BIOINFORMATICA: *2008/2009*

- Seminario:

  Individuazione di famiglie di repeat
  in sequenze genomiche incomplete

- Autori:
  - Giuseppe Zichittella
  - Claudio Porta

Indice degli argomenti

- I repeat:
  - Definizione
  - L'importanza della ricerca dei repeat
- Obiettivi dell'articolo
- Metodi per l'analisi di repeat
  - Il grafo *de Bruijn*
  - Il grafo sparso *de Bruijn*
  - Il grafo di sequenza

Indice degli argomenti

I repeat (1)

- Un repeat è una sequenza di due o più basi ripetuta all'interno di una sequenza genomica

...GATTGCACTATTGACCG...

- La sua lunghezza può variare da una singola coppia a diverse migliaia di basi.

I repeat (2)

- Si classificano in due categorie:
  - Interspersed repeat: distribuite in maniera apparentemente casuale lungo il genoma

    ...ATCATTCGGTACACCTAGATT...
  - Tandem repeat: le cui unità ripetute sono localizzate una a fianco all'altra

    Ad esempio la sequenza:

    ATTCGATTCGATTCGATTCG

    è una ripetizione in tandem di ATTCG

    (ripetuta quattro volte)

L'importanza della ricerca dei repeat (1)

- Per quanto non costituiscano sempre parte attiva, i repeat possono, tuttavia, essere causa di modiche del genoma sia nei batteri sia negli organismi eucarioti:
  - *Mutazioni puntiformi*: nel genoma dei batteri attraverso l'inserimento di singole basi.
  - *Mutazioni cromosomiche*: negli eucarioti attraverso la duplicazione o la traslocazione di intere sequenze genomiche

L'importanza della ricerca dei repeat (2)

- Queste mutazioni possono essere causa di diversi disordini genetici quali:
  - *Beta-Talassemia*: causata generalmente dalla sostituzione, cancellazione, o inserimento di una singola base del gene beta-globina
  - *Distrofia muscolare*: causata dalla mutazione di un gene all'interno del cromosoma 21

Obiettivi

- Introduzione di algoritmi per la ricerca e caratterizzazione di repeat nel caso in cui il genoma, oggetto di studio, sia ancora non sia del tutto noto.

- L’applicabilità di questi metodi verrà mostrata su dataset genomici simulati e reali.

Introduzione agli algoritmi

- Gli algoritmi per la ricerca di famiglie di repeat si basano sulla costruzione la modifica e l'analisi di particolari grafi:
  - Costruzione di un grafo di sequenza

    (una variante del grafo di "de Brujin")
  - Analisi della sua struttura

Il grafo *de Bruijn*

- Un grafo *d-dimensionale* di *de Bruijn*

$$G=(V,A)$$

Sull'alfabeto Σ è definito come segue:

$$V=\Sigma^d$$

$$A=\{(u,v):u,v\in V\wedge u_{i+1}=v_i,\forall i,1\leq i<d\}$$

dove u_i è l'i-esimo carattere della stringa u

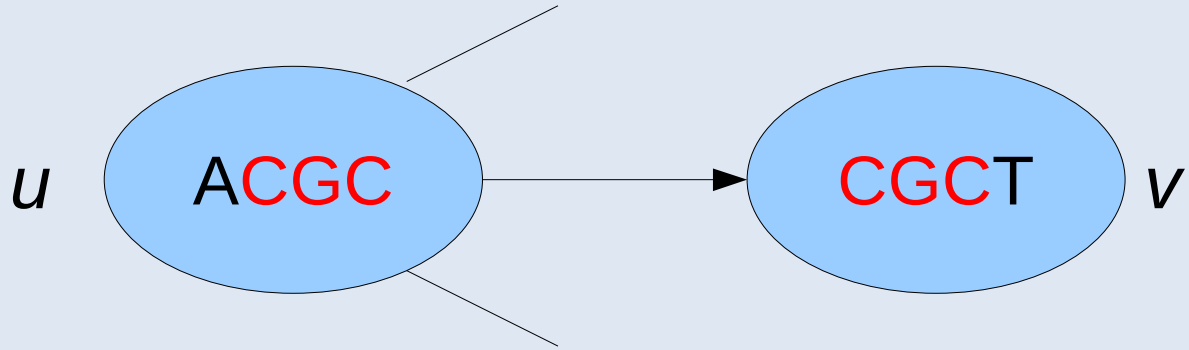
Il grafo *de Bruijn*: esempio

Per esempio siano:

$\Sigma = \{A, C, G, T\}$

$d = 4$

allora:



Stringhe, sullo stesso alfabeto di lunghezza almeno d-caratteri costituiscono cammini d-dimensionali sul grafo de Bruij*n.*

Considerazioni sul grafo

- Inefficiente in spazio

  (se d = 1 tutti i vertici sono connessi con tutti formando |Σ|^2 archi)

- Ovviamo al problema introducendo un'implementazione più efficiente del grafo, così da ridurre il numero di archi:

Introduzione al sottografo sparso di *de Bruijn*

- Sia *s* una stringa definita su Σ.
- Lo *spettro* d-dimensionale di *s*

spectrum(s, d)

è l'unione di tutte le *sottostringhe* di *s* di lunghezza *d*

Es: Σ={A,T,C,G} ; *d = 3*

s = ATCGA

spectrum(s,d) = {ATC,TCG, CGA}

Il sottografo sparso di *de Bruijn*

- Dato un insieme *S* di stringhe

  il sottografo sparso di *de Bruijn* è definito da

$$G_S = \{V_S, A_S\}$$

dove:

$$V_S = spectrum(S, d)$$

$$A_S = \{(u, v) : u, v \in V_S \wedge u_1 \ldots u_d v_d \in spectrum(S, d+1)\}$$

Il sottografo sparso di *de Bruijn: esempio*

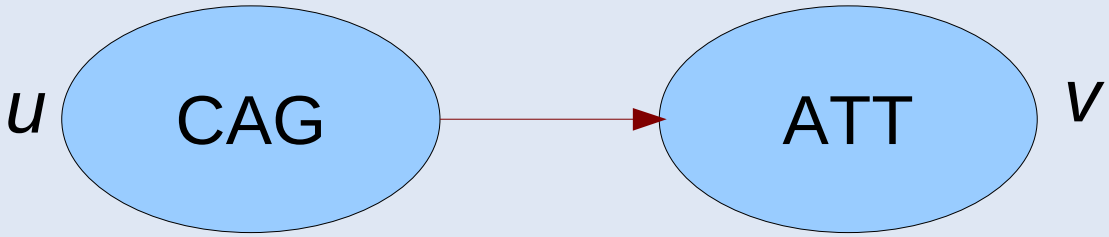
- Siano:

  $\Sigma = \{A, C, G, T\}$

  $d = 3$

  s1 = CAGTCA ; s2 = ATTGGA due stringhe in *S*

  presi $u, v \in V_S$



$(u, v) \in A_S$ in quanto $CAGT \in spectrum(S, d+1)$

Efficienza del sottografo sparso di *de Bruijn*

- Scala bene con la dimensione dell'insieme *S*.
  - (relativamente al numero di sequenze genera un numero di tuple limitato)
- Il numero dei nodi cresce linearmente con la dimensione delle sequenze in input

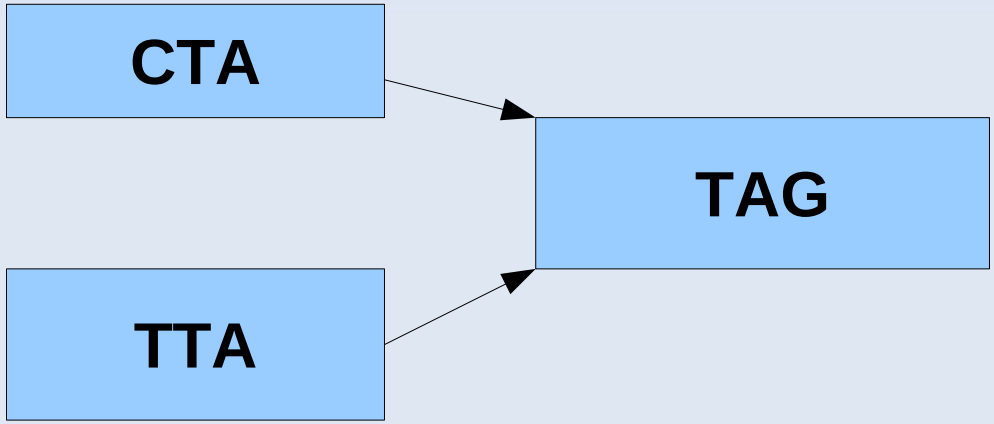
Il grafo di sequenza (1)

- Versione compatta del grafo di *de Bruijn:*
  - *lunghi cammini "non ramificati" sono uniti in singoli nodi in maniera da migliorarne la complessità.*
- Gli elementi dell'insieme V_S sono detti *d-tuple*
- Sull'alfabeto Σ ogni *d-tupla* è rappresentata al massimo da un vertice
- La lunghezza di ogni vertice è limitata da

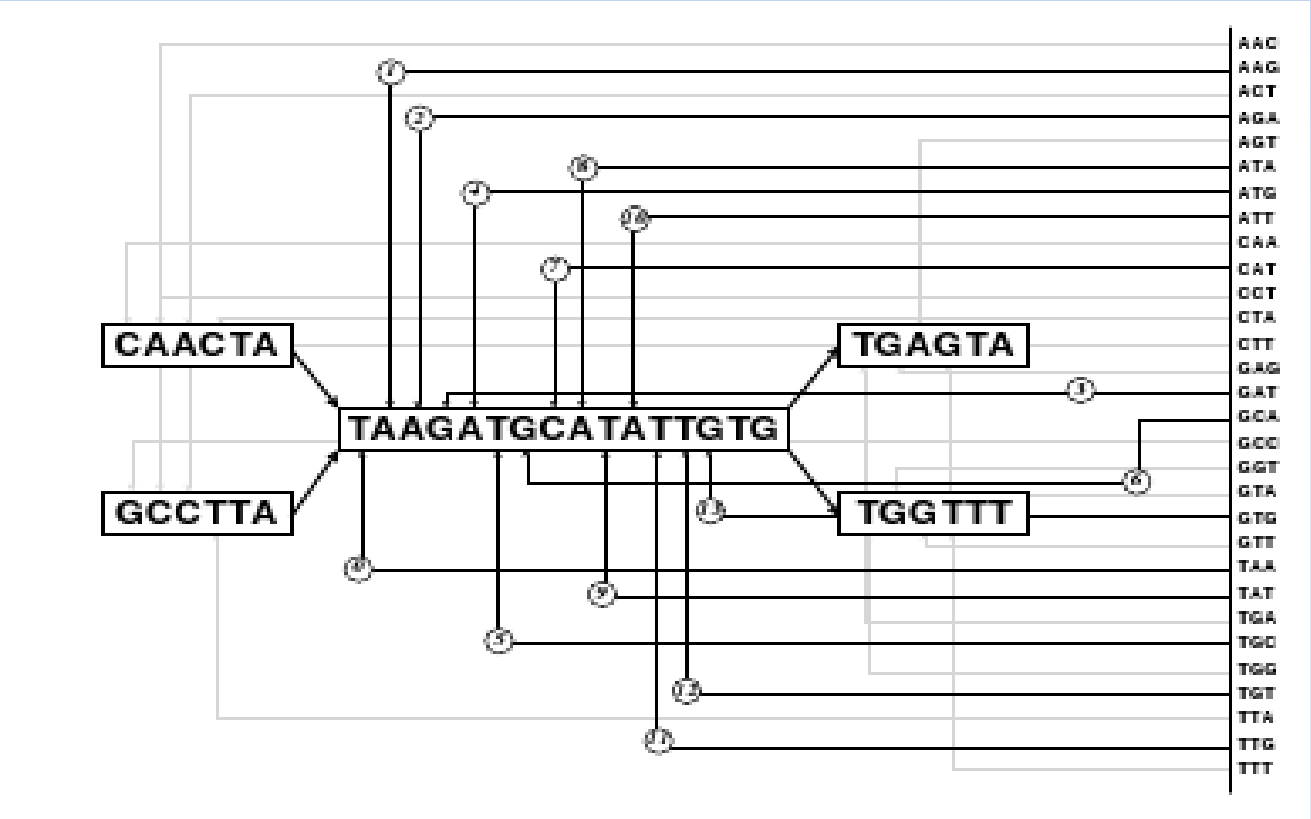
$$|\Sigma|^d + d - 1$$

Il grafo di sequenza (2)

- Due nodi si dicono *vicini* se esiste un arco tale che il suffisso di *u* di lunghezza *d-1* matcha con il prefisso di *v*.

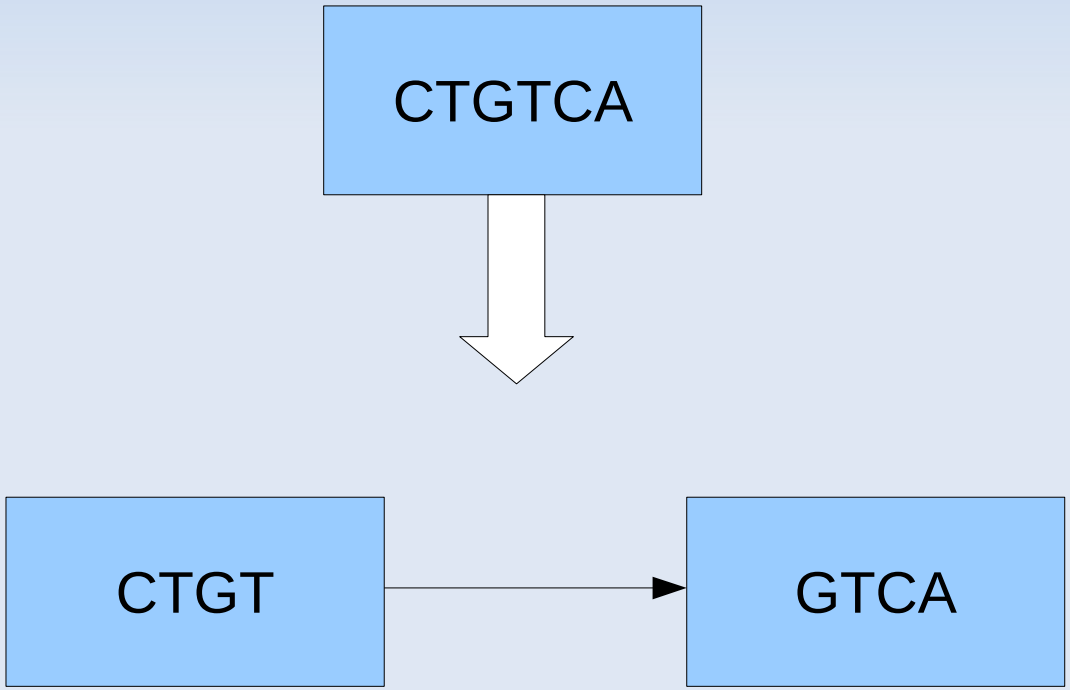


Il grafo di sequenza: esempio



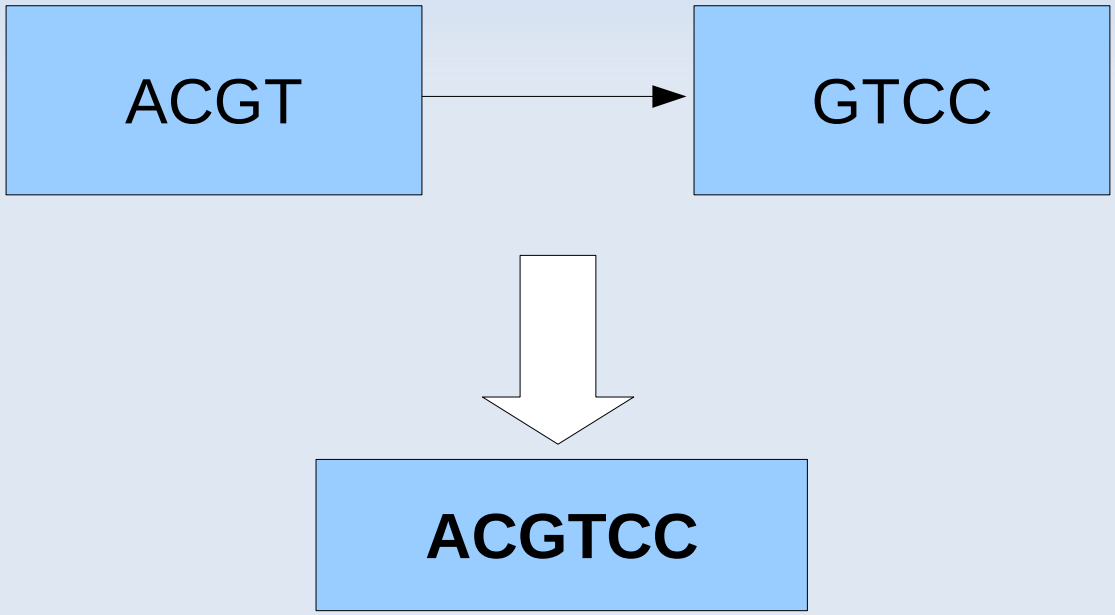
Operazioni sul grafo di sequenza (1)

- CUT : Trasforma un singolo nodo in due nodi vicini



Operazioni sul grafo di sequenza (2)

- MERGE : Trasforma due nodi vicini *(con in/out degree = 1)* in un singolo nodo



Le famiglie di repeat nei grafi di sequenze

- Il grafo di sequenza rileva, principalmente, due tipi di repeat
  - Tandem repeat (di dimensione piccola)

    CGATATATATATCTAC
  - Interspersed repeat

    ...ACTG...<basi>...ACTG...
- Spesso i grafi di sequenza di famiglie di repeat sono grafi diretti aciclici.

Scopo della ricerca

- Scopo finale: ricerca negli organismi eucarioti di famiglie di repeat
  - Famiglia di repeat: collezione di sequenze simili.

- Primo soggetto di studio: genomi batterici.
  - *Più semplici e facili da capire rispetto a quelli eucarioti.*
  - *Composti in genere da 6 – 7 milioni di coppie di basi, e al'incirca un ordine simile di l- tuple, quando l è scelto relativamente piccolo.*

Osservazioni

- *Il numero di possibili stringhe di lunghezza l per un alfabeto di dimensione 4, è già enorme per piccoli valori di l.*

  Per le applicazioni di analisi di sequenze (string matching) l è scelto abbastanza grande, per permettere l'assunzione che ci siamo pochissime stringhe che appaiono nel genoma due volte per caso.

- Il numero di famiglie di repeat, all'interno di un genoma, è abbastanza piccolo.

- Il numero degli elementi appartenenti a una stessa famiglia è ugualmente piccolo.

Ricerca di famiglie di repeat

- Assumendo che le copie siano uniformemente spalmate nella sequenza genomica, ci aspettiamo di trovare ripetizioni, separate da lunghe sequenze non ripetute.
  - In particolare:
    - I nodi corrispondenti a sequenze ripetute saranno scoperte ed etichettate durante la costruzione del grafo delle sequenze.
    - Avremo poi altri nodi, rappresentanti lunghe sequenze non ramificate, oppure il risultato di piccole dissimilarità tra elementi della stessa famiglia di repeat.

Ricerca di famiglie di repeat

Il grafo di sequenza sarà quindi composto da cluster di piccoli nodi ripetuti, interconnessi con nodi singoli di lunghezza maggiore

La cancellazione dei nodi singoli, porta alla decomposizione del grafo in poche componenti connesse, contenenti una o più famiglie di repeat.

Grazie a questo semplice principio, mostriamo un primo metodo per l'individuazione di famiglie di repeat.

Componenti Connesse

- Input:
  - Un insieme S di letture di un qualche genoma
  - Un valore di lunghezza di soglia L
- Output:
  - Le componenti connesse risultanti

Algoritmo

Algorithm 1 Connected Components

```
1: function IsolateComponents(S, l)
2:     Build the sequence graph for S
3:     Merge all possible pairs of nodes
4:     Remove all single nodes of length ≥ l
5:     Merge all possible pairs of nodes
6:     Remove all small components
7:     return the resulting connected components
8: end function
```

Costruzione del grafo di sequenza

- I nodi formati da differenti insiemi di sequenze non possono essere uniti. (*Per eseguire il merge, i nodi devono essere vicini*)

- Ogni fine lettura coincide con la fine di un nodo.

- Questo porta alla formazione generalmente,di path non ramificati nel grafo di sequenza.

Merging dei nodi

- Ignoriamo questa restrizione, e facciamo il merge dei nodi, finché vengono marcati come ripetizioni oppure no, se i loro insiemi di sequenze sono differenti

- Così facendo otteniamo dei nodi, non marcati come ripetizioni, con una lunghezza maggiore della soglia L scelta in input.

Rimozione nodi e merging

- Cancelliamo i nodi che superano la soglia L
- Eseguiamo il merging tra i nodi ripetuti, che non stati uniti in precedenza a causa dei nodi ora cancellati.
- A questo punto, nessun altro nodo può essere unito o cancellato, e il grafo e già una collezione di componenti connesse.

Rimozione delle componenti non significative

- Alcune componenti possono però essere il risultato di match perfetti tra piccole parti, a causa di operazioni di merge precedenti.
- Tali componenti sono formate da pochi nodi (al massimo 5 nodi), con un singolo nodo ripetuto al centro.
- Rimuoviamo tali componenti, e lasciamo nel grafo solo i componenti corrispondenti alle famiglie più grandi.

Combining nodes

- Nel caso in cui elementi della stessa famiglia di repeat differiscano in molte basi, la procedura precedentemente descritta può tralasciare alcune famiglie meno rappresentative.
- Nel caso in cui le sequenze condividono poche tuple, possiamo rimpiazzare i nodi che contengono le tuple più significative, con altri nodi ottenuti per combinazione di quelli precedenti

Allineamento di Nodi

- Nella maggior parte dei casi, i due nodi che vengono combinati sono di differenti lunghezze.
- In tali caso li allineiamo usando un algoritmo di allineamento semi – globale.
- La matrice dei costi per l’allineamento dà un punteggio di zero, in caso di match tra i nucleotidi (A,T,G,C), e 1 in caso di mismatch.

Allineamento di Nodi

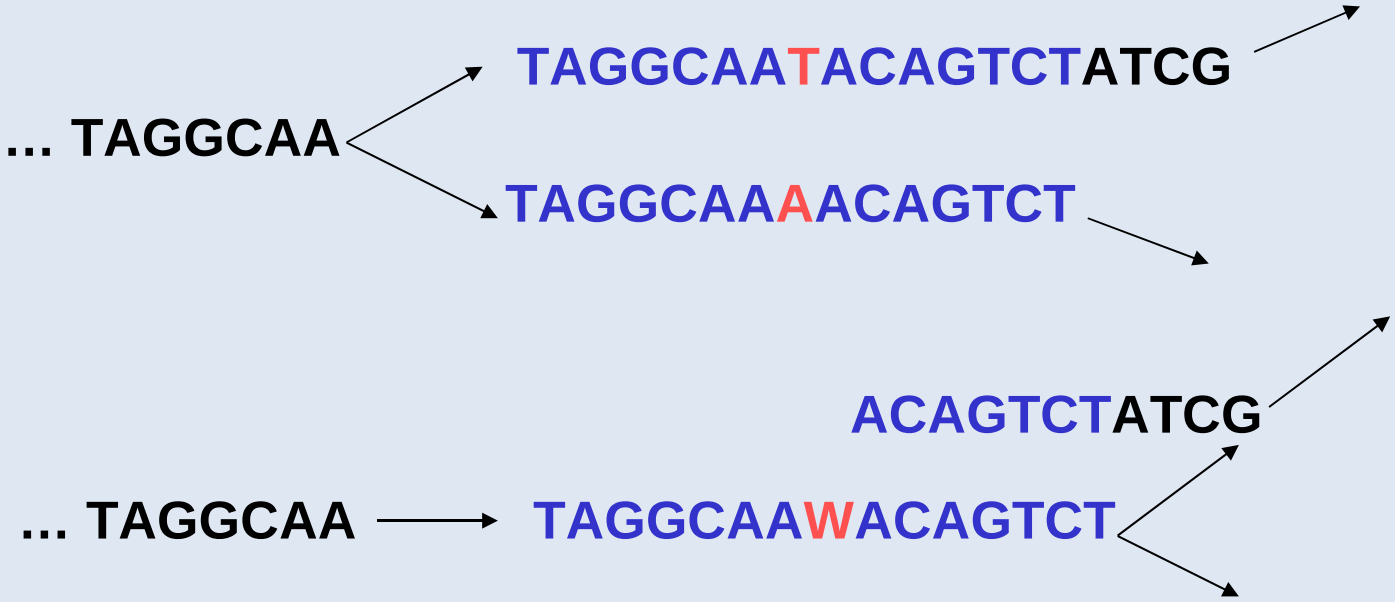
- Per match che coinvolgono insiemi di nucleotidi, come W = (A,T), il punteggio è zero se uno contiene l'altro, altrimenti è dato dal minimo numero di rimpiazzi e cancellazioni per trasformare un insieme nell'altro
- Esempio:

W = (A,T) - G score = 2

una cancellazione e un rimpiazzo

Definizione

- Combinare due nodi: sostituire entrambi con un unico nodo, la cui sequenza è l'unione delle sequenze dei due nodi.



Dove W = A,T

Algoritmo

Algorithm 2 Combine

1: **procedure** COMBINE(n_1, n_2, t)
2: Let n_1 be the longer of the two nodes
3: Align the sequences of n_1 and n_2, creating a semi-global alignment of length l
4: **if** the alignment score is smaller than $t \times \left\lceil \frac{l-d+1}{d} \right\rceil$ **then**
5: Cut the node n_1 at the end of the aligned prefix
6: Let $\overline{n_1}$ be the left portion of the cut node n_1
7: Create a new node n with the consensus of $\overline{n_1}$ and n_2
8: Bind the nodes in the neighborhood of $\overline{n_1}$ and n_2 to n
9: Remove $\overline{n_1}$ and n_2
10: **end if**
11: **end procedure**

t = fattore di scala

$m = \left\lceil \frac{l-d+1}{d} \right\rceil$ = minimo numero di mismatch necessari per separare i nodi

Combine: Conclusioni

- Generalmente, combinare due nodi non riduce la dimensione del grafo, ma ne migliora la complessità.
- Una serie di combinazioni può ridurre sottografi annidati, in singoli cammini, i quali verranno uniti in seguito in un singolo nodo più lungo.
- Abbiamo quindi dei maggiori vantaggi, rispetto all’approccio semplice per la ricerca di componenti connesse visto in precedenza.

Riassumendo:

I metodi hanno successo nel caso in cui:

- *Abbiamo ogni famiglia in una differente componente connessa.*
- *Ogni componente connessa contiene solo membri di una singola famiglia*

Confronti con altri metodi

- Tali metodi sono stati utilizzati sia su dati reali, che su dati creati artificialmente.

- Non è stato possibile comparare tali metodi con altri (in particolare metodi “de novo”), in quanto differiscono sia per l’input che per l’output.
- Infatti i nostri metodi
  - non prendono come input porzioni continue di genoma, ma un insieme di letture.
  - Restituiscono come output componenti connesse.

Risultati: Connected Component

- Per testare il nuovo metodo, abbiamo utilizzato dei cromosomi creati artificialmente, con un diverso numero di famiglie di repeat.
- In particolare:
- Le sequenze di background (sequenze di genoma, non ripetute), non contengono nessuna sottostringa duplicata di lunghezza 19
- Le famiglie di repeat inserite hanno dimensione 2,4,16 e 256
- Un cromosoma artificiale può avere da 0 a 2 famiglie di ogni tipo.

Risultati

- Abbiamo utilizzato 15 tipi di cromosomi differenti, e da questi abbiamo creato letture con una lunghezza media di 250 coppie di basi.
- Per ogni cromosoma, abbiamo utilizzato rispettivamente il 25, 50, 75 e 100 % della sua copertura
- Sapendo quali sequenze corrispondevano ai membri di una famiglia, siamo stati capaci di associare ogni componente del grafo risultante, alle famiglie in esso contenute.
- Nel caso ideale, dovremmo trovare ogni famiglia contenuta in un singolo componente connesso.

Tabella dei risultati

Components per Family				Families per Component				Discovered Families (%)			
25	50	75	100	25	50	75	100	25	50	75	100
6.33	5.57	4.41	5.68	1.00	1.00	1.00	1.00	20	40	70	73
4.50	3.50	4.33	4.70	1.00	1.00	1.00	1.00	63	97	97	100
5.26	4.78	4.82	4.32	1.00	1.00	1.00	1.00	43	70	85	92
4.27	4.37	4.18	4.42	1.00	1.00	1.00	1.00	100	100	100	100
3.93	4.48	4.31	4.12	1.00	1.00	1.00	1.02	58	78	87	93
4.41	4.01	4.31	4.47	1.00	1.01	1.00	1.02	83	95	100	100
4.31	4.83	4.25	4.20	1.01	1.01	1.02	1.05	64	82	91	94
4.08	3.80	3.89	4.44	1.93	1.93	1.93	1.93	100	100	100	100
4.68	4.56	3.98	4.51	1.78	1.93	1.61	1.51	65	70	80	90
4.38	4.93	4.45	4.72	1.78	1.50	1.58	1.91	85	100	100	100
4.88	4.36	4.28	4.14	1.64	1.55	1.40	1.48	60	77	89	96
4.83	4.69	5.08	4.23	2.18	2.89	2.98	3.31	100	100	100	100
4.48	4.89	4.81	4.50	2.29	2.19	2.36	2.28	72	89	90	98
4.28	4.62	4.63	4.24	1.66	1.82	2.35	2.50	89	98	99	100
4.26	4.51	4.91	4.98	1.60	1.66	1.78	1.87	68	93	97	98

Risultati: Combine

- Per testare il metodo, abbiamo creato un dataset di sequenze di genomi batterici, e le loro inserzioni di sequenze conosciute.
- *Inserzioni di sequenza:*
  - *fanno parte degli elementi trasponibili*
  - *hanno brevi sequenze ripetute invertite all'estremità del loro DNA*
- Abbiamo utilizzato 15 insiemi di letture differenti, che coprivano in media il 25, 50, 75 e 100% del genoma in media.
- Ognuno di questi insiemi, è stato usato due volte, con un fattore di scala t=1 prima, e t = 3 poi.

Tabella dei risultati

Combine Factor (t)	1.0				3.0			
Sequencing Coverage (%)	25	50	75	100	25	50	75	100
*Bacillus anthracis (plasmid PX01)**	0	7	0	40	17	58	92	100
Bifidobacterium longum	14	54	63	64	39	68	81	83
Burkholderia xenovorans	44	67	73	78	50	67	75	81
*Colwellia psychrerythraea**	40	100	100	100	83	100	100	100
*Desulfitobacterium hafniense**	67	80	97	100	71	96	100	100
*Desulfovibrio desulfuricans**	33	47	93	100	33	75	92	100
Escherichia coli	17	50	62	70	44	65	85	92
Geobacter uraniumreducens	32	62	67	70	46	68	75	80
Gloeobacter violaceus	30	70	60	83	54	75	88	100
*Granulibacter bethesdensis**	7	7	40	53	25	33	42	75
Haloarcula marismortui	3	12	22	28	13	25	50	63
Halobacterium sp-plasmid pNRC100	37	57	56	61	37	42	53	57
Legionella pneumophila-Paris	0	13	20	7	8	0	0	17
Legionella pneumophila-Philadelphia 1	27	63	93	93	54	63	63	92
Methanosarcina acetivorans	88	98	98	100	93	100	99	100
Methylococcus capsulatus	22	65	77	83	44	71	90	96
*Nitrosospira multiformis**	53	93	100	100	92	100	100	100
Photobacterium profundum	87	100	100	100	100	100	100	100
Pseudomonas syringae	92	99	100	100	97	100	100	100
Pyrococcus furiosus	47	58	71	73	50	72	81	89
Ralstonia solanacearum	38	60	75	89	53	78	93	95
Rhodopirellula baltica	82	98	100	100	97	100	100	100
*Roseobacter denitrificans**	40	80	87	100	42	92	100	92
*Salinibacter ruber**	100	100	100	100	100	100	100	100
Shewanella oneidensis	10	26	18	23	18	38	15	41
Sulfolobus solfataricus	94	99	100	99	94	100	100	100

Conclusioni

- Abbiamo presentato due metodi per la ricerca di famiglie di repeat in genomi non completamente sequenziati.
- Abbiamo verificato, che il principale ostacolo per entrambi i metodi, è la divisione delle famiglie di repeat in componenti separate.
- Questo è anche il problema principale da risolvere, prima di utilizzarli per l’analisi di genomi di complessità maggiore.