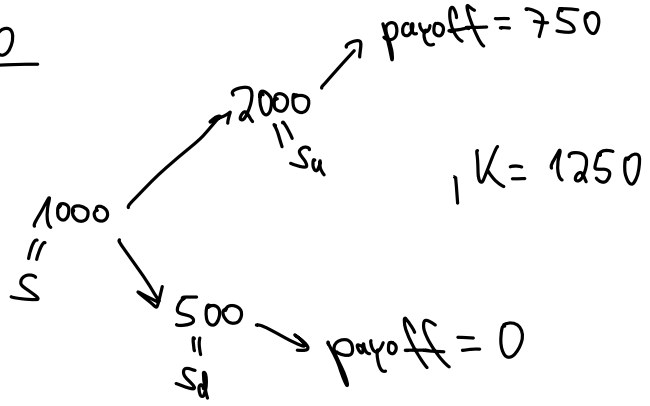

Quiz 2 Q. 3 solution:

$r=0$



money borrowed: x_1 ; # of stocks in the replicating portfolio: x_2

$$x_1 + S_u x_2 = 750 \quad \Rightarrow \quad x_1 + 2000 x_2 = 750 \quad \Rightarrow \quad 1500 x_2 = 750$$

$$x_1 + S_d x_2 = 0 \qquad x_1 + 500 x_2 = 0 \qquad x_2 = \frac{1}{2}$$

$$\Rightarrow x_1 = -250$$

price $C = x_1 + S x_2 = -250 + 1000 \cdot \frac{1}{2} = 250$

Alternatively: $p_d = \frac{a - e^r}{a - d} \overset{\text{here}}{=} \frac{a-1}{a-d} = \frac{S_u - S}{S_u - S_d} = \frac{2000 - 1000}{2000 - 500} = \frac{1000}{1500} = \frac{2}{3}$

$$= \frac{2-1}{2-\frac{1}{2}} = \frac{1}{\frac{3}{2}} = \frac{2}{3}$$

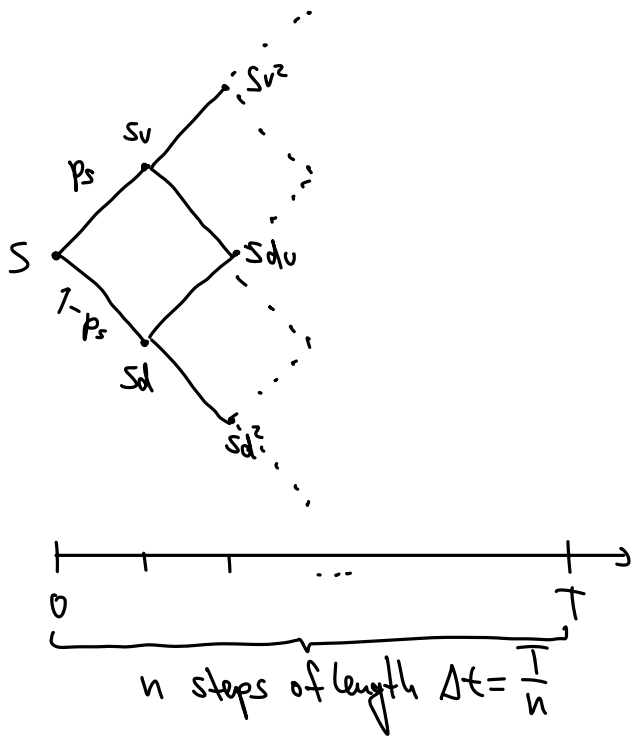
$$p_u = 1 - p_d = \frac{1}{3}$$

price $C = p_u C_u = e^0 \left(0 + \frac{1}{3} \cdot 750\right) = 250 \quad \left(C = e^{-r} \left(p_d C_d + p_u C_u \right) \right)$

2.3 Binomial Tree Models

We repeat the binary model with n steps:

1. Stock price model:



$\Rightarrow$ stock price after j upward movements after n steps: $S_T(j \text{ up's}) = S v^j d^{n-j}$

Probability for j up movements for n steps is $P(j,n) = \binom{n}{j} p_s^j (1-p_s)^{n-j}$

$$\binom{n}{j} = \frac{n!}{(n-j)!\,j!} = \frac{n(n-1)\cdots(n-j+1)}{j!} \quad (\text{"}n \text{ choose } j\text{"})$$

(Recall: $(a+b)^n = \sum_{j=0}^{n} \binom{n}{j} a^j b^{n-j}$, the binomial theorem.)

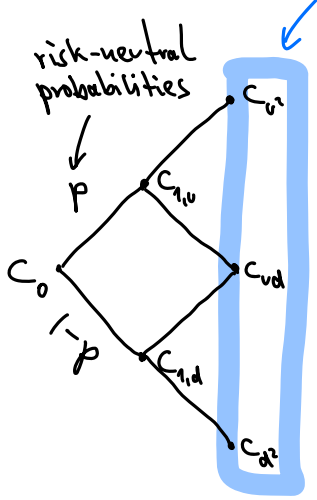
Consistency check: Is this really a probability?

$$\sum_{j=0}^{n} P(j;n) = \sum_{j=0}^{n} \binom{n}{j} p_s^j (1-p_s)^{n-j} = \left(p_s + (1-p_s)\right)^n = 1 \Rightarrow \text{yes (all possibilities add up to 1)}$$

We will come back to reasonable choices of parameters u, d, p_s later.

2. Option Price Model:

Ex.: $n = 2$



here, at expiration, we know the payoff (given the possible $S(T)$)

$$C_{u^2} = \max(0, Su^2 - K)$$

$$C_{ud} = \max(0, Sud - K)$$

$$C_{d^2} = \max(0, Sd^2 - K)$$

} for European calls

K = strike price, $p = \frac{e^r - d}{u - d}$ ($C_{1,u}, C_{1,d}$ are "intermediate payoffs", "option value at step 1", C_0 = option price)

We know from binary model how to do one step:

$$\Rightarrow C_{1,u} = e^{-r}\left(p\,C_{u^2} + (1-p)\,C_{ud}\right)$$

$$\Rightarrow C_{1,d} = e^{-r}\left(p\,C_{ud} + (1-p)\,C_{d^2}\right)$$

r = interest rate for one step

next step (from 1 to 0):

$$C_0 = e^{-r}\left(p\,C_{1,u} + (1-p)C_{1,d}\right)$$

$$= e^{-2r}\left(p^2 C_{u^2} + p(1-p)C_{ud} + (1-p)pC_{ud} + (1-p)^2 C_{d^2}\right)$$

$$= e^{-2r}\left(p^2 C_{u^2} + 2p(1-p)C_{ud} + (1-p)^2 C_{d^2}\right) \rightarrow \text{option price at time/step } 0$$

In this case (European call options without dividend payments) we get the closed-form formula (for n steps):

$$C_0 = e^{-nr}\sum_{j=0}^{n}\binom{n}{j}p^j(1-p)^{n-j}\max(0,\, Su^j d^{n-j} - K)$$

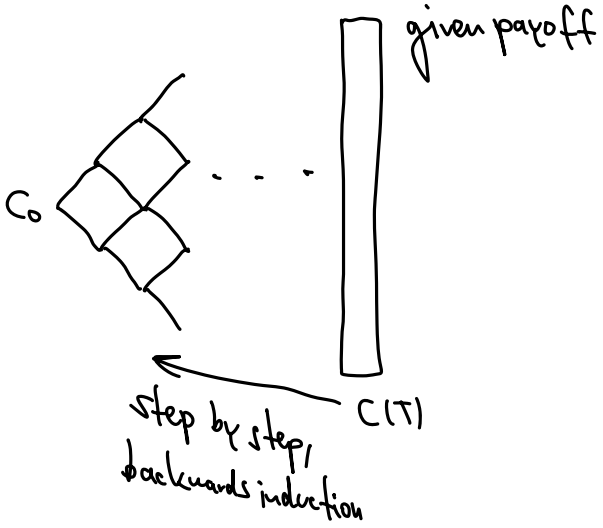
Important note: in this notation r is the interest rate for one step; for the period interest rate r_p (annual if T is in years) we have $r = r_p\,\Delta t = r_p\frac{T}{n}$

(so also $p = \frac{e^{r_p\frac{T}{n}} - d}{u-d}$)

Note: In the general case and for more complicated models (e.g., puts or dividend payments or discontinuous interest compounding) there might not be closed-form formulas, so it is better to have an algorithm available using "backwards induction" (meaning: start from last column, go to step $n-1$, then $n-2$, ..., until at time 0 you get the result C_0); still based on binomial tree.

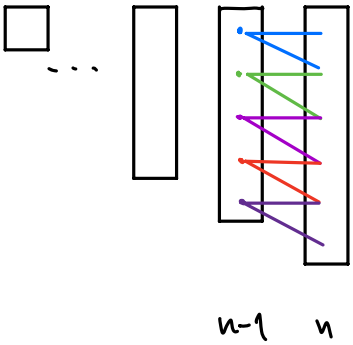
This is a very general and versatile way of option pricing.

For several steps:



Implementation in python: (Problem 2, HW 4)

- use one "for loop" to go through all the steps
- but: computation of vector/array C(n-1) from vector C(n) should be implemented vectorized



recall the notation vector[a:b:increment]

- to store data one could use:
  - one vector (length n+1); memory efficient
  - an (n+1)x(n+1) matrix; if all data is needed, e.g., for visualization