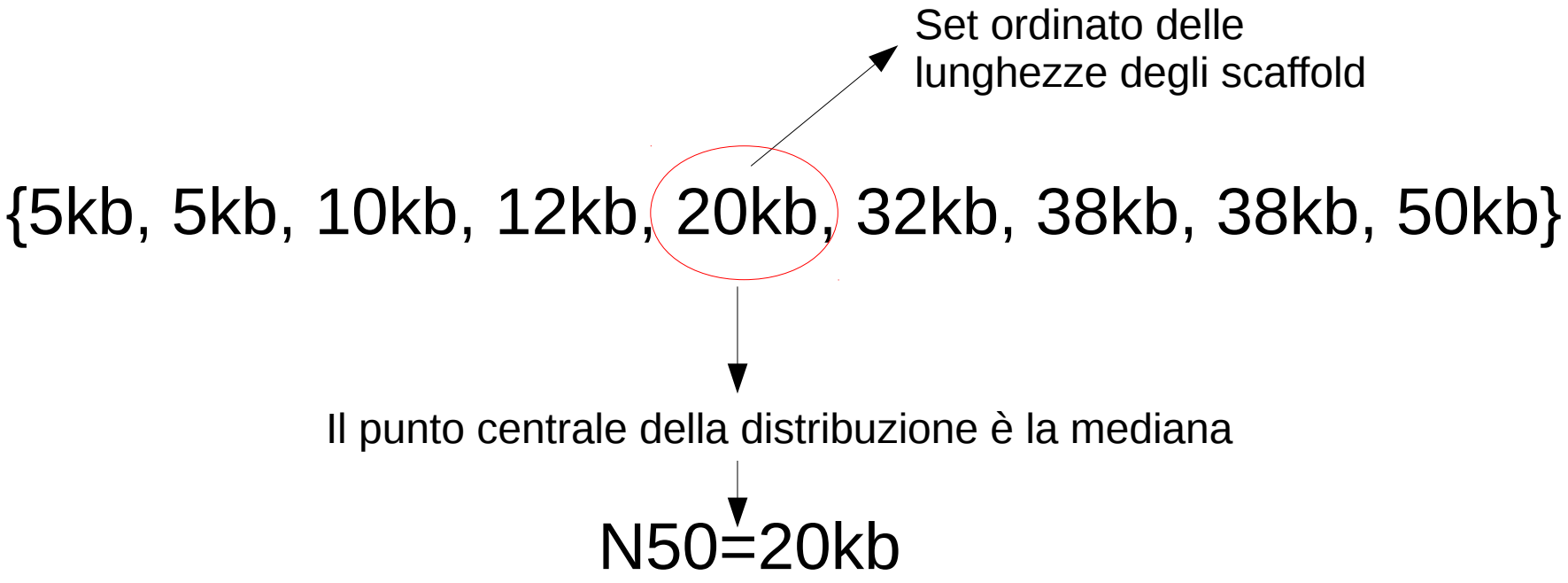


Strategia di sequenziamento ottimale

- Selezione di individui omozigoti (se possibile)
- Paired-end library, piccolo inserto (300-800pb), sequenziata ad alto coverage (40-80x)
- Mate-pair library, medio inserto (2-10kb), sequenziata a medio/basso coverage (1-5x)
- Mate-pair library, lungo inserto (10-40Kb), sequenziata a medio/basso coverage (1-5x)
- PacBio/Nanopore library, lunghezza 5-30kb, basso coverage (1x)

Qualità dell'assemblaggio

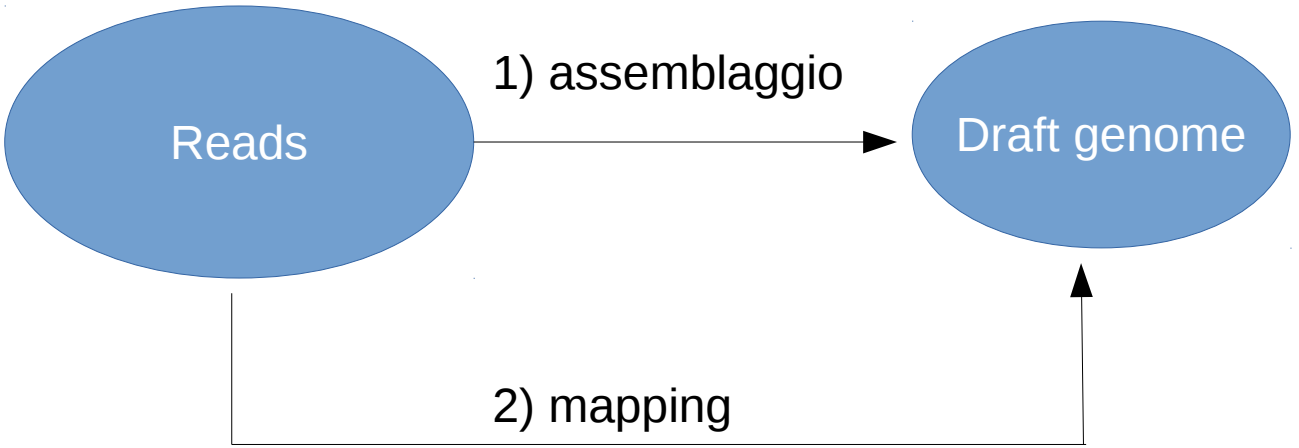
- Tre parametri: contiguità, completezza e accuratezza
  - Contiguità → Numero di contigs (meno sono meglio è)
  - Contiguità → **N50**: lunghezza mediana del set di contig (o scaffold) ricostruiti nella fase di assembly



ES: IL 50% del genoma è composto da scaffold lunghi >=20kb

Qualità dell'assemblaggio

- Completezza → proporzione delle reads utilizzate per assemblare il genoma che mappa sul genoma assemblato



Qualità dell'assemblaggio

- Accuratezza → confronto dell'assembly con un riferimento:
  - Se è disponibile un genoma di riferimento:
    - Accuratezza nella chiamata delle basi
    - Posizione e orientamento dei contig/scaffold
  - Se NON è disponibile un genoma di riferimento:
    - Riallineamento delle reads all'assembly e controllo dell'uniformità del coverage
    - Sequenziamento di RNA (RNAseq) e mapping sull'assembly

Performance computazionali

Table 3. Comparison of computational costs of short read assemblers

Assemblers	Algorithm type	Genome[a]	Sequencing depth	Peak memory[b] (in GB)	Relative speed[c]	Relative computational cost[d]	References[e]
ALLPATHS-LG	Eulerian	Human	~100×	>512	1	597	[29]
SOAPdenovo2	Hamiltonian	Human	~50×	35	10.8	4	[25]
ABySS	Hamiltonian	Human	~35×	<16	11.9	<8	[41]
SGA[f]	Hamiltonian	Human	~100×	56	65	1	[32]
JR-Assembler	Greedy like	Human	~130×	418	1.8	279	[33]
MaSuRCA	Eulerian+OLC	Loblolly Pine	~80×	800	1.4	644	[37, 38]

[a]Estimated genome size: Human, 3.1 Gb; Loblolly Pine, 22 Gb; bird (parrot), 1.2 Gb.
[b]RAM memory of ALLPATHS-LG is a minimum requirement, while the others are peak.
[c]Relative speed is estimated by (genome size)/(CPU time), and normalized by ALLPATHS-LG, where the CPU time is approximately (running time)/(number of threads). Note that the number of threads of Intel processors is twice the number of cores, whereas the number of threads of AMD processors is equal to the number of cores.
[d]Relative computational cost, (peak memory)/(relative speed), was normalized to SGA.
[e]ALLPATHS-LG took 3.5 weeks with Dell R815, 48 processors [29]; SOAPdenovo2 took 81 h with eight Quad-core AMD (2.3 GHz) [25]; SGA took 24 h with single Hexa-core XEON X5650 (2.66 GHz) [32]; JR-Assembler took 1.5 CPU weeks with four Octa-core Xeon E7-4820 (2 GHz) [33]; MaSuRCA took about 3 months with 64-core IA64 computer [37, 38].
[f]The relative speed of SGA was estimated with the time to contig assembly.

Alcuni draft

Global statistics

Total sequence length	2,135,083,061
Total assembly gap length	30,732,878
Gaps between scaffolds	331
Number of scaffolds	598
Scaffold N50	10,525,104
Scaffold L50	63
Number of contigs	2,789
Contig N50	1,279,870
Contig L50	506
Total number of chromosomes and plasmids	12

Global statistics

Total sequence length	2,004,063,690
Total assembly gap length	22,596,073
Gaps between scaffolds	0
Number of scaffolds	32,573
Scaffold N50	4,188,677
Scaffold L50	132
Number of contigs	105,348
Contig N50	69,131
Contig L50	7,850
Total number of chromosomes and plasmids	1