I giochi e l'intelligenza artificiale

Di come i giochi
hanno accompagnato
l'evoluzione
dell'intelligenza artificiale

La vita artificiale

Il golem di Praga (homunculus magicus)

Gli automi meccanici

Automa che scrive
18º secolo

Automa che serve il tè
19º secolo

Linguaggio, giochi di abilità

1956: nasce formalmente il primo gruppo di ricerca sull'intelligenza artificiale

I primi progressi sono stupefacenti: programmi che formano frasi sensate in linguaggio naturale, o che giocano a dama

1968: John McCarthy, ricercatore, scommette 3000$ col maestro internazionale di scacchi David Levy che un computer l'avrebbe battuto entro 10 anni

Simulare l'intelligenza: ELIZA

```
I am the psychotherapist.  Please, describe
your problems.  Each time you are finished
talking, type RET twice.

> Hello, how are you?

How do you do?  What brings you to see me?

> I am depressed, it's a gloomy raining day

Are you depressed often?

> Sometimes.  More often recently

Why do you say more often recently?

> Something happened to me

Earlier you said you felt depressed?

> Yes, that's it.  I have a serious problem
```

La grande delusione

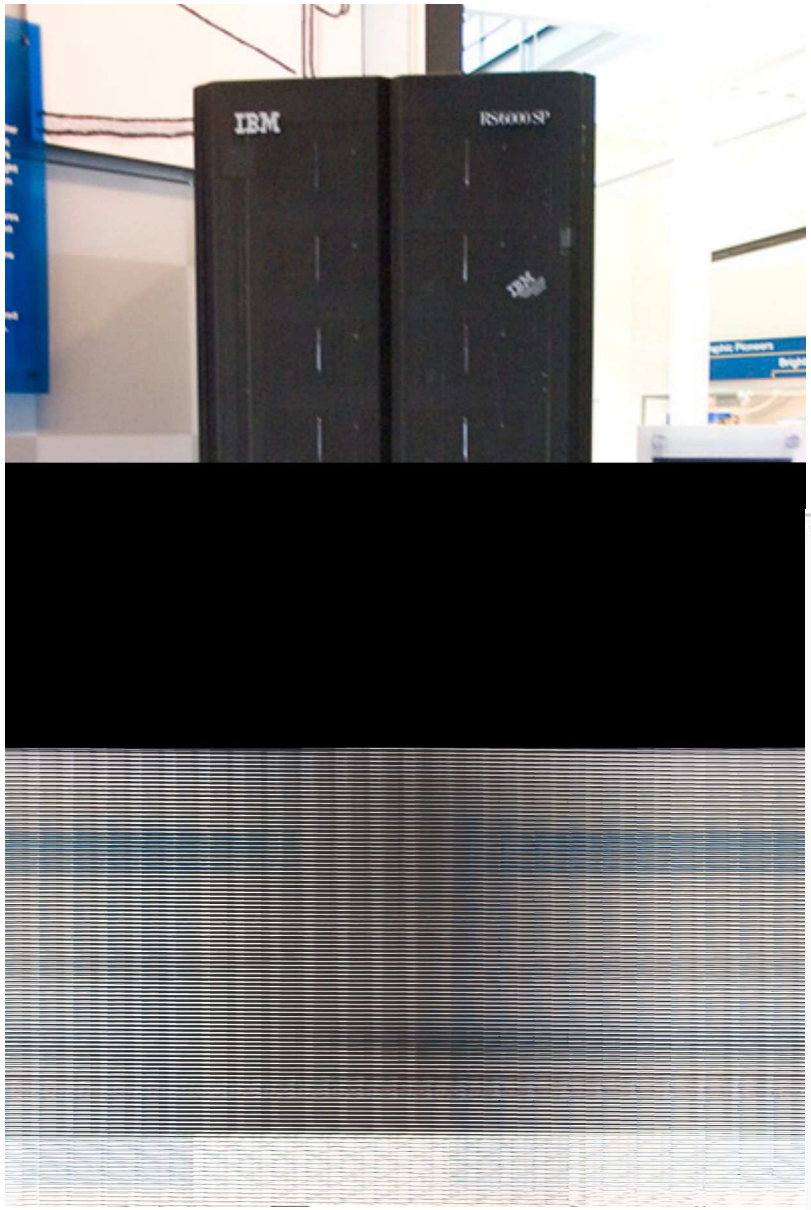
I problemi sono più grandi del previsto, la comprensione del linguaggio segna il passo

L'esplosione combinatoria si rivela un limite arduo da superare

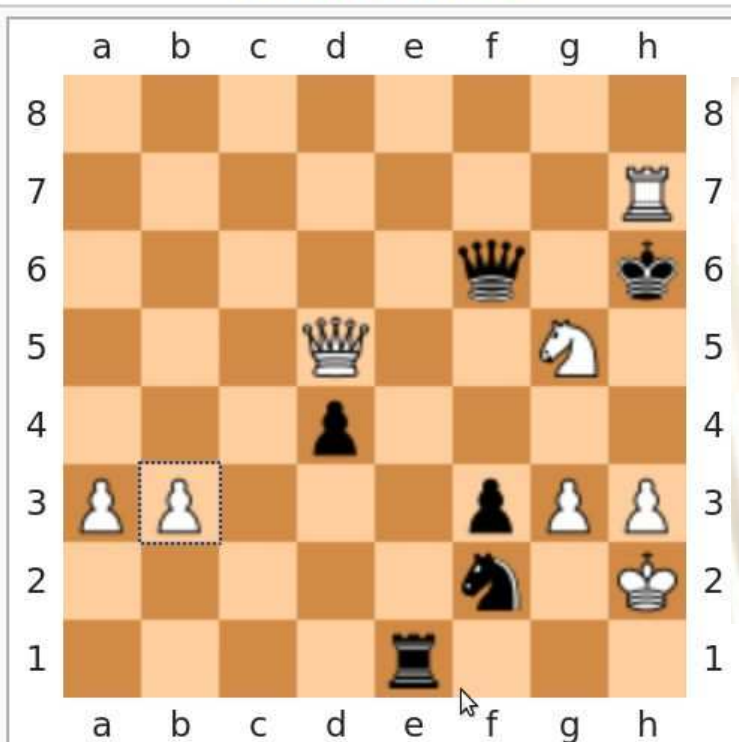
Gli anni '70 sono l'inverno dell'intelligenza artificiale, i finanziamenti languono

Ma la forza bruta si fa sentire: pur senza progressi concettuali, computer sempre più potenti cominciano a battere a scacchi giocatori dilettanti

Computer e scacchi



1997: valutando 200 milioni di posizioni al secondo, Deep Blue sconfigge il campione del mondo Kasparov



Scacchi e go

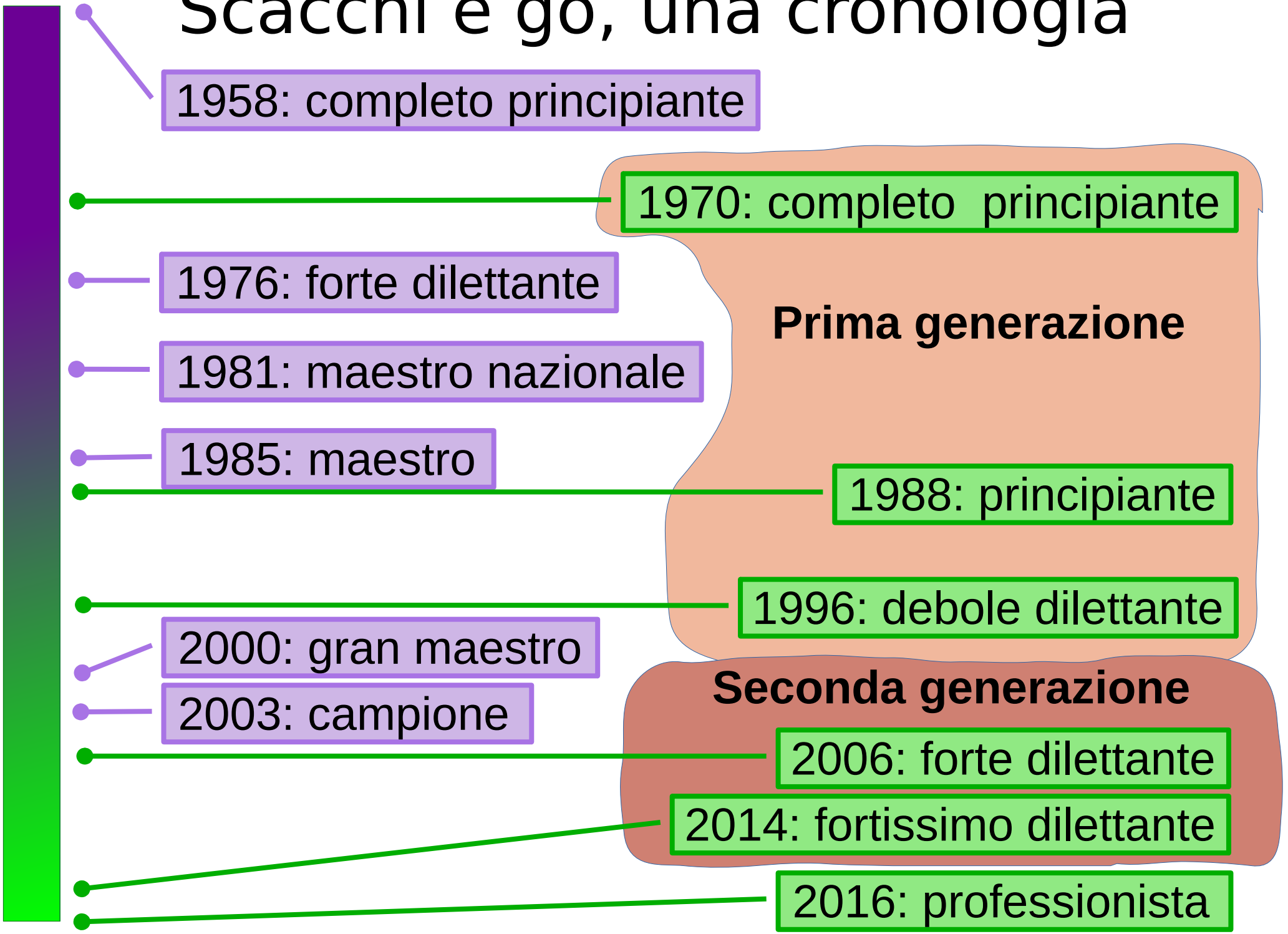
Negli scacchi l'approccio di forza bruta è vincente:

- la potatura alfa-beta dell'albero consente esplorazioni profonde
- una funzione valore semplice è già accurata

Nel go la forza bruta incontra problemi:

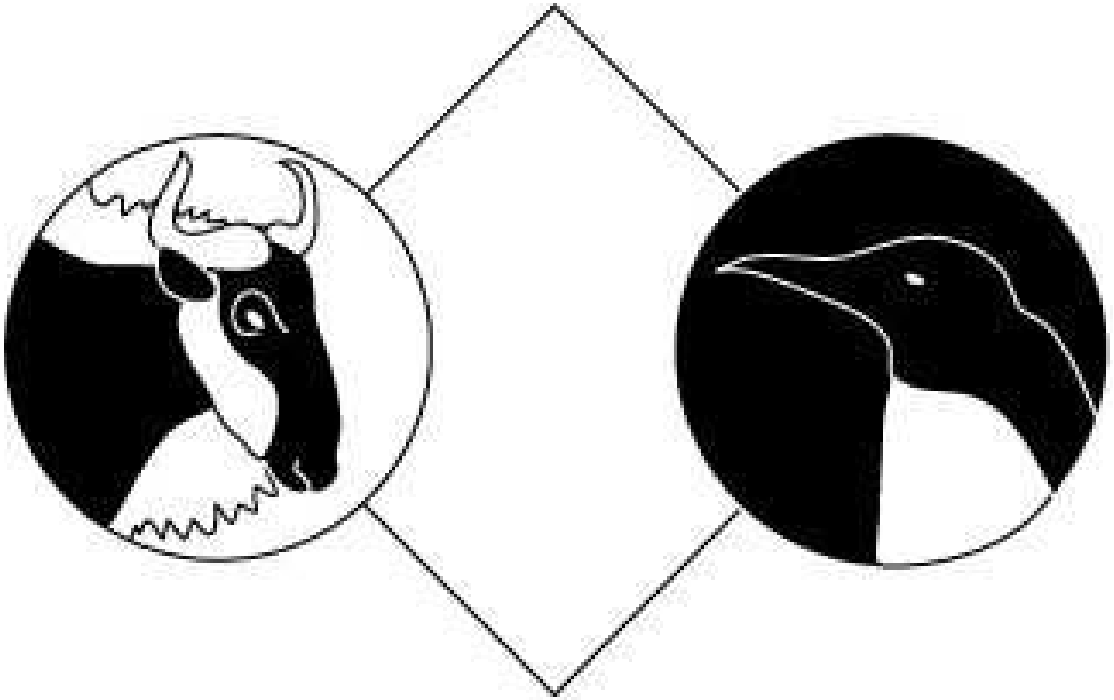
- l'albero è molto più ampio e profondo
- una funzione valore semplice è poco accurata

Scacchi e go, una cronologia



Computer e go, prima generazione

2000: Gnu Go, il primo programma libero che gioca a go, è a livello di debole dilettante



- solo euristiche
- niente ricerca sull'albero

Altri programmi di prima generazione, oltre alle euristiche, fanno una ricerca poco profonda sull'albero

Tecniche di prima generazione

Le tecniche sono varie e numerose, il che rende I programmi complessi

- librerie di aperture (joseki)
- librerie di forme significative (tesuji)
- calcolo delle battaglie locali (vita e morte)
- calcolo delle mosse alternate (ko)
- stima del punteggio
- ricerche su alberi ristretti

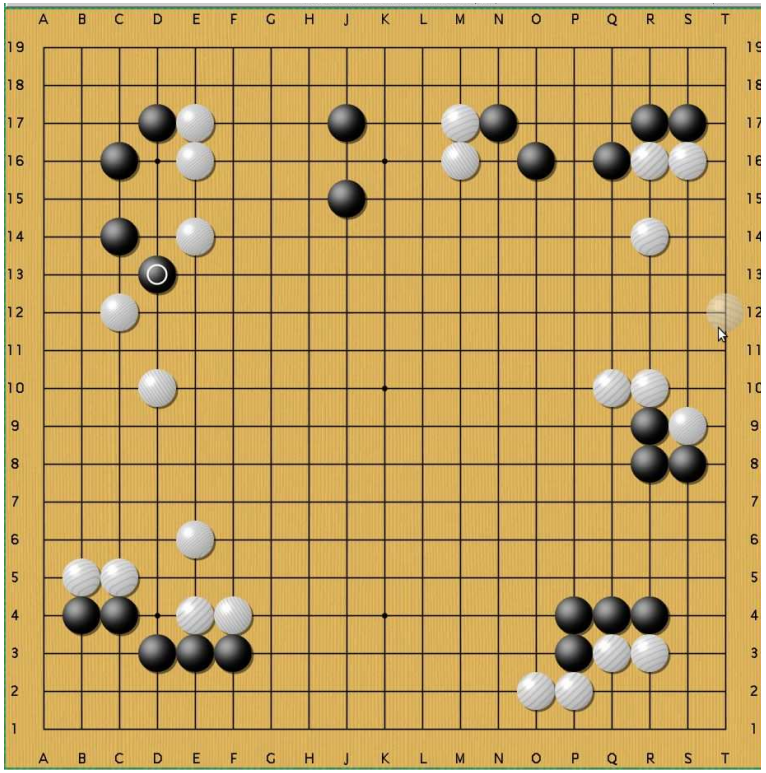
La seconda generazione

MoGo introduce per primo un nuovo algoritmo

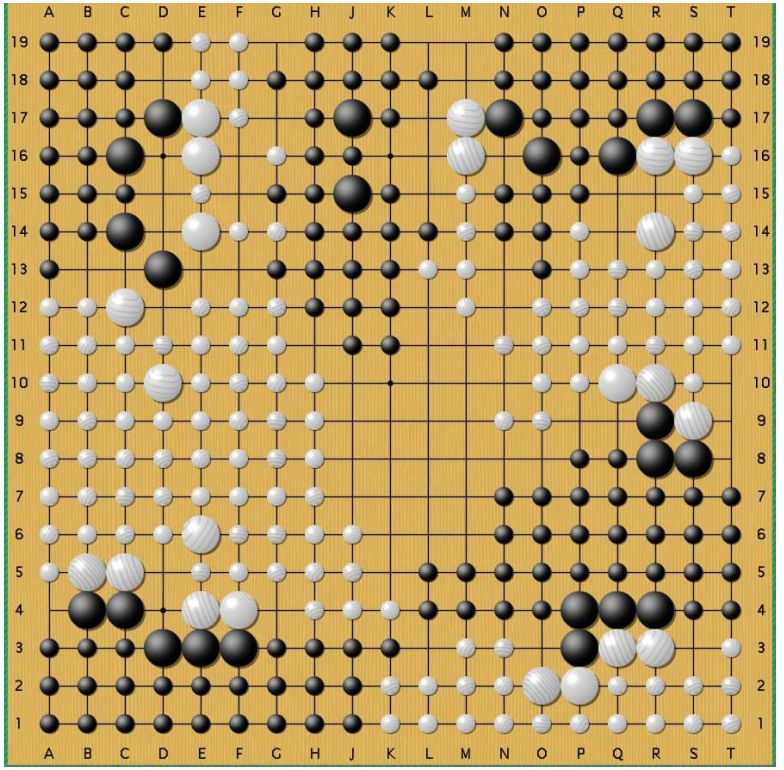
- La **ricerca ad albero**, finora poco utilizzata perché costosa, diventa fattibile grazie a due nuovi criteri
- **selezione dei rami** promettenti con tecniche di apprendimento per rinforzo (multi-armed bandit)
- **valutazione della posizione** con esplorazione casuale delle partite possibili (montecarlo)

Valutazione della posizione

La funzione valore dà in generale una stima della situazione di una partita: vittoria o sconfitta



Valore



Valutazione montecarlo

Negli scacchi la valutazione è veloce e accurata

Nel go è più difficile: *Many Faces of Go* nel 2000 valutava 10 posizioni al secondo, contro le 100.000 dei programmi di scacchi

Montecarlo: partendo dalla posizione da valutare si giocano a caso numerose partite, e si fa la media delle partite vinte sul totale

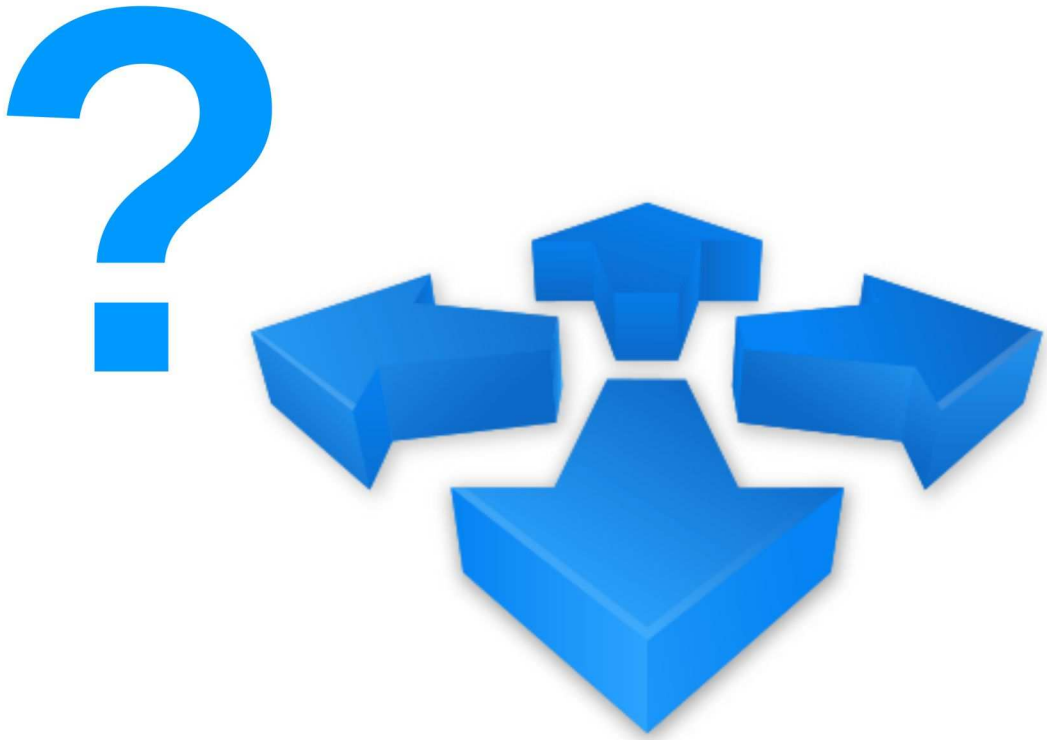
Semplice e veloce, ma non ancora abbastanza

Selezione dei rami

Pur valutando velocemente la mossa, la ricerca ad albero porta all'esplosione combinatoria

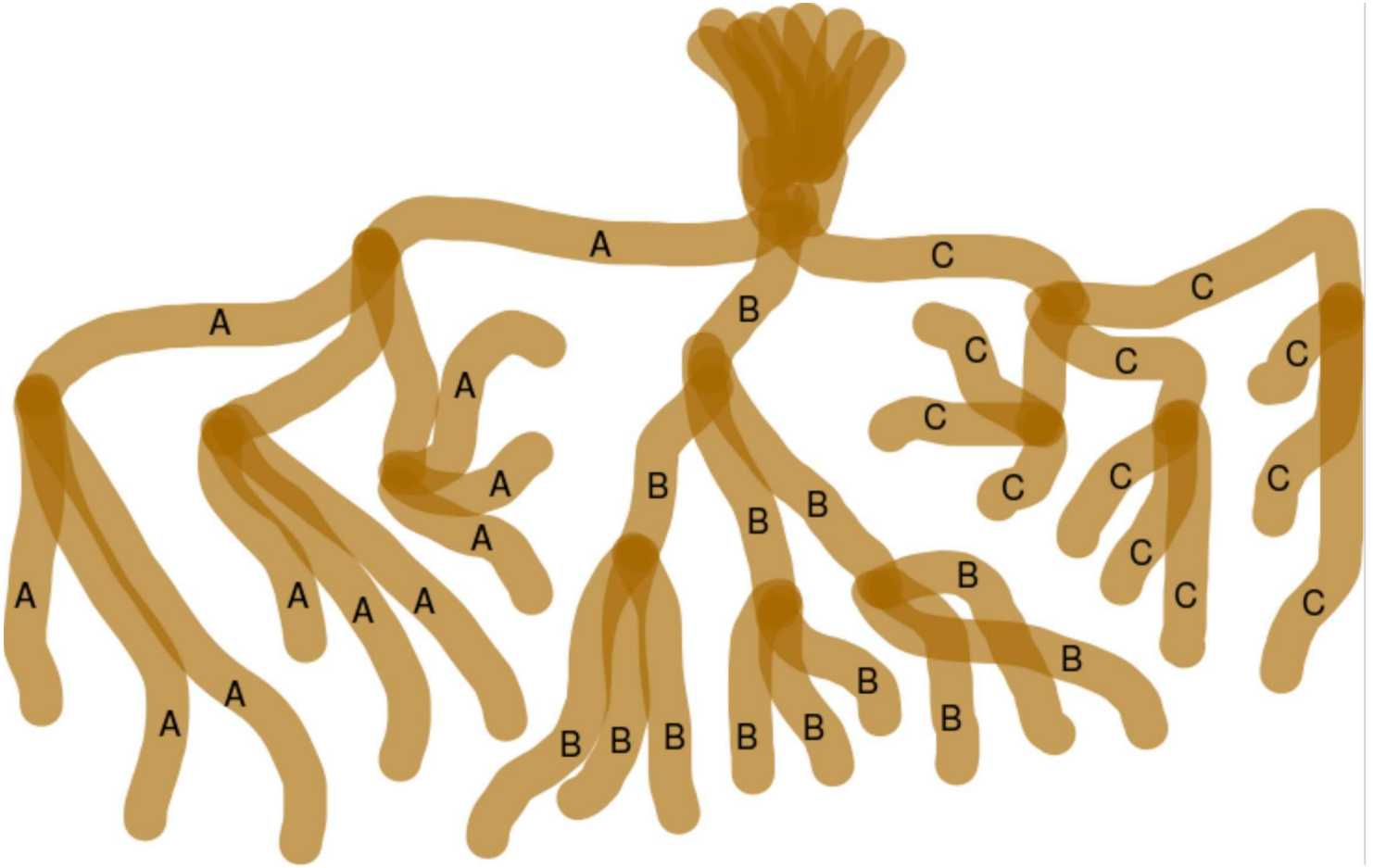
La potatura alfa-beta la controlla, ma non abbastanza per il go

Un nuovo criterio
di selezione dei rami:
l'algoritmo
UCT (Upper bound Confidence for Tree)



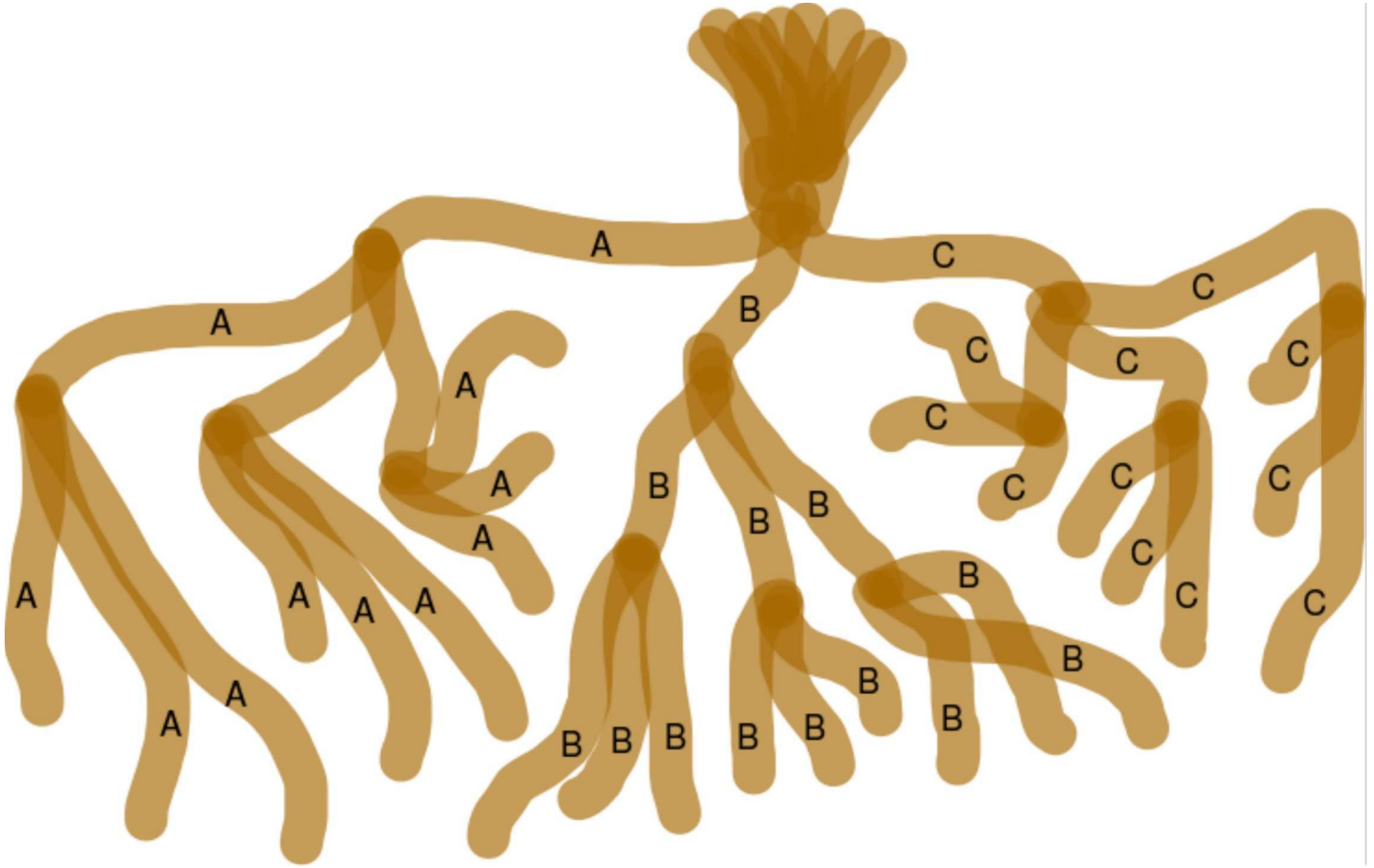
Il dilemma della scelta: la miniera

Una miniera parte con tre cunicoli (**A**, **B**, **C**), che dopo una giornata di scavo si dipartono in altri tre cunicoli e così via.



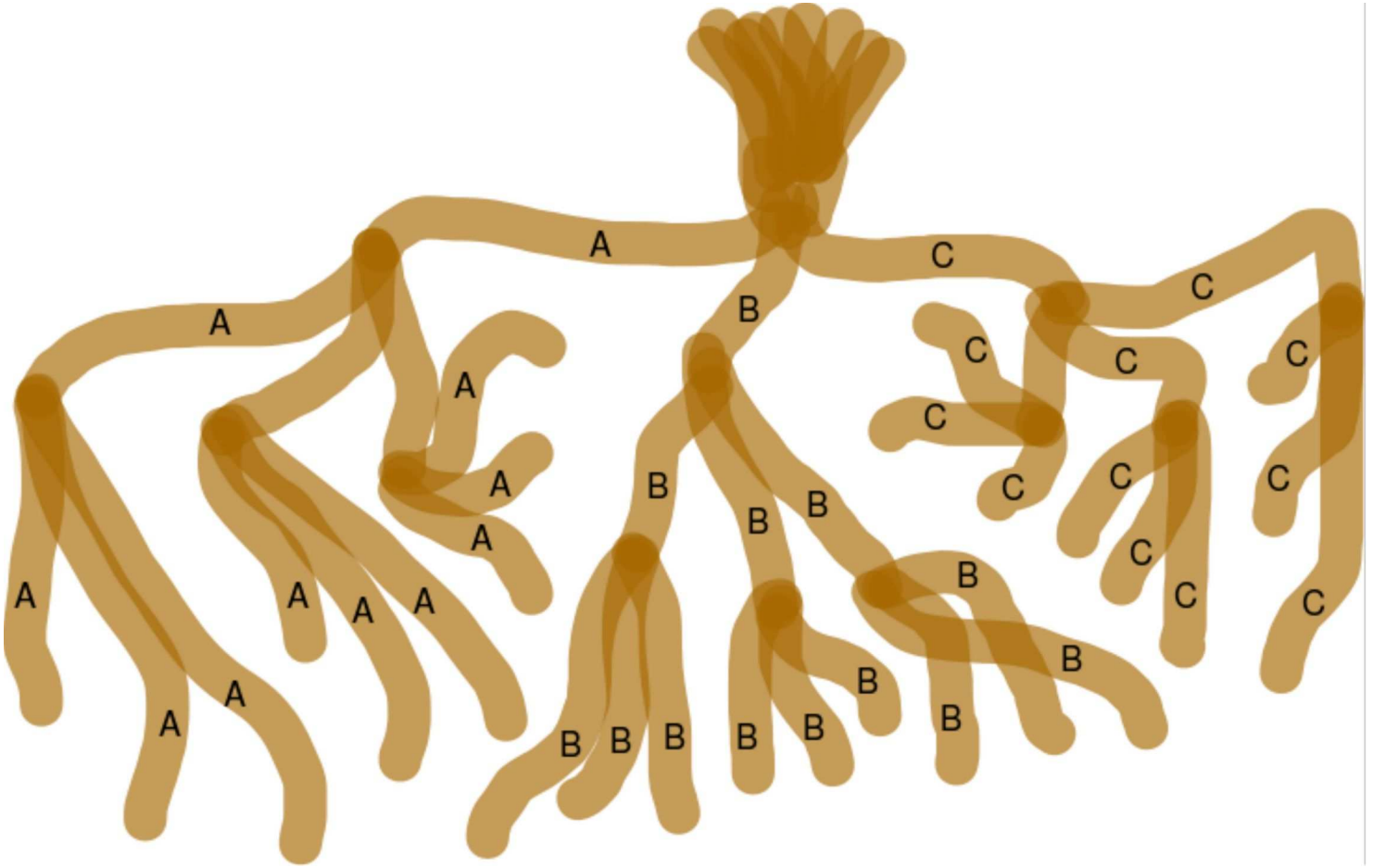
Il dilemma della scelta: le pepite

Ogni tratto di cunicolo richiede una giornata di scavo, durante la quale si può trovare una singola pepita (oppure nessuna).



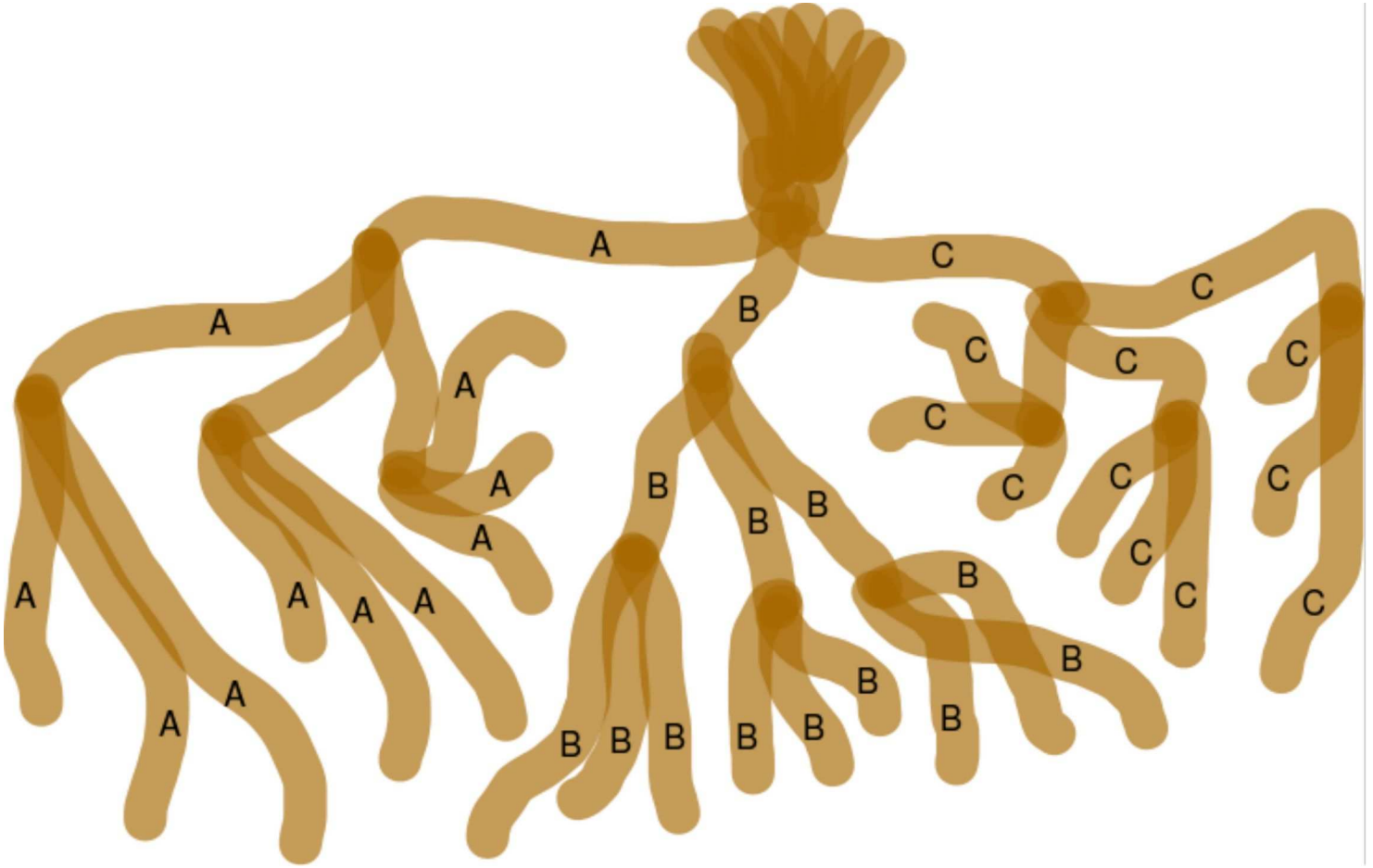
Il dilemma della scelta: esplorazione

Ognuna delle tre zone della miniera (**A**, **B**, **C**) può dare più pepite di altre, ma senza scavarci per esplorarle non si può sapere.



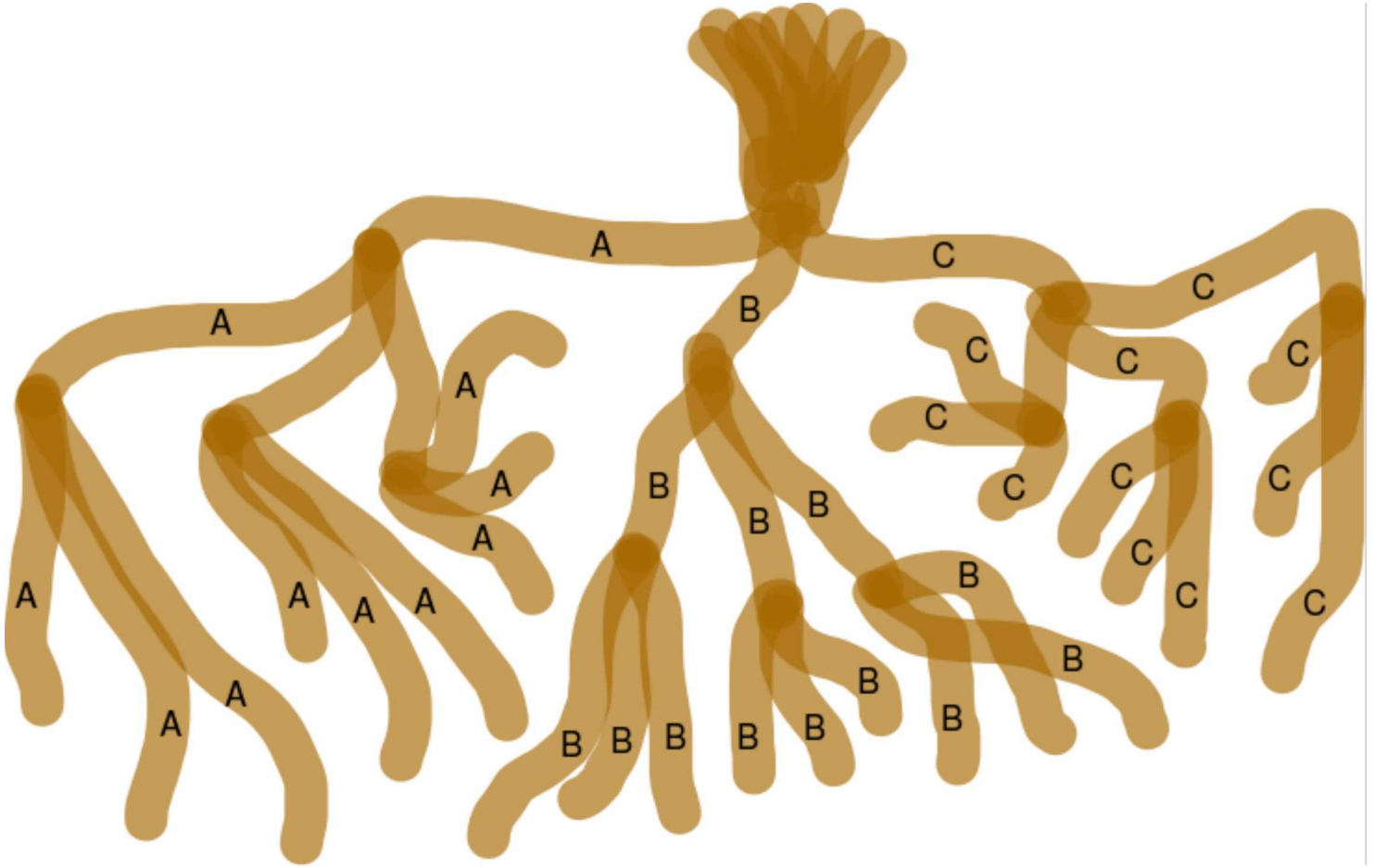
Il dilemma della scelta: sfruttamento

D'altra parte, ogni giorno o si scava per esplorare, o si scava per sfruttare la zona che finora è apparsa più ricca.



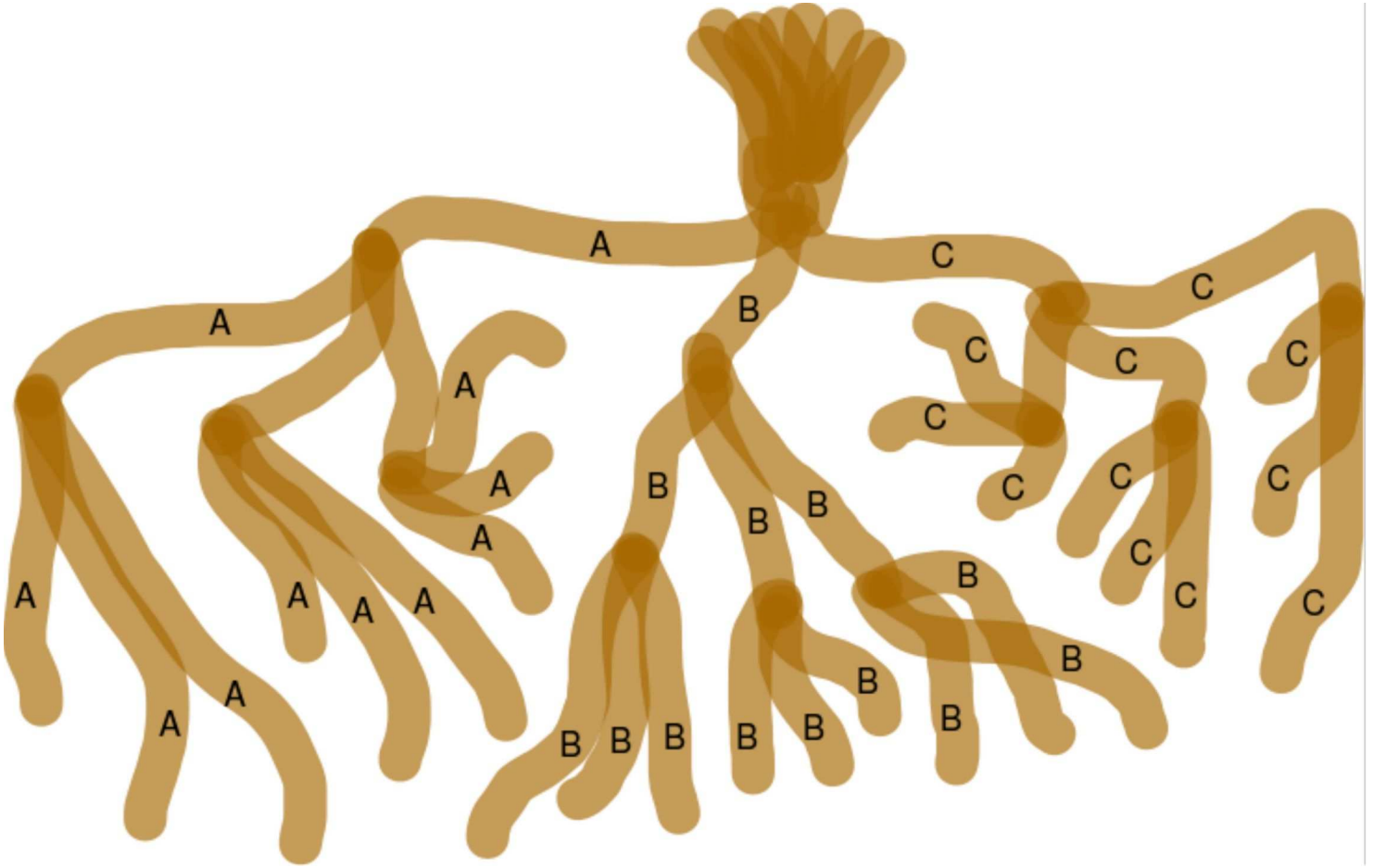
Il dilemma della scelta: criterio

Serve un criterio per decidere ogni giorno se sfruttare la zona migliore o esplorare le altre due. Vediamo due possibili criteri alternativi.



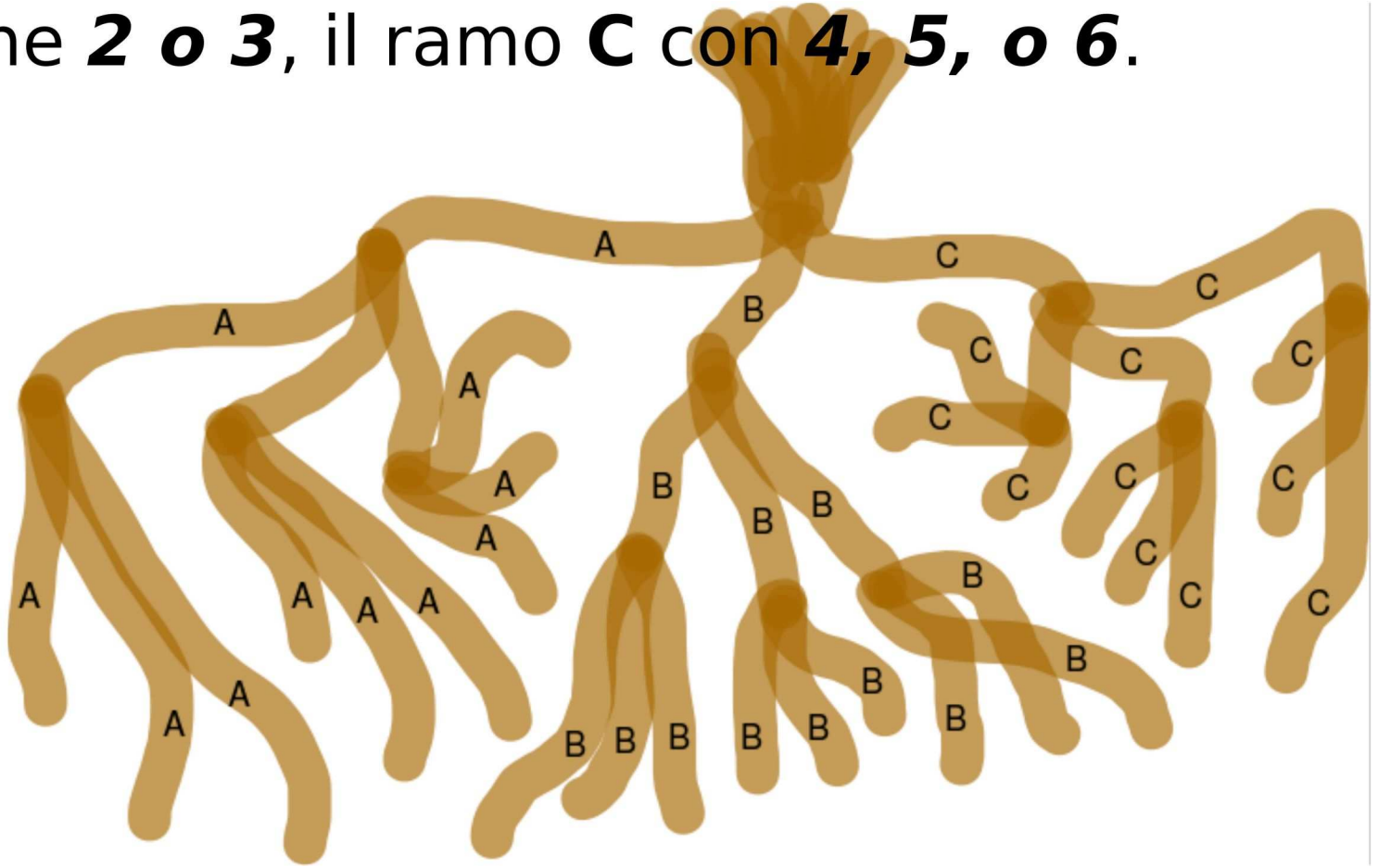
Selezione *probabilistica*

Per ogni ramo (**A**, **B**, **C**) valuto la probabilità di trovare una pepita. Estraggo a sorte uno dei tre rami privilegiando i rami più promettenti.



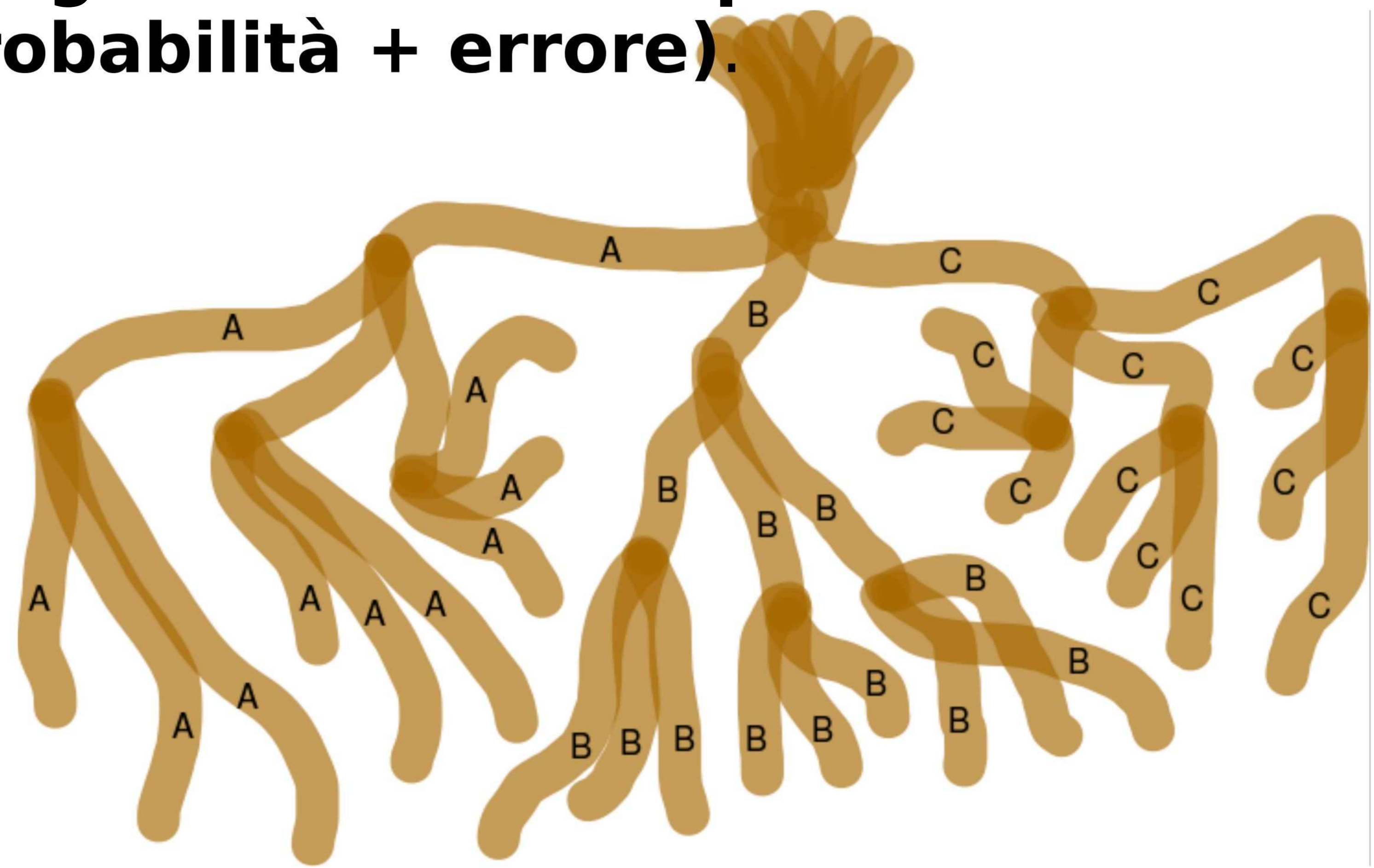
Selezione *probabilistica*: esempio

Se partendo dai rami **A**, **B**, **C** ho trovato pepite con probabilità ***1/6, 1/3, 1/2***, tirando un dado sceglierò il ramo **A** se viene ***1***, il ramo **B** se viene ***2 o 3***, il ramo **C** con ***4, 5, o 6***.

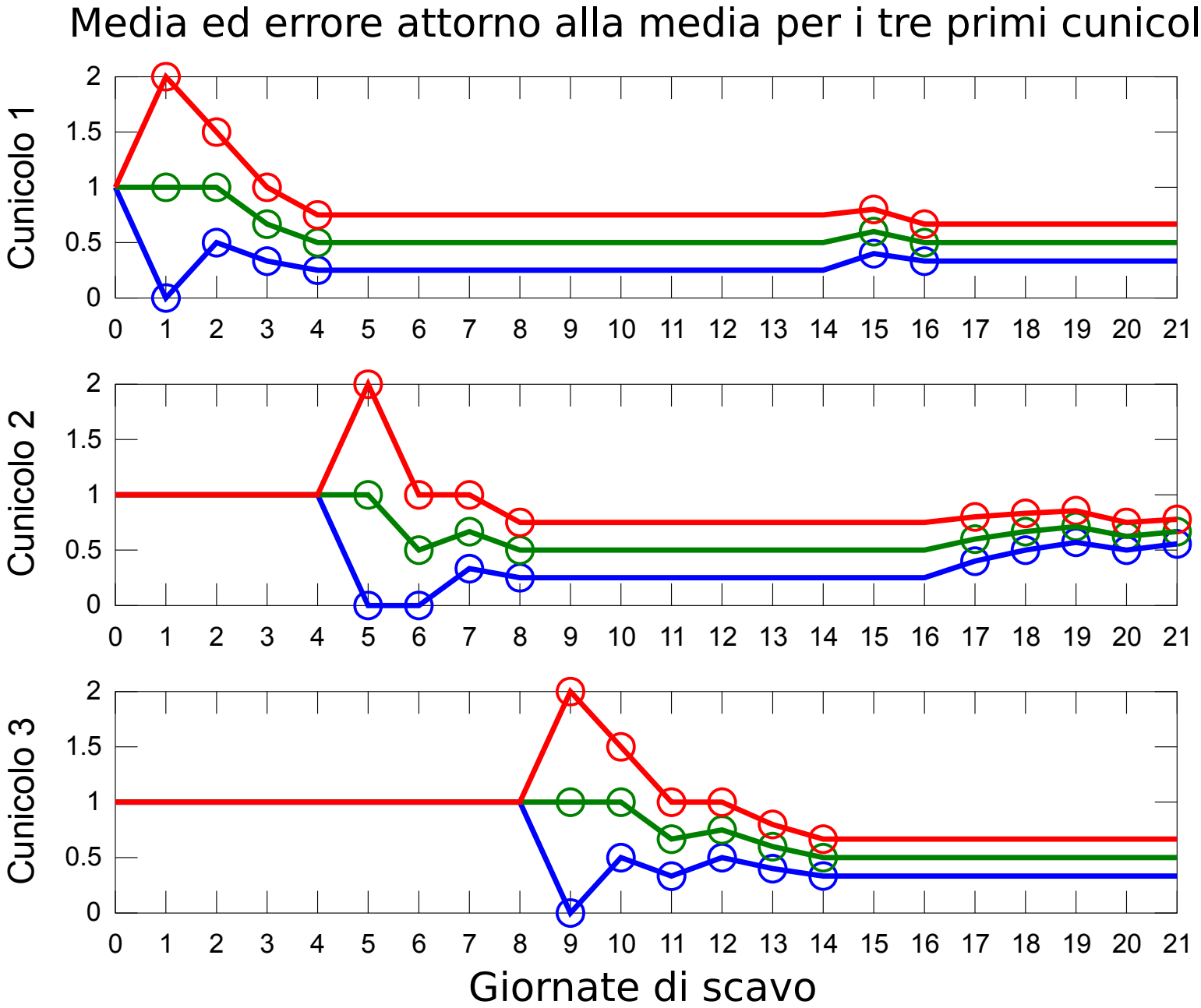


Selezione *deterministica* con UCT

Per ogni ramo valuto la **probabilità** di trovare una pepita e **l'errore** di questa valutazione. **Scelgo il ramo con la più alta somma (probabilità + errore)**.

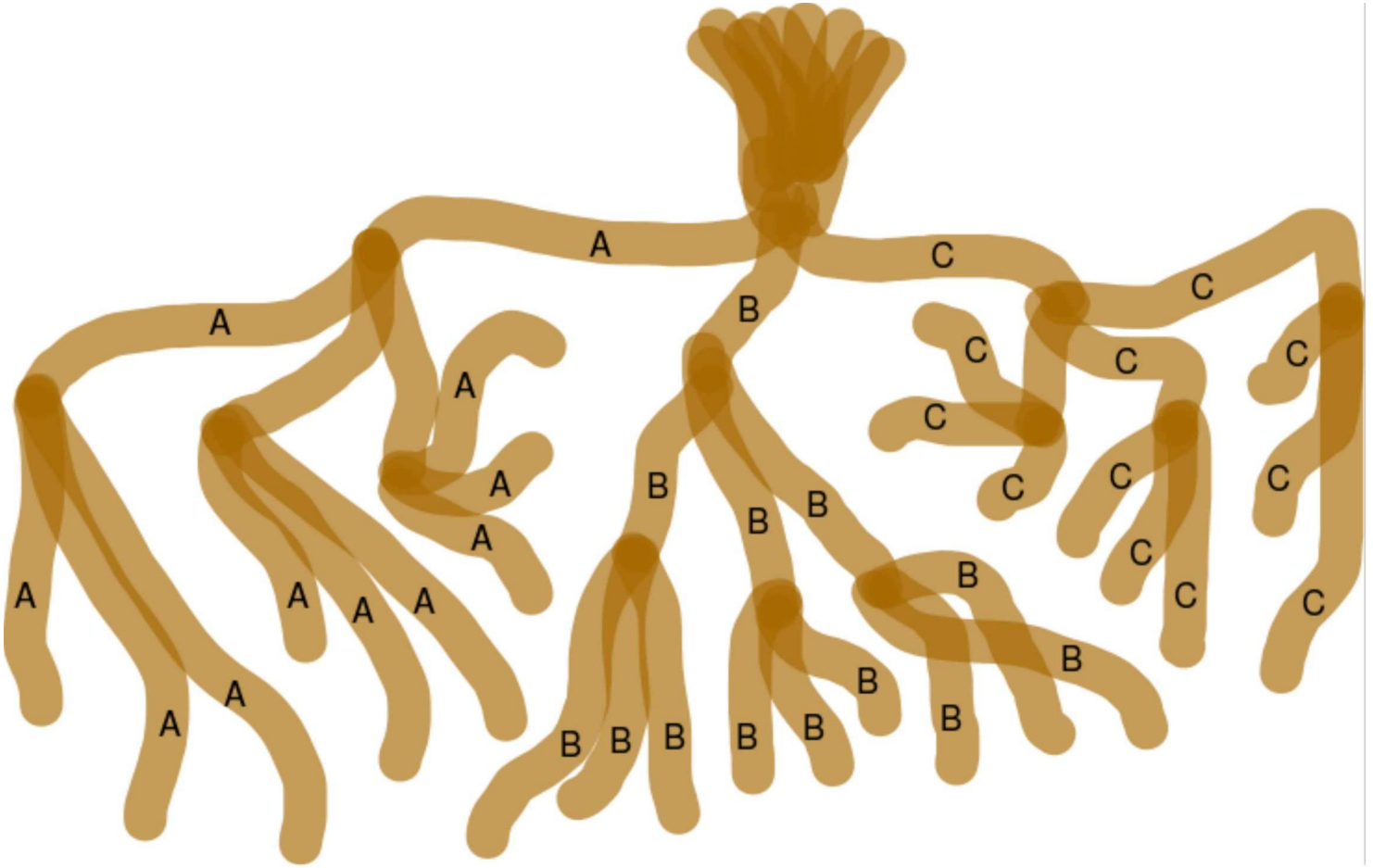


Selezione *deterministica*: esempio



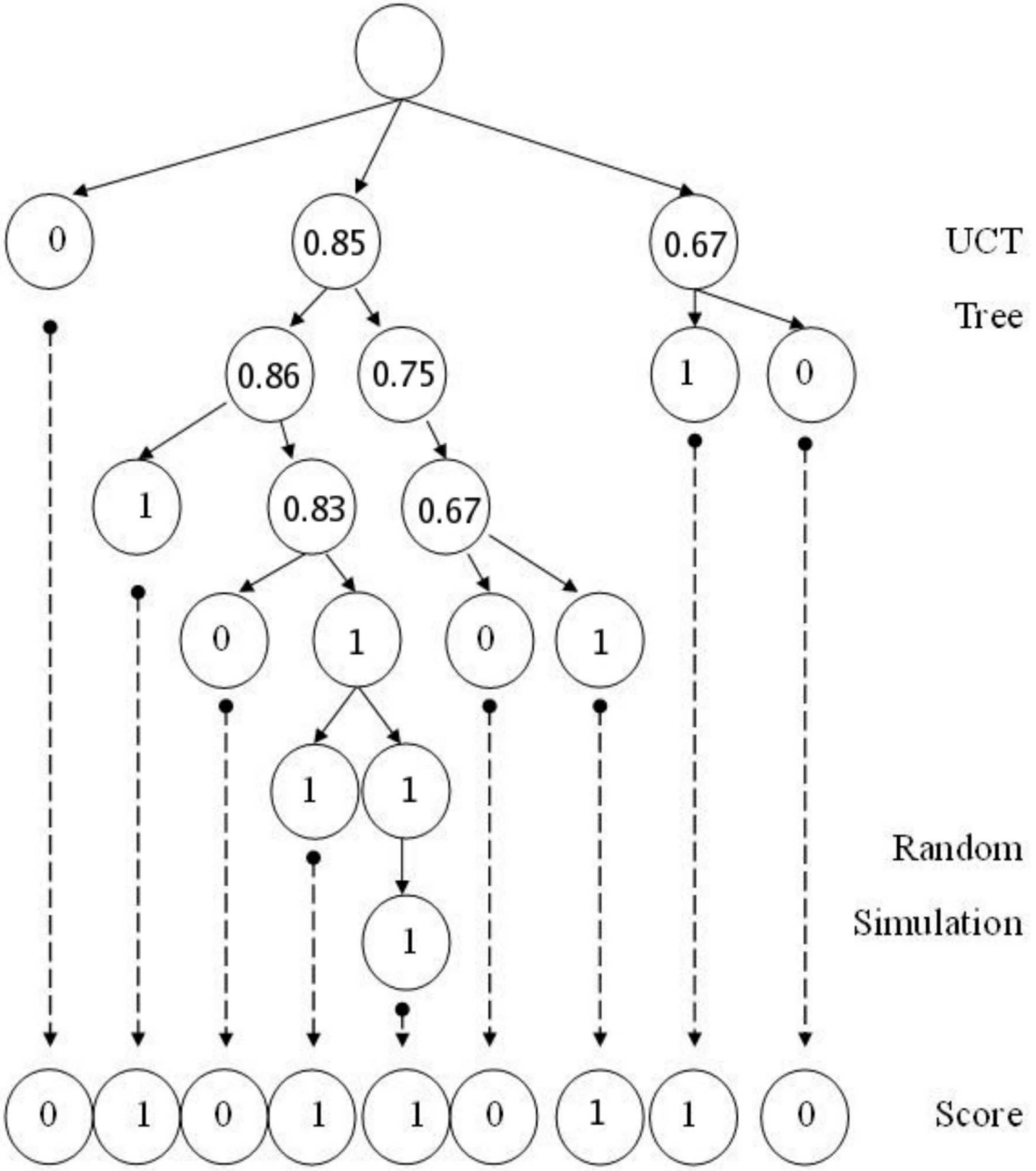
Discendere l'albero con UCT

Ogni volta che si incontra un nuovo trivio (un **nodo**), si applica UCT dando un punteggio ad ogni ramo



MoGo, UCT e montecarlo

Nell'albero delle mosse del go, ad ogni **nodo** (ogni mossa) si usa UCT per scegliere la successiva. Se quella mossa non ha ancora un valore, si gioca una partita a caso da quel punto per darle un valore



La terza generazione

1970-1996
Prima generazione: euristiche, discesa nell'albero molto superficiale

1997-2015
Seconda generazione: discesa profonda nell'albero con UCT, valutazione della posizione con montecarlo

2016
Terza generazione: valutazione della posizione e della mossa con reti neurali, discesa profonda nell'albero con UCT