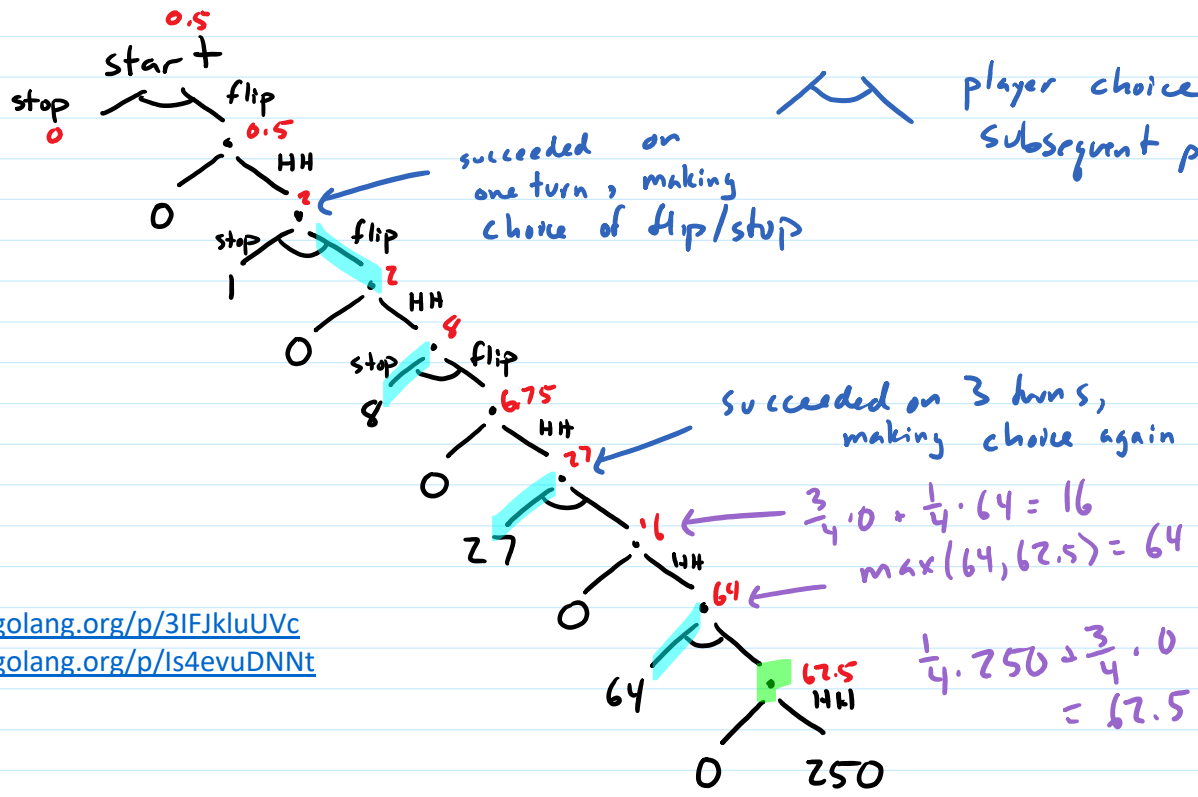


https://play.golang.org/p/3IFJkluUVc
https://play.golang.org/p/Is4evuDNNt

E[pos] = expected winnings having reached position pos

For final positions pos, E[pos] = determined by rules

For non-final choice positions

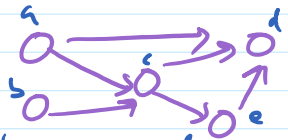
position that results from making choice c at position pos

$$E[pos] = \max_{\text{choice } c} E[next(pos, c)]$$

For non-final random event positions

position resulting from outcome σ occurring at position pos

$$E[pos] = \sum_{\text{outcome } \sigma} P(\sigma) \cdot E[next(pos, \sigma)]$$



a b c e d (or any topological sort)

```
for every final position pos
    E[pos] ← payout(pos)

for every non-final position pos   in order of ↑ distance from end
                                       max turns to end
    if pos is a choice position
        max ← −∞
        argmax ← NIL
        for every choice c
            e ← E[next(pos, c)]
            if e > max
                max ← e
                argmax ← c
        E[pos] ← max
        OPT[pos] ← argmax
    else
        e ← 0.0
        for every outcome σ
            e ← e + P(σ) · E[next(pos, σ)]
        E[pos] ← e
```

determined by rules

Coins: Start with n coins.
On each turn, flip as many of your remaining coins as you wish.
If #T ≥ #H, lose all the T
Else earn #H points
Win at X points
Lose if no coins left and < X points

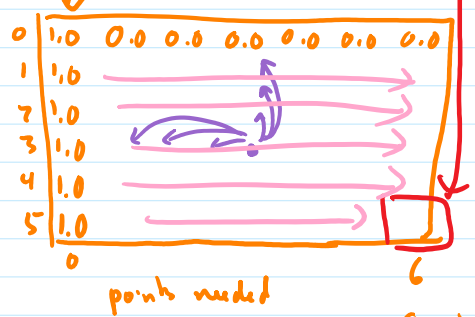
Low if no coins left and < points

Coins position = # coins left, # points left to target

5 coins, play to 6

start = (5, 6)

end = (x, 0) win
or (0, y) loss




```
init row 0 to 0.0
init col 0 to 1.0

for coins = 1 to start.C
    for needed = 1 to target
        max ← -∞
        argmax ← NIL
        for flip = 1 to coins
            val ← 0.0
            for heads = 0 to flip
                if heads > flip - heads
                    prob ← P(get heads H when flipping flip coins)
                    val ← val + prob · E[coins][max(0, needed - heads)]
                else
                    prob ← P(get heads H when flipping flip coins)
                    val ← val + prob · E[coins - (flip - heads)][needed]
            if val > max
                max ← val
                argmax ← flip
        E[coins][needed] ← max
```

$\binom{flip}{heads} / 2^{flip}$

anchor: positions at start of turn

component: positions during a single turn

number of anchors: every state of upper
every state of FH, SS, LS
every state of Y
3K, 4K, C

unused, 0, 1, 2, 3, 4, 5
↓ ← 6 categories
7^6
3^3
15 unused, 0, 50, 150, ..., 1250
28^3

≈ 1 trillion anchors
• 1600 pos/comp
≈ 1.6 quadrillion pos
≈ 32 years @ 1 million pos/sec

modification: E[pos] =

For non-final choice positions

$$E[pos] = \max_{\text{choice } c} E[next(pos, c)]$$

For non-final random event positions

$$E[pos] = \sum_{\text{outcome } \sigma} P(\sigma) \cdot E[next(pos, \sigma)]$$

anchors =

Yahtzee Graph