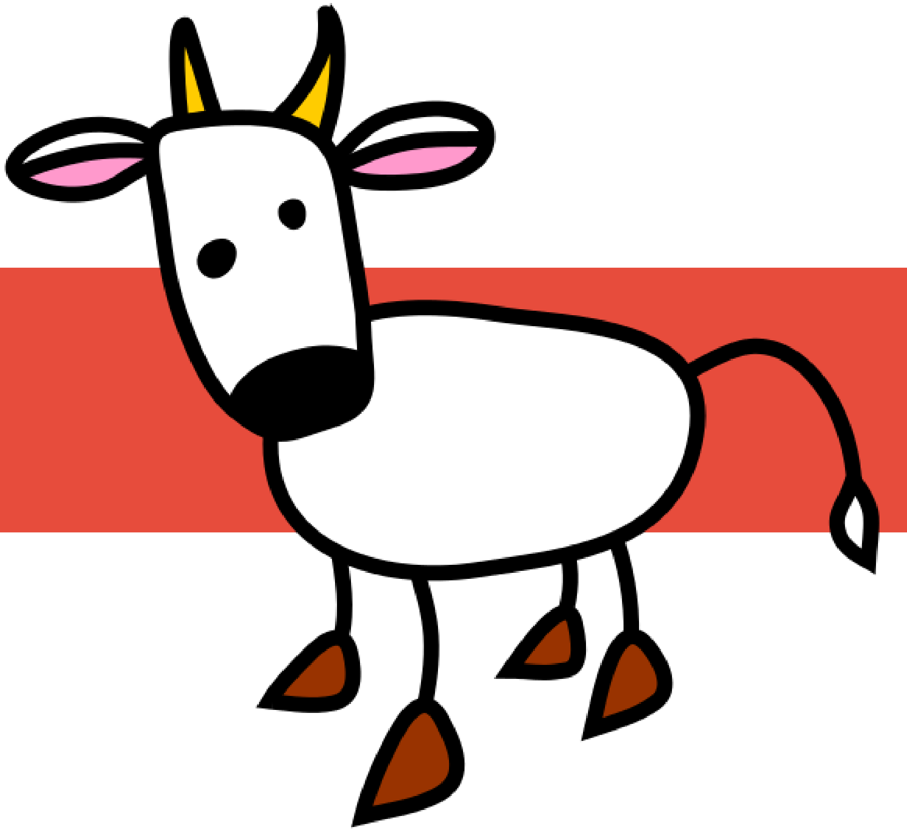
The Cow Path Problem



Francesco Galla` e Francesco Mecca

Il Problema

Una mucca Bessie si trova ad un incrocio di *w* cammini che portano a terre sconosciute.

Lungo uno solo dei cammini abbiamo un campo fiorito che consideriamo la destinazione. Questo cammino e` a distanza *n*. Tutti gli altri cammini sono interminabili.

Bessie, che ha una scarsa vista, non ha modo di sapere che ha raggiunto la destinazione finche` non riesce effettivamente a calpestare il prato.

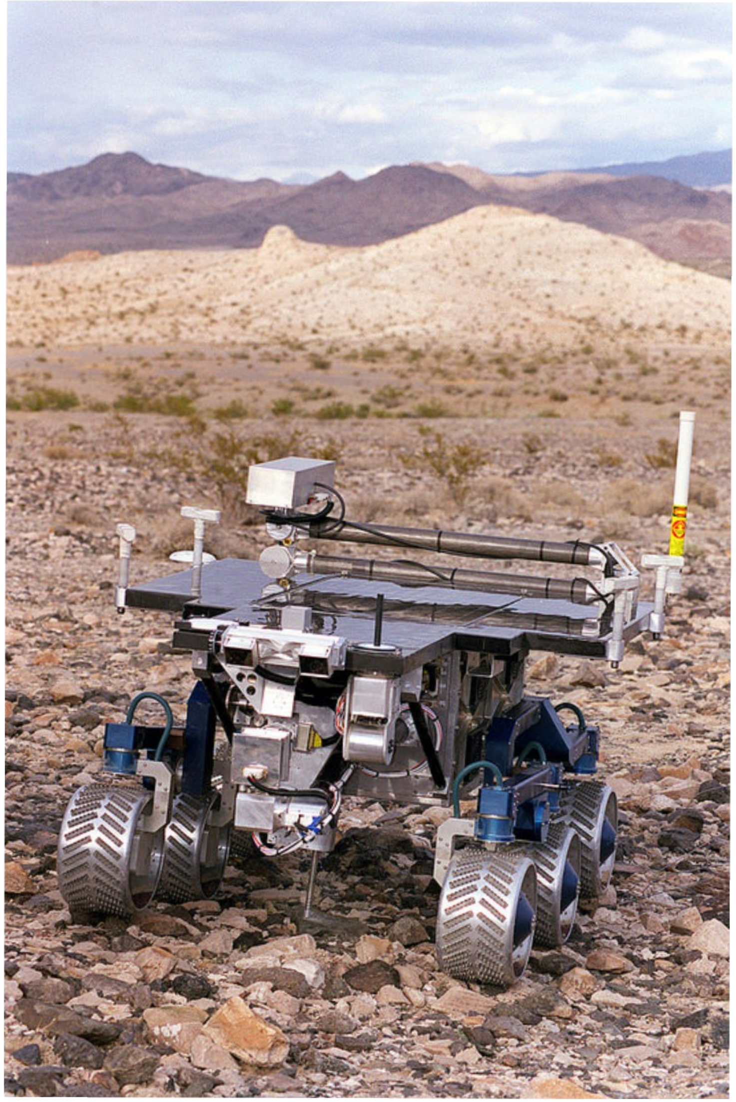
Possiamo quindi affermare che Bessie dovra` camminare almeno una distanza *n*; non assumendo una conoscenza a priori dei cammini possibili, vogliamo considerare un algoritmo ottimale per percorrere la minore distanza possibile.

Applicazione #1 – Robot e ostacolo

Un robot in uno spazio bidimensionale sconosciuto si trova davanti ad un ostacolo.

Deve trovare l’angolo piu` vicino per aggirarlo.

$w = 2$



Applicazione #2 - Search space in AI algorithms

Un albero di ricerca composto da *k* sottoalberi dev'essere esplorato alla ricerca della soluzione.

Supponiamo di avere una funzione che ci permette di capire se il nodo sul quale ci troviamo ci portera` alla soluzione.

Possiamo nuovamente considerare il cow-path problem con

$w = k$ per raggiungere il sottoalbero corretto.

Soluzione Deterministica – Primo Approccio

- Con $w = 2$, ci muoviamo a destra di un passo, poi torniamo all’origine e ci muoviamo a sinistra di un passo.
- A questo punto torniamo all’origine e incrementiamo il numero di passi di 1, ripetendo quindi il procedimento per 2 passi.
- Se il punto e` a una distanza finita n dall’origine per raggiungere il punto definito sara` necessario un numero di passi al piu`:

$$\boldsymbol{O(n^2)}$$

Soluzione Deterministica – Spiral Search

Definiamo una funzione *Spiral(i)* che rappresenta il numero di passi prima del turno i.

- I valori di *i* pari indicano il numero di passi verso destra
- I valori di *i* dispari indicano il numero di passi verso sinistra

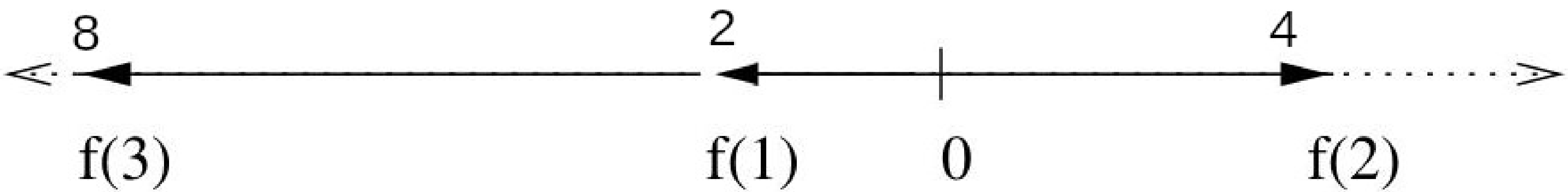
Per ottenere una progressione:

$$Spiral(i) \geq Spiral(i-2) + 1$$

La **distanza raggiunta** e` data da:

$$Spiral(i) = 2^i, \forall i \geq 1$$
$$Spiral(0) = 0$$

Spiral Search



O(Spiral (i)) $= 2 \sum_{i=1}^{\lfloor \log n \rfloor + 1} 2^i + n < 2\left(2^{\lfloor \log n \rfloor + 2}\right) + n$

$$\leq 2\left(2^2 \cdot 2^{\lfloor \log n \rfloor}\right) + n$$

$$= 2^3 n + n$$

$$= 9n$$

Algoritmi online e offline

Algoritmo offline

- I dati sono conosciuti a priori
- Nella versione offline del cow-path Bessie percorre n passi per raggiungere l'obbiettivo

Algoritmo online

- L'input viene processato poco alla volta
- Gli algoritmi greedy ne sono un esempio

Competitive ratio

Rapporto fra versione online dell'algoritmo e la versione ottimale offline.

Nel caso del cow-path:

$$R = \frac{9n}{n} = 9$$

La Randomizzazione

Un algoritmo randomizzato effettua alcune scelte in maniera **probabilistica**, utilizzando la funzione *RANDOM(A)* che restituisce con probabilita` uniforme un elemento dell'insieme *A* scelto casualmente.

Gli algoritmi randomizzati possono reagire in modo diverso con pari istanze di input.

Soluzione Randomizzata - SmartCow

L'algoritmo e` definito in termini di σ, ovvero l'insieme dei percorsi numerati da *0* a $w - 1$ ed una costante $r > 1$ che definiremo in seguito.

L'uso della randomizzazione e` molto limitato in quanto necessario esclusivamente per avere:

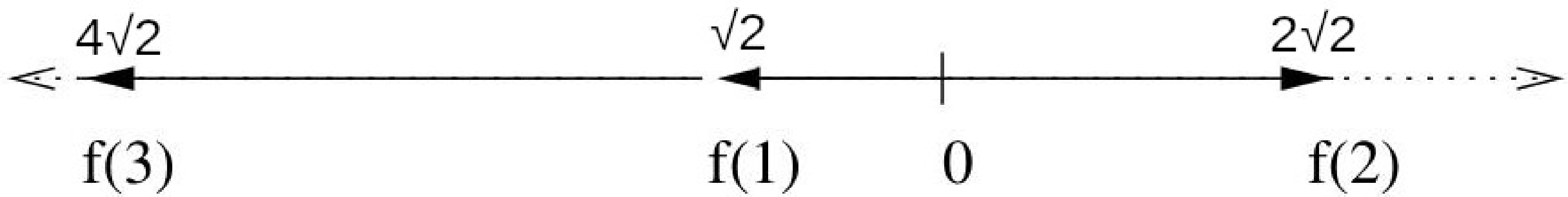
- una permutazione casuale dei possibili percorsi
- una distanza di ricerca iniziale

Algoritmo

$\sigma \leftarrow$ A random permutation of $\{0, 1, 2, \cdots, w-1\}$;
$\epsilon \leftarrow$ A random real uniformly chosen from $[0, 1)$;
$d \leftarrow r^{\epsilon}$;
$p \leftarrow 0$;
repeat
    Explore path $\sigma(p)$ up to distance d;
    if goal not found then return to origin;
    $d \leftarrow d \cdot r$;
    $p \leftarrow (p+1) \bmod w$;
until goal found;

Algoritmo

$r = 2,\ \varepsilon = 0.5$



E con $w > 2$?

Studio di Complessita`

- L'algoritmo SmartCow procede in step successivi, nei quali percorre la distanza ***r*** $^{i+\varepsilon}$ sul path:

  $$\sigma(\, i \bmod w\,)$$

- ***c*** e` il turno in cui percorriamo per la prima volta una distanza sufficiente a raggiungere il goal a distanza *n.*

  Caso limite:

  $$r^c \geq n$$

- Allora il turno corretto e`:

$$\log_r n \leq c \leq log_r n + w - 1$$

Studio di Complessita`

- Se definiamo k = $\log_r n$

$$E[D] = \sum_{i=k}^{k+w-1} \text{Prob}(c = i)\, E[D \mid c = i]$$

- Ovvero: il valore atteso della distanza percorsa e` la sommatoria del prodotto della probabilita` che il percorso sia corretto e la probabilita` che la distanza sia corretta.

- La probabilita` di trovarsi sul cammino corretto e`: $\frac{1}{w}$

Studio di Complessita`

- La distanza percorsa e` definita come:

$$D = 2 \sum_{i=0}^{c-1} r^{i+\varepsilon} + n = \frac{2r^{\varepsilon}(r^{c}-1)}{r-1} + n,$$

La formula della distanza percorsa dipende dal valore di r^{ε}, che e' una **variabile casuale** dato che ε e' uniformemente distribuito tra 0 e 1. Sapendo che E[r^{ε} | c] si puo` calcolare come:

$$E[r^{\varepsilon}] = \int_{1}^{r} x \cdot \frac{1}{x \ln r}\, dx = \frac{r-1}{\ln r}$$

Studio di Complessita`

- Dobbiamo quindi considerare l'**expected value** della distanza, che puo' essere espresso come:

$$E[D \mid c] = \frac{2(r^c - 1)}{r - 1} E[r^{\varepsilon} \mid c] + n$$

- Otteniamo:

$$E[D \mid c] = \frac{2(r^c - 1)}{\ln r} + n.$$

Studio di Complessita`

- Il valore atteso della distanza e`:

$$E[D] \leqslant \left[\frac{2(r^w - 1)}{w(r-1)\ln r} + 1 \right] n$$

- Da cui otteniamo il competitive ratio:

$$R(r, w) = 1 + \frac{2(r^w - 1)}{w(r-1)\ln r} = 1 + \frac{2}{w} \cdot \frac{1 + r + r^2 + \cdots + r^{w-1}}{\ln r}$$

Come scegliere r ?

E’ possibile dimostrare che la soluzione all’equazione:

$$\ln r = \frac{1 + r + r^2 + \cdots + r^{w-1}}{r + 2r^2 + 3r^3 + \cdots + (w-1)r^{w-1}}$$

Fornisce il valore ottimale di r per **minimizzare il competitive ratio**.

Confronto con Algoritmo Deterministico

Approximate Values for Small *w*

w	r_w^*	Competitive ratio of SmartCow	Optimal deterministic ratio
2	3.59112	4.59112	9.00000
3	2.01092	7.73232	14.5
4	1.62193	10.84181	19.96296
5	1.44827	13.94159	25.41406
6	1.35020	17.03709	30.85984
7	1.28726	20.13033	36.30277

Bibliografia

- Baeza-Yates, Culberson, Rawlins; *Searching in the Plane;* 1993
- Kao, Reif, Tate; *Searching in an Unknown Environment: An Optimal Randomized Algorithm for the Cow-Path Problem;* 1996
- Bullington, Dudek; *Spiral Search as an Efficient Mobile Robotic Search Tecnique;* 1999