Engineer-aligned Artificial Intelligence

Sebastijan Dumančić, TU Delft

Systems Workforce Summit, Delft 2024

TU Delft

Robustly designed systems are reliable, verifiable, and guarantee desired properties

Robustly designed systems are reliable, verifiable, and guarantee desired properties

Modern-day AI, though effective, is black-box, unverifiable, and offers no guarantees

A different perspective...

On building interpretable, reliable, and verifiable AI

through programmatic representations and program synthesis

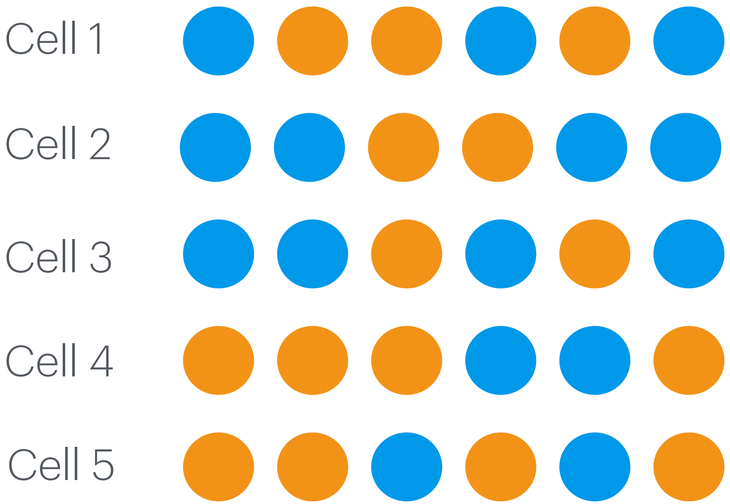
Read a text file into memory

```
def read(fname):
    With open(fname, 'r') as f:
        contents = f.readlines()
    return contents
```

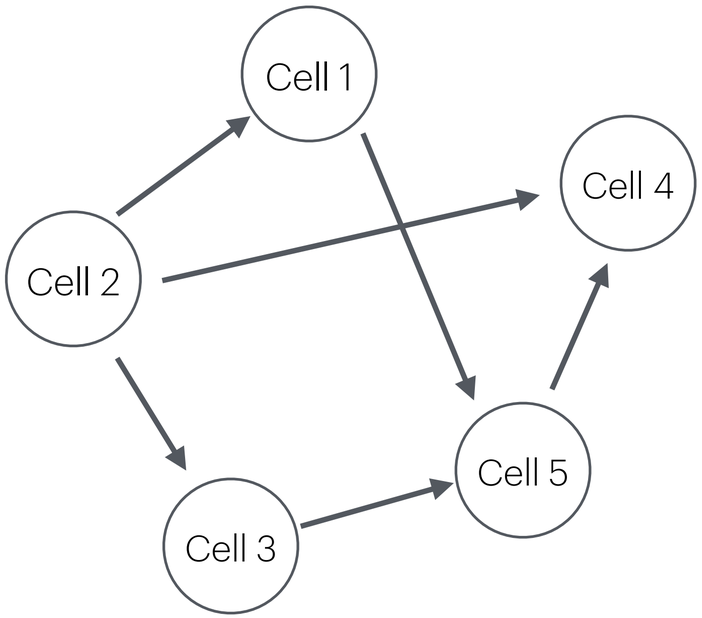
```
(Union
 (Cylinder [1, 5, 5])
 (Fold Union
  (Tabulate (i 6)
   (Rotate [0, 0, 60i]
    (Translate [1, −0.5, 0]
     (Cuboid [10, 1, 1])))))))
```

Discovering dynamical systems

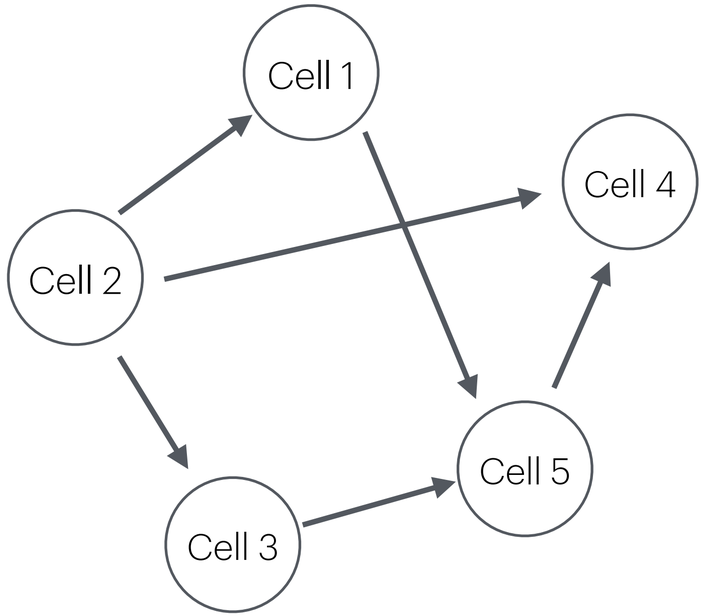
From observations of
cell/protein/gene behaviour



To a program describing
a dynamical system



Programs as digital twins



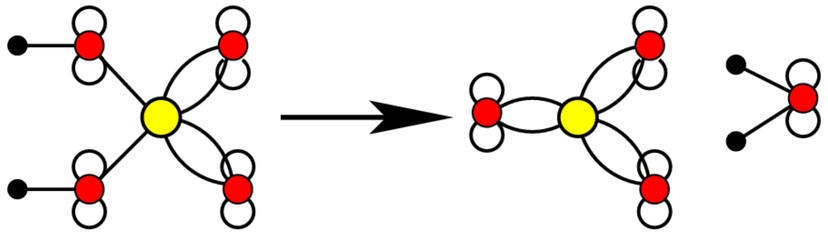
The discovered dynamical system is a **digital twin**

Understandable and controllable

Easy to verify if requirements are satisfied

There are many languages in the wild

Chemical reaction networks



System engineering languages

OMG SYSTEMS MODELING LANGUAGE SysML™

AADL

Programming language

```
def say_hello(who):
    print("Hello", who)
```

Graphical design languages

AUTOCAD®

Program synthesis for system engineering?

Which formalisms do you use you describe design on complex systems?

Many formalisms can be considered programming languages

“Programs" can be constructed with costs, safety, feasibility in mind

Program repair: repairing designs to satisfy requirements