

Markov chains

Preface

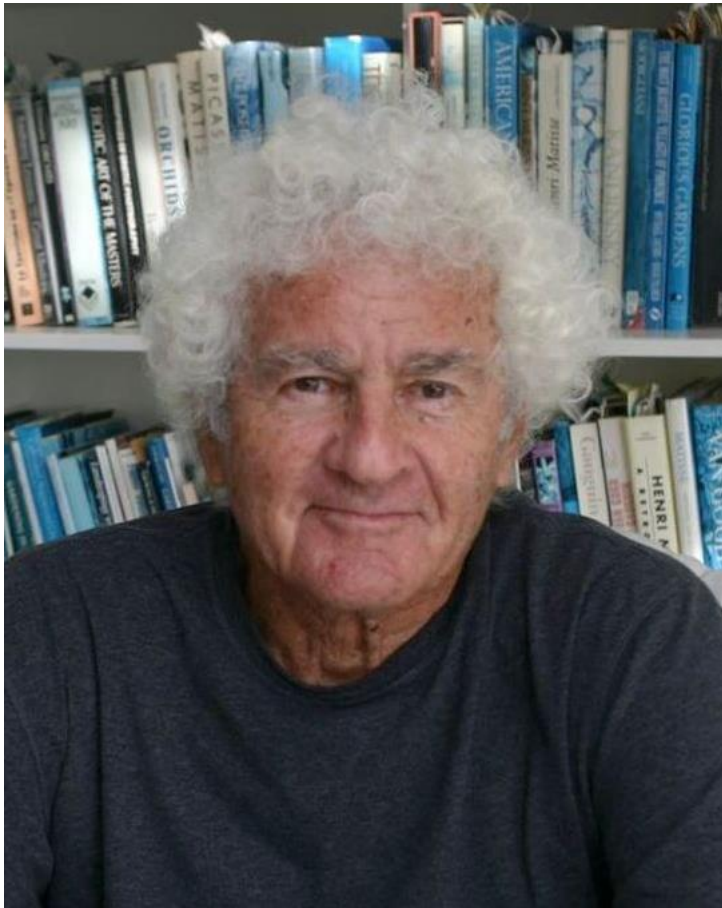
John Lennon had a difficult relationship with his mother - Julia Lennon.



When their father, Alfred Lennon, left them, Julia began to look for a life partner and began to pay less attention to John Lennon, which caused him psychological trauma.

The injury worsened when Julia Lennon was killed by a drunken police officer in a car accident.

Lennon was helped by psychologist Artur Yanov with his "[Primal Scream](#)" therapy.



Artur Janov helped Steve Jobs with a similar problem.

He was an honest psychologist who treated the sick (and successfully) for more than 30 years,

and not even for 3 years,

and not even 3 months

but in three weeks.

And very cheap.

For the first 2 weeks, the patient recalled his grievances towards his mother and expressed them as best he could.

Sometimes he cursed, called bad names.

And on the third week, Yanov explained to the patient that his mother could and injured him and was bad, but this is already in the past.

And now the whole future depends only on him.

We just need to act, not looking back at the past.

Don't waste your time and energy on this.

And, as a mathematician, I will now try to prove the same to you with the help of an automaton called Markov chains.

Maybe just like Janov helped John Lennon and Steve Jobs to successfully fulfill themselves, so today I will help you to successfully fulfill your natural potential – do you best!

Do Your Best!



This mechanism was used by Shannon for approximation.

<https://people.math.harvard.edu/~ctm/home/text/others/shannon/entropy/entropy.pdf>

4. GRAPHICAL REPRESENTATION OF A MARKOFF PROCESS

Stochastic processes of the type described above are known mathematically as discrete Markoff processes and have been extensively studied in the literature.⁶ The general case can be described as follows: There exist a finite number of possible “states” of a system; S_1, S_2, \dots, S_n . In addition there is a set of transition probabilities; $p_i(j)$ the probability that if the system is in state S_i it will next go to state S_j . To make this Markoff process into an information source we need only assume that a letter is produced for each transition from one state to another. The states will correspond to the “residue of influence” from preceding letters.

The situation can be represented graphically as shown in Figs. 3, 4 and 5. The “states” are the junction

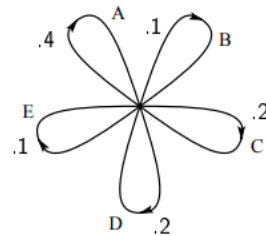
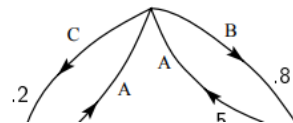


Fig. 3—A graph corresponding to the source in example B.

points in the graph and the probabilities and letters produced for a transition are given beside the corresponding line. Figure 3 is for the example B in Section 2, while Fig. 4 corresponds to the example C. In Fig. 3

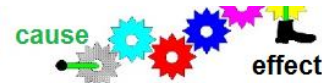
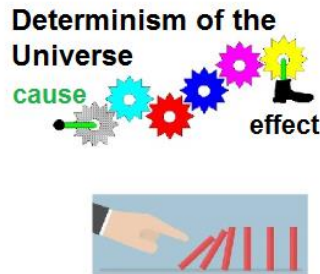


And we'll do it a little differently. This automaton, which has become so popular for modeling

various processes, arose as a result of a metaphysical philosophical dispute about free will

Six proof of Kant [Critique of Practical Reason]

Everything in this universe is subject to cause and effect.



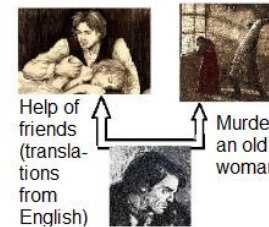
Lion is hungry ⇒ lion kills zebra



The lion is innocent. He has no choice



Man has freedom of choice



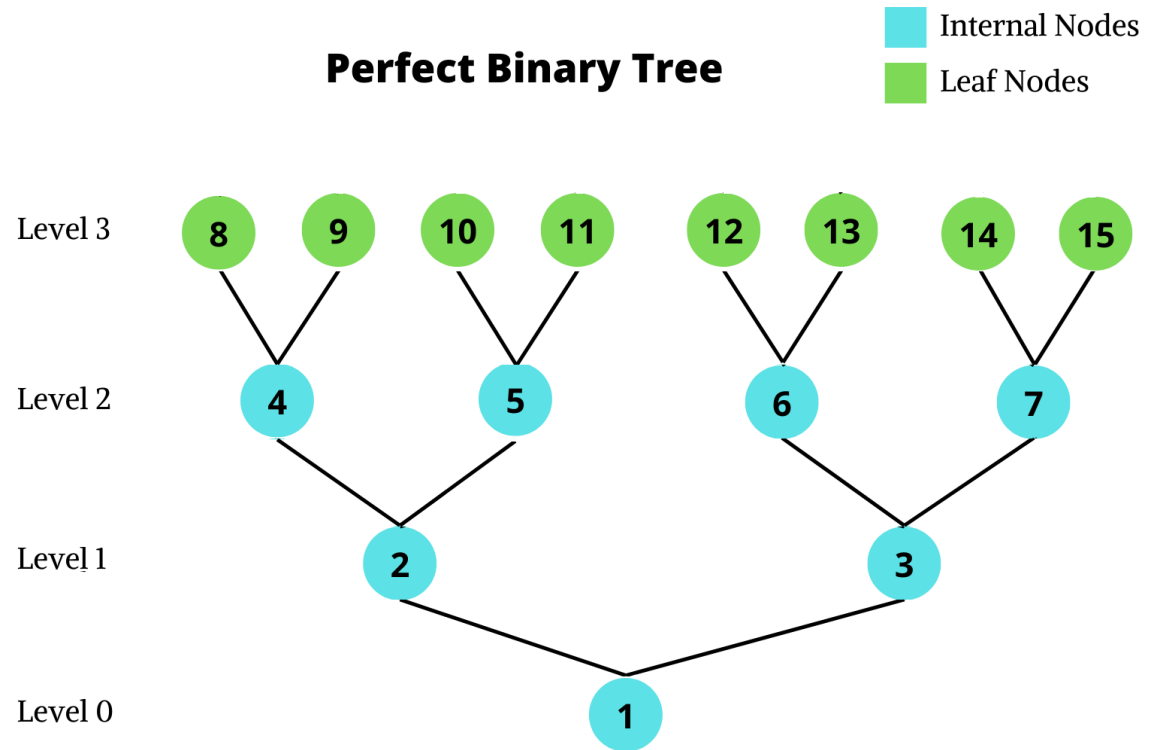
Man is not determined! He is free

Practice shows that a person is subject to trial according to:

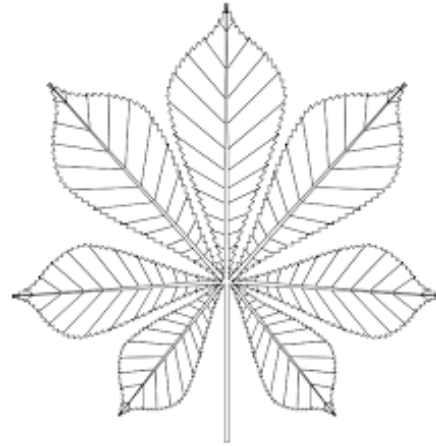
- criminal laws
- moral,
- ethical



The fact is that, when observing the natural world, many of us notice a somewhat beautiful dichotomy [dai'kɒtəmi].



No two things are ever exactly alike, but they all seem to follow some underlying form.



Plato believed that the **true forms** of the Universe were hidden from us (usual people).

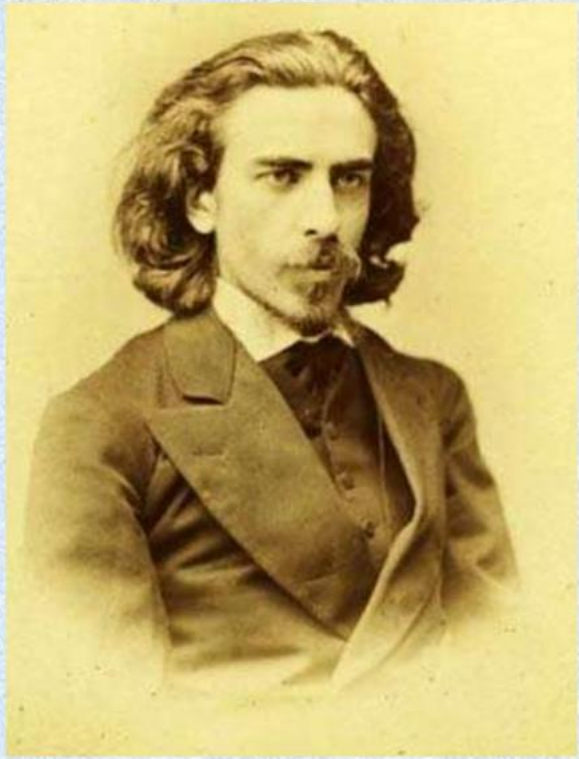


Plato

(Greek: Πλάτων *Plátōn*;

428/427 or 424/423 – 348/347

BC)

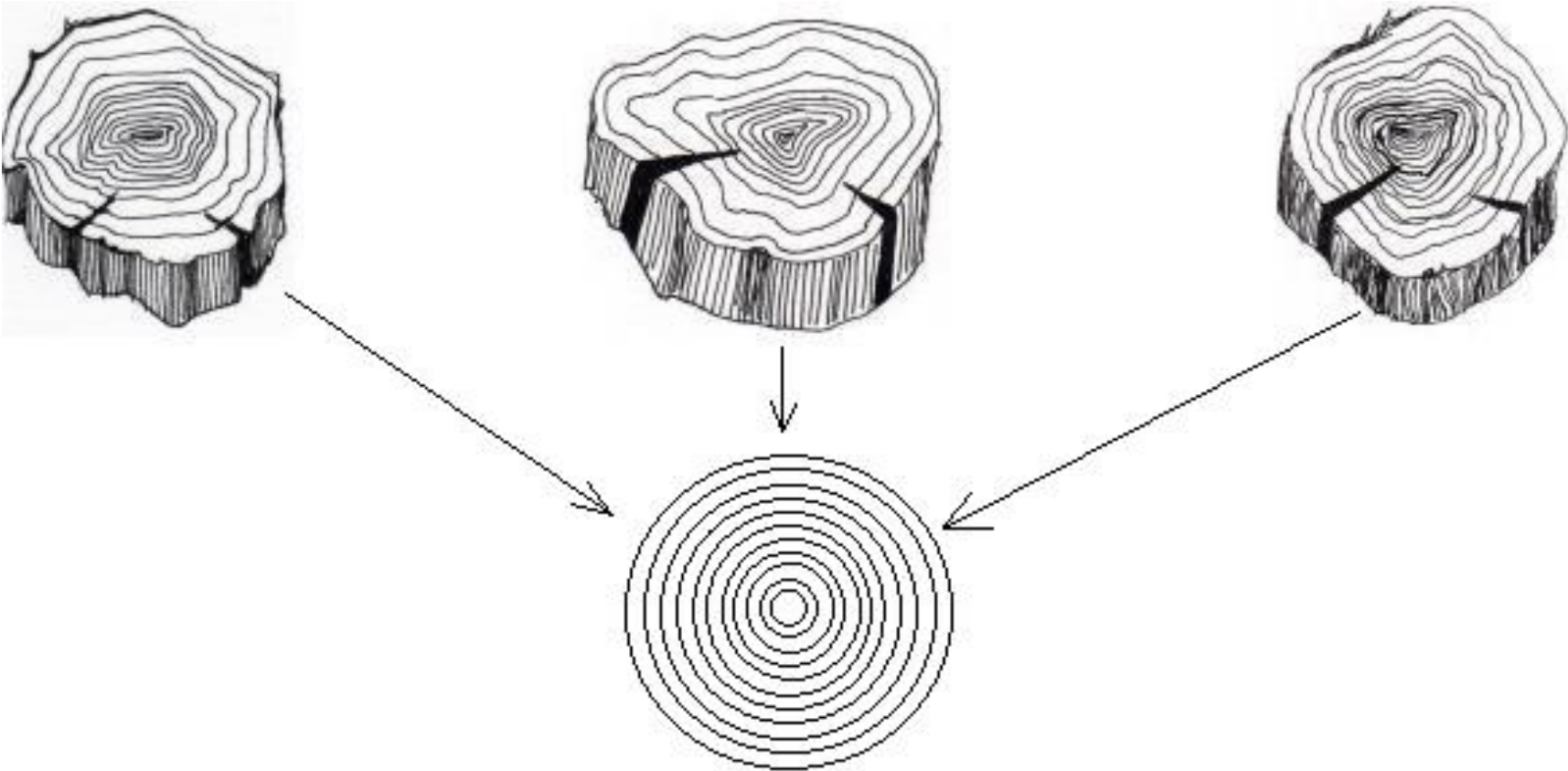


**ВЛАДИМИР
СЕРГЕЕВИЧ СОЛОВЬЕВ
1858 - 1900**

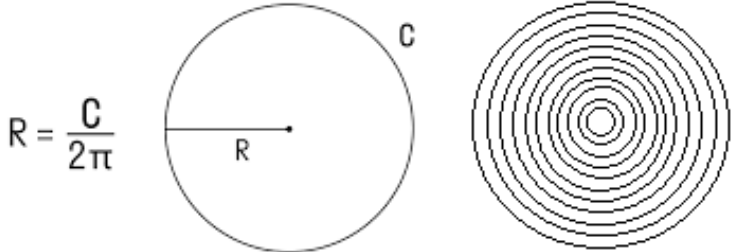
**Милый друг, иль ты не видишь,
Что всё видимое нами –
Только отблеск, только тени
От незримого очами?**

**Милый друг, иль ты не слышишь,
Что житейский шум трескучий –
Только отклик искаженный
Торжествующих созвучий?**

Through observation of the natural world, we could merely acquire approximate knowledge of them.



They were hidden blueprints (pure forms).



hidden blueprint



implementations

принц на белом коне (prince charming)

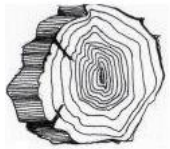


hidden blueprint
(pure form)

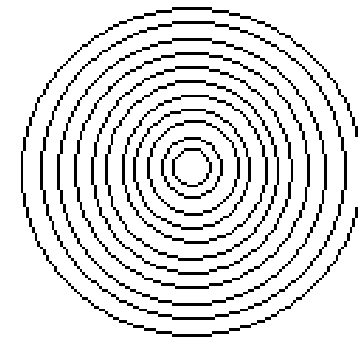
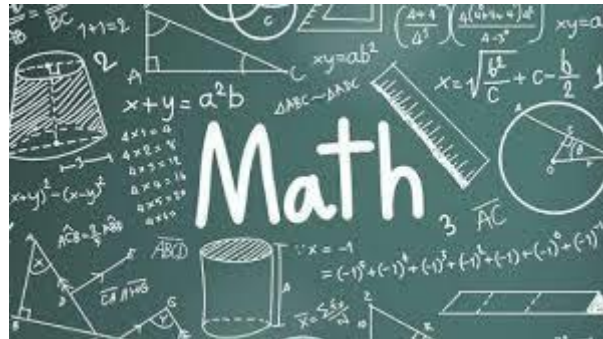
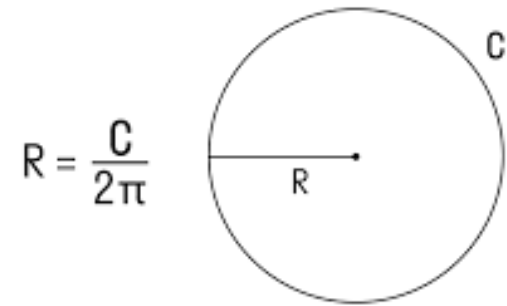


implementation

The pure forms were only accessible through abstract reasoning of philosophy and mathematics.

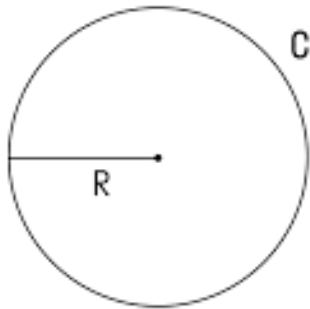


Abstract reasoning

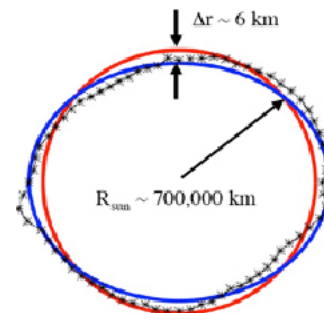
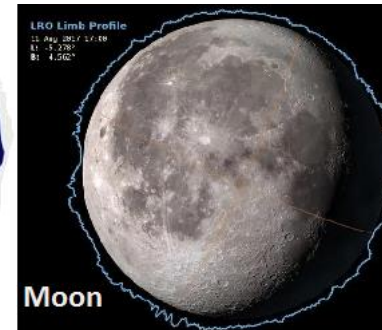


For example, the circle he describes as that which has the distance from its circumference to its center everywhere equal.

$$R = \frac{C}{2\pi}$$



Yet we will never find a material manifestation of a perfect circle

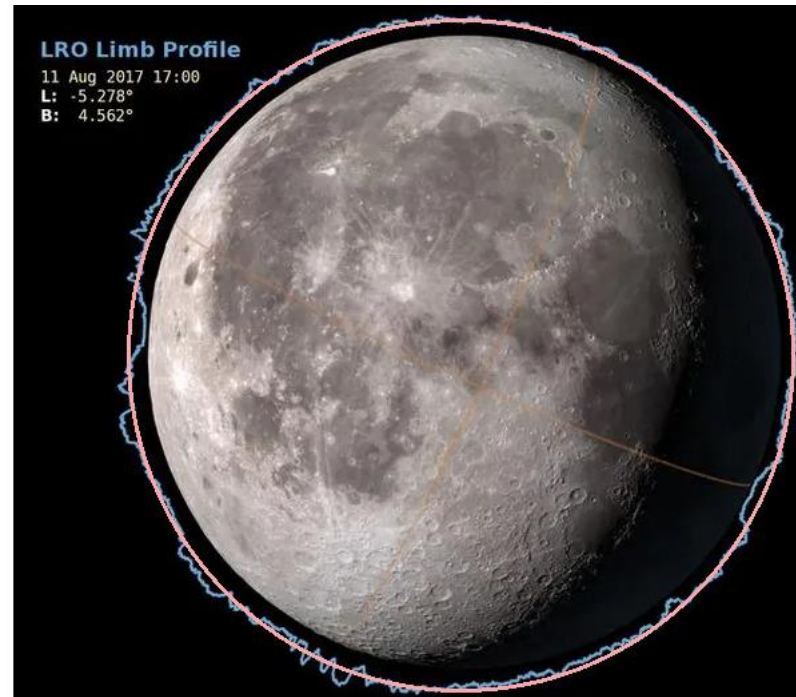


NASA Spacecraft Finds the Sun is Not a Perfect Sphere

https://www.nasa.gov/topics/solarsystem/features/oblate_sun.html

This Platonic focus on abstract pure forms (hidden blueprints) remained popular for centuries. It wasn't until the 16th century when people tried to embrace the messy variation in the real world and apply mathematics to reveal underlying patterns.

Moon
(pure
form)



Bernoulli refined the idea of expectation.

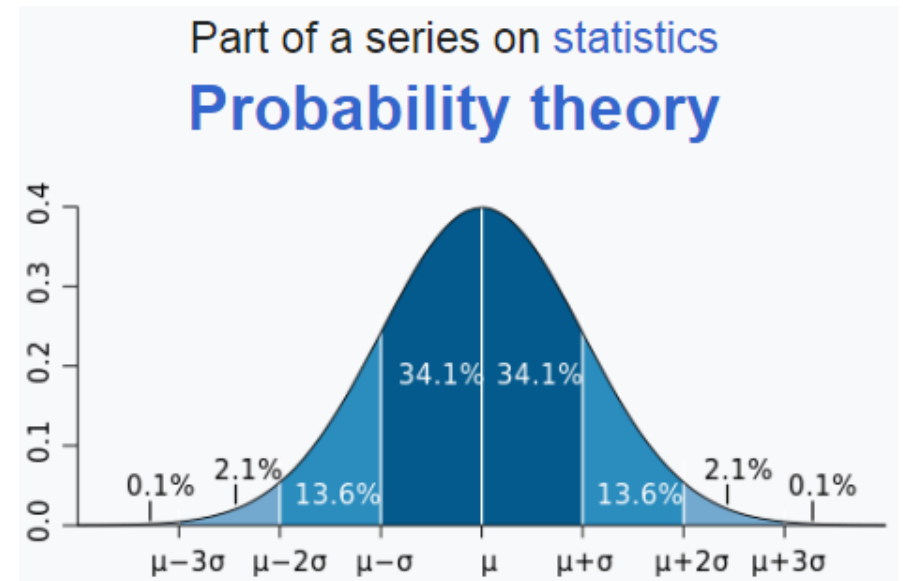


(1700 – 1782)

He was focused on a method of accurately estimating the unknown probability of some event based on the number of times

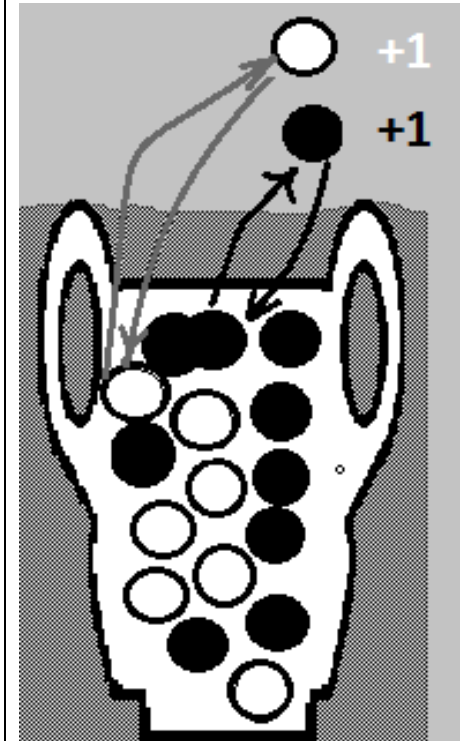
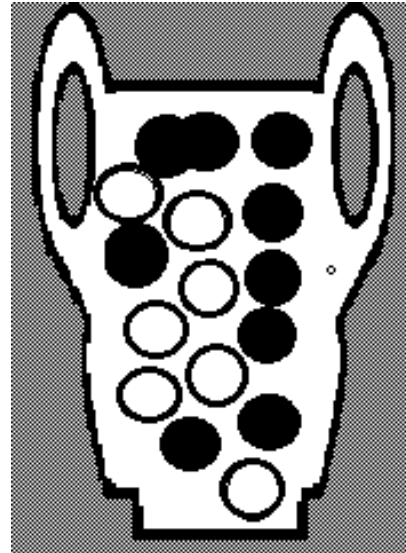
the event occurs in independent trials.

He uses a simple example.



Suppose that without your knowledge, 3,000 light beans and 2,000 dark beans are hidden in an urn,
and that to determine the ratio of white versus black by experiment,
you draw one bean after another, with replacement, and note how many times a white bean is drawn versus black.

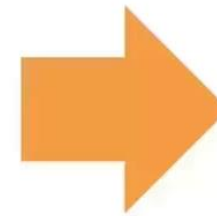
He went on to prove that the expected value of white versus black observations will converge on the actual ratio as the number of trials increases, known as the weak law of large numbers.



--	--

He concluded by saying, "If observations of all events be continued for the entire infinity, it will be noticed that everything in the world is governed by precise ratios and a constant law of change."

This idea was quickly extended as it was noticed that not only did things converge on an expected average,



Of those, over
16%
reported having a
mental illness in the
past year²

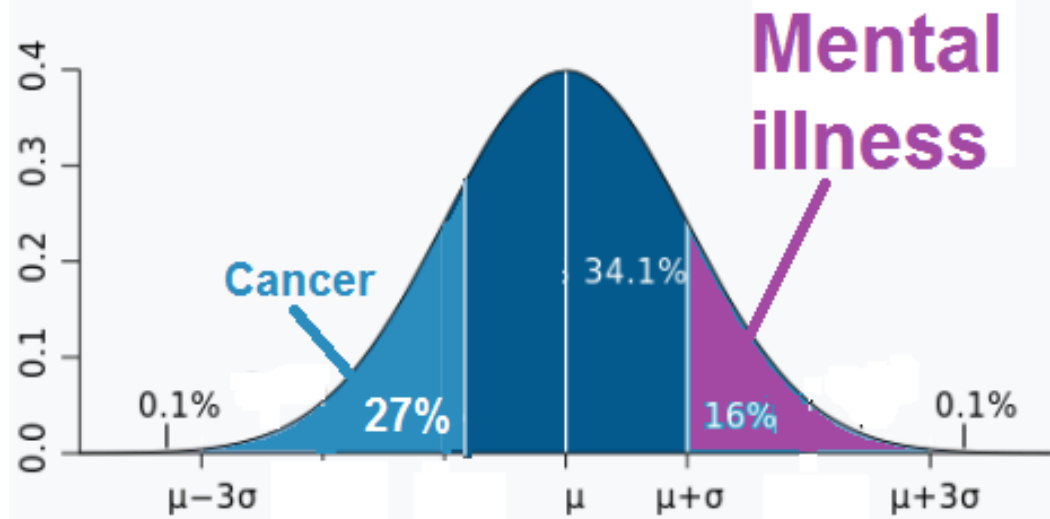
but the probability of variation away from averages also follow a familiar, underlying shape, or distribution.

CANCER DEATHS IN THE UNITED STATES

Cancer death rates
dropped 27% from
2001 to 2020.

27%

Part of a series on statistics Probability theory



A great example of this is Francis Galton's bean machine.

https://upload.wikimedia.org/wikipedia/commons/transcoded/d/dc/Galton_box.webm/Galton_box.webm.720p.vp9.webm



Imagine each collision as a single independent event, such as a coin flip.

After 10 collisions or events, the bean falls into a bucket representing the ratio of left versus right deflection, or heads versus tails.

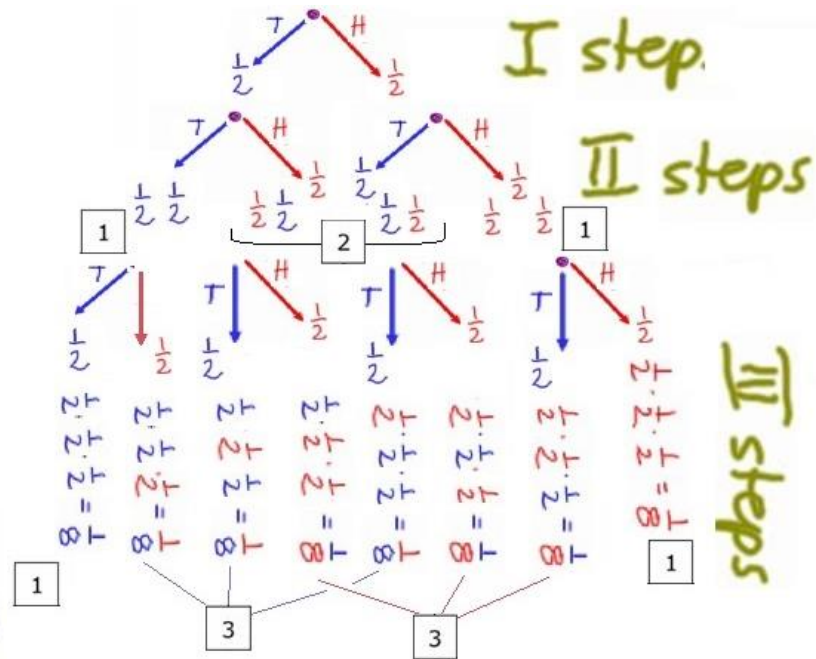
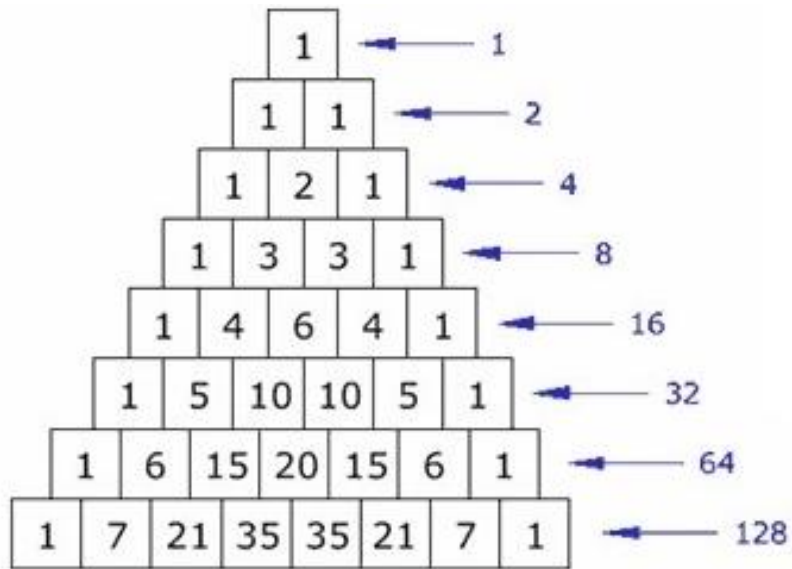
After 10 collisions or events, the component falls into a basket representing ratio of left and right deflection or heads and tails.



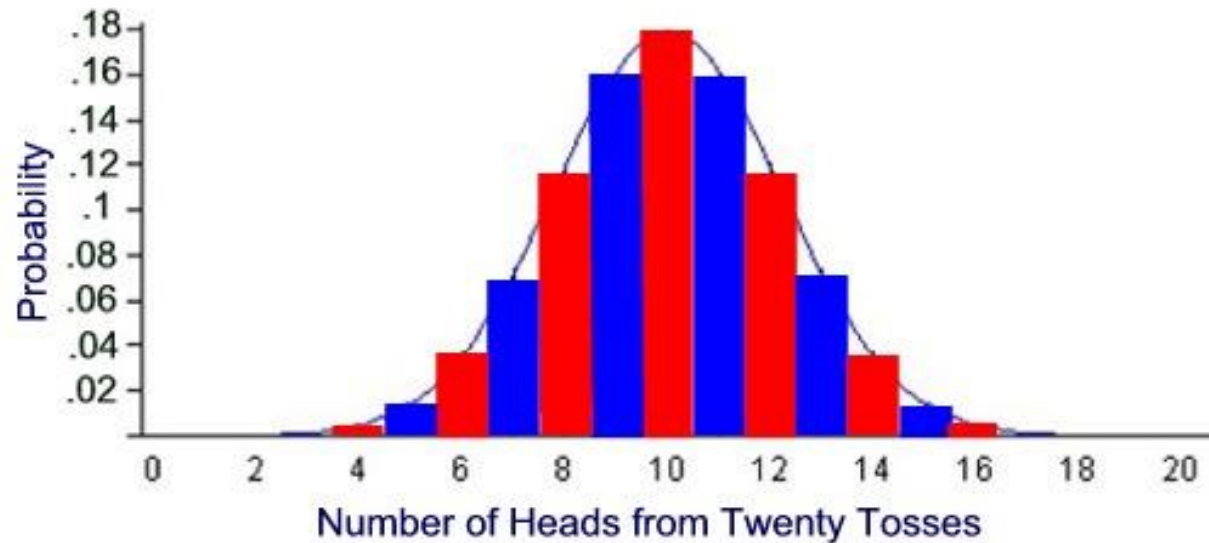
Binomial Distribution

[bī-'nō-mē-əl ,di-strə-'byü-shən]

The likelihood of observing a certain outcome when performing a series of tests for which there are only two possible outcomes, such as getting heads or tails in a coin toss.



This general curvature, known as the binomial distribution, seems to be the ideal shape, as it appears everywhere whenever you look at the variation of a large number of random trials.



It seems the average fate of these events is somehow predetermined, known today as the central limit theorem.

This was a dangerous philosophical idea to some.



Некрасов
Павел Алексеевич
(1853—1924)

Pavel Nekrasov, originally a theologian by training, later took up mathematics and was a strong proponent of the doctrine of free will.

He didn't like the idea of us having this predetermined statistical fate.

He made a famous claim that independence is a necessary condition for the law of large numbers, since **independence** just describes these toy examples using beans or dice, where the outcome of previous events doesn't change the probability of the current or future events.

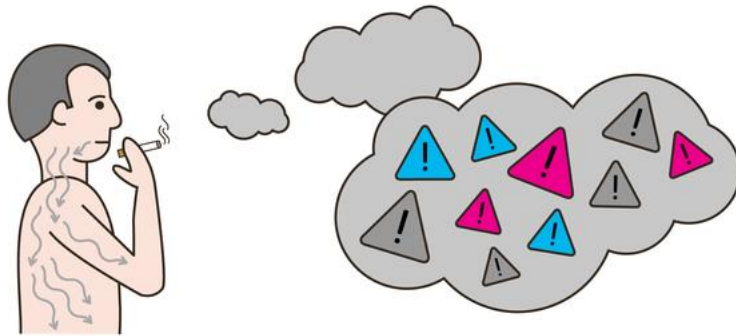
However, as we all can relate, most things in the physical world are clearly dependent on prior

outcomes, such as the chance of fire or sun or even our life expectancy. When the probability of some event depends, or is conditional, on previous events, we say they are dependent events, or dependent variables.



smoking cause cancer

1 Cigarette smoke releases over 5,000 different chemicals.

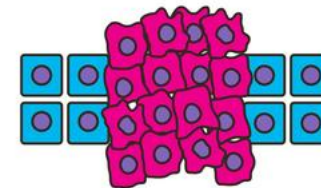
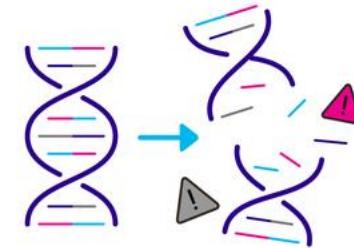


2 Harmful chemicals enter our lungs and can then affect the entire body.

3 Chemicals damage our DNA, including parts that protect against cancer.

4 Other chemicals make it harder for cells to repair DNA damage.

5 This DNA damage can cause cancer in cells.



Cause



Effect



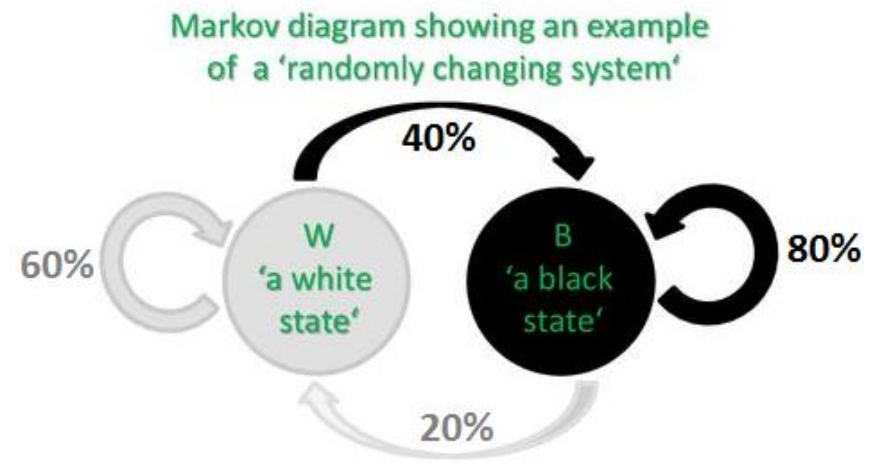
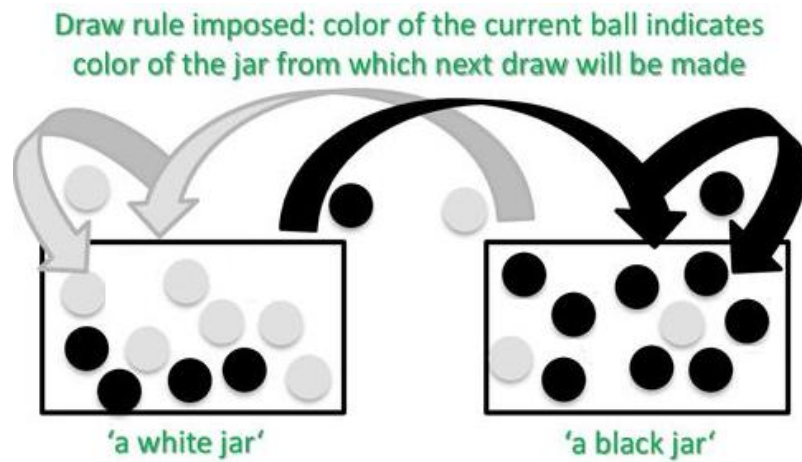
Марков Андрей Андреевич
(1856-1922)

This claim angered another Russian mathematician, Andrey Markov, who maintained a very public animosity towards Nekrasov (Because of the writer Leo Tolstoy, who was then excommunicated from the Orthodox Church - Markov, in protest, asked him to be excommunicated too).

He goes on to say in a letter that "this circumstance prompts me to explain in a series of articles that the law of large numbers can apply to dependent variables," using a construction which he brags, Nekrasov cannot even dream about.

Markov extends Bernoulli's results to dependent variables using an ingenious construction.

Imagine a coin flip which isn't independent, but dependent on the previous outcome, so it has short-term memory of one event.

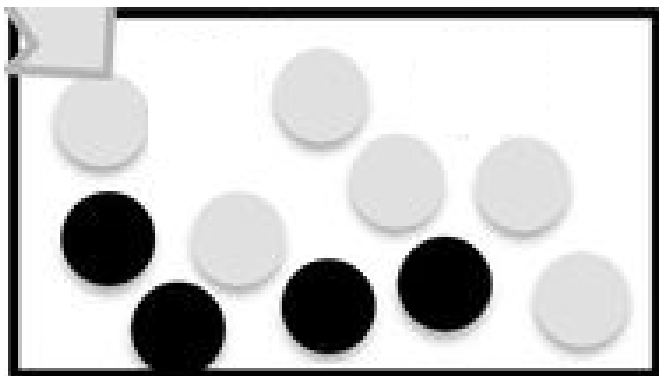


Markov matrix (simplified, based on colors)

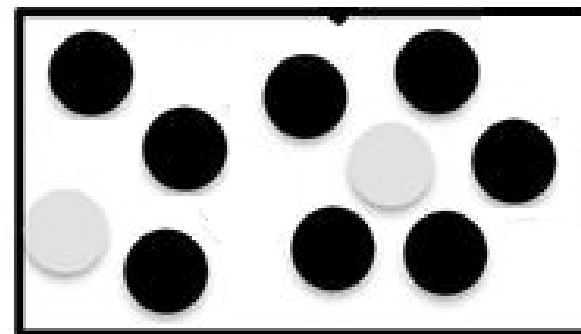
	To W	To B
From W	60%	40%
From B	20%	80%

In one state we have a 60%-40% mix of light versus dark beans

in the other state we have more dark versus light beans.



'a white jar'



'a black jar'

One jar we can call state **zero**.

Other jar we can call state **one**.

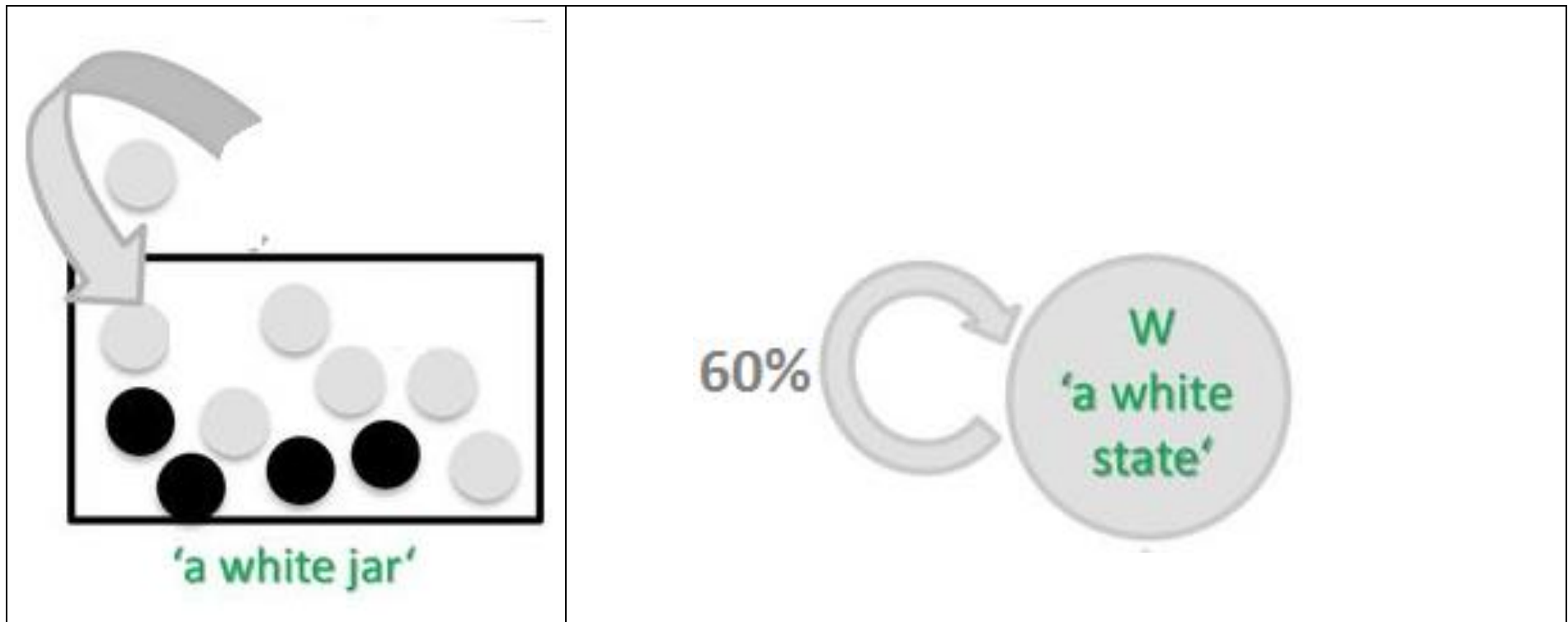
,

With this two-state machine, we can identify four possible transitions.

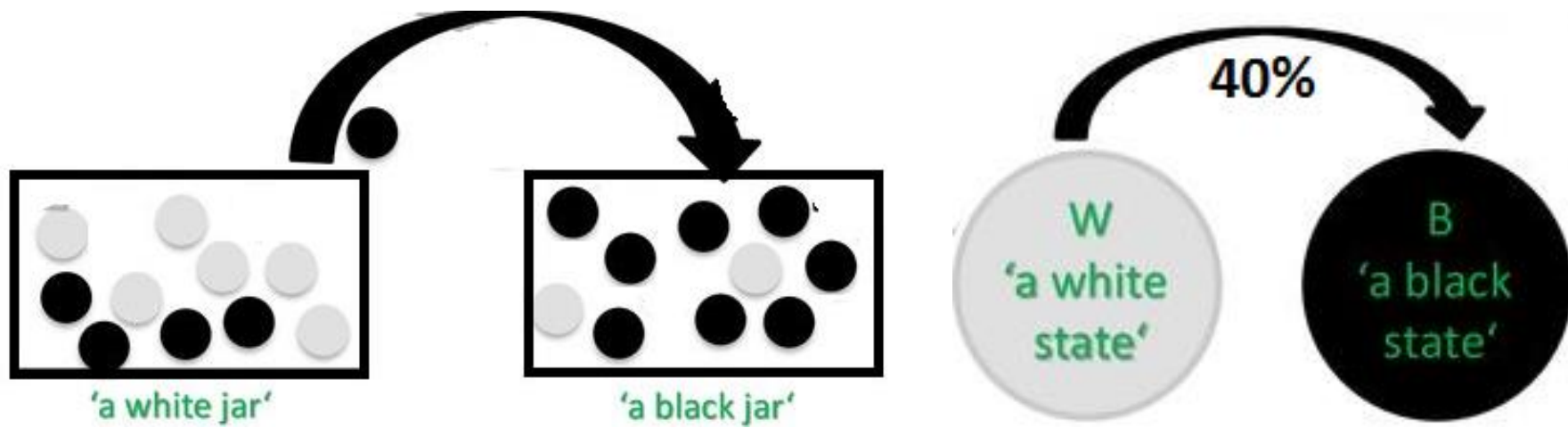
while

while in the other state we have more dark versus light.

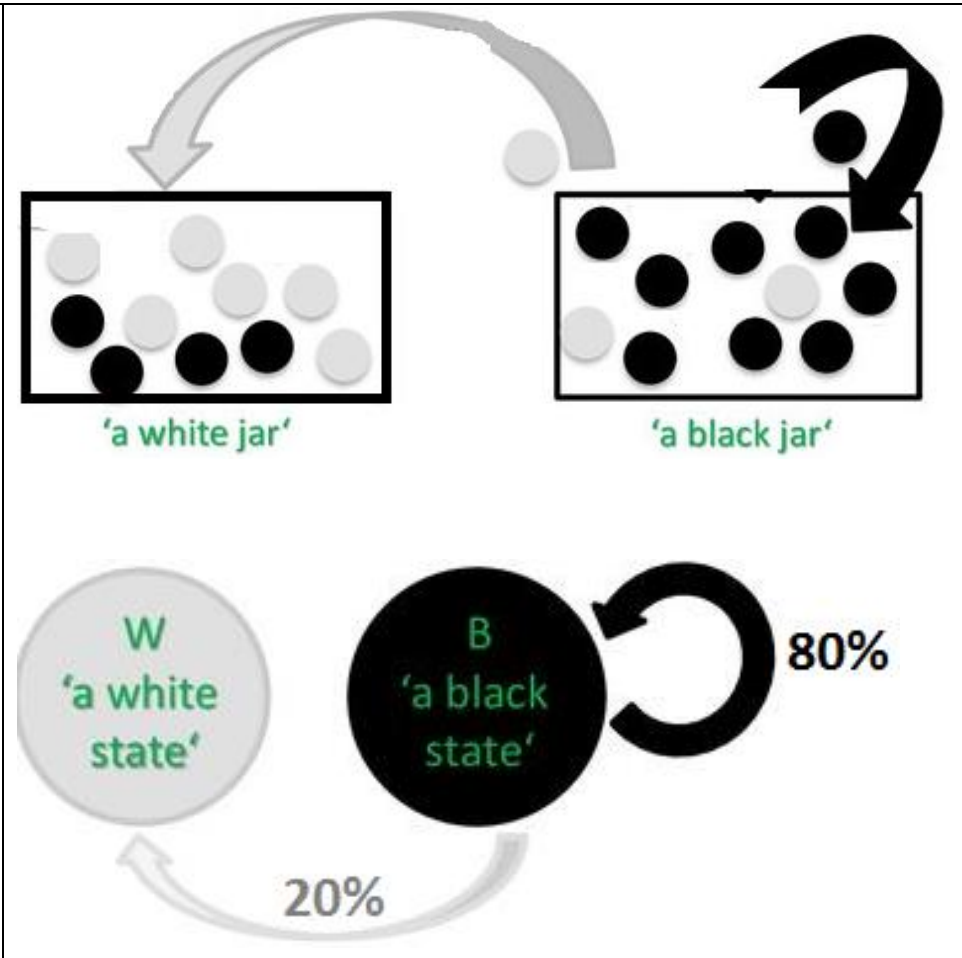
If we are in state zero and a white occurs, we loop back to the same state and select again.



If a black bead is selected, we jump over to state one

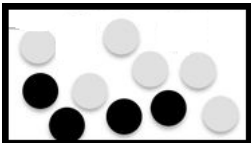
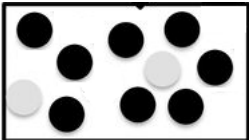




If a dark been is selected, we jump over to state one, which can also loop back on itself, or jump back to state zero if a dark is chosen.

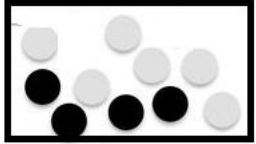
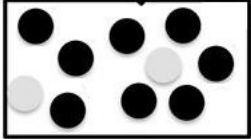

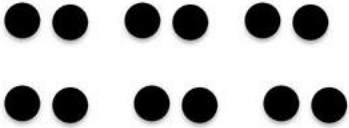


The probability of a light versus dark selection is clearly not independent here, since it depends on the previous outcome.

But Markov proved that as long as every state in the machine is reachable, when you run these machines in a sequence, they reach equilibrium.

 'a white jar'	 'a black jar'
 - 8	 - 12
40%	60%

That is, no matter where you start, once you begin the sequence, the number of times you visit each state converges to some specific ratio, or a probability.

 <p>'a white jar'</p>	 <p>'a black jar'</p>
 <p>8</p>	 <p>- 12</p>
<p>40%</p>	<p>60%</p>

That is, no matter where you start (white or black jar) *(what the unfavorable conditions in your childhood no matter what mistakes your parents make in upbringing (and no matter what injuries your classmates inflict on you))*

the number of times (after 20-30 months of study at the university), all converges to some specific ratio.

Your Probability (chances of success) are the same as your more successful peers [piθ]

This simple example disproved Nekrasov's claim that only independent events could converge on predictable distributions.

But the concept of modeling sequences of random events using states and transitions between states became known as a Markov chain.

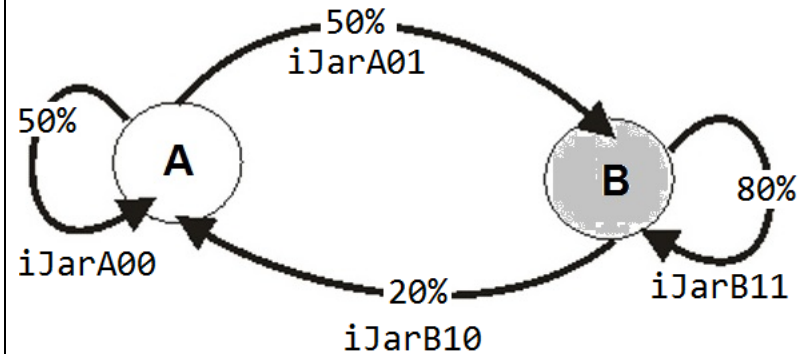
One of the first and most famous applications of Markov chains was published by Claude Shannon.

--	--

1. Writing a console program

1.1. Create a Markov chain class

```
class MarkovChain
{
    internal int iJarA00; //50%
    internal int iJarA01; //50%
    internal int iJarB10; //20%
    internal int iJarB11; //80%
```



```
public MarkovChain(int A00, int A01, int B10, int B11)
```

```
{
    iJarA00 = A00;        iJarA01 = A01;
    iJarB10 = B10;       iJarB11 = B11;
}
```

```
static void Main(string[] args)
{
    myMC = new MarkovChain(50, 50, 20, 80);
    Console.ReadLine();
}
```

1.2. Let's display the result of initialization of class `MarkovChain`

```
Console.WriteLine(myMC.iJarA00 + "," + myMC.iJarA01);
Console.WriteLine(myMC.iJarB10 + "," + myMC.iJarB11);
```

1.3. Let's divide the code into interface (input-output) and business logic

```
static void Read()
```

```
{
```

```
    myMC = new MarkovChain(50, 50, 20, 80);
```

```
        r = new Result();
```

```
    }
```

```
    static void Write()
```

```
    {
```

```
        Console.WriteLine(myMC.iJarA00 + "," + myMC.iJarA01);
```

```
        Console.WriteLine(myMC.iJarB10 + "," + myMC.iJarB11);
```

```
    }
```

```
    static void Main(string[] args)
```

```
    {
```

```
        Read();
```

```
        Write();
```

```
        Console.ReadLine();
```

```
    }
```

1.4. Let's write the Markov network operation method

MarkovChain.Run()

Let's create the Result class (for saving results)

```
class Result
{
    internal int white;    //white - number of randomly selected whites beans for iSteps
    internal int black;   //black- number of randomly selected whites beans for iSteps
    internal string s;
}
```

1.5. Let's supplement the data section of the class `MarkovChain` with new variables (service)

```
internal int iSteps; - number of steps
```

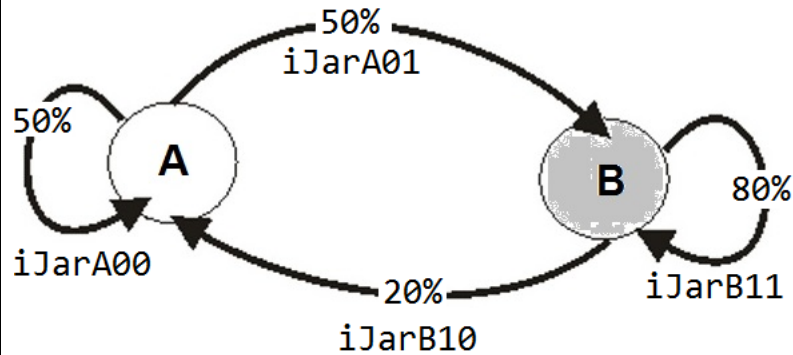
```
internal Result res; - variable (field) for storing results
```

`internal Random r;` - object (instance) of the Random class for implementing random selection

`int white;` - number of white beans pulled per `iSteps`

`int black;` - number of black beans pulled per `iSteps`

```
jarNo = 1; //Jar number A-1 B-2
int p = -1; - random number from 1 to 100,
dropping out on a step  $i \in [0, iSteps]$ .
int i = -1; - step number  $i \in [0, iSteps]$ .
```



1.6. Let's write the method JarA()

that determines whether the next choice will be made from the white jug (A - returns one), or the transition to the black jug will be made (B - returns 2)

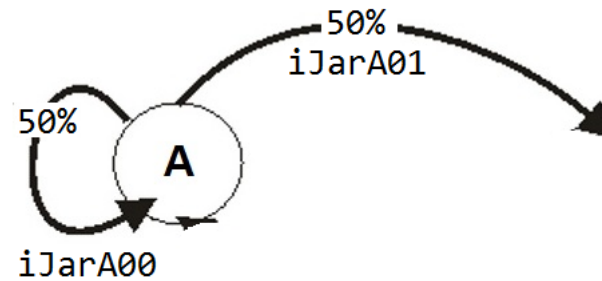
```
int JarA()
```



```

{
    p = r.Next(1, 100);
    if (p < iJarA00) return 1; //iJarA00
    else return 2;
}

```



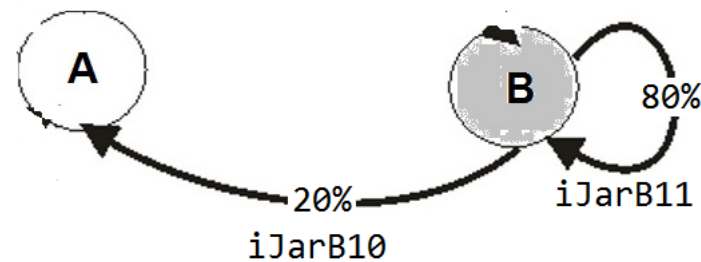
1.7. Let's write the method JarB()

that determines whether the next choice will be made from the black jug (B - returns 2), or the transition to the white jug will be made (B - returns 1)

```

int JarB()
{
    p = r.Next(1, 100);
    if (p < iJarB11) return 2; //iJarB11
}

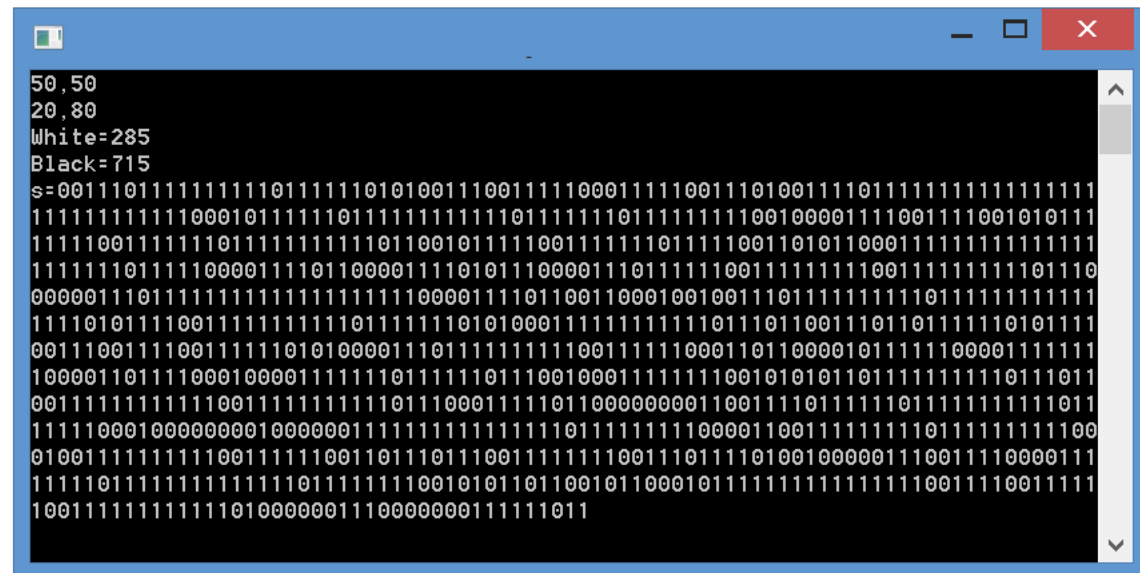
```



```
else return 1;
}
```

1.8. Let's write the method Run()

```
public Result Run()
{
    i = 0;
    jarNo = 1; //Jar A-1 B-2
    white = 0;
    black = 0;
    string s = "";
    while (true)
    {
        i++;
        s += (jarNo - 1).ToString();
        if (jarNo == 1)
        {
            white++;
            jarNo = JarA();
        }
        else
```



```
50,50
20,80
White=285
Black=715
s=001110111111110111110101001110011111000111100111010011110111111111111111
1111111111000101111110111111111111101111110111111110010000111100111100101011
1111100111110111111111011001011110011111101111100110101100011111111111111
11111101111100001111011000011110101110000111011111100111111111110111110110
0000011101111111111111111100001110110011000100100111011111111101111111111
11110101110011111111110111110101000111111111101110110011101101111110101111
001110011110011111010100001110111111111001111100011011000010111110000111111
10000110111100010000111111011111011100100011111110010101010111111110111011
0011111111110011111111101110001111011000000011001111011111101111111111011
1111000100000000100000011111111111111011111111000011001111111101111111100
0100111111111001111100110111011100111111100111011110100100000111001111000011
11111011111111111101111111001010110110010110001011111111111111111110011110011111
1001111111111101000000111000000111111011
```

```
    {
        black++;
        jarNo = JarB();
    }
    if (i >= iSteps) break;
}
}
```

```
    res.white = white;
    res.black = black;
    res.s = s;
    return res;
}
```

1.9. Let's change the method Write()

```
static void Write()
{
    Console.WriteLine(myMC.iJarA00 + "," + myMC.iJarA01);
    Console.WriteLine(myMC.iJarB10 + "," + myMC.iJarB11);
    Console.WriteLine("White=" + r.white);
    Console.WriteLine("Black=" + r.black);
    Console.WriteLine("s=" + r.s);
}
```

1.10. Start Console Program

Draw a table 3x3 and insert the drawing in the center

```
<table>
```

```
  <tr>
```

```
    <td></td>
```

```
    <td></td>
```

```
    <td></td>
```

```
  </tr>
```

```
  <tr>
```

```
    <td></td>
```

```
    <td>
```

```
      
```

```
    </td>
```

```
    <td></td>
```

```
  </tr>
```

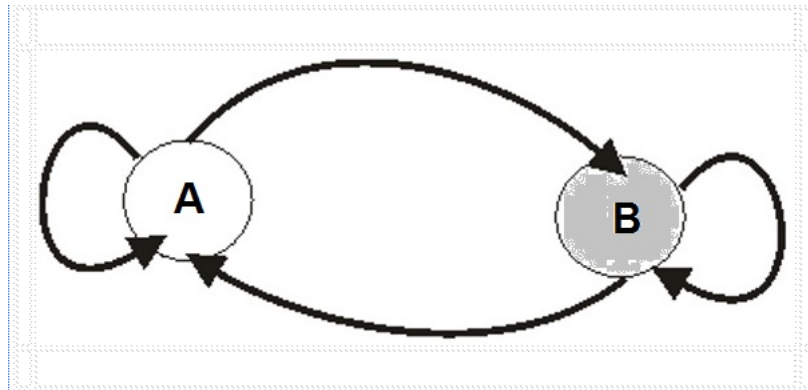
```
  <tr>
```

```
    <td></td>
```

```
    <td></td>
```

```
    <td></td>
```

```
  </tr>
```



We carry out code migration - from a console application to a WEB application

1. **Business Logic** - classes `Result` and `MarkovChain` - 1:1

Console	Default.aspx.cs
<pre>class Result { internal int white; //white internal int black; //black internal string s; } class MarkovChain { ... }</pre>	<pre>using System; using System.Web; using System.Web.UI; using System.Web.UI.WebControls; class Result { } class MarkovChain { ... }</pre>

```
}
```

2. Method Read() – get values from textboxes

Console	Default.aspx.cs
<pre>static void Read() { myMC = new MarkovChain(50, 50, 20, 80, 1000); r = new Result(); }</pre>	<pre>public partial class Default2 : System.Web.UI.Page { MarkovChain myMC; Result r; protected void Read() { int a00 = int.Parse(TextBoxJarA00.Text); int a01 = int.Parse(TextBoxJarA01.Text); int b10 = int.Parse(TextBoxJarB10.Text); int b11 = int.Parse(TextBoxJarB11.Text); int steps = int.Parse(TextBox_iSteps.Text); myMC = new MarkovChain(a00, a01, b10, b11, steps); r = new Result(); } }</pre>

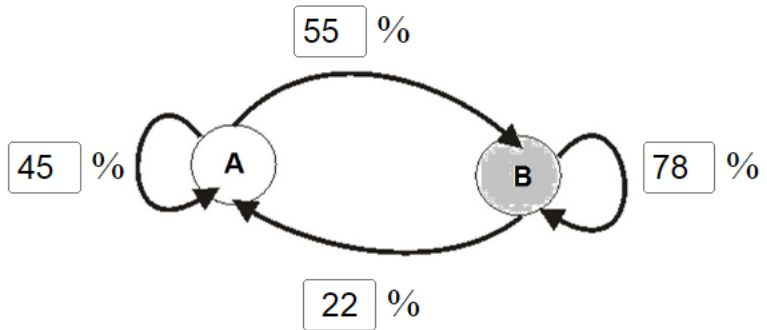
3. Method Write() – put values to textboxes

Console	Default.aspx.cs
<pre data-bbox="286 683 1079 1013">static void Write() { Console.WriteLine(myMC.iJarA00 + "," + myMC.iJarA01); Console.WriteLine(myMC.iJarB10 + "," + myMC.iJarB11); Console.WriteLine("White=" + r.white); Console.WriteLine("Black=" + r.black); Console.WriteLine("s=" + r.s); }</pre>	<pre data-bbox="1160 715 1998 1038">protected void Write() { //Console.WriteLine(myMC.iJarA00 + "," + myMC.iJarA01); //Console.WriteLine(myMC.iJarB10 + "," + myMC.iJarB11); LabelWhite.Text = (r.white).ToString(); LabelBlack.Text= (r.black).ToString(); LabelS.Text="s=" + r.s; }</pre>

4. The code from the Main() method is placed in the method cmbRun_Click()

Console	Default.aspx.cs
<pre>static void Main(string[] args) { Read(); r = myMC.Run(); Write(); Console.ReadLine(); }</pre>	<pre>protected void cmbRun_Click(object sender, EventArgs e) { Read(); r = myMC.Run(); Write(); }</pre>

<https://hwaet.bsite.net/Projects/5/Markov/>



1000 steps

Run()

304 696

s=0000011111110100110000110111110100011111011100100