

Dove finisce una lettera? I codici a prefisso

Informatica · scuola secondaria di primo grado · 11-14 anni

Un messaggio fatto soltanto di 0 e 1 non contiene necessariamente spazi tra una lettera e l'altra. Come possiamo leggerlo senza indovinare? La risposta dipende dalle regole con cui lo abbiamo scritto, non dalla nostra abilità nel riconoscere parole italiane.

Revisione docente e prova in classe: aperte

Il problema: due messaggi, gli stessi bit

Immagina questa tabella inventata:

Lettera	Codice
A	0
B	01
C	1

Per scrivere un messaggio sostituiamo ogni lettera con il suo codice e attacchiamo i codici uno dopo l'altro. La lettera B diventa 01. Ma anche il messaggio AC diventa 01: 0 per A, seguito da 1 per C. Il destinatario non può sapere quale dei due messaggi sia stato inviato. Abbiamo trovato un'ambiguità concreta.

Non basta quindi che ogni lettera abbia un codice diverso. Dobbiamo controllare anche che cosa succede quando i codici vengono concatenati, cioè uniti in una sola sequenza. [1, «Variable-length codes»; 2, p. 2, «Uniquely Decodable & Instantaneous Codes»]

Tre parole utili prima di cominciare

Un **bit** può assumere il valore 0 oppure 1. Qui lo rappresentiamo scrivendo una cifra sul foglio. Una **parola-codice** è la sequenza assegnata a un singolo simbolo: per esempio 110. Un **prefisso** è una parte iniziale di una sequenza: 1 e 11 sono prefissi di 110, mentre 10 non lo è.

Il termine «parola-codice» non indica una parola italiana: è il nome tecnico di un blocco di bit. Anche A, B, C e D sono soltanto i quattro simboli del nostro piccolo alfabeto; non stiamo descrivendo il modo in cui un computer memorizza tutto l'italiano.

Una tabella che non lascia dubbi sui confini

Da ora in poi, salvo diversa indicazione, usiamo questa tabella:

Simbolo	Parola-codice	Quanti bit
A	0	1
B	10	2
C	110	3
D	111	3

Nessuna parola-codice è l'inizio di un'altra parola-codice distinta. Questo è il significato di **codice a prefisso**, chiamato anche prefix-free: un codice «libero da prefissi» tra le sue parole-codice. Il nome italiano può trarre in inganno: non significa che dobbiamo aggiungere un prefisso al messaggio. [1, «Variable-length codes»; 2, p. 3, «A Check for Instantaneous Codes»]

Controlliamolo insieme. 0 non è l'inizio di 10, 110 o 111, perché queste iniziano con 1. 10 non è l'inizio di 110 o 111, perché il secondo bit è diverso. Infine 110 e 111 sono distinti e hanno la stessa lunghezza: nessuno può essere un inizio più corto dell'altro.

Codificare: dalla tabella alla striscia

Per scrivere **CAB**, sostituiamo una lettera alla volta:

C → 110 · A → 0 · B → 10

La striscia da consegnare è **110010**. Possiamo scrivere 110 | 0 | 10 durante il ragionamento, ma le barrette servono soltanto a mostrare i passaggi: non fanno parte del messaggio trasmesso.

La tabella deve essere nota sia a chi scrive sia a chi legge. Inoltre dobbiamo sapere dove comincia e dove finisce la striscia. Nel laboratorio questi confini coincidono con il primo e l'ultimo bit sul foglio. Non stiamo risolvendo il problema di riconoscere l'inizio di un messaggio dentro una trasmissione continua sconosciuta. [2, p. 2, «What Codes are Decodable?»]

Decodificare: un bit alla volta, da sinistra

Segui questa procedura:

1. Parti dal primo bit non ancora letto e raccogli i bit da sinistra a destra.
2. Appena il gruppo raccolto coincide con una parola-codice della tabella, scrivi il simbolo corrispondente.
3. Riparti dal bit successivo, con il gruppo vuoto.
4. Alla fine controlla di aver consumato tutti i bit senza lasciare un gruppo incompleto.

Proviamo con 110010.

Bit raccolti	Che cosa so?	Azione
1	Non è ancora un codice completo.	Continuo.
11	Non è ancora un codice completo.	Continuo.
110	È C.	Scrivo C e riparto.
0	È A.	Scrivo A e riparto.
1	Non è ancora un codice completo.	Continuo.
10	È B.	Scrivo B; non restano bit.

Otteniamo **CAB**. Perché possiamo fermarci appena troviamo un codice completo? Perché quel codice non può essere l'inizio di un altro codice più lungo. Non dobbiamo aspettare il simbolo successivo per decidere quale lettera abbiamo appena riconosciuto. Questa proprietà viene chiamata anche decodifica **istantanea**: riguarda l'assenza di attesa per i bit della lettera seguente, non una velocità fisica infinita. [2, pp. 2-3, «Uniquely Decodable & Instantaneous Codes» e «A Check for Instantaneous Codes»]

Perché il risultato è unico

Supponiamo che due letture dello stesso messaggio diano una prima lettera diversa. Le due parole-codice partirebbero dallo stesso punto della striscia. Se avessero la stessa lunghezza dovrebbero coincidere; se una fosse più corta, sarebbe il prefisso dell'altra. La prima possibilità non dà lettere diverse nella nostra tabella; la seconda è vietata dalla regola. La prima lettera è dunque determinata. Ripetendo il ragionamento sui bit rimasti, sono determinate anche le successive.

Questo vale per una sequenza completa e corretta, costruita con la tabella concordata. Non significa che qualsiasi fila casuale di bit rappresenti un messaggio completo. Per esempio 11, con la nostra tabella, finisce troppo presto: potrebbe completarsi in 110 oppure 111, ma non possiamo inventare il bit mancante.

Tre cose da non confondere

Leggibilità e segretezza. Conoscere la tabella basta a leggere il messaggio: questa attività non è un metodo per tenere segreti i dati. Non chiameremo «password» o «cifatura sicura» i nostri esempi.

Leggibilità e controllo degli errori. Un bit cambiato può trasformare un messaggio valido in un altro messaggio valido. Per esempio 0 è A, mentre 10 è B: la perdita del primo bit di B può produrre A senza segnalare da sola l'errore. Un codice a prefisso non garantisce rilevazione o correzione degli errori. [2, p. 1, «Lossless Data Compression», distinzione fra codifica della sorgente e del canale]

Leggibilità e risparmio di spazio. Il nostro obiettivo è capire dove finiscono le parole-codice, non costruire il file più corto. Non scegliamo le lunghezze in base alla frequenza dei simboli e non studiamo l'algoritmo di Huffman. Una parola-codice breve non dimostra che l'intero messaggio occupi meno spazio: per un confronto reale dovremmo specificare anche il formato e come viene comunicata la tabella.

Anche i codici tutti della stessa lunghezza possono andare bene

Con P=00, Q=01, R=10, S=11, basta dividere il messaggio in coppie, partendo dal confine iniziale noto. Le quattro parole-codice hanno tutte due bit e sono distinte: nessuna è l'inizio più corto di un'altra. Anche questo è un codice a prefisso. Non è necessario che le lunghezze siano diverse. [2, p. 3, «More Examples», Code E]

Un'attenzione in più: «non a prefisso» non è una condanna

La proprietà a prefisso è una garanzia sufficiente per la decodifica immediata da sinistra. Non è l'unico modo possibile per rendere un codice univocamente decodificabile. Se una tabella non supera il controllo dei prefissi, non possiamo concludere automaticamente che esistano due messaggi uguali in bit: per dichiarare l'ambiguità dobbiamo dimostrarla, come abbiamo fatto con B e AC all'inizio. Esistono codici univocamente decodificabili che richiedono di aspettare altri bit; non li classifichiamo in questa lezione. [1, esercizio 39; 2, p. 2, «Examples», Code C e Code D]

Laboratorio: il decodificatore umano

Occorrente: foglio, matita e tabella A-D. Nessun account, dispositivo o dato personale. Lavoro a coppie, con possibilità di svolgimento individuale.

Una persona scrive un messaggio inventato di quattro o cinque simboli, tutti scelti tra A, B, C e D. Lo conserva coperto, codifica la sequenza e consegna al compagno soltanto la striscia senza spazi, insieme alla tabella. Il destinatario segna le barrette durante la lettura e ricostruisce i simboli. Confrontate poi il risultato con l'originale, una posizione alla volta, e scambiatevi i ruoli.

Non serve ottenere una parola con significato: **DACA** va bene quanto **ABCD**. Indovinare dal senso nasconderebbe proprio il problema che vogliamo studiare. Se qualcosa non torna, conservate la striscia originale e controllate prima la codifica, poi i confini segnati nella lettura.

Per lavorare con più calma: tenete la tabella visibile, usate caratteri grandi e coprite i bit non ancora letti con un foglio. Il laboratorio non dipende da colori, udito o rapidità. Le barrette di lavoro aiutano a tenere il segno senza diventare separatori della striscia trasmessa.

Esercizi

Per ciascun esercizio, mostra i passaggi: non basta riportare il risultato.

1. Codifica e decodifica guidata

Usa **A=0, B=10, C=110, D=111**. Codifica **BAD**. Poi decodifica **0110110**. Segna le divisioni in parole-codice.

2. Un messaggio e la verifica

Con la stessa tabella, codifica **DACA** e decodifica la sequenza ottenuta senza guardare la parola di partenza. Indica perché la lettura da sinistra non lascia alternative.

3. Un errore da correggere

Con la tabella **A=0, B=10, C=110, D=111**, Luca deve decodificare **10110**. Scrive: "10 è B; poi leggo 11, ma 11 non è un codice: allora il messaggio è impossibile". Individua l'errore e completa la decodifica corretta.

4. Non a prefisso: ambiguità concreta

Considera la tabella **A=0, B=01, C=1**. Qui una parola-codice è prefisso di un'altra. Mostra due parole diverse che producono entrambe la sequenza **01**. Spiega perché questo è un esempio effettivo di ambiguità, non soltanto un codice non a prefisso.

5. Attenzione alla conclusione troppo rapida

Considera **X=0, Y=01**. Il codice non è a prefisso perché 0 è prefisso di 01. Tuttavia, usando solo X e Y e concatenando i codici, **01** ha una sola decodifica possibile. Verifica questa affermazione e spiega perché non è corretto dichiarare ambiguo un singolo messaggio soltanto guardando il prefisso. Qui controlliamo una sola striscia, non dimostriamo una proprietà generale del codice.

6. Caso a lunghezza fissa

Usa la tabella **P=00, Q=01, R=10, S=11**. Codifica **RPS** e decodifica **010011**. Perché è naturale dividere la sequenza in coppie? Spiega perché questo codice è anche a prefisso, pur non avendo codici di lunghezza diversa.

Soluzioni commentate

1. BAD = B | A | D = **10 | 0 | 111**, quindi **100111**. Per decodificare 0110110 si legge: 0 | 110 | 110, cioè **A | C | C**. Dopo lo 0, il bit successivo inizia una nuova parola-codice; 110 è completo e non può essere l'inizio di un codice più lungo nella tabella.

2. DACA = 111 | 0 | 110 | 0 = **11101100**. Rileggendo da sinistra si ottiene 111 (D), 0 (A), 110 (C), 0 (A): DACA. Non ci sono codici che iniziano con un altro codice: ogni volta che la lettura raggiunge un codice completo, nessuna alternativa può prolungarlo.

3. Dopo aver letto 10 (B), la parte ancora da leggere è **110**. Il gruppo 11 non è ancora un codice completo: bisogna proseguire. La scansione corretta è **10 | 110**, cioè **B | C**. L'errore è aver separato la coda prima di controllare se il bit successivo completa un codice valido.

4. La parola **B** si codifica 01. Anche **AC** si codifica 0 | 1 = 01. Due parole diverse producono la stessa sequenza, quindi la decodifica è davvero ambigua. B=01 comincia con il codice A=0: il codice non è a prefisso.

5. La sequenza **01** può essere Y, con il codice 01. Se invece proviamo a partire da X, consumiamo lo 0 e lasciamo 1, che non è un codice della tabella. Quindi **01** si decodifica soltanto come Y. Il codice non è a prefisso, ma questa sequenza non è ambigua. Questo controllo su una striscia non dimostra da solo che tutte le strisce generate dal codice abbiano una sola lettura: per una conclusione generale servirebbe un ragionamento ulteriore.

6. RPS = 10 | 00 | 11 = **100011**. La sequenza proposta **010011** si divide in 01 | 00 | 11, ossia **QPS**. I codici hanno tutti due bit, dunque i confini sono ogni due bit; nessuna parola-codice di due bit può essere l'inizio di un'altra parola-codice distinta di due bit. Quindi è a prefisso. Nota: 010011 non è la codifica di RPS, che è 100011.

Per chi insegna: obiettivi e conduzione

Prerequisiti: riconoscere 0 e 1, leggere una tabella e seguire una sequenza da sinistra a destra. Non sono richieste conversioni fra basi, programmazione o potenze.

Al termine, l'alunno dovrebbe saper codificare un breve messaggio; decodificare mostrando i confini; individuare un prefisso tra due parole-codice; esibire due messaggi diversi quando sostiene che un codice è ambiguo; distinguere decodifica, segretezza e controllo degli errori.

Una possibile ora di lavoro: 5 minuti sul messaggio ambiguo iniziale, 10 sulle regole e sulla lettura guidata, 15 sul laboratorio, 15 sugli esercizi 1-4, 10 sul confronto con soluzioni e caso a lunghezza fissa, 5 per la risposta di uscita. L'approfondimento sul codice non a prefisso può essere rinviato.

Risposta di uscita: «Perché con la tabella A-D posso fermarmi appena riconosco una parola-codice?». Una risposta adeguata collega la fine del blocco all'assenza di altre parole-codice che lo prolunghino; non si limita a «perché ho imparato la tabella».

Criterio di controllo: valutare soprattutto il procedimento. Una striscia finale corretta senza passaggi non mostra ancora come lo studente abbia riconosciuto i confini. Errori di copia e difficoltà concettuali vanno distinti nel confronto orale o scritto.

Fonti per approfondire

[1] Robert Sedgewick e Kevin Wayne, Algorithms, 4ª edizione, Addison-Wesley, 2011; sito del libro, §5.5 «Data Compression». URL: <https://algs4.cs.princeton.edu/55compression/> . Localizzatori: sottosezione «**Variable-length codes**», passaggio «no string is a prefix of another», per la definizione; **esercizio 39, «Uniquely decodable code**», per l'esistenza di codici univocamente decodificabili non a prefisso. Usiamo questi passaggi, non le indicazioni storiche o le sintesi sui formati della pagina.

[2] Sam Roweis, Lecture 2: Uniquely Decodable and Instantaneous Codes, CSC310, 15 settembre 2005, copia conservata sul sito della New York University. PDF: https://cs.nyu.edu/home/people/in_memoriam/roweis/csc310-2005/notes/lec2x.pdf . Localizzatori per **pagine fisiche del PDF**: **p. 1**, «Encoding One Symbol at a Time» e «Lossless Data Compression»; **p. 2**, «What Codes are Decodable?», «Uniquely Decodable & Instantaneous Codes», «Examples»; **p. 3**, «More Examples» e «A Check for Instantaneous Codes». Le pagine contengono più diapositive: i titoli permettono di individuare il passaggio senza confonderle con una numerazione di slide.

Le due fonti hanno autori e progetti editoriali distinti. La spiegazione, le consegne e i messaggi del laboratorio sono rielaborati per la scuola media; le tabelle rappresentano piccoli esempi matematici, non uno standard di codifica del testo.

Revisione docente e prova in classe: aperte