Assignment 2, Bioinformatica

Davide Cozzi, 829827

Indice

Capitolo 1

Esercizio 1

Per la specifica del problema abbiamo la seguente situazione:

- **input**: una matrice di caratteri M costruita su $\{0, 1, *\}$, con al più due occorrenze di $*$. Si specifica che M è una matrice $|S| \times |C|$, con ogni riga che rappresenta una specie dell'insieme S e ogni colonna che rappresenta un carattere dell'insieme C
- **output**: un'eventuale matrice binaria di filogenesi perfetta M_P, anch'essa $|S| \times |C|$, o eventualmente un vettore di matrici binarie di filogenesi perfetta

Il ragionamento verte sulla presenza, all'interno della matrice M in input, eventualmente anche a priori rispetto alle possibili sostituzioni dei caratteri $*$, della cosiddetta **matrice proibita**, che indico con M_F:

$$M_F = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}$$

La presenza della matrice proibita infatti impedisce la costruzione di un albero di filogenesi perfetta in quanto, con questa teoria, si ha un solo *gain* e nessun *loss* per ogni carattere. In termini di albero di filogenesi questa caratteristica si verifica avendo che ogni carattere etichetta un solo ramo dell'albero e, una volta guadagnato un certo carattere, esso non verrà mai perso. In altri termini, preso un nodo interno w dell'albero di filogenesi, si ha che tutti i caratteri, che etichettano i rami che dalla radice portano a quel nodo, saranno presenti in tutte le foglie del sottoalbero che w come radice. Si ha quindi che un alberi di filogenesi perfetta esiste sse la matrice da cui viene calcolato non contiene la matrice proibita in quanto dovrei avere la "perdita" di uno dei due caratteri per poter rappresentare, in presenza delle prime due

specie, la terza, quella con entrambi i caratteri.
La verifica dell'assenza della matrice proibita può essere effettuata tramite il controllo che la collezione delle colonne della matrice sia **laminare**.
Nel caso in esame però la matrice in input può contenere una o due posizioni mancanti. Si ha che:

- nel caso si abbia un solo $*$ esso può essere sostituito con o 0 o 1
- nel caso si abbiano due $*$ essi possono essere sostituiti, rispettivamente, da una delle seguenti coppie di valori binari:

$$(0,0),(0,1),(1,0),(1,1)$$

Prima di effettuare le sostituzioni è possibile fare un controllo preliminare. Si prenda la sottomatrice di M, che chiamiamo M', composta da tutte le colonne di M in cui non si ha un simbolo $*$. Qualora M' non abbia la collezione di colonne laminare, avendo quindi all'interno la matrice proibita, posso già concludere dicendo che non esiste alcuna sostituzione che permetta la costruzione di una matrice di filogenesi perfetta. Questo controllo permette di evitare diversi controlli di laminarità, di costo quadratico, in caso che la matrice non ammetta filogenesi perfetta a priori rispetto alle sostituzioni. Il controllo viene passato "automaticamente" qualora M' sia di una sola colonna.

Esempio 1. *Vediamo un esempio di sottomatrice M'. Data:*

$$M = \begin{bmatrix} 1 & 0 & 0 \\ 0 & * & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

Si ottiene, rimuovendo la seconda colonna, la matrice:

$$M' = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}$$

che, in questo caso, non solo contiene la matrice proibita ma è la matrice proibita stessa. Quindi nessun valore di $*$ porterebbe ad una matrice di filogenesi perfetta.
Come anticipato la verifica della presenza della matrice perfetta, a livello algoritmico, viene effettuata tramite il test di laminarità. Nel caso si analisi si produrrebbe la seguente matrice L:

$$L = \begin{bmatrix} -1 & 0 \\ 0 & -1 \\ -1 & 1 \end{bmatrix}$$

Dove si ha la seconda colonna con due valori non nulli diversi tra loro, avendo quindi che M' non è laminare, contenendo la matrice proibita.

Questo primo controllo è utile anche per calcolare l'esistenza della matrice di filogenesi perfetta, e di conseguenza del relativo albero, qualora non si abbiano correzioni da fare.
Passiamo ora ai casi in cui sia effettivamente possibile effettuare le sostituzioni.
Negli esempi successivi ho scelto di non mostrare ogni volta le matrici L utili a verificare algoritmicamente la laminarità in quanto, avendo a che fare con toy examples, era facilmente individuabile la presenza della matrice proibita o meno. Il test viene comunque utilizzato nel piccolo codice usato come appoggio per questo esercizio.

1.1 Singola Sostituzione

Avendo già verificato che la matrice, privata della singola colonna in cui è presente il simbolo $*$, non contiene la matrice proibita non resta che verificare se questo accade anche sostituendo con 0 e con 1. A priori rispetto al resto della matrice in input non è verificabile se una sostituzione sia valida o meno e, essendo il controllo tramite laminarità in tempo $\mathcal{O}(|S| \cdot |C|)$ sulla matrice M in input dopo aver sostituito $*$, pare sensato procedere con entrambe le sostituzioni. A seconda della matrice potrei ottenere una matrice di filogenesi solo sostituendo $*$ con 1 o solo con 0 ma in determinati casi anche con entrambi.

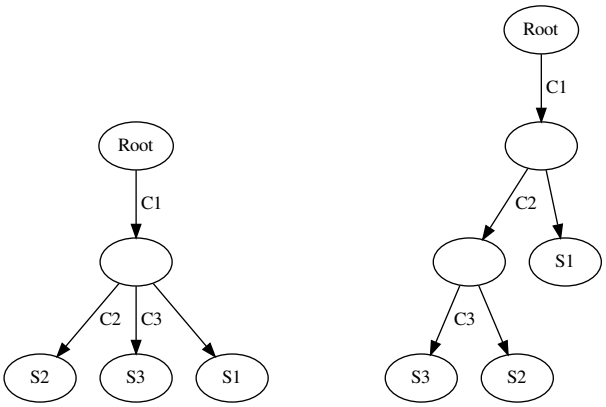
Esempio 2. *Vediamo un esempio.*
Sia data in input la matrice (e si noti che la relativa matrice M' non presenta all'interno la matrice proibita):

$$M = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & * & 1 \end{bmatrix} \qquad M' = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 1 \end{bmatrix}$$

Si ha quindi una matrice con 3 specie, in ordine per riga $S1, S2, S3$, e 3 caratteri, in ordine per colonna $C1, C2, C3$.
Si hanno quindi le due matrici, sostituendo prima 0 e poi 1:

$$M_0 = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \qquad M_1 = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

In entrambi i casi è possibile effettuare con successo il test di linearità avendo quindi che in entrambi i casi è possibile costruire l'albero di filogenesi. Possiamo quindi chiamare tali matrici M_{P_0} e M_{P_1}, avendo i rispettivi alberi di filogenesi perfetta:



1.2 Doppia Sostituzione

Il caso con la doppia sostituzione è leggermente più complicato da trattare. Come già anticipato, avendo due * nella matrice M in input, posso avere 4 combinazioni possibili di correzioni:

$$(0,0),(0,1),(1,0),(1,1)$$

Si può quindi notare che in questo caso possiamo fare qualche ragionamento aggiuntivo prima di effettuare il test di laminarità sull'intera matrice.
In questo caso infatti, qualora i due simboli $*$ non si trovino sulla stessa colonna, è possibile effettuare un test di laminarità preliminare sulle due colonne in cui sono presenti i $*$, ovviamente dopo aver effettuato la sostituzione. Solo nel momento in cui questo primo test, di costo $\mathcal{O}(|S| \cdot 2) \approx \mathcal{O}(|S|)$, venga superato si procede al più dispendioso test di laminarità sull'intera matrice, che ricordiamo essere di costo $\mathcal{O}(|S| \cdot |C|)$. Vengono così limitati i test più dispendiosi.
Anche in questo caso non pare possibile stabilire a priori quali sostituzioni possano portare ad una matrice di filogenesi perfetta, senza passare per la "batteria" di test di laminarità sopra descritta.

Esempio 3. *Vediamo un esempio.*
Sia data in input la matrice M, il cui controllo di laminarità su M', ottenuta

da M senza le colonne con i simboli $*$, non segnala problemi:

$$M = \begin{bmatrix} 1 & 0 & * & 0 & 0 \\ * & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad M' = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

Si può quindi procedere con le varie sostituzioni, ricordando che, avendo i simboli $*$ su due colonne diverse, potremo effettuare i controlli di laminarità in primis sulla coppia di colonne con le sostituzioni.
Si comincia con la coppia $(0, 0)$, avendo quindi:

$$M_{0,0} = \begin{bmatrix} 1 & 0 & \mathbf{0} & 0 & 0 \\ \mathbf{0} & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Il primo test, quello di linearità sulla sottomatrice composta dalle due colonne con $*$, già segnala come si abbia la matrice proibita, infatti:

$$M'_{0,0} = \begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{0} & 1 \\ 1 & 1 \\ 0 & 0 \end{bmatrix}$$

Si può quindi concludere che la sostituzione $(0, 0)$ non porterà ad una matrice di filogenesi perfetta.
Si analizza quindi la seconda sostituzione, $(0, 1)$:

$$M_{0,1} = \begin{bmatrix} 1 & 0 & \mathbf{0} & 0 & 0 \\ \mathbf{1} & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad M'_{0,1} = \begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{1} & 1 \\ 1 & 1 \\ 0 & 0 \end{bmatrix}$$

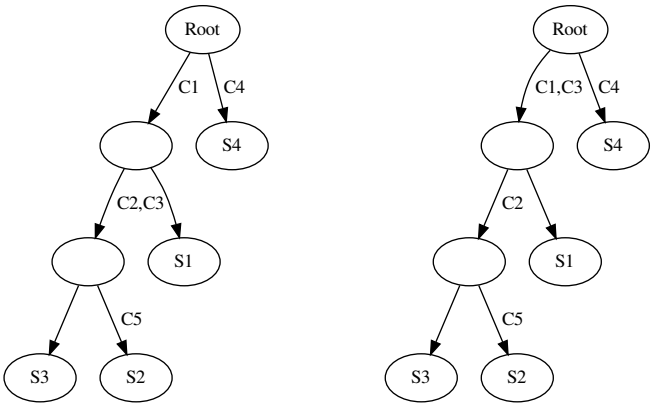
Dove si nota che né il primo controllo sulla sottomatrice né il test di laminarità sulla matrice completa segnalano problemi quindi la matrice $M_{0,1}$ è una matrice di filogenesi perfetta, $M_{P_{0,1}}$.
Si passa ad analizzare la terza sostituzione, $(1, 0)$:

$$M_{1,0} = \begin{bmatrix} 1 & 0 & \mathbf{1} & 0 & 0 \\ \mathbf{0} & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad M'_{1,0} = \begin{bmatrix} 1 & \mathbf{1} \\ \mathbf{0} & 1 \\ 1 & 1 \\ 0 & 0 \end{bmatrix}$$

In questo caso il primo test di linearità sulla sottomatrice non segnala problemi ma il test di linearità sulla matrice segnala come, a causa delle prime due colonne, la matrice $M_{1,0}$ non sia una matrice di filogenesi perfetta.
Si arriva infine all'ultima correzione possibile, $(1,1)$*:*

$$M_{1,1} = \begin{bmatrix} 1 & 0 & \mathbf{1} & 0 & 0 \\ \mathbf{1} & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \qquad M'_{1,1} = \begin{bmatrix} 1 & \mathbf{1} \\ \mathbf{1} & 1 \\ 1 & 1 \\ 0 & 0 \end{bmatrix}$$

Anche in questo caso né il test di laminarità preliminare sulla sottomatrice né quello sulla matrice intera segnalano problemi, avendo infatti che $M_{1,1}$ è una matrice di filogenesi perfetta, $M_{P_{1,1}}$.
Si ha quindi che sia la sostituzione $(0,1)$ *che la sostituzione* $(1,1)$ *portano ad avere due matrice di filogenesi perfetta, $M_{P_{0,1}}$ e $M_{P_{1,1}}$, coi rispettivi alberi:*



1.3 Note Conclusive

Come si è notato non sembra essere possibile prevedere le sostituzioni corrette, in entrambi i casi, prima di effettuarle e verificarle, possibilmente con controlli di laminarità su sottomatrici. L'unico caso per cui si può evitare di procedere è quello in cui la matrice in input M includa la matrice proibita a priori rispetto alle sostituzioni. Una volta ottenuta la singola matrice di filogenesi perfetta si segnala che l'albero di filogenesi perfetta relativo è **unico**. In questo contesto con "unico" si intende che dal nodo radice ad ogni foglia avrò sempre e comunque lo stesso insieme di etichette per gli archi. Si segnala

comunque che diverse sostituzioni possono portare a diverse matrici valide e quindi altrettanti alberi (come visto negli esempi).
Vediamo quindi una bozza di pseudocodice che codifica la procedura seguita. Si hanno in primis i due pseudocodici relativi ai due tipi di sostituzione:

Algorithm 1 Procedura per la correzione di un singolo errore

```
1: function OneCorrection(M)
2:     perfectPhylogenyVec ← [ ]
3:     corrections ← [0, 1]
4:     for corr in corrections do
5:         M_corr ← correctWith(M, corr)
6:         if isLaminar(M_corr) then
7:             push(perfectPhylogenyVec, M_corr)
8:         end if
9:     end for
10:    return perfectPhylogenyVec
11: end function
```

Algorithm 2 Procedura per la correzione di un doppio errore

```
1: function TwoCorrection(M)
2:     perfectPhylogenyVec ← [ ]
3:     corrections ← [(0, 0), (0, 1), (1, 0), (1, 1)]
4:     for corr in corrections do
5:         M_corr ← correctWith(M, corr)
6:         M'_corr ← columnsCorrectedWith(M, corr)
7:         if isLaminar(M'_corr) ∧ isLaminar(M_corr) then
8:             push(perfectPhylogenyVec, M_corr)
9:         end if
10:    end for
11:    return perfectPhylogenyVec
12: end function
```

Nel secondo caso, quello con la doppia sostituzione, si noti come l'operazione di *and* all'interno dell'*if* verifichi in primis la laminarità delle colonne, qualora siano due (avendo che il controllo di laminarità su singola colonna è automaticamente passato), su cui sono state effettuate le correzioni.

Si ha quindi lo pseudocodice con l'unione delle due possibili categorie di correzione, in presenza di singolo errore o doppio, che calcola le varie possibili matrici di filogenesi perfetta e restituisce un array contenente esse e il conto dei possibili alberi:

Algorithm 3 Procedura per la verifica di filogenesi perfetta con al più due sostituzioni

```
1: function CheckPerfect(M)
2:     errCount ← countError(M)
3:     M' ← subMatrixNoError(M)
4:     if ¬isLaminar(M') then
5:         return ∅
6:     end if
7:     perfectPhylogenyVec ← [ ]
8:     if errCount == 0 then
9:         return [perfectPhylogeny(M)]
10:    else if errCount == 1 then
11:        perfectPhylogenyVec ← OneCorrection(M)
12:    else
13:        perfectPhylogenyVec ← TwoCorrection(M)
14:    end if
15:    treeCount ← length(perfectPhylogenyVec)
16:    return (perfectPhylogenyVec, treeCount)
17: end function
```

Capitolo 2

Esercizio 2

Per il problema in analisi si ha:

- **input**: una matrice M_G, $n \times m$, di n genotipi costruita su $\{0, 1, *, ?\}$, dove $0/1$ rappresentano i rispettivi siti omozigoti, $*$ rappresenta i siti eterozigoti e ? rappresenta i dati mancanti
- **output**: una matrice M_H, $2 \cdot n \times m$, di aplotipi, costruita su $\{0, 1\}$ tale che ogni genotipo dello matrice in input M_G sia risolto da una coppia di aplotipi della matrice M_H e che M_H sia matrice di filogenesi perfetta

Definizione 1. *Si ha che la coppia di aplotipi $\langle h_1, h_2 \rangle$, visti come vettori su alfabeto $\{0, 1\}$,* ***risolve*** *il genotipo g, visto come vettore su alfabeto $\{0, 1, *\}$, sse, $\forall i$:*

$$\begin{cases} h_1[i] \neq h_2[i] & \text{se } g[i] = * \\ g[i] = h_1[i] = h_2[i] & \text{altrimenti} \end{cases}$$

Definizione 2. *La matrice di aplotipi M_H* ***risolve*** *la matrice di genotipi M_G sse:*

$$\forall g \in M_G, \ \exists h_1, h_2 \in M_H \text{ t.c. } \langle h_1, h_2 \rangle \text{ risolve } g$$

La strategia per risolvere la matrice di genotipi con la matrice di aplotipi consiste quindi, in primis, nell'effettuare le possibili correzioni. Qualora si corregga un dato mancante con un sito omozigote 0 o 1 si ha che in fase di costruzione della matrice degli aplotipi verranno costruite due righe con 0 o 1, rispettivamente, in quella precisa posizione. Qualora il dato venga invece corretto tramite un sito eterozigote si avrà, nella matrice di aplotipi, una coppia di righe con due valori complementari per quella posizione.

Si intuisce quindi come la complessità del problema cresca in primis all'aumentare dei dati mancanti nella matrice M_G, aumentando le possibili combinazioni di 0, 1 e * per poter correggere la matrice prima di poter effettivamente tentare di risolverla con una matrice di aplotipi. Inoltre, in fase di risoluzione con la matrice di aplotipi, un'elevata presenza di siti eterozigoti nel medesimo genotipo, quindi nella stessa riga, comporta che si abbiano potenzialmente molte matrici di aplotipi da controllare, per trovare quella che effettivamente risolve la matrice di genotipi e ammette anche filogenesi perfetta.
Una possibile strategia di ottimizzazione del problema potrebbe essere quella di avere una sorta di *reference*, tramite il quale escludere, in prima battuta, alcune strategie di correzione dei dati mancanti e, in seconda battuta, poter velocizzare la scelta di coppie di aplotipi che risolvano un genotipo con vari siti eterozigoti.

2.1 Conclusioni ed Esempio

Come si è descritto il problema non sembra prevedere una soluzione "facile". Si propone quindi un piccolo toy example con un solo dato mancante.

Esempio 4. *Sia data la matrice:*

$$M_G = \begin{bmatrix} * & ? \\ 1 & 1 \\ 0 & 1 \end{bmatrix}$$

Le possibili correzioni sono le seguenti:

$$M_{G_0} = \begin{bmatrix} * & \mathbf{0} \\ 1 & 1 \\ 0 & 1 \end{bmatrix} M_{G_1} = \begin{bmatrix} * & \mathbf{1} \\ 1 & 1 \\ 0 & 1 \end{bmatrix} M_{G_*} = \begin{bmatrix} * & \mathbf{*} \\ 1 & 1 \\ 0 & 1 \end{bmatrix}$$

Dopo aver effettuato le correzioni bisogna costruire le possibili matrici di aplotipi, secondo le regole sopra definite.
Iniziamo con M_{G_0}*:*

$$M'_{G_0} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 1 & 1 \\ 1 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

Dove si nota la matrice proibita quindi sicuramente la sostituzione con il sito omozigote 0 non è una sostituzione accettabile.
Passiamo a M_{G_1}*:*

$$M'_{G_1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 1 \\ 1 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

Che si nota non contenere la matrice proibita quindi la sostituzione con il sito omozigote 1 sarebbe una sostituzione accettabile. Si ottiene quindi una matrice di aplotipi valida, che, essendo ottenuta sostituendo 1, chiamiamo M_{H_1}*.*
Si ha infine M_{G_*} *che, a causa del doppio sito eterozigote nello stesso genotipo comporta due possibili matrici di aplotipi:*

$$M'_{G_*} = \begin{bmatrix} 1 & 1 \\ 0 & 0 \\ 1 & 1 \\ 1 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \quad M''_{G_*} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \\ 1 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

Notando che solo M'_{G_*} *ammette effettivamente filogenesi perfetta quindi la sostituzione del dato mancante con un sito eterozigote* $*$ *non sempre porta ad una matrice di aplotipi di filogenesi perfetta.* M''_{G_*} *infatti contiene la matrice proibita. Chiamo tale matrice di aplotipi valida, essendo ottenuta sostituendo* $*$*,* M_{H_*}*.*

Si è visto quindi come il problema della ricostruzione di matrici di aplotipi a partire da una matrice di genotipi con errori sembri essere un problema complesso, a causa, in primis, della moltitudine di correzioni possibili e, in seconda battuta, a causa della procedura di calcolo di valide matrici di aplotipi a partire da una matrice di genotipi senza dati mancanti.

Capitolo 3

Codice per la Filogenesi Perfetta

Si segnala che nei primi due esercizi è stato usato come appoggio il seguente codice: `https://github.com/dlcgold/perfect-phylogeny-rs`.
Questo piccolo tool in Rust contiene i due algoritmi visti a lezione per la verifica della laminarità e la creazione dell'albero di filogenesi. È anche presente una bozza di correzioni secondo la procedura indicata per il primo esercizio. Per quanto il codice fosse solo abbozzato è stato comodo per la creazione degli alberi di filogenesi perfetta del primo esercizio e per alcuni test durante lo svolgimento del secondo e quindi mi è sembrato giusto segnalarlo.

Capitolo 4

Esercizio 3

Per questo problema si hanno:

- **input**: una *matrice di n frammenti di aplotipi* M, per la quale sono eventualmente ammessi indel e spazi. La matrice quindi è costruita su $\{0, 1, -\}$ e, essendo costruita su m SNPs, è di dimensioni $n \times m$. Tale matrice è assunta poter non essere *error-free*
- **output**: una bipartizione delle righe di M in due insiemi di frammenti, H_1 e H_2, tali che, all'interno del singolo insieme, non si abbiano *conflitti*. Tali insiemi rappresentano quindi due aplotipi, avendo quindi che il conflitto tra frammenti avviene solo tra due aplotipi differenti. Tale bipartizioni potrebbe non esistere

Definizione 3. *Preso un insieme di frammenti di aplotipi* $F = \{f_1, f_2, \ldots, f_n\}$ *si ha che* $f' \in F$ *e* $f'' \in F$ *sono in* ***conflitto*** *sse* $f' \in H_1$ *e* $f'' \in H_2$*, ovvero sse:*

$$\exists i \in [1, m] \text{ } t.c. \text{ } f'[i] \neq f''[i] \text{ } con \text{ } f'[i] \neq \text{"-"} \wedge f''[i] \neq \text{"-"}$$

Quindi se prese due righe esiste almeno una posizione (su indice di colonna in pratica) in cui non si ha lo stesso valore su entrambe. Si noti come spazi e indel, indicati sempre con "-", vengono "ignorati" in quanto rappresentano una mancanza di informazione e possiamo quindi "immaginarli" come un valore desiderato. Questo ultimo aspetto si applica in tutte le situazioni dell'esercizio in cui si specifica che i "-" vengono ignorati.

Avendo assunto che la matrice potrebbe non essere *error-free* bisogna procedere con la correzione degli stessi, tramite, ad esempio, l'algoritmo k-cMEC. Tale algoritmo risolve il problema di ottimo di trovare il minimo numero di correzioni per l'intera matrice di frammenti, con al più k correzioni per colonna, necessarie per ottenere una matrice che ammette bipartizione.

Da specifica, inoltre, si ha che il numero massimo di correzioni per colonna è uno, quindi si ha, per l'algoritmo k-cMEC, $k = 1$. Nel dettaglio con il termine *correzione* si intende l'operazione di *flip* di un valore, da 0 a 1 o viceversa, trascurando quindi gli indel e gli spazi, per gli stessi motivi espressi sopra.
Da specifica, in aggiunta, si assume che la matrice M in input è priva di gap interni.
Si ha quindi, per l'algoritmo di correzione:

- **input**: una matrice M di n frammenti di aplotipi non *error-free*, come definita sopra
- **output**: una matrice M', qualora esista, di dimensioni uguali a M, anch'essa costruita su $\{0, 1, -\}$, *error-free*, costruita a partire da M con al più una singola correzione per colonna

Prima di dare un'idea dell'algoritmo bisogna dare qualche definizione.

Definizione 4. *Una colonna j di una matrice M di n frammenti di aplotipi è detta* ***omozigote*** *sse:*

$$\forall i \in [1, n], \;\; M_j[i] = 0 \vee M_j[i] = \text{"-"}$$

oppure sse:

$$\forall i \in [1, n], \;\; M_j[i] = 1 \vee M_j[i] = \text{"-"}$$

In altri termini se la colonna, al più degli indel e degli spazi, è formata o da soli zeri o da soli uni.
Una colonna priva di questa caratteristica, avendo quindi valori diversi oltre a quello di indel e spazi, è detta ***eterozigote****.*

Definizione 5. *Prese due colonne, j' e j'' di una matrice M di n frammenti di aplotipi si ha che esse sono dette* ***colonne concordanti*** *se vale una delle seguenti condizioni:*

- *entrambe le colonne sono omozigote*
- *entrambe le colonne sono eterozigote e si ha che, a parità di indice, tutti i valori della prima colonna sono il complementare dei valori della seconda*
- *una colonna è omozigote e l'altra eterozigote*

Per la seconda condizione (anche se implicitamente questo discorso, come visto nella definizione precedente, si applica anche agli altri due casi) si trascurano indel e spazi, quindi la condizione vale:

$$\forall i \in [1, n] \;\; t.c \; M_{j'}[i] \neq \text{"-"} \wedge M_{j''}[i] \neq \text{"-"}$$

Tutti e tre i casi permettono quindi la bipartizione delle righe della coppia di colonne.

Si noti che una colonna omozigote non presenta particolari informazioni in merito alla questione della bipartizione, avendo tutti valori uguali.

Teorema 1. *Si può dimostrare che si può ottenere una bipartizione da una matrice M di n frammenti di aplotipi senza gap interni sse ogni coppia di colonne concorda.*

Si propone quindi un'idea di base per l'ottenimento della matrice M', *error free*, a partire da M, con al più una correzione per colonna.
L'idea di base è che si procede scorrendo da sinistra a destra le colonne di M, correggendo la colonna j basandosi eventualmente sulle prime $j-1$ colonne, con eventuali branch per le possibili correzioni.
La correzione della colonna j, qualora venga effettivamente fatta, può comportare la costruzione di una colonna omozigote o di una colonna eterozigote. In questo secondo caso la colonna j deve essere concordante con almeno un'altra colonna tra 1 e $j-1$.
Per tenere traccia del minimo numero di correzioni fino alla colonna j si usa $D(1,j)$ e diciamo che esso viene calcolato a partire da $D(1,j-1)$, qualora possibile. Il limite di una sola correzione per colonna infatti potrebbe comportare il non ottenimento della matrice M' e per indicare il fatto in termini matematici diciamo che in tal caso si ha:

$$D(1,j) = \infty$$

Chiamiamo $d(j', j'')$ una funzione equivalente alla distanza di Hamming, adattata a conteggiare le differenze tra colonne due colonne, j' e j''.
Definiamo quindi una funzione che calcola il numero di correzioni sulla colonna j'' necessarie per renderla concorde con la colonna j:

$$c(j', j'') = \begin{cases} 0 & \text{se } d(j', j'') = 0 \\ 1 & \text{se } d(j', j'') = 1 \wedge j' \text{ concorde } j'' \\ \infty & \text{altrimenti} \end{cases}$$

Possiamo quindi definire una bozza di equazione di ricorrenza per il calcolo del minimo numero di correzioni necessarie ad ottenere M', con il vincolo di avere al più una correzione per colonna:

$$D(1,j) = \min_{j \in [2,n]} \begin{cases} D(1,j-1) + \min \begin{cases} c(\vec{0}, j) \\ c(\vec{1}, j) \end{cases} & \text{, con } j \text{ omozigote} \\ D(1,j-1) + \min_{i \in [1,j-1]} c(i,j), \text{ con } i, j \text{ eterozigote} \end{cases}$$

Avendo quindi il minimo tra due casi, in ordine:

- il caso omozigote, che viene calcolato sommando a $D(1, j-1)$ il minimo numero, che potrebbe essere 0, 1 o ∞, di flip atti a ottenere una matrice omozigote

- il caso eterozigote, che viene calcolato sommando a $D(1, j-1)$ il minimo numero, che potrebbe essere 0, 1 o ∞, di flip atti a ottenere una colonna eterozigote concordante con una delle prime $|j-1|$ colonne. Qualora non esista una colonna $i \in [1, j-1]$ eterozigote si assume $c(i, j) = 0$ (in quanto una colonna omozigote ed una eterozigote concordano e quindi non sono necessarie correzioni)

Ovviamente se entrambi i casi restituiscono che il minimo di flip è ∞ si conclude che non è possibile ottenere una matrice M' *error free* che ammetta bipartizione. Eventuali colonne di soli "-" vengono ignorate.
In base a quanto detto possiamo dire che vale la *sottostruttura ottima*. Questo è dovuto al fatto che di procede partendo da $D(1, j-1)$, che quindi viene assunto ottimo, per il calcolo di $D(1, j)$, che o lascia uguale $D(1, j-1)$ o lo incrementa di uno (l'incremento di ∞ invece segnalerebbe l'impossibilità di avere un ottimo).

Si noti che che l'algoritmo tiene conto delle ipotetiche correzioni, tenendo appunto conto del minor numero di correzioni con al più un flip per colonna, senza però effettuarle veramente, limitandosi a calcolare il numero minimo di correzioni. È comunque una bozza è non ho dimostrazioni tangibili del fatto che effettivamente funzioni ma sembra essere comunque un punto di partenza accettabile. D'altro canto la ricostruzione effettiva della matrice M' è un problema non banale che deve tenere conto di più branch possibili di correzione e non viene analizzato in questo assignment.

Una volta ottenuta la matrice M' di n frammenti di aplotipi *error-free* possiamo ridurre il problema della ricerca della bipartizione in un problema sui grafi:

- **input**: una matrice M' di n frammenti di aplotipi *error-free* che si è visto ammettere bipartizione. Si chiami $F = \{f_1, f_2, \ldots, f_n\}$ l'insieme delle righe, ovvero l'insieme degli n frammenti

- **output**: un grafo bipartito, non orientato, indicato con $G = (V, E)$ e detto **grafo dei conflitti**, dal quale si estraggono le due partizioni, ovvero H_1 e H_2

La costruzione di tale grafo è la seguente:

- l'insieme dei nodi del grafo è formato dall'insieme dei frammenti, ovvero $V = F$
- l'insieme degli archi del grafo è formato dagli archi che collegano frammenti di F in conflitto, ovvero, presi due nodi f' e f'' (che ricordiamo rappresentano anche due frammenti):

$$(f', f'') \in E \iff f' \text{ in conflitto con } f''$$

Una volta costruito il grafo si hanno algoritmi polinomiali per la verifica della bipartizione e per l'identificazione della stessa. Si ottengono quindi, qualora esistano, i due aplotipi H_1 e H_2.

Ricapitolando l'intero problema:

1. **input**: una matrice M di n frammenti di aplotipi, eventualmente non *error-free*, come già definita
2. si effettua, se possibile, la correzione di M in M' tramite *1-cMEC*
3. si costruisce l'eventuale grafo dei conflitti $G = (V, E)$, a partire da M', come definito sopra. Si noti che, qualora il grafo esista, esso ammette bipartizione in quanto costruito ottenuto da una matrice *error-free* che ammette bipartizione
4. **output:** l'eventuale bipartizione nei due aplotipi H_1 e H_2, calcolata a partire da G. Qualora la matrice non possa essere corretta con 1-cMEC sicuramente non si avrà bipartizione per le ipotesi fatte e quindi l'algoritmo segnalerà l'errore

Come discusso a lezione, si potrebbe ragionare anche in ottica di grafo dei conflitti per poter procedere con la correzione degli errori. Ipoteticamente, una volta costruito il grafo $G = (V, E)$ a partire dalla matrice non error-free *in input, qualora non ammetta bipartizione, si potrebbe procedere con un algoritmo che elimina determinati archi al fine di ottenere una bipartizione. Tale soluzione non viene però qui approfondita ma una prima analisi sembra portare a pensare che anche questa soluzione sia, almeno in parte, esponenziale.*