a cura di Francesco Romani

Risoluzione Automatica di Parole Crociate Crittografate

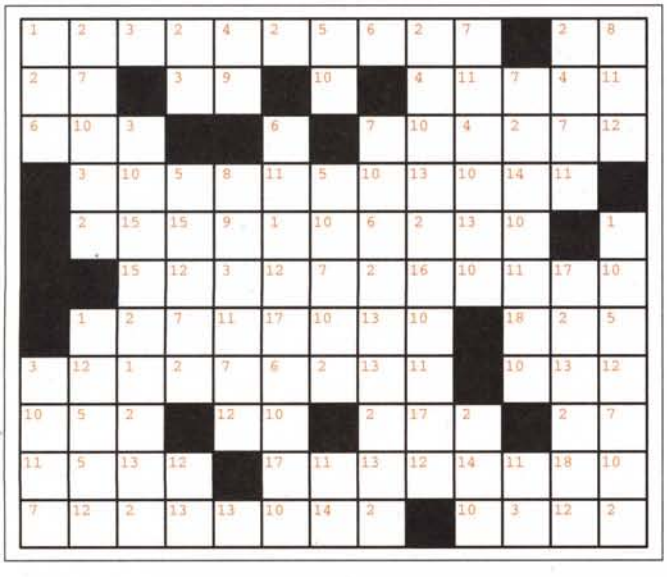
Nella puntata precedente abbiamo visto come risolvere un problema *di parole crociate crittografate* in modo manuale utilizzando la macchina solo come ausilio grafico, al posto della penna e della rivista. In questo articolo cerchiamo di essere più bravi e proponiamo un metodo di soluzione automatica basato sull'uso di un dizionario.

di Federico Curcio e Francesco Romani

Il problema

Il problema seguente è comparso nel numero 3431 de *La Settimana Enigmistica*. A numero uguale corrisponde lettera uguale e il lettore deve completare lo schema sfruttando un suggerimento.

Siccome vogliamo essere **molto** bravi, stavolta abbiamo ignorato il suggerimento.



Mentre risolviamo uno schema di parole crociate crittografate diventiamo un po' crittanalisti perché la soluzione di tale gioco consiste proprio nel risolvere - forzare, nel gergo della Crittologia - un codice ideato dal redattore dello schema. Dal momento che l'associazione numeri-caratteri - come dice la didascalia del gioco - fa sempre corrispondere a numero uguale lettera uguale, ci troviamo a forzare un codice monoalfabetico.

Non utilizzeremo l'attacco classico ai codici monoalfabetici, cioè non sfrutteremo il fatto che ogni lingua ha una propria distribuzione statistica delle occorrenze di caratteri singoli o gruppi di essi; per chiarire, accade, ad esempio, che in Italiano i caratteri E, I, A in un testo tendano a presentarsi ognuno con una frequenza pari all'11%, così come è più facile che in un testo vi sia in totale una maggioranza di L rispetto alle R e così via, dando un notevole aiuto a chiunque voglia ricostruire il testo originale.

L'attacco che sferreremo sarà differente perché negli schemi che appaiono sulle riviste di enigmistica non tutte le distribuzioni statistiche vengono rispettate, sia a causa della limitatezza del campione di caratteri coinvolti in ogni schema, sia per rendere più difficile il gioco. Cercheremo sequenze di numeri che presentino un pattern al quale corrisponda il minor numero di voci possibile. Per spiegare cosa si intende per pattern consideriamo la parola:

PACCHETTO

Essa è formata da 7 caratteri diversi (A,C,E,H,O,P,T) ai quali associamo un numero, secondo l'ordine con cui incontriamo i caratteri stessi: P=1, A=2, C=3, H=4, E=5, T=6, O=7; il pattern per la nostra parola risulta essere:

[1 , 2 , 3 , 3 , 4 , 5 , 6 , 6 , 7]

La parola PACCHETTO non è l'unica ad avere questo *pattern*: ad esempio parole come DIFFRATTO, RICCHEZZA e TACCHEGGI hanno il medesimo *pattern* (sono lunghe nove caratteri dei quali 7 diversi fra loro e disposti come indicato). In effetti la ricerca di voci con il *pattern* visto darebbe anche SOFFRIGGE e SUPPLIMMO, fra le altre, ma essendo flessioni verbali diverse dall'infinito presente e dal participio passato non è possibile trovarle in uno schema (classico) di parole crociate, crittografate o meno.

Per poter affrontare con successo la risoluzione automatica delle parole crociate crittografate deve essere certa l'unicità della soluzione dello schema proposto. Tale condizione permette di ridurre la ricerca (una volta individuata una soluzione non bisogna proseguire per individuarne un'altra) e - nel caso non si terminasse lo schema - ci informa dell'eventuale incompletezza del dizionario consultato.

A proposito del dizionario da utilizzare è bene sottolineare come un buon dizionario enigmistico sia un ibrido fra un vocabolario e un'enciclopedia, arricchito di locuzioni e sigle, ripulito delle voci inaccettabili (come le parolacce). Inoltre è da considerarsi come una vera e propria creatura in continua evoluzione, con inevitabili perdite di voci ormai desuete (p. es. PANINARO) e frequenti ingressi o cambiamenti (neologismi, nuovi personaggi, nuove denominazioni geografiche). La manutenzione di tale oggetto richiede impegno e ricerche continui, con l'ausilio di staff specializzato (il cui lavoro non viene reso di dominio pubblico né messo in commercio - se non in versioni ridotte - in quanto risorsa vitale per gli editori di periodici enigmistici).

Creazione del pattern

Vediamo una funzione per costruire il *pattern* in *Mathematica*:

```
In[1]:=
sign[x_]:=x/.Flatten[
 Rule@@#&/@Transpose[{x,Range[Length[x]]}]]
```

Per capire come funziona, proviamola pezzo per pezzo. L'argomento di ingresso deve essere una lista di caratteri:

```
In[2]:=
x=Characters["PACCHETTO"]

Out[2]=
{P, A, C, C, H, E, T, T, O}
```

Si accoppia ogni carattere con la sua posizione:

```
In[3]:=
Transpose[{x,Range[Length[x]]}]

Out[3]=
```

L'affare Superenalotto

Nel mese di gennaio 1998 sono stati vinti circa 13 miliardi al Superenalotto. La cosa più curiosa è che proprio il giorno precedente vi erano state forti polemiche, anche da parte di fonti autorevoli, che affermavano che a quel gioco non si poteva vincere. Vediamo con i nostri soliti mezzi qual è la probabilità che in una data estrazione vi sia **almeno** un vincitore del premio maggiore.

Il gioco consiste nell'indovinare i primi estratti delle sei ruote di Bari, Firenze, Milano, Napoli, Palermo, Roma. Se il primo estratto di una ruota è uguale al primo estratto di una delle precedenti, subentra il secondo estratto e così via.

Se nessuno ha indovinato i 6 numeri, il primo estratto della ruota di Venezia funge da Jolly potendo sostituire uno qualsiasi dei numeri non indovinati.

Considerando che l'ordine dei numeri non conta vi sono 90*89*88*87*86*85/6! combinazioni giocabili:

```
In[1]:=
ncomb = 90 89 88 87 86 85 /6!

Out[1]=
622614630
```

Definiamo vincita possibile il fatto che una combinazione giocata abbia 6 numeri in comune con i sette estratti, indifferentemente dall'ordine. In realtà una vincita possibile diventa una vincita vera se i numeri indovinati sono quelli delle 6 ruote primarie o se nessuno ha conseguito una vincita, di questo tipo e si può utilizzare il Jolly. Poiché siamo interessati alla probabilità di avere "almeno" una vincita possiamo studiare solo la possibilità che vi sia almeno una vincita possibile. In questo caso per ogni combinazione vi sono 7 possibilità di vittoria, quella giocata più le 6 sostituzioni della ruota di Venezia. La probabilità di "vincita possibile" con una giocata è quindi `7/ncomb` = 1.12429... 10^{-8}.

Se tutti i giocatori si mettessero d'accordo per giocare un colossale sistema, basterebbe giocare circa 90 milioni di combinazioni perché almeno un giocatore vinca con certezza il montepremi.

Supponendo invece che tutte le colonne giocate siano generate dai vari giocatori estraendole in modo casuale ed indipendente, perché nessuno vinca bisogna che tutte le `n` colonne giocate non azzecchino nessuna delle 7 possibilità. Tale probabilità vale `(1-7/ncomb)^n`.

In genere i sistemisti giocano molte colonne tutte diverse

Continua a pag. 296

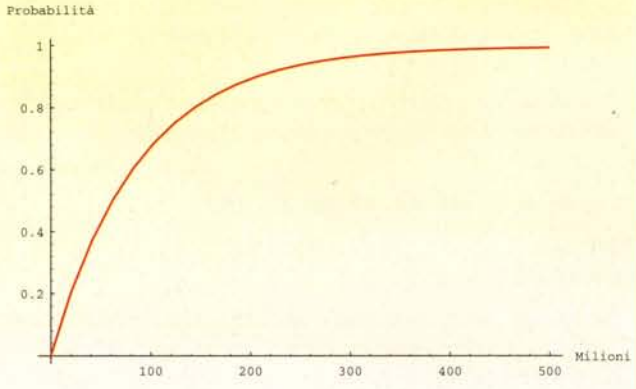
Segue da pag. 295

quindi le ripetizioni di colonne sono meno frequenti di quanto previsto da questo modello e la probabilità che ci sia almeno un vincitore è maggiore di:

In[2]:=
```
p[n_]:= 1 - (1-7/ncomb)^n
```

Vediamo questa funzione:

In[3]:=
```
Plot[p[1000000 x],{x,0,500},
  PlotStyle->Red,
  AxesLabel->{"Milioni",
              "Probabilità"}];
```



Dal grafico è evidente che, se le combinazioni giocate sono pochi milioni la vincita del premio grosso è difficile, ma mano a mano che il premio si accumula è verosimile che molti milioni di persone, attratti dalla enorme posta in palio, giochino un gran numero di combinazioni (spesso organizzate in sistemi) e qualcuno vinca.

Dimenticavamo! Siccome solo una parte delle puntate viene messa in palio, in questo gioco (come nel Totocalcio) è matematico che quello che vince sempre è lo Stato!

```
{{"P", 1}, {"A", 2}, {"C", 3}, {"C", 4},
{"H", 5}, {"E", 6}, {"T", 7}, {"T", 8},
{"O", 9}}
```

... e si trasforma il tutto in una lista di regole:

In[4]:=
```
Rule@@#&/@%
```

Out[4]=
```
{P -> 1, A -> 2, C -> 3, C -> 4, H -> 5, E
-> 6, T -> 7, T -> 8, O -> 9}
```

... che si applicano all'insieme di partenza:

In[5]:=
```
x/.Flatten[%]
```

Out[5]=
```
{1, 2, 3, 3, 5, 6, 7, 7, 9}
```

Si noti che nel *pattern* non compaiono le cifre 5 e 8 perché in posizione 5 e 8 c'erano lettere già presenti in precedenza, quindi il *pattern* risultante non è esattamente quello che avevamo introdotto prima, ma il tutto funziona lo stesso perché parole con la stessa struttura generano lo stesso *pattern*:

In[6]:=
```
sign[Characters["DIFFRATTO"]]
```

Out[6]=
```
{1, 2, 3, 3, 5, 6, 7, 7, 9}
```

Trattamento del dizionario

Supponiamo di avere un "ricco" dizionario di parole (magari proprio di quelle enigmisticamente significative), eleggiamolo in una lista **lemmi**. Per fare un "piccolo" esempio supponiamo che **lemmi** sia una "piccola" lista di parole di 5 lettere:

In[1]:=
```
lemmi=ReadList["lemmi", String]
```

Out[1]=
```
{"CACCA", "COCCO", "MAMMA", "NANNA", "NONNO",
"PAPPA", "BABBO", "COCCA", "LILLA", "NINNA",
"NONNA", "POPPA", "SASSO", "SESSO", "TATTO",
"TETTA", "TETTO", "TUTTO", "ZOZZA", "ARARE",
"CACAO", "AGATA", "ALATA", "AMACA", "AMARA",
"ARABA", "AVANA", "AVARA", "EBETE", "EDERE",
"EREDE", "ETERE", "INIZI", "IRITI", "OBOLO",
"OVOLO", "OZONO", "GIGLI", "ABACO", "ABATE",
"ABATI", "ACARI", "ACARO", "AGAPE", "AGATE",
"ALANI", "ALARE", "ALATE", "ALATI", "ALATO",
"AMARE", "AMARI", "AMARO", "AMATO", "ARABE",
"ARABI", "ARABO", "AVARO", "CACHI", "CACIO",
"EBETI", "EDEMA", "EDERA", "EREDI", "EREMI",
"EREMO", "ETERI", "GOGNA", "IRIDE", "IRITE",
"MAMBO", "NENIA", "NINFA", "ODORE", "ONORE",
"PEPLO", "PEPSI", "SOSIA", "USURA"}
```

La funzione **dodiz[s]** aggiunge **s** alla lista dei lemmi con lo stesso *pattern* di **s**.

In[2]:=
```
dodiz[s_String]:=
```

```
AppendTo[diz[sign[Characters[s]]],s]
```

Applichiamola a tutto il dizionario e vediamo cosa succede:

```
In[3]:=
Clear[diz];
diz[___]:={};
Scan[dodiz,lemmi];
?diz

Out[3]=
diz[{1, 2, 1, 1, 2}] =
{"CACCA", "COCCO", "MAMMA", "NANNA", "NONNO", "PAPPA"}
diz[{1, 2, 1, 1, 5}] =
 {"BABBO", "COCCA", "LILLA", "NINNA", "NONNA", "POPPA", "SASSO", "SESSO", "TATTO",
"TETTA", "TETTO", "TUTTO", "ZOZZA"}
diz[{1, 2, 1, 2, 5}] =
 {"ARARE", "CACAO"}
diz[{1, 2, 1, 4, 1}] =
 {"AGATA", "ALATA", "AMACA", "AMARA", "ARABA", "AVANA", "AVARA", "EBETE", "EDERE",
"EREDE", "ETERE", "INIZI", "IRITI", "OBOLO", "OVOLO", "OZONO"}
diz[{1, 2, 1, 4, 2}] =
 {"GIGLI"}
diz[{1, 2, 1, 4, 5}] =
 {"ABACO", "ABATE", "ABATI", "ACARI", "ACARO",
"AGAPE", "AGATE", "ALANI", "ALARE", "ALATE",
"ALATI", "ALATO", "AMARE", "AMARI", "AMARO",
"AMATO", "ARABE", "ARABI", "ARABO", "AVARO",
"CACHI", "CACIO", "EBETI", "EDEMA", "EDERA",
"EREDI", "EREMI", "EREMO", "ETERI", "GOGNA",
"IRIDE", "IRITE", "MAMBO", "NENIA", "NINFA",
"ODORE", "ONORE", "PEPLO", "PEPSI", "SOSIA",
"USURA"}
diz[___] := {}
```

Trattando nello stesso modo un dizionario più completo, è possibile avere subito la lista di tutte le parole con un certo *pattern*.

Ricerca della soluzione

La funzione **str** scorre la matrice **A** per righe isolando le parole tra le caselle nere. Applicando **str** alla trasposta di **A** si fa lo stesso per le colonne. Mettendo tutto insieme si ottengono i vincoli del problema:

```
In[1]:=
str[x_]:=(
  AF=Flatten[Append[#,"*"]&/@x];
pas=Flatten[Position[AF,"*"]];
pas1=Transpose[{Prepend[Drop[pas,1],0]+1,pas-1}];
  Select[AF[[Range@@#]]&/@pas1,
      ((Length[#]>2)&&(! TrueQ[And@@LetterQ/@#]))&])
vincoli:=(or=str[A];
          ver=str[Transpose[A]];
          Union[or,ver])
```

Fin qui tutto come nella puntata precedente. Ora selezioniamo i vincoli lunghi almeno 9 caratteri:

```
In[2]:=
v=Select[vincoli,Length[#]>=9&];
%//ColumnForm

Out[2]=
{3, 10, 15, 15, 2, 1, 2, 13, 2}
{3, 12, 1, 2, 7, 6, 2, 13, 11}
{6, 11, 1, 12, 17, 6, 10, 17, 10}
{7, 10, 6, 2, 13, 13, 2, 13, 2}
{1, 2, 3, 2, 4, 2, 5, 6, 2, 7}
{2, 4, 10, 13, 2, 16, 10, 11, 17, 12}
{2, 15, 15, 9, 1, 10, 6, 2, 13, 10}
{3, 10, 5, 8, 11, 5, 10, 13, 10, 14, 11}
{15, 12, 3, 12, 7, 2, 16, 10, 11, 17, 10}
```

La funzione **pair** costruisce un insieme di regole a partire da una lista di *pattern* e di soluzioni proposte:

```
In[3]:=
pair[a_,b_] :=
Union[Rule@@#&/@Transpose[{b,Characters[a]}]]
```

Costruiamo una regola di sostituzione cercando nel dizionario le parole che soddisfano i vincoli e adottando come buone quelle che presentano una sola possibilità:

```
In[4]:=
rule1[x_,y_]:=Union[Flatten[pair[#[[1,1]],#[[2]]]&/@

Select[Transpose[{x,y}],Length[#[[1]]]==1&]]]

In[4]:=
rule=rule1[diz/@sign/@((ToString/@#)&/@v),v]

Out[3]=
{1->M, 2->A, 3->D, 4->G, 5->S, 6->C, 7->R, 8->P, 10->I, 11->O, 12->E, 13->T, 14->V, 17->N}
```

Applicando queste regole al nostro schema si vede che passo da gigante abbiamo compiuto verso la soluzione:

```
In[5]:=
A=A/.rule;
showtab
```

Vedi Figura 2

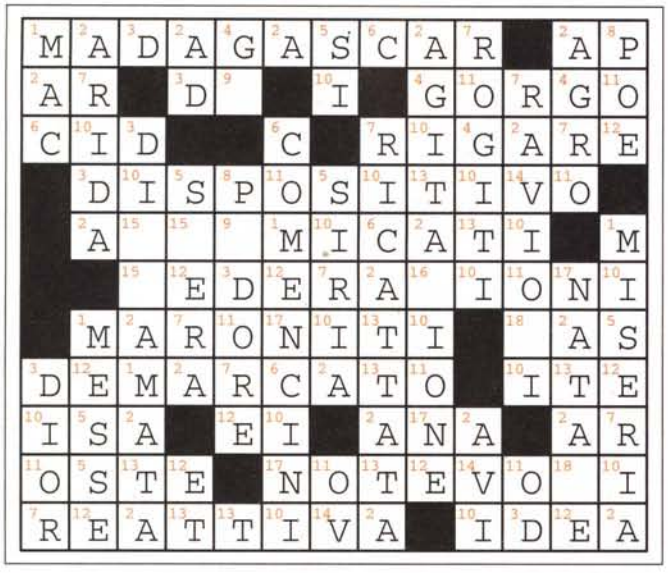


Figura 2

Tra i vincoli restanti ora selezioniamo quelli lunghi almeno 5 caratteri:

In[6]:=
```
(v=Select[vincoli,Length[#]>=5&])//ColumnForm
```

Out[6]=
```
{S, 15, E, R, A}
{N, A, T, A, 18, E}
{P, 9, D, O, R, E}
{R, A, V, I, O, 18, I}
{N, O, T, E, V, O, 18, I}
{D, I, 15, 15, A, M, A, T, A}
{A, 15, 15, 9, M, I, C, A, T, I}
{A, G, I, T, A, 16, I, O, N, E}
{15, E, D, E, R, A, 16, I, O, N, I}
```

La nuova regola di sostituzione si ottiene cercando tra le soluzioni proposte dal dizionario quelle che soddisfano anche le lettere già presenti:

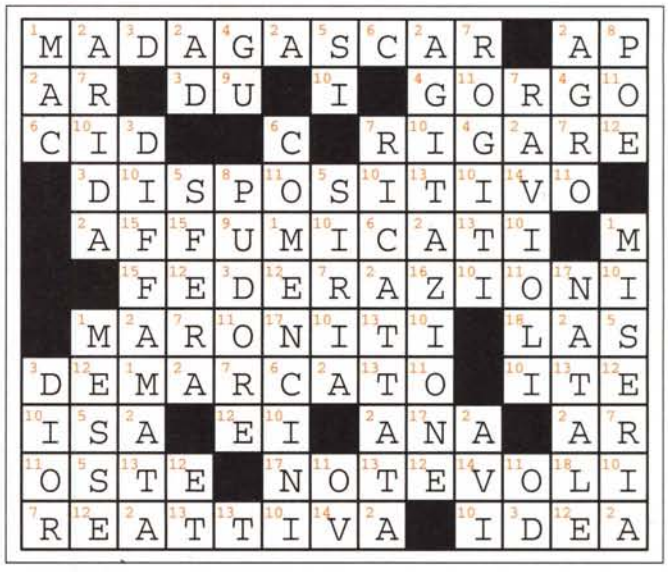
In[7]:=
```
match1[{x_?LetterQ,y_}]:=(x==y);
match1[{_?NumberQ,_}]:=True;
match[a_,b_]:=
  And@@match1/@
    Transpose[{a,Characters[b]}]
cerca[x_]:=Select[ddd=diz[sign[ToString/@#]&[x]],
  match[x,#]&]
rule2[x_]:=Select[x,NumberQ[#[[1]]]&]
```

In[8]:=
```
rule=rule2[rule1[cerca/@v,v]]
```

Out[8]=
```
{9->U, 15->F, 16->Z, 18->L}
```

Con questa sostituzione il problema è risolto. Si noti che la potenza di questo metodo (che toglie ogni soddisfazione al risolutore umano) sta tutta nella ricchezza del dizionario:

In[9]:=
```
A=A/.rule;
showtab
```



MC

Bibliografia

La Settimana Enigmistica, n. 3431, 27 Dicembre 1997, problema n. 3141, pag. 7.

CoFax Telematica®

da oltre 10 anni
al servizio della comunicazione "veloce".

I prodotti ISDN leader del mercato, la più vasta scelta di schede con supporto ISA, PCI, USB, PCMCIA, complete di driver per Windows NT, '95, Linux, Unix, Novell.

ZyXEL

Router, Modem e TA, per connessioni ISDN Internet ed Intranet per il Personal Computer e le reti locali.

VCON VIDEOCONFERENCING

Piattaforme di videoconferenza standard (H.320/H.323) per il Personal Computer, sale di videoconferenza, sistemi portatili ed OEM.

Per centrare i nostri obiettivi, ci siamo affidati ai migliori marchi internazionali.

La nostra gamma di prodotti è in grado di soddisfare tutte le esigenze di comunicazione veloce, in maniera pratica ed affidabile: connessioni LAN to LAN, connessioni PC to PC, accesso remoto a LAN IP/IPX ed Internet/Intranet, videocomunicazione.

CoFax® TELEMATICA

http://www.cofax.it

Roma - V.le dei Colli Portuensi, 110/a
Tel. 06/58201362 r.a.
Fax 06/58201550

Milano - C.so Buenos Aires, 37
Tel. 02/29526100 r.a.
Fax 02/29520884

ISDN world® by CoFax Telematica

ISDN® point

Primi a credere nell' ***ISDN***
Primi ad investire nell' ***ISDN***
Primi ad integrare il ***Networking*** *con l'* ***ISDN***

design by filax

Desidero ricevere maggiori informazioni sulla Vs. gamma di prodotti **ISDN**.
Vi prego di inviare caratteristiche e listini aggiornati al seguente indirizzo:

Nome

Cognome

Indirizzo

Città CAP

Con riferimento alle vigenti normative sulla riservatezza dei dati personali, Vi autorizzo ad utilizzare le informazioni contenute nel presente coupon per la sola finalità di essere aggiornato sulle Vs. iniziative commerciali.

Firma

barrare qui sotto se si desidera ricevere informazioni riservate ai S.ri Rivenditori

☐ sono un Rivenditore, inviatemi Listini ed Offerte Speciali
☐ sono un Rivenditore interessato al Vs. programma **ISDN Point.**

MC-MIC