Markov chains and their applications in AI

I have chosen Markov chain as a subject for my project since I really enjoy more theoretical, mathematical, somewhat easy to understand topics in computer science, and this is definitely one of them. On top of that Markov chains can be seen in a lot of places, sometimes in unexpected ones, and having some knowledge about them would help anyone understand computer science better.

First of all, a bit of history on Markov chain. As I have already mentioned, Markov chain comes from mathematics, and as any standard mathematical subject it has deep historical roots. The story starts from 16th century with the Swiss mathematician Jacob Bernoulli, who was one of the more famous members of his family (most of them were mathematicians). One of the things he worked on and proven was the famous Law of Large Numbers. It describes the result of performing the same experiment a large number of times. According to the law, the average of the results obtained from a large number of trials should be close to the expected value and will tend to become closer to the expected value as more trials are performed. As any curious human being, he tried to apply what he has proven to explain something in real world. He famously said “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”. He implied that fate of events is predetermined, which was a dangerous idea in the old religious world. Many years later, in early 20th century, Russian mathematician Pavel Nekrasov, formerly theologist, tried to oppose that idea. He believed in the doctrine of free will and didn’t like the idea of us having this predetermined statistical fate. He made a claim that independence is a necessary condition for the law of large numbers, since independence only describes theoretical examples and doesn’t transition well to the real world. Everything that happens in real world happens for some reason and is reason for provoking something else to happen. This claim angered another Russian mathematician, Andrey Markov, who maintained a very public animosity towards Nekrasov. 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 states Nekrasov cannot even dream about. Markov proceeds to extend Bernoulli’s results to dependent variable using an ingenious construction. He then proved that in this construction the probabilities of each state happening converge to some specific ratio. He disproved Nekrasov’s claim and cemented his legacy in the science with Markov chain.

So what is the Markov chain? It is a process for which predictions can be made regarding future outcomes based solely on its present state and—most importantly—such predictions are just as good as the ones that could be made knowing the process's full history. For example, in the Figure 1 we can see such Markov chain. The states here are different weather conditions: sunny or rainy. There are probabilities with which tiny weather can stay rainy or become sunny and vice versa. This simple chart attempts to predict weather, and potentially could do it very well. What’s more interesting is from this graph we can confidently say that the weather will be sunny with probability of 67% or rainy with 33%.

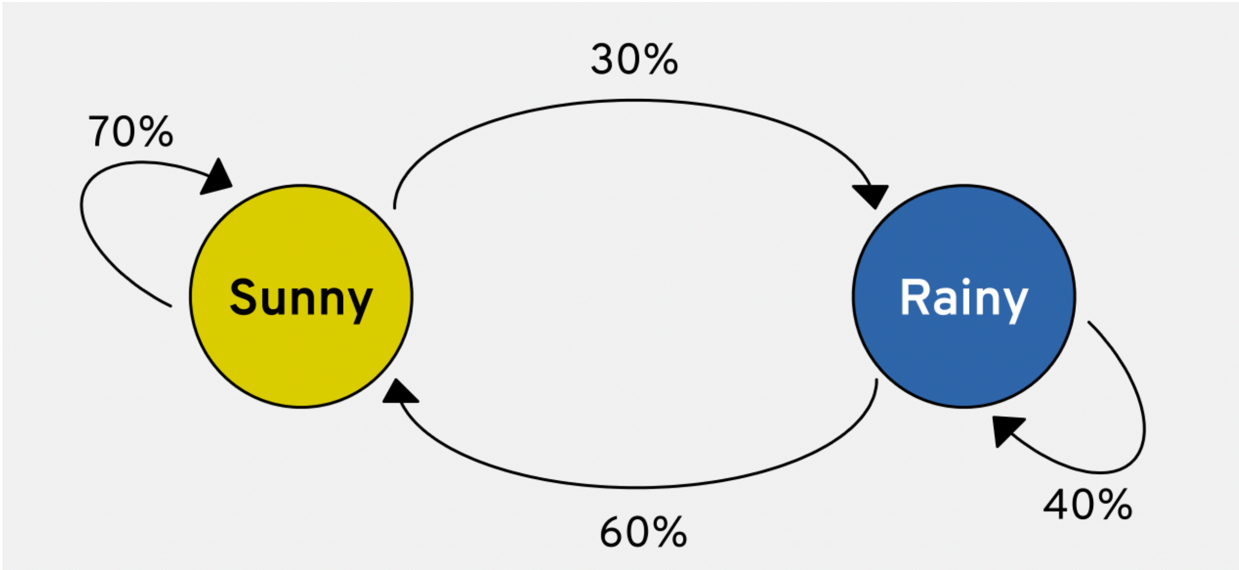


Figure 1. Markov chain predicting weather

That was a relatively abstract example, and Markov chains are used a lot in real life. One of such usages is winning games. I am going to use an example of Monopoly, but it's going to be clear that this can be used on more board games. Monopoly is a game in which a player rolls dice to determine on which field he lands (Figure 2). Which field he lands can be very important, and in some cases he can buy the property on the field, and in others he would have to pay tax for visiting someone else's property. Owning a property (or a set of properties) on which players will land more times to give you more taxes will more often than not lead you to victory. As a player rolls the dice, there are 11 outcomes - from 2 (1 on each dice) to 12 (6 on each dice). There are certain probabilities for each of these outcomes, and we can use these to determine which field the player would land on after his dice roll. From each of those fields the player can roll dice again and determine probabilities to land on next fields and so on. This is a premier example of a Markov chain in which the states are the fields and the probabilities are the dice rolls. From Markov's findings we know that we are bound to come to an equilibrium of probabilities after enough rolls, and we indeed do arrive to a concrete result. On Figure 3 the probabilities to land on any field are displayed. If we use these probabilities and combine them with other knowledge such as which taxes are more profitable compare to investments of real estate, the fact that you want to own sets of properties to build on them and collect more taxes and more, we can arrive to a strategy that will win its user some games. Despite such robust maths, this (or any) method cannot guarantee a win since there are not a lot of moves in a game of Monopoly, and the result still comes down to some luck. Additionally, this method can be used for any other board game involving dice and fields, since the Markov chain is going to be very similar to the Markov chain used here, and a winning strategy can be developed from such a Markov chain.

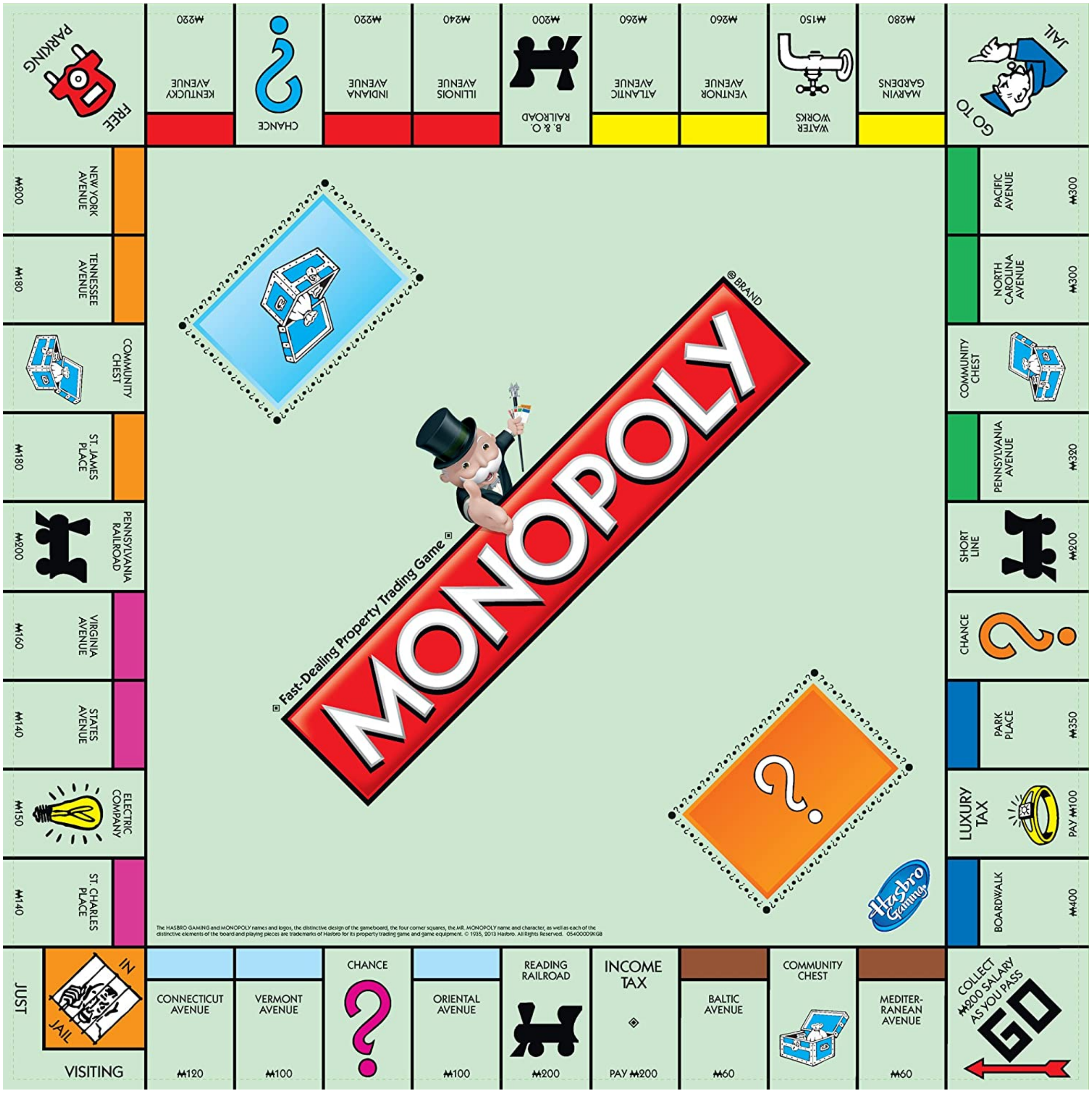


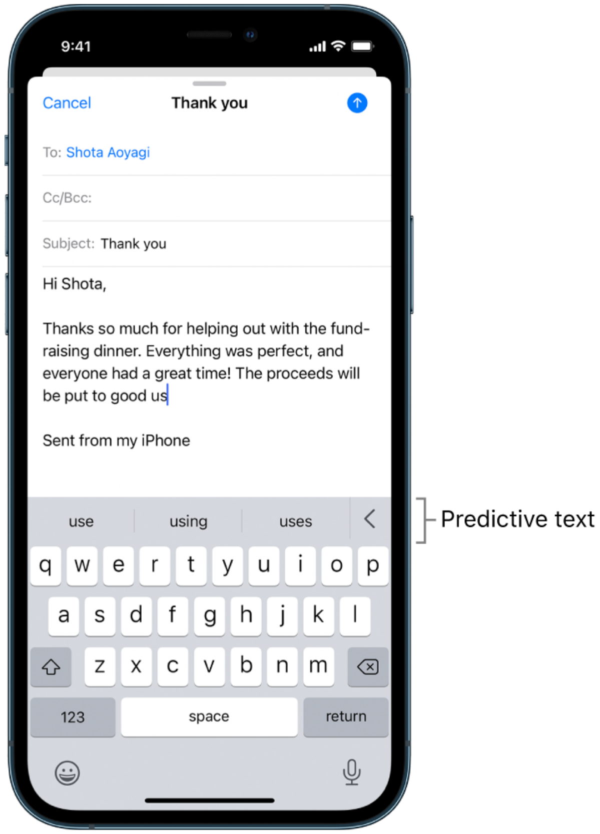
Figure 2. Monopoly board

Square	Property	Probability
0	Go	0.02914
1	Mediterranean Avenue	0.02007
2	Community Chest	0.01775
3	Baltic Avenue	0.02037
4	Income Tax	0.02193
5	Reading Railroad	0.02801
6	Oriental Avenue	0.02132
7	Chance	0.00815
8	Vermont Avenue	0.02187
9	Connecticut Avenue	0.02168
10	Just Visiting (Jail)	0.02139
11	St. Charles Place	0.02556
12	Electric Company	0.02614
13	States Avenue	0.02174
14	Virginia Avenue	0.02426
15	Pennsylvania Railroad	0.02635
16	St. James Place	0.02680
17	Community Chest	0.02296
18	Tennessee Avenue	0.02821
19	New York Avenue	0.02812
20	Free Parking	0.02825
21	Kentucky Avenue	0.02614
22	Chance	0.01045
23	Indiana Avenue	0.02567
24	Illinois Avenue	0.02993
25	B&O Railroad	0.02893
26	Atlantic Avenue	0.02537
27	Ventnor Avenue	0.02519
28	Water Works	0.02651
29	Marvin Gardens	0.02438
30	Go To Jail	0.09457
31	Pacific Avenue	0.02524
32	North Carolina Avenue	0.02472
33	Community Chest	0.02228
34	Pennsylvania Avenue	0.02353
35	Short Line Railroad	0.02291
36	Chance	0.00816
37	Park Place	0.02060
38	Luxury Tax	0.02052
39	Boardwalk	0.02483

Figure 3. Probabilities to land on each field in Monopoly.

Despite this being a very powerful use of Markov chains, this does not involve much computer science. The next application is going to show how Markov chains are used in modern computer science. But first an explanation is needed on how to store a Markov chain. Similarly to the concept itself, it is relatively easy - it can be done with a 2D array. In a node [i,j] a probability to transition from state i to state j is stored. The size of the array would be n^2, where n is number of states in the Markov chain, and all probabilities in each row or column add up to 1. This is a very simple data structure that can be used by even the beginner coders.

We use these applications of Markov chain on our phones every day, and some even say they can’t live without them - they are Predictive Text and Autocorrect (Figures 4 and 5). For both of these features, a Markov chain is created by the manufacturer and trained on many pairs of words in a language and pairs of letters in words to determine probabilities with which letter/word comes next depending on the previous one. The resulting feature is then used by the owner of the phone, and with the experiences of the owner these Markov chains are edited. For example, in a standard phone after word “ice” a phone might suggest “skating” or “cream”, but if the use types “Ice T” a lot more than “ice cream” or “ice skating”, the phone will edit the probability of “T” coming after “Ice” to be a lot higher than “cream” or “skating”, so the phone will suggest “T” more. It works in a very similar way with predicting letters or user mistakes - if you make a lot more of a similar error, the phone will remember it and start suggesting more of the solution the owner is looking for using the Markov chain.



Figures 4 and 5. Predictive Text and Autocorrect on iPhone.

Another frequent application of Markov chains is in data compression. One of the algorithms used to compress data is the Lempel-Ziv-Markov chain algorithm, or LZMA in short. This is an algorithm used to perform lossless data compression. It has been under development since late 20th century by Igor Pavlov and was first used in the 7z format of the 7-Zip archiver. This algorithm uses a scheme somewhat similar to the LZ77 algorithm published by Abraham Lempel and Jacob Ziv in 1977. LZMA combines this scheme with the Markov chain to achieve very high compression ratios while still maintaining decompression speed similar to other commonly used compression algorithm.

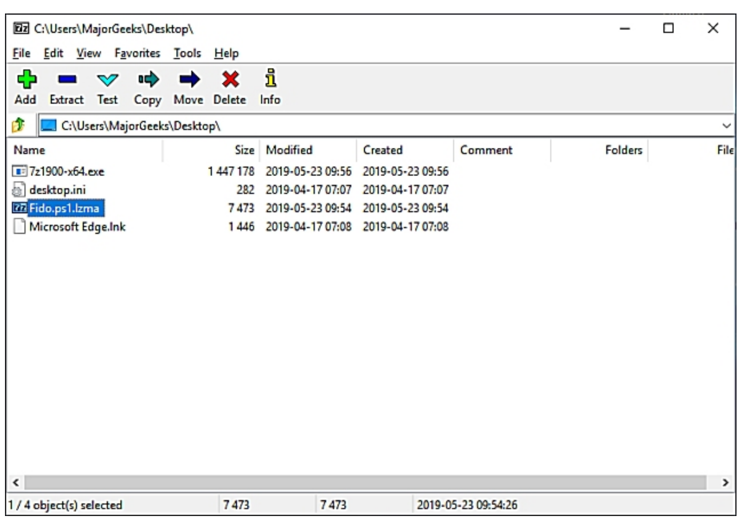


Figure 6. Screenshot of 7-zip with LZMA compressed file highlighted.

Markov chains are also used in some Internet applications. For example, the PageRank of a webpage as used by Google is defined by a Markov chain (Figure 7). PageRank ranks all webpages as how probable is a page to be visited from other pages. Google uses this data to better predict the page a user wants in a Google query. Markov models have also been used to analyze web navigation behaviour of users. A user's web link transition on a particular website can be modelled using Markov models and can be used to make predictions regarding future navigation and to personalize the web page for an individual user. Markov chains are also a basis for more powerful concept like Bayesian networks or Hidden Markov chain, which is an important tool in such diverse fields as telephone networks, speech recognition and bioinformatics.

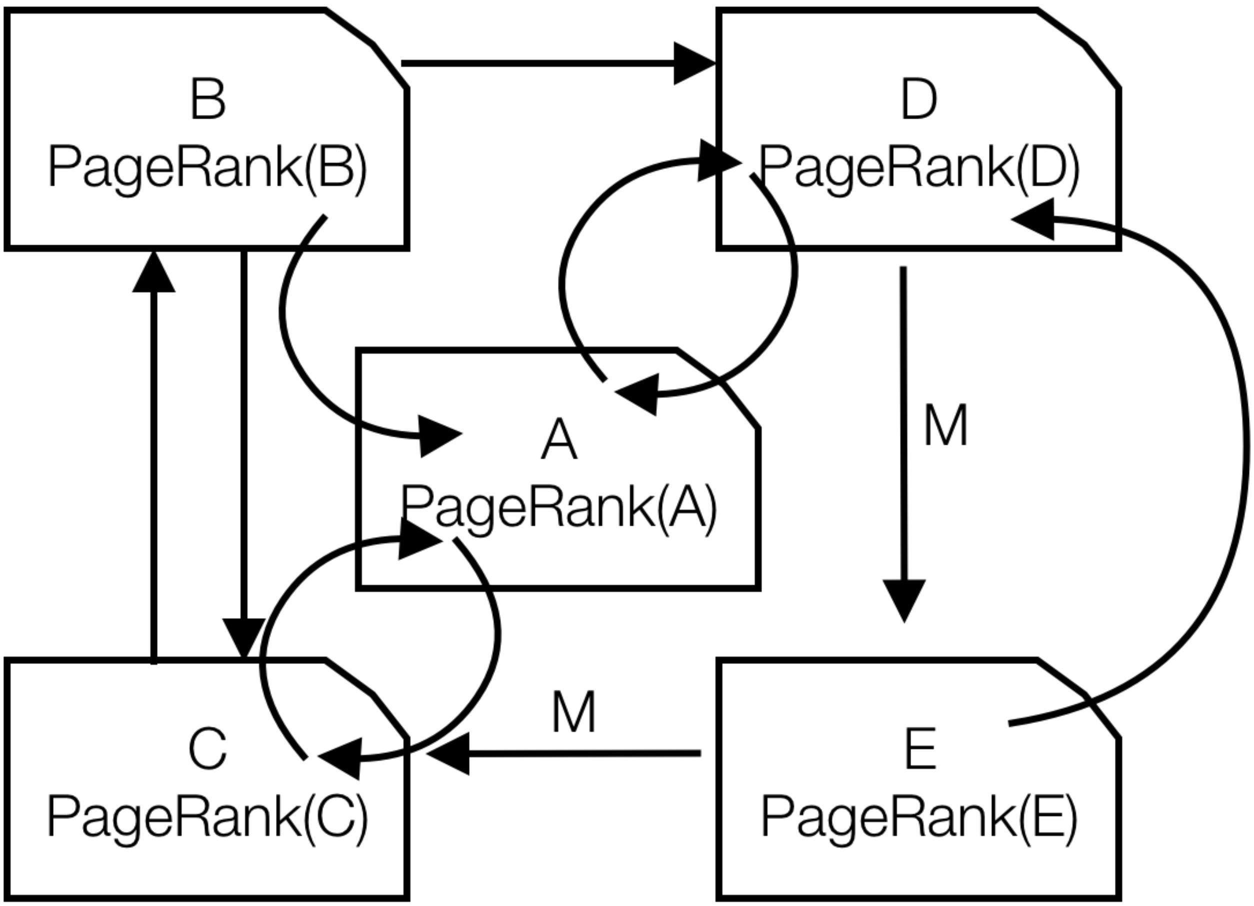


Figure 7. A state diagram that represents the PageRank algorithm.

In conclusion, Markov chains is a simple, yet powerful concept which has many applications. I believe that with enough knowledge about this concept people will understand computers a little better, which is a noble goal to strive for.

References:

1) Dekking, Michel (2005). A Modern Introduction to Probability and Statistics (https://archive.org/details/modernintroducti00fmde)
2) Gagniuc, Paul A. (2017). Markov Chains: From Theory to Implementation and Experimentation.
3) Page, Lawrence; Brin, Sergey; Motwani, Rajeev; Winograd, Terry (1999). *The PageRank Citation Ranking: Bringing Order to the Web* (Technical report).
4) Ziv, Jacob; Lempel, Abraham (May 1977). "A Universal Algorithm for Sequential Data Compression”.
5) Li, Ben (April 2013). “Markov Chains in the Game of Monopoly”. (https://web.williams.edu/Mathematics/sjmiller/public_html/hudson/Li_Markov%20Chains%20in%20the%20Game%20of%20Monopoly.pdf)