Agent based models in Python

Mario Castro
Universidad Pontificia Comillas

marioc@comillas.edu

Anatomy of an agent

Business and Organizations

- Manufacturing Operations
- Supply chains
- Consumer markets
- Insurance industry

Economics

- Artificial financial markets
- Trade networks

Infrastructure

- Transportation/traffic
- Electric power markets
- Hydrogen infrastructure

Crowds

- Pedestrian movement
- Evacuation modeling

Society and Culture

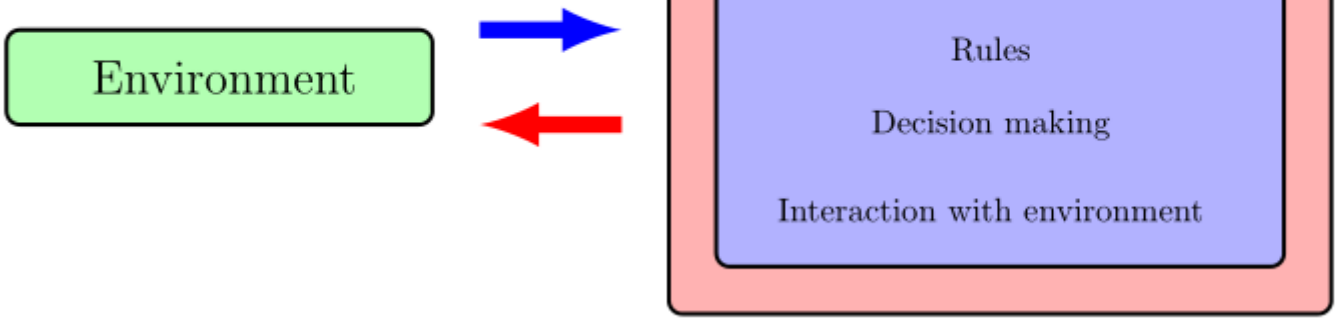
- Ancient civilizations
- Civil disobedience
- Social determinants of terrorism
- Organizational networks

Military

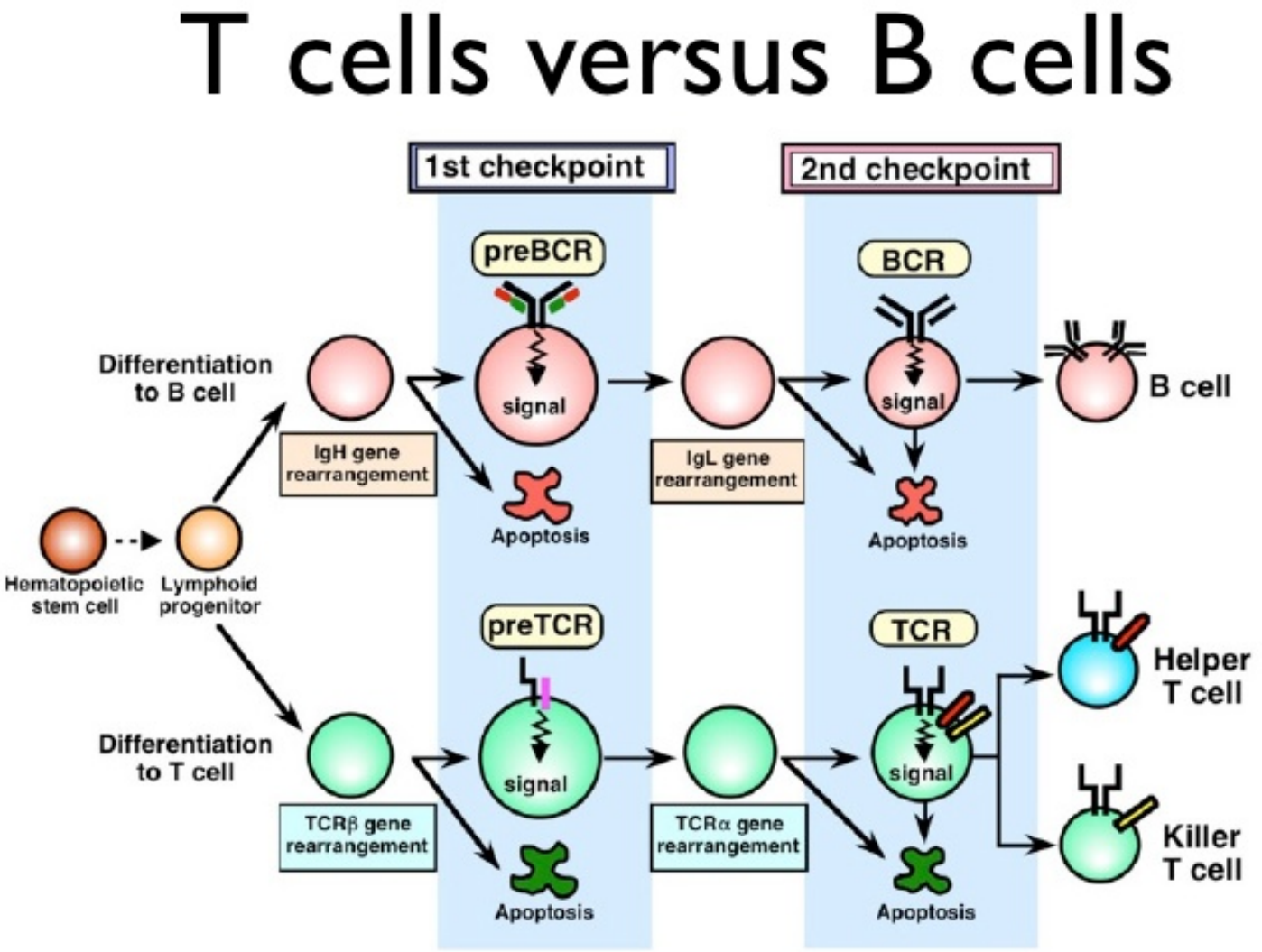
- Command & control
- Force-on-force

Biology

- Population dynamics
- Ecological networks
- Animal group behavior
- Cell behavior and sub cellular processes



Tutorial exampke:
The lifecycle of lymphocytes



But how?

Object oriented programming

Overview of OOP Terminology

- **Class**:
  - A user-defined prototype for an object that defines a set of attributes that characterize any object of the class. The attributes are data members (class variables and instance variables) and methods, accessed via dot notation.
- **Class variable**:
  - A variable that is shared by all instances of a class. Class variables are defined within a class but outside any of the class's methods. Class variables aren't used as frequently as instance variables are.
- **Function overloading**:
  - The assignment of more than one behavior to a particular function. The operation performed varies by the types of objects (arguments) involved.
- **Instance variable**:
  - A variable that is defined inside a method and belongs only to the current instance of a class.
- **Inheritance :**
  - The transfer of the characteristics of a class to other classes that are derived from it.
- **Instance:**
  - An individual object of a certain class. An object obj that belongs to a class Circle, for example, is an instance of the class Circle.
- **Method :**
  - A special kind of function that is defined in a class definition.
- **Object :**
  - A unique instance of a data structure that's defined by its class. An object comprises both data members (class variables and instance variables) and methods.

Overview of OOP Terminology

- **Class**: A "block of code" that contains variables (*atributes*) and functions that work on those variables (*methods*)

- **Inheritance :** The transfer of the characteristics of a class to other classes that are derived from it.

Creating Classes

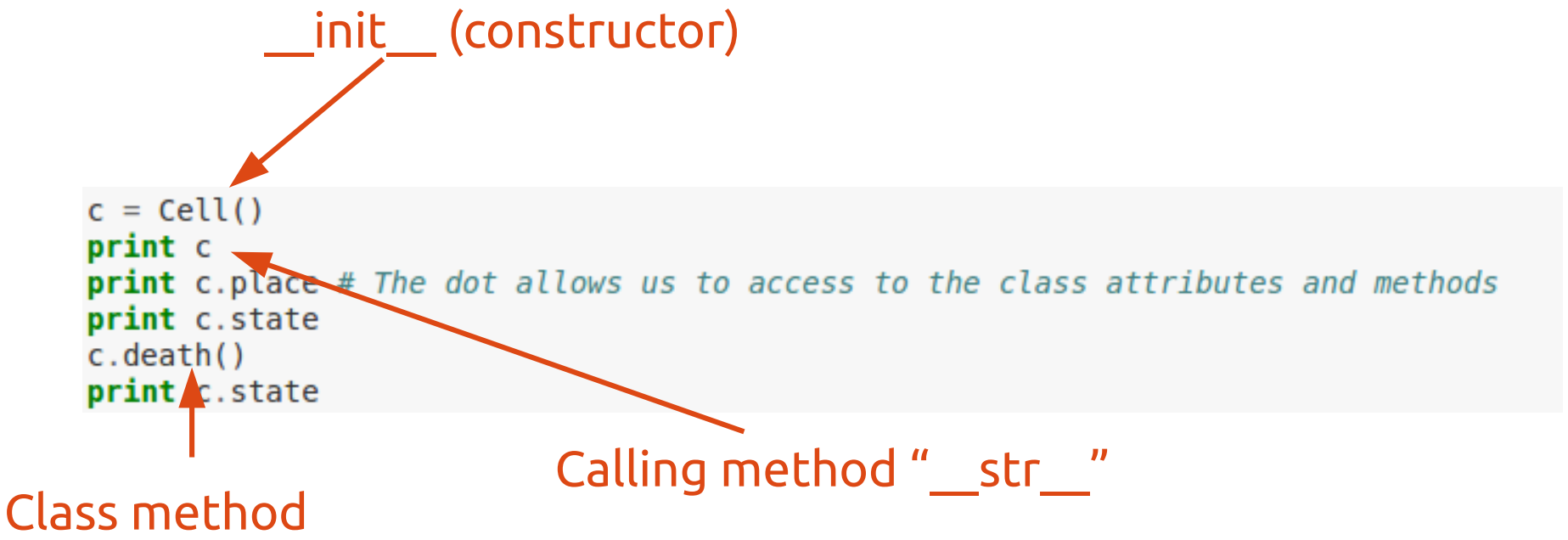
__init__ (constructor)

```
class Cell(object):
    ''' The mother of all the cells'''
    def __init__(self):
        ''' The constructor'''
        self.type = 'generic'
        self.place = 'wherever'
        self.state = 'undefined'
    def __str__(self):
        ''' When you call using print'''
        return "Type: %s Place: %s State: %s"%(self.type,self.place,self.state)
    def death(self):
        ''' Kill a cell'''
        print "It's the end of the world as we know it!"
        self.state='dead'
```

__str__ (to be used with “print”)

Object methods are **self**-ish

Instanciating classes and accessing methods



Inheritance

This means “inherit from the class Cell”

```
class lymphoid(Cell):
    def __init__(self):
        self.type = 'lymphoid progenitor'
        self.place = 'bone marrow'
        self.state = 'immature'
```

```
l = lymphoid()
print l
print l.state
l.death() # lymphoid has inherited the method "death" from its ancestor
print l.state
```

```
Type: lymphoid progenitor Place: bone marrow State: immature
immature
I'ts the end of the world as we know it!
dead
```

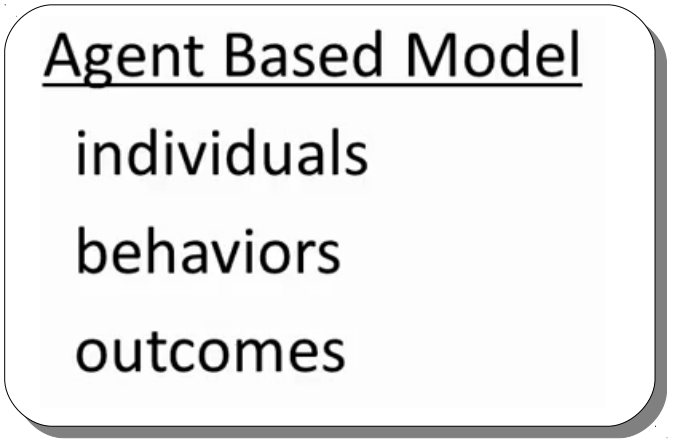
Parent (ancestor in general) method

General considerations:
answer to these questions?

- Are there several functions for the same parameters (e.g., coordinates, cells, ...)?
  - If many: use classes; If not: use functions
- Are there categories with subcategories of “species”
  - If yes: use classes; If no: use functions
- Do you have variable number of elements?
  - If yes: Use lists; If no: use tuples (faster but fixed)
- Can you vectorize the data?
  - Try always to answer yes in any case

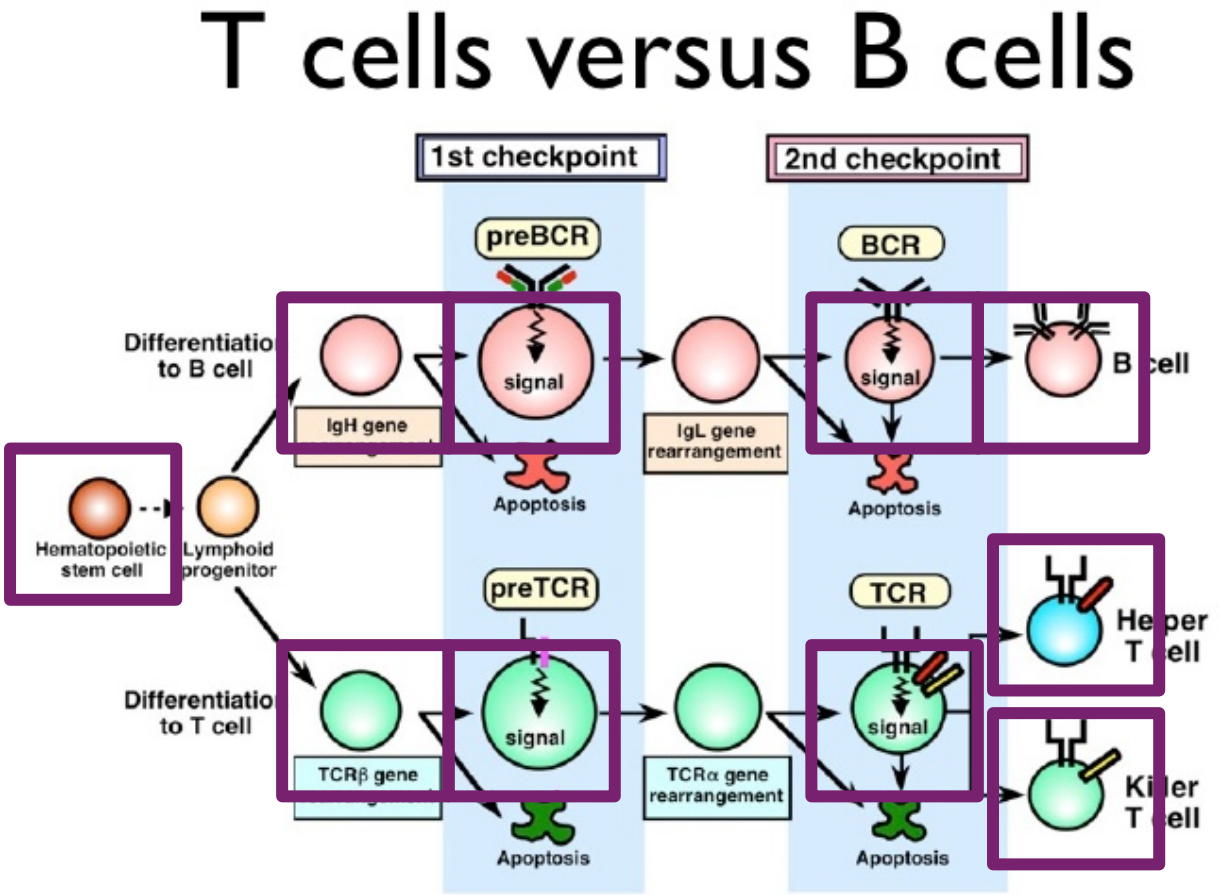
How to create an ABM

1) Name the parts (agents): Cells, humans, viruses, … and their attributes
2) Identify the relationships
3) Sketch the “logic”
4) Code
5) Understand/evaluate the outcome

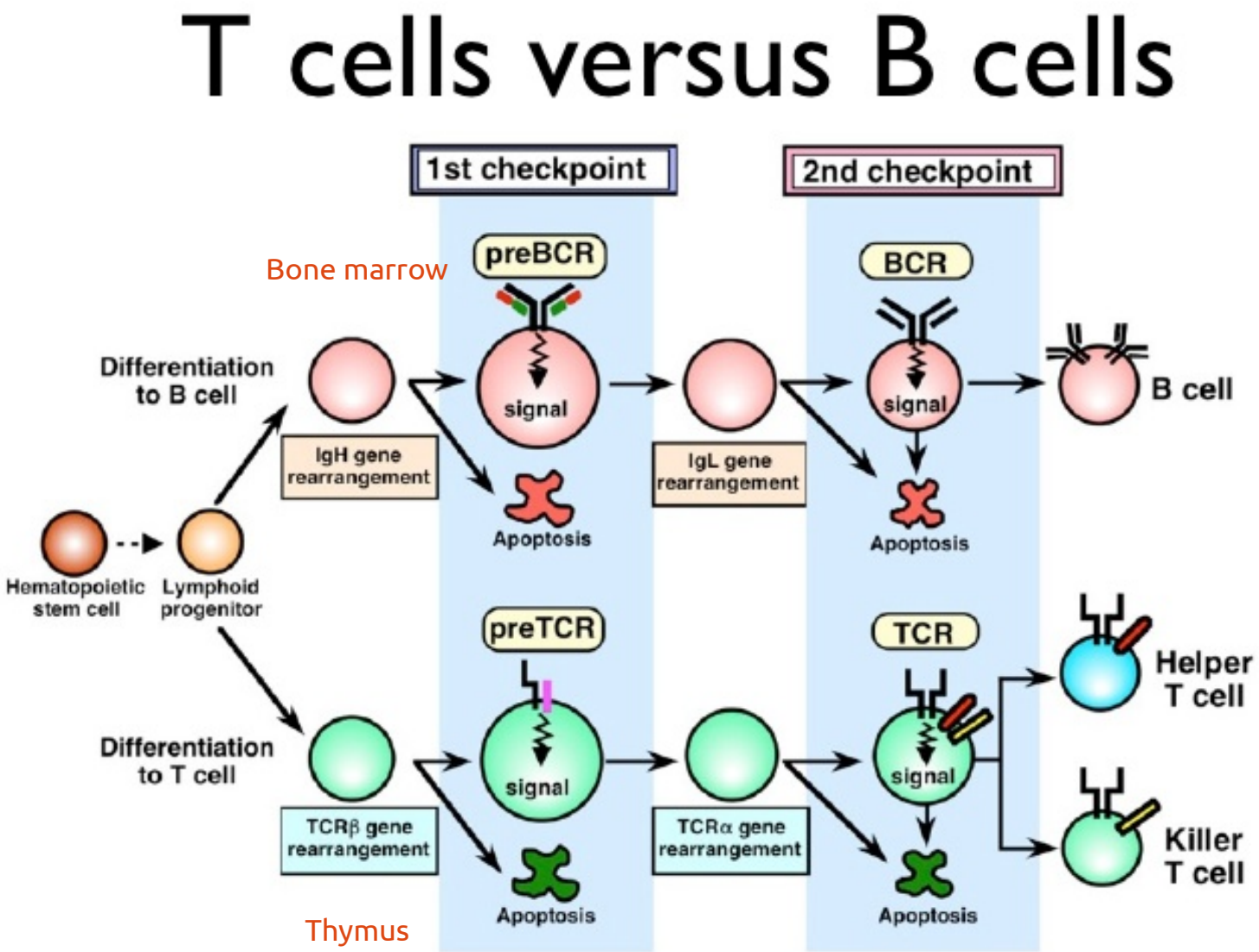


1) Name the partes

Cells, humans, viruses, …and their attributes

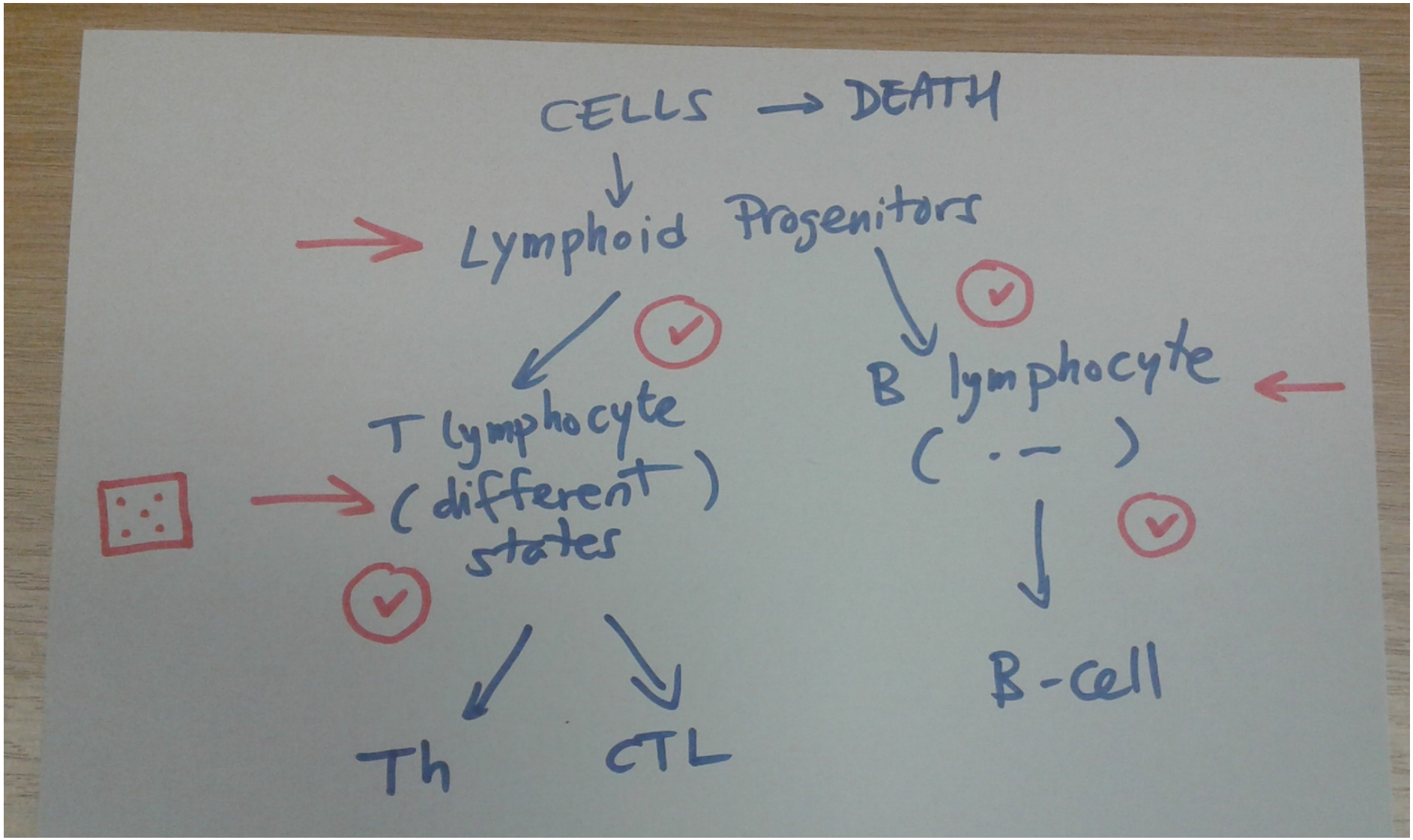


2) Identify the relationships



Bone marrow

3) Sketch the “logic”



4) (let’s) Code!!!!!

Scientific software engineering

1) Get it right.
2) Test it's right.
3) Profile if slow.
4) Optimise.
5) Repeat from 2.

Problems with motion:
T-cells targetting Dendritic cells

- Random search

https://www.youtube.com/watch?v=jgJKaP0Sj5U

- Chasing

https://www.youtube.com/watch?v=KxTYyNEbVU4