Computational Finance with C++

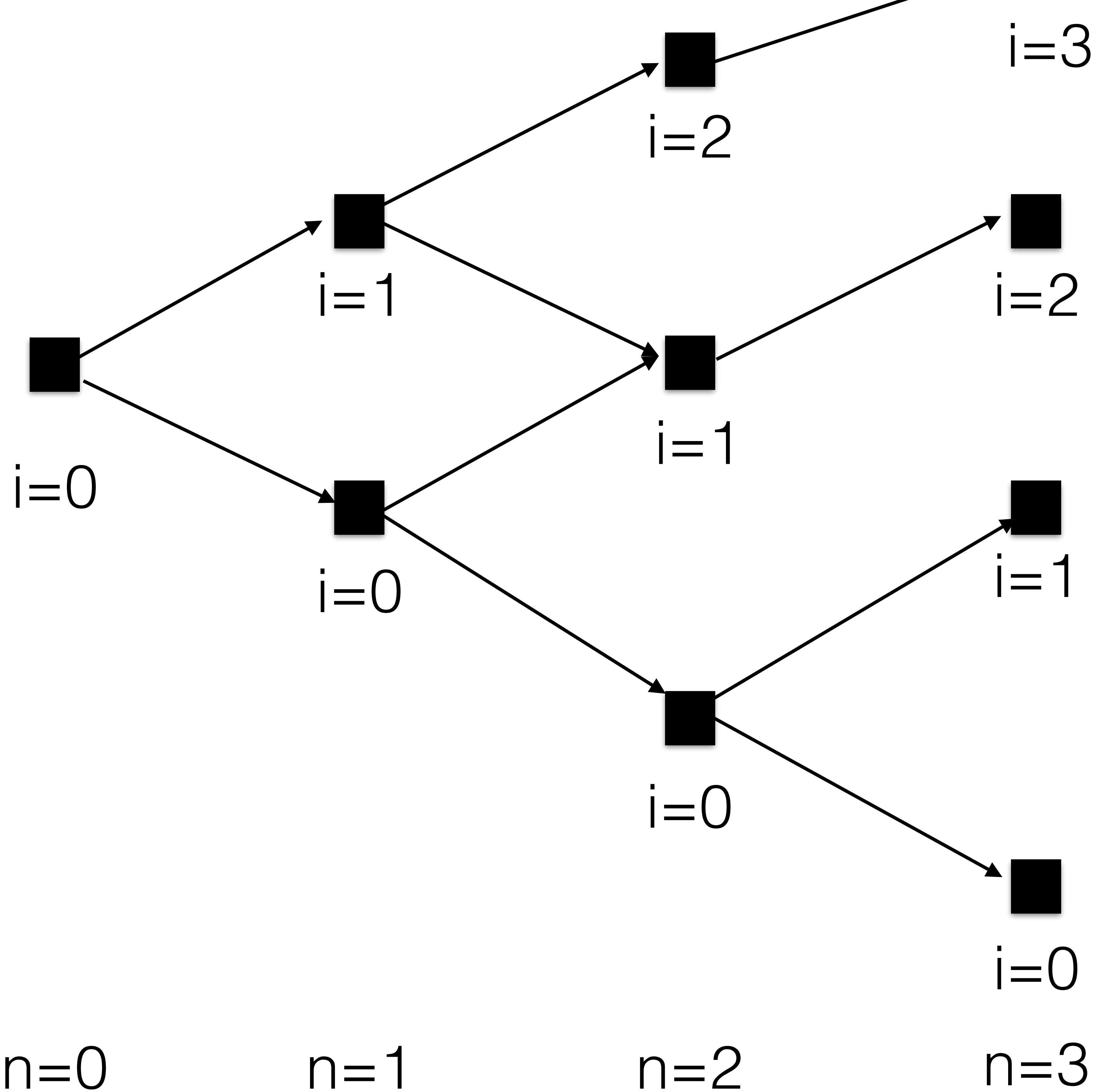
Lecture 3: More on Inheritance and Template programming

Panos Parpas
Imperial College London
p.parpas@imperial.ac.uk

Outline

- **Advanced** Inheritance concepts using American Options
  - ✦**Multiple** Inheritance
  - ✦**Virtual** Inheritance
- Introduction to **templates**
- Reading:
  - ✦Chapter 3 Capinski+Zastawniak,Numerical Methods in finance with C++

Pricing with a binomial tree



U: up factor >-1

D: down factor >-1

S(n,i): Asset Price

At time n, state i

$S(0)(1+U)^i(1+D)^{n-i}$

S(0)>0

R: risk free rate

No Arbitrage:

$D<R<U$

Reminder: American Options

- The holder of an American option can exercise their right at any time up to and including the expiry date.
- Asset Price at time n, node i is $S(n,i)$
- If option is exercised then payoff is $h(S(n,i))$
- Can be priced by backwards induction.

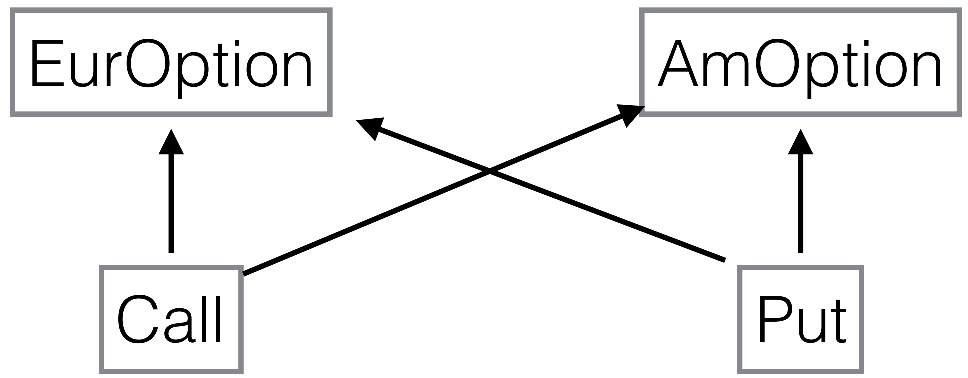
Backwards Induction for Pricing American Options

- H(n,i)=price of a American option at **time** n, **node** i
- At expiry time=N and payoff is H(S(N,i))=h(S(N,i)) e.g. h(S(N,i))=max(S(N,i)-K,0) for American call option with strike K.
- Work by backward induction: H(n+1,i) known and then:

$$H(N,i) = \max\left(\frac{qH(n+1,i+1)+(1-q)H(n+1,i)}{1+R}, h(S(n,i))\right)$$

Risk Neutral Probabilities: $q = \dfrac{R-D}{U-D}$

Multiple Inheritance: `Options07.h`, `Options07.cpp`, `Main12.cpp`



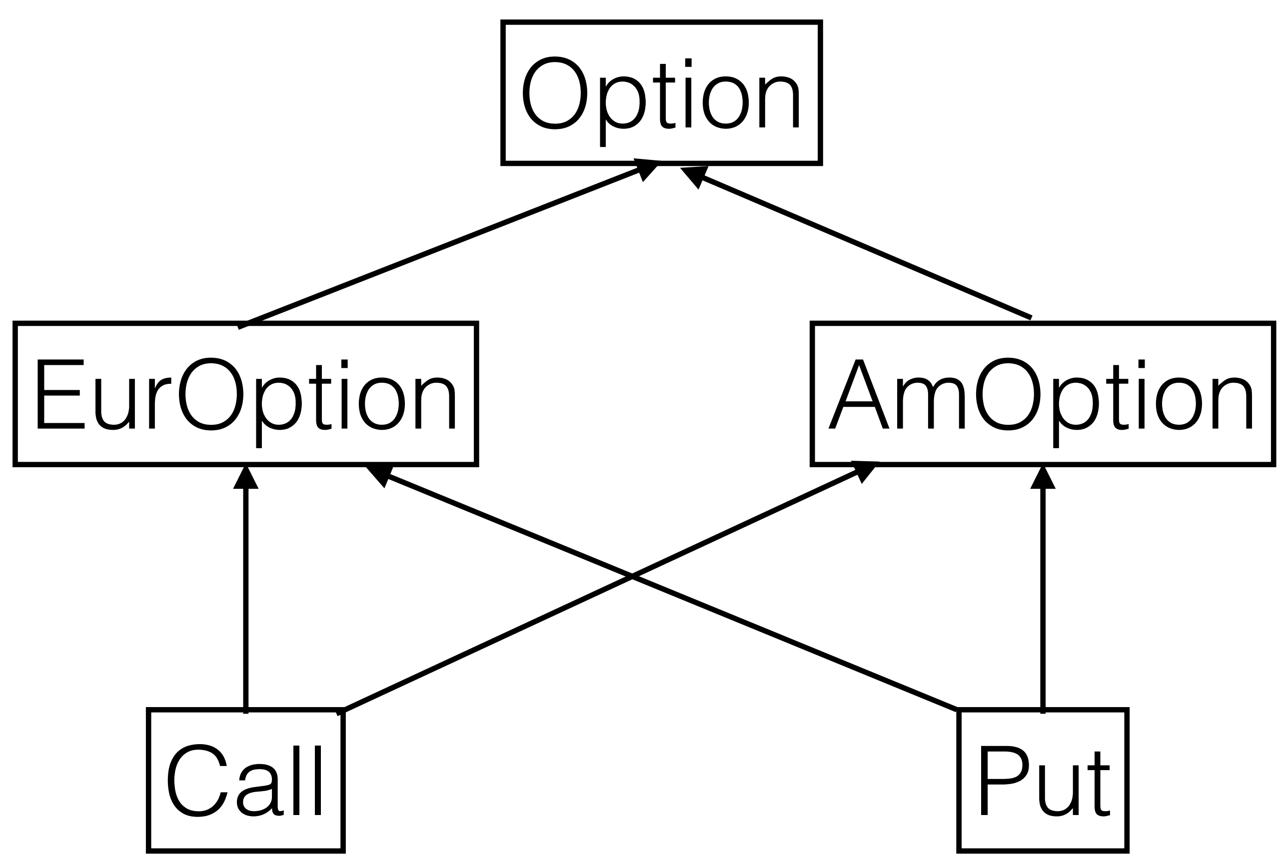
Things to note:

- `EurOption` and `AmOption` are similar apart from pricing algorithm
- A call/put can be either American or European. Put and Call classes inherit features from both
- Public vs private inheritance
- Note use of templates and STL library (more on templates later)

Virtual Inheritance: `Options08.h, Options08.cpp`

Things to note:

- What is virtual inheritance?
- Why use virtual inheritance?



Class Templates: `BinLattice01.h`, `BinLattice02.h` `Options09` `Main14.cpp`

Things to note:

- What problems do templates solve? Compare `BinLattice01.h with BinaLattice02.h`
- How are templates used? See `Options09.h` and `Options09.cpp`

Exercise 3.1: Modify the PriceByCRR() function in Options09.h and Options09.cpp to compute the replicating strategy for a European option in the binomial tree model using the BinLattice<> class template to store the stock and money market account positions in the replicating strategy at the nodes of the binomial tree.

The portfolio belonging to the replicating strategy created at time n-1, node i and help during the n-th time step, that is, until time n, consists of stock and money market account positions

$$x(n,i) = \frac{H(n,i+1) - H(n,i)}{S(n,i+1) - S(n,i)} \quad y(n,i) = \frac{H(n-1,i) - x(n,i)S(n-1,i)}{(1+R)^{n-1}}$$

for n=1,2,…,N and i=0,1,….n-1. where S(n,i) and H(n,i) denote the stock and option prices at time n, node i.

Exercise 3.2: The binomial model can be employed to approximate the Black-Scholes model. One of the several possible approximation schemes is the following. Divide the time interval [0,T] into N steps of length h=T/N, and set the parameters of the binomial model to be

$$U = \exp\left(\left(r + \sigma/2\right)h + \sigma\sqrt{h}\right) - 1$$

$$D = \exp\left(\left(r + \sigma/2\right)h - \sigma\sqrt{h}\right) - 1$$

$$R = \exp\left(rh\right) - 1$$

where σ is the volatility and r is the continuously compounded interest rate in the Black-Scholes model.

Develop code to compute the appropriate price for an American put option in the Black-Scholes model by means of the binomial tree approximation.