Metodo per prove ed errori

Prerequisiti

- Conoscere le basi fondamentali del ragionamento logico.
- Conoscere e saper utilizzare i connettivi logici.
- Saper determinare il valore di verità di una proposizione logica.

Scopo

Un metodo di risoluzione per prove ed errori è generalmente considerato come "l'ultima spiaggia" a cui attingere nell'ambito del problem solving. Tuttavia, nel caso di problemi come quello mostrato nel video non vi è alternativa all'utilizzo di questa strategia.

Si tratta di un problema di logica piuttosto celebre: una versione dell'enigma del lupo, della capra e del cavolo.

Questa volta i protagonisti sono una volpe, un'oca e un sacco di granturco. Un contadino deve trasportare i due animali e il sacco con la sua barchetta attraverso il fiume. La barchetta può trasportare solo un animale o il sacco oltre al contadino. Inoltre non possono essere lasciati da soli né la volpe con l'oca, né l'oca con il granturco. Il problema richiede di stabilire quali sono i passaggi che permettono al contadino di passare da una parte all'altra del fiume con i due animali e il sacco di granturco.

Visualizzazione operativa: qualche proposta

Dopo aver mostrato la soluzione del video, è opportuno cercare di schematizzare la strategia risolutiva. Può essere utile tenere conto anche dei tentativi che non portano alla risoluzione del problema.

> «Organizzate uno schema, aiutandovi con frecce e disegni per rappresentare le prove eseguite nel video per risolvere il problema.
> Quale tentativo non è andato a buon fine? Quale tentativo è stato invece efficace?»

Nel video non sono state analizzate tutte le possibilità per quanto riguarda il primo passaggio.

> «Riuscite a trovare un'altra soluzione? Quale tentativo non è stato effettuato? »

Il tentativo iniziale non eseguito è quello di trasportare per primo il sacco di granturco. Ciò però permette alla volpe di rimanere incustodita con l'oca; si può concludere che questo tentativo non va a buon fine. Non ci sono altre possibilità per il primo passaggio.

> «Vi sono altre possibilità per il secondo passaggio? E per il terzo passaggio? »

C'è soltanto un'altra soluzione, del tutto analoga a quella considerata nel video, che prevede di scambiare le posizioni della volpe con quella del sacco di granturco.

Approfondimenti

Una variante del problema analizzato è quella del celebre enigma "Missionari e Cannibali". Questo tipo di problema può essere schematizzato come segue.

N missionari e *N* cannibali si trovano sulla sponda di un fiume. Vi è solo una barchetta che permette a tutti i personaggi di attraversare il corso d'acqua, ma la barchetta può trasportare solo *M* persone alla volta. Sapendo che il numero di cannibali non può mai superare il numero di missionari e che la barchetta non può mai essere vuota, in quante mosse tutti i personaggi possono passare sull'altra sponda del fiume?

Al variare dei due parametri *N* e *M* si ottengono diverse varianti dello stesso problema. L'origine di questo enigma si può far risalire al VII-VIII secolo d.C.: la prima testimonianza di un simile problema, che vedeva in azione coppie di fratelli e sorelle, si può trovare nelle *Propositiones ad juvenes acuendos*, attribuita ad Alcuino, filosofo e teologo anglosassone protagonista del Rinascimento carolingio.

L'enigma che viene proposto nella prossima sezione è di questo tipo (con $N = 3$, $M = 2$).

Struttura del video	Tempo
testo del problema	1:23
primo tentativo	1:27
secondo tentativo	1:37

Altri problemi

1. *Tre sciacalli e tre coyote devono attraversare un fiume su una barchetta che può contenere solo due animali per volta. Il numero degli sciacalli non può mai superare quello dei coyote, e ci deve sempre essere un animale sulla barchetta.*

Prerequisiti

- Conoscere le basi fondamentali del ragionamento logico.
- Conoscere e saper utilizzare i connettivi logici.
- Saper determinare il valore di verità di una proposizione logica.

Commenti e soluzione

Vi sono 4 possibili soluzioni in 11 mosse.

Tutte le soluzioni devono prevedere un numero di mosse dispari, perché il percorso si conclude sempre con un "arrivo".

Le soluzioni sono riassunte nelle tabelle che seguono (s = sciacallo, c = coyote).

Si può notare che:

- in tutte le soluzioni, le mosse da 3 a 9 sono uguali (quelle che cambiano sono invece le prime due o le ultime due)
- le soluzioni hanno i primi due passaggi uguali a coppie
- le soluzioni hanno gli ultimi due passaggi uguali a coppie.

Per esemplificare il percorso risolutivo segue un'analisi dettagliata dei ragionamenti che portano al primo passaggio.

Poiché la barchetta può trasportare solo due animali, vi sono tre possibilità per il primo passaggio: due sciacalli, due coyote oppure uno sciacallo e un coyote.

Nel primo caso sulla sponda di partenza rimarrebbero uno sciacallo e un coyote: situazione possibile.

Nel secondo caso sulla sponda di partenza rimarrebbero tre sciacalli e un coyote: situazione impossibile.

Nel terzo caso sulla sponda di partenza rimarrebbero due sciacalli e due coyote: situazione possibile.

Quindi solo il primo e il terzo caso sono possibili.

Per gli altri passaggi si procede in modo analogo, ricordando sempre di controllare il numero di animali sia sulla sponda di partenza che su quella di arrivo.

	azione	partenza	arrivo
0		sssccc	-
1	-> ss ->	sccc	ss
2	<- s <-	ssccc	s
3	-> ss ->	ccc	sss
4	<- s <-	sccc	ss
5	-> cc ->	sc	sscc
6	<- sc -<	sscc	sc
7	-> cc ->	ss	sccc
8	<- s <-	sss	ccc
9	-> ss ->	s	ssccc
10	<- s <-	ss	sccc
11	-> ss ->	-	sssccc

azione	partenza	arrivo
	sssccc	-
-> ss ->	sccc	ss
<- s <-	ssccc	s
-> ss ->	ccc	sss
<- s <-	sccc	ss
-> cc ->	sc	sscc
<- sc -<	sscc	sc
-> cc ->	ss	sccc
<- s <-	sss	ccc
-> ss ->	s	ssccc
<- c <-	sc	sscc
-> sc ->	-	sssccc

azione	partenza	arrivo
	sssccc	-
-> sc ->	sscc	sc
<- c <-	ssccc	s
-> ss ->	ccc	sss
<- s <-	sccc	ss
-> cc ->	sc	sscc
<- sc -<	sscc	sc
-> cc ->	ss	sccc
<- s <-	sss	ccc
-> ss ->	s	ssccc
<- s <-	ss	sccc
-> ss ->	-	sssccc

azione	partenza	arrivo
	sssccc	-
-> sc ->	sscc	sc
<- c <-	ssccc	s
-> ss ->	ccc	sss
<- s <-	sccc	ss
-> cc ->	sc	sscc
<- sc -<	sscc	sc
-> cc ->	ss	sccc
<- s <-	sss	ccc
-> ss ->	s	ssccc
<- c <-	sc	sscc
-> sc ->	-	sssccc