POLITECNICO
MILANO 1863

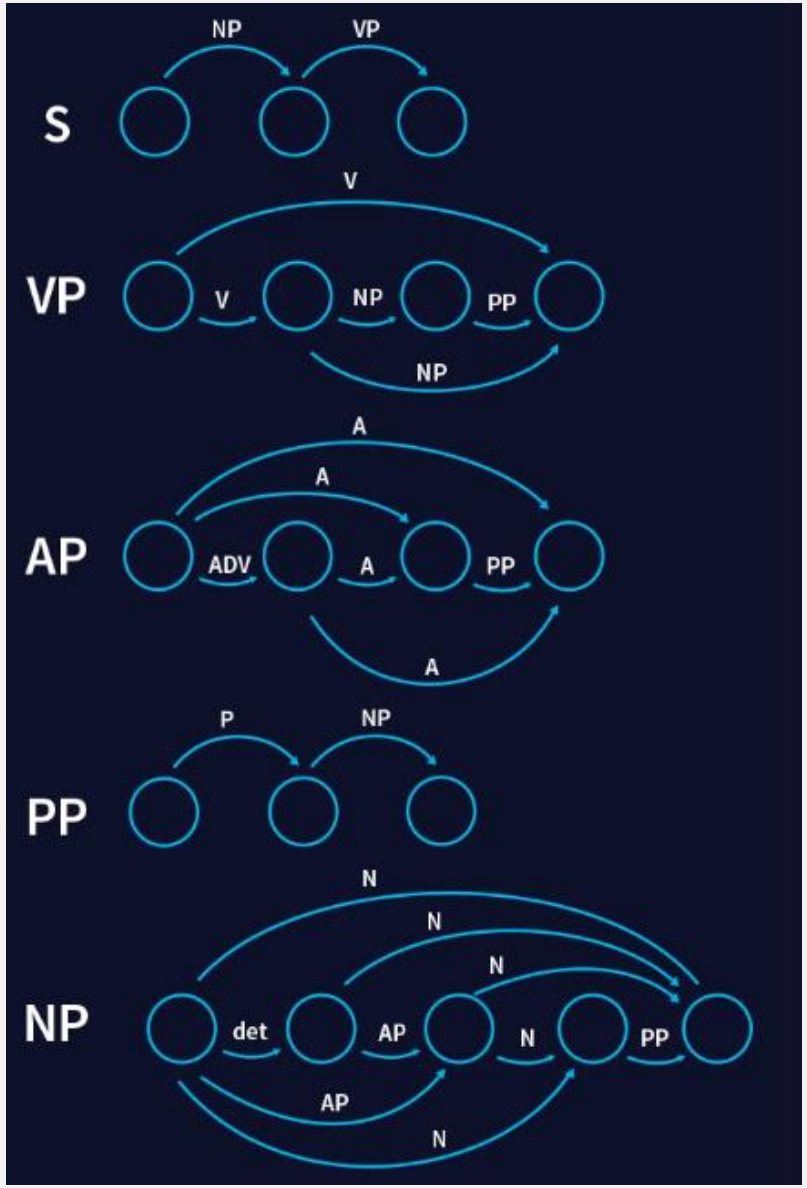
IMAGE AND SOUND
ISPG
PROCESSING GROUP

CREATIVE PROGRAMMING AND COMPUTING

From transitional networks to hybrid agents

TRANSITIONAL NETWORKS

TRANSITIONAL NETWORKS

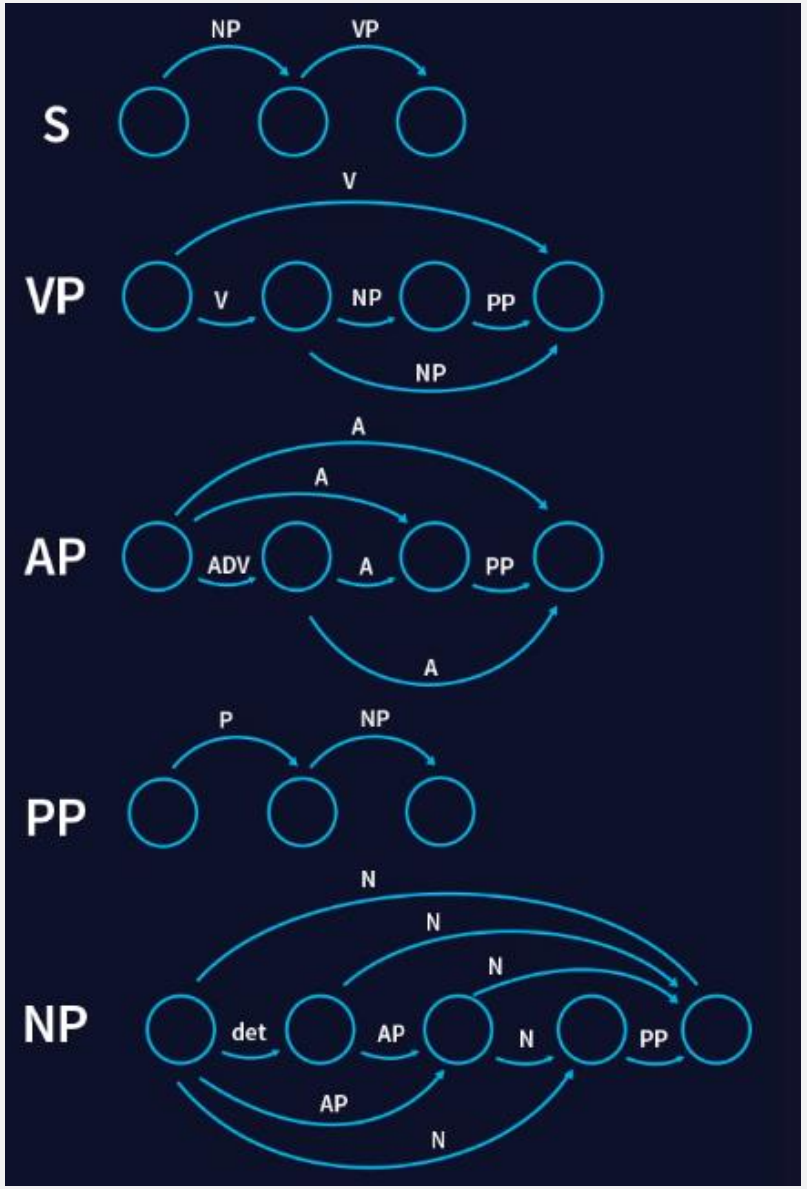


- Another formalism to model grammar are **Transitional Networks**

- Their functioning can be represented by a set of linked **state** where the links are **transitions** between states
- Each state can have an output and a transition link
- The states represent a generated phrase
- Transitions produces Non-terminal symbols
- Output produces Terminal symbols

TRANSITIONAL NETWORKS

- More in general
  - They are finite state automata
  - Transitions can be deterministic or stochastic
  - In the stochastic approach, transitions are based of probabilities
  - Probabilities can be learnt by a corpus of data
  - Any computable function can be represented in terms of transitional network

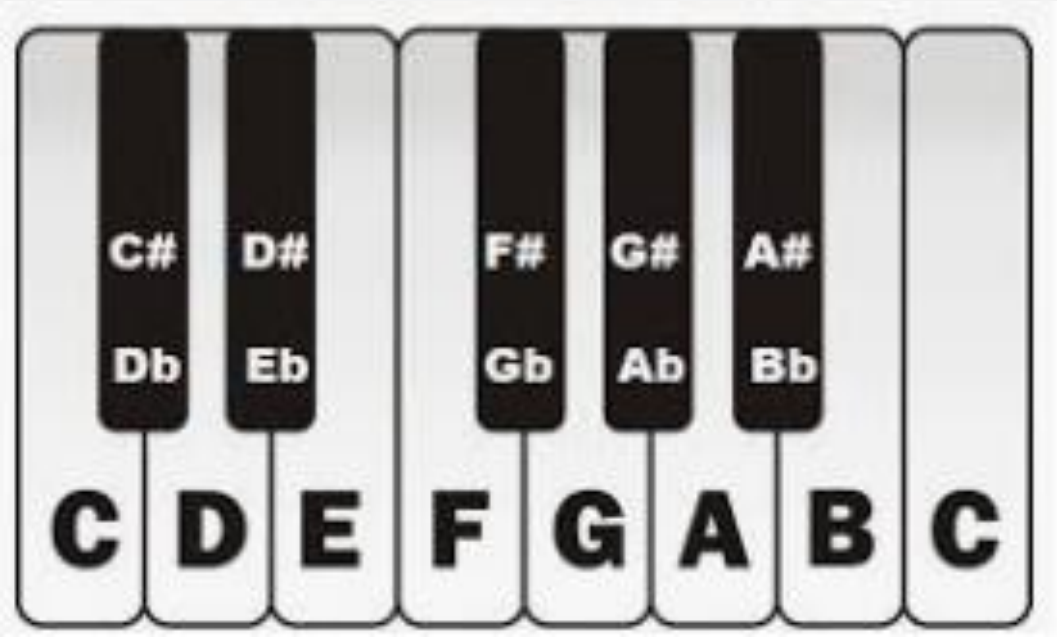


LINKED AUTOMATA

- Transitional networks applied to automatic music composition are called **automaton**
- An **automaton** is a procedure whose output depends on its internal state and its input.
- An automaton is able to model the evolution in time of musical compositions
- The basic elements of a musical score are partitioned into a number of parameters (e.g. pitch, amplitude,...). An automaton models the behaviour of each parameter
- Modelling the evolution of
  - Pitch -> melody
  - Amplitude -> envelope
  - Chords -> harmonic progression
  - …

MUSIC THEORY - REVIEW

- There are twelve different notes in traditional western music:
- C, C#/Db, D, D#/Eb, E, F, F#/Gb, G, G#/Ab, A, A#/Bb, B
- This sequence is called the **chromatic scale**



MUSIC THEORY - REVIEW

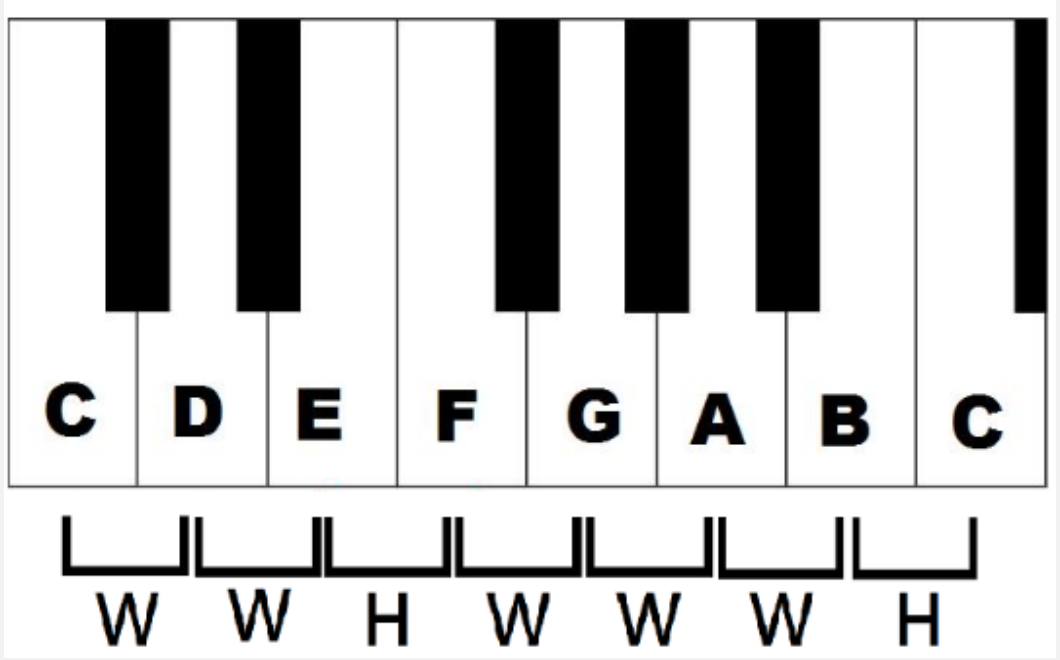
- All other scales are simply subsets of the chromatic scale
- The simplest scale is the C major scale, which is formed by the notes C, D, E, F, G, A, B

MUSIC THEORY - REVIEW

- An **interval** is a combination of two notes and is defined by the number of half steps between them
- **Major scale** is defined by the intervals between these notes:

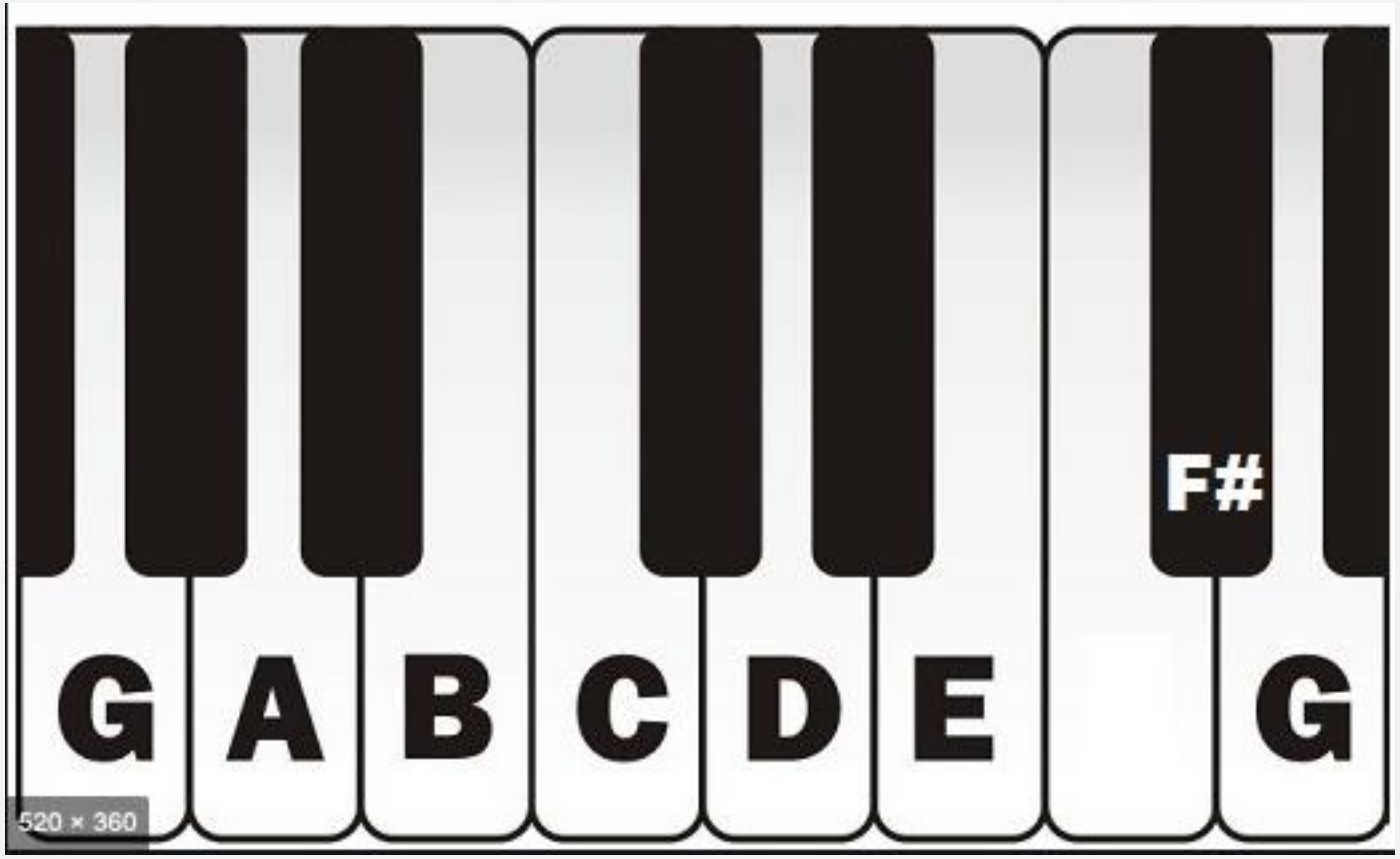
W W H W W W H

- Where "W" indicates a whole step (**tone**) and "H" a half (**semi-tone**)



MUSIC THEORY - REVIEW

- Maintaining the pattern we can generate other major scales
- A G major scale is **G, A, B, C, D, E, F#**

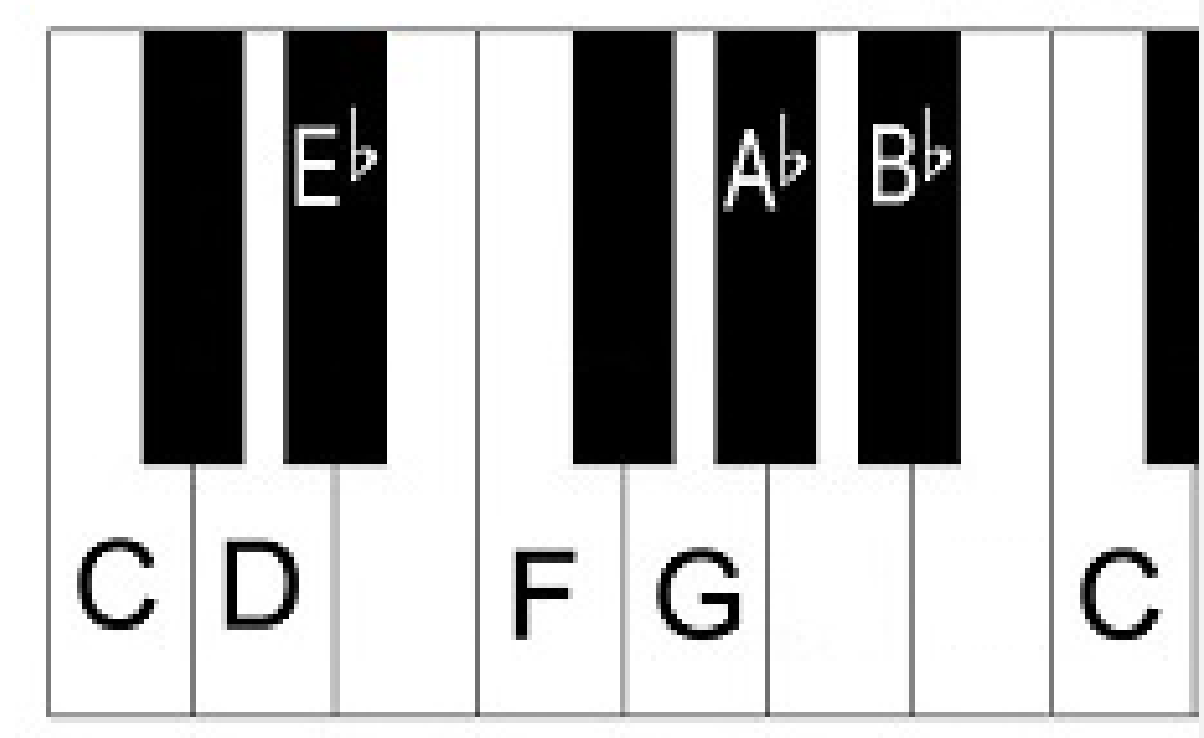


MUSIC THEORY - REVIEW

- **Minor scale** is defined by the intervals between these notes:

W H W W H W W

Where "W" indicates a whole step (**tone**) and "H" a half (**semi-tone**)

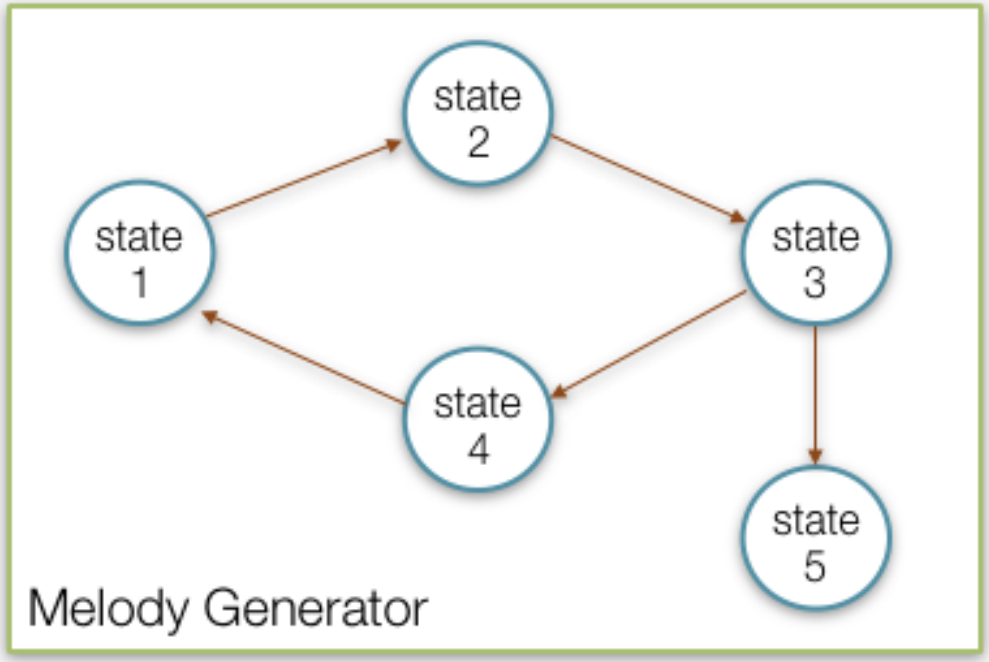


LINKED AUTOMATA

```
procedure harmonize(input_note, output_note);
    integer input_note, output_note;
    static boolean harmony_state;
    if harmony_state = true then
    begin
        {* Harmonize by a fifth (7 semitones) *}
        output_note := input_note + 7;
        harmony_state:= false;
    end
    else
    begin
        {* Harmonize by a perfect fourth (5 semitones) *}
        output_note := input_note + 5;
        harmony_state := true;
    end
end procedure;
```

- A simple automaton that harmonizes an input note alternatively by a perfect fourth or fifth.
- The *boolean* variable *harmony_state* stores the internal state of the automaton.

LINKED AUTOMATA




```
Next_state(state_1, pitch, count) → state_2
Next_state(state_2, pitch, count) → state_3
Next_state(state_3, pitch, 1) → state_4
Next_state(state_3, pitch, 2) → state_4
Next_state(state_3, pitch, 3) → state_5 (HALT)
Next_state(state_4, pitch, count) → state_1

Current_output(1, pitch, count) → pitch, count
Current_output(2, pitch, count) → pitch + 5, count
Current_output(3, pitch, 0) → pitch + 7, 1
Current_output(3, pitch, 1) → pitch + 7, 2
Current_output(3, pitch, 2) → pitch + 7, 3
Current_output(4, pitch, count) → pitch + 2, count
Current_output(5, pitch, count) → pitch, 0
```

- next_state(*current_state*, *count*) —> new_*state*
- current_output(*current_state*, *note*, *count*) —> (*new_note*, *new_count*)

- ***count*** counts the number of times it cycles from state 1 to state 4 before to reach the state 5 and exit

ex3 - automata

LINKED AUTOMATA

- In the above examples, if we know the initial state and the input data of the automata, we can predict their behavior with certainty -> **deterministic automata**

- **Stochastic automata:** some aspects of their behaviour is determined by a random or probabilistic procedure.

- We can introduce some random decisions in either **Next_state** or **Current_output functions**

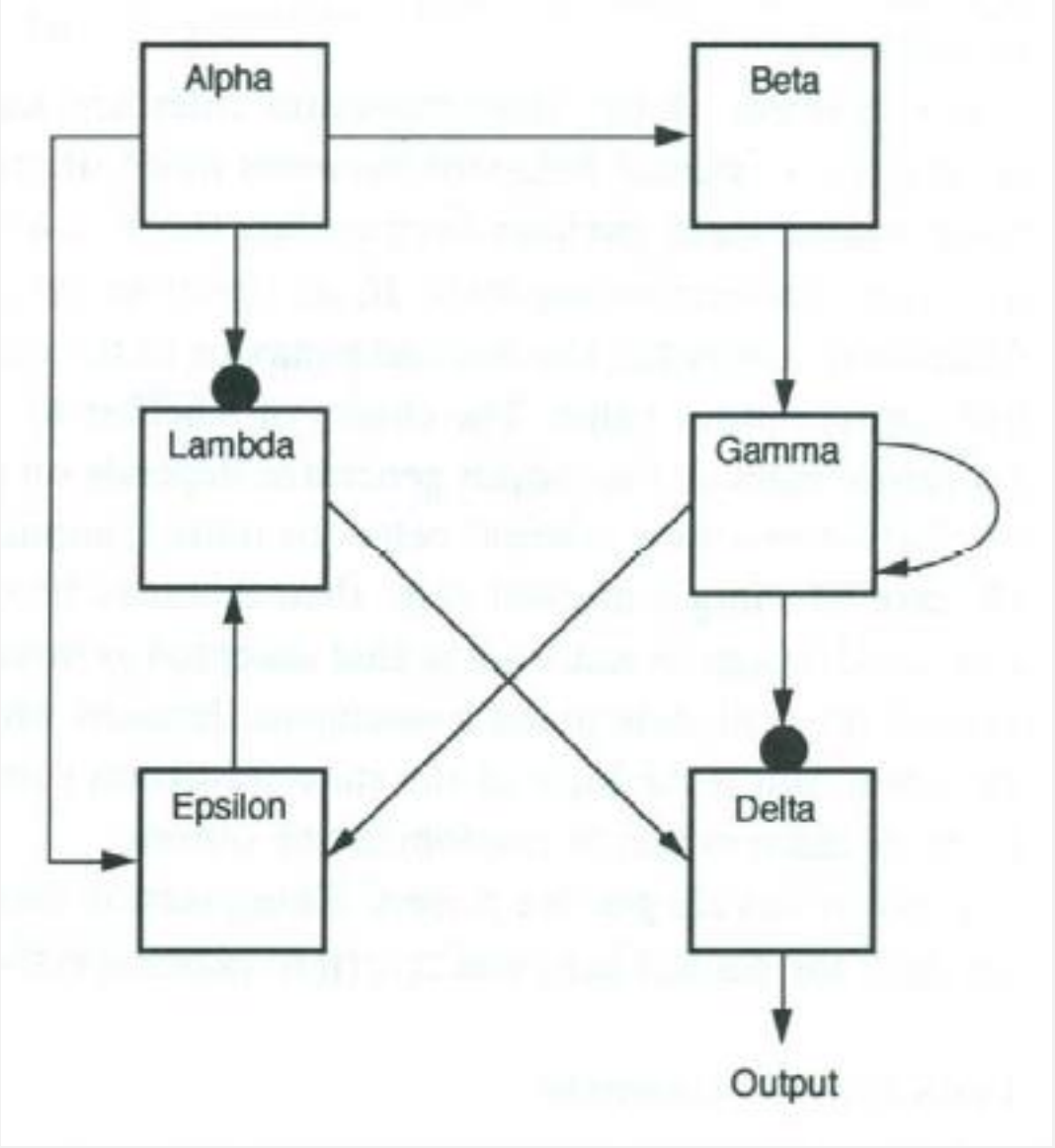
automataStocOutput

automataStocNextState

LINKED AUTOMATA

- Automata can be linked with one another in a network, constituting a system. The links transmit the output of one automata to the input of other automata.

- A system of six automata interconnected via direct links
- The output may be taken from a single automaton or from multiple automata.



MARKOV MODEL

MARKOV MODEL

- Markov models are used to model sequence of discrete events
- Es: e sequence of musical notes are a sequence of discrete events
- The simplest model (**previously seen in this course**)
  - Harry F. Olson (1976) analysed 11 monophonic melodies by Stephen Foster (Oh susanna)
  - Obtained the frequency of appearance of each note

Note	B_3	$C_4^{\#}$	D_4	E_4	$F_4^{\#}$	G_4	$G_4^{\#}$	A_4	B_4	$C_5^{\#}$	D_5	E_5
Relative Frequency	17	18	58	26	38	23	17	76	42	29	30	17

MARKOV MODEL

Note	B_3	$C_4^{\#}$	D_4	E_4	$F_4^{\#}$	G_4	$G_4^{\#}$	A_4	B_4	$C_5^{\#}$	D_5	E_5
Relative Frequency	17	18	58	26	38	23	17	76	42	29	30	17

- Obtain the frequency distribution of all the notes: **$frequency_i$/Total_frequency**
- x_t radom variable representing the note at time t
- $P(x_t)$ is the probability distribution of the event represented by x at time t
- The generation problem in this model is:
  - **Given a certain time t -> how is $P(x_t = N)$ where N is a fixed note ?**

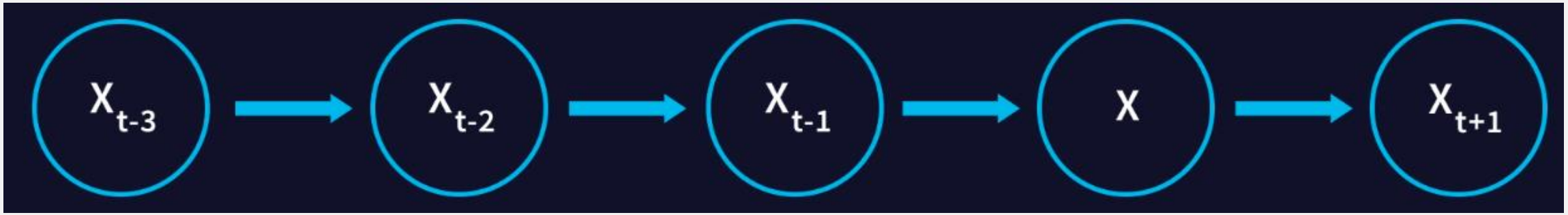
$$P(X_t=A_4)=67/382=0.17539267\ (+-=17.5\%)$$

MARKOV MODEL

- CONS
  - The model assumes there is no relation between the note in the sequence
  - This is not true in music where the **progression** of note is important (**harmonic progression** and **melodic progression**)
  - Notes in a progression have a relation

- Markov model attempts to model the **transition** from one event to another

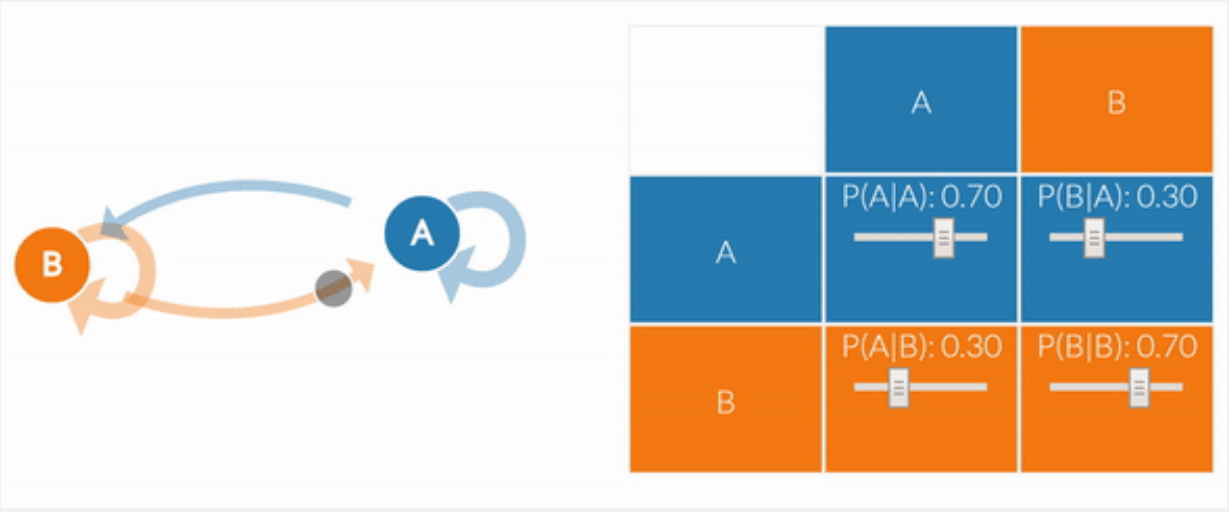
MARKOV MODEL



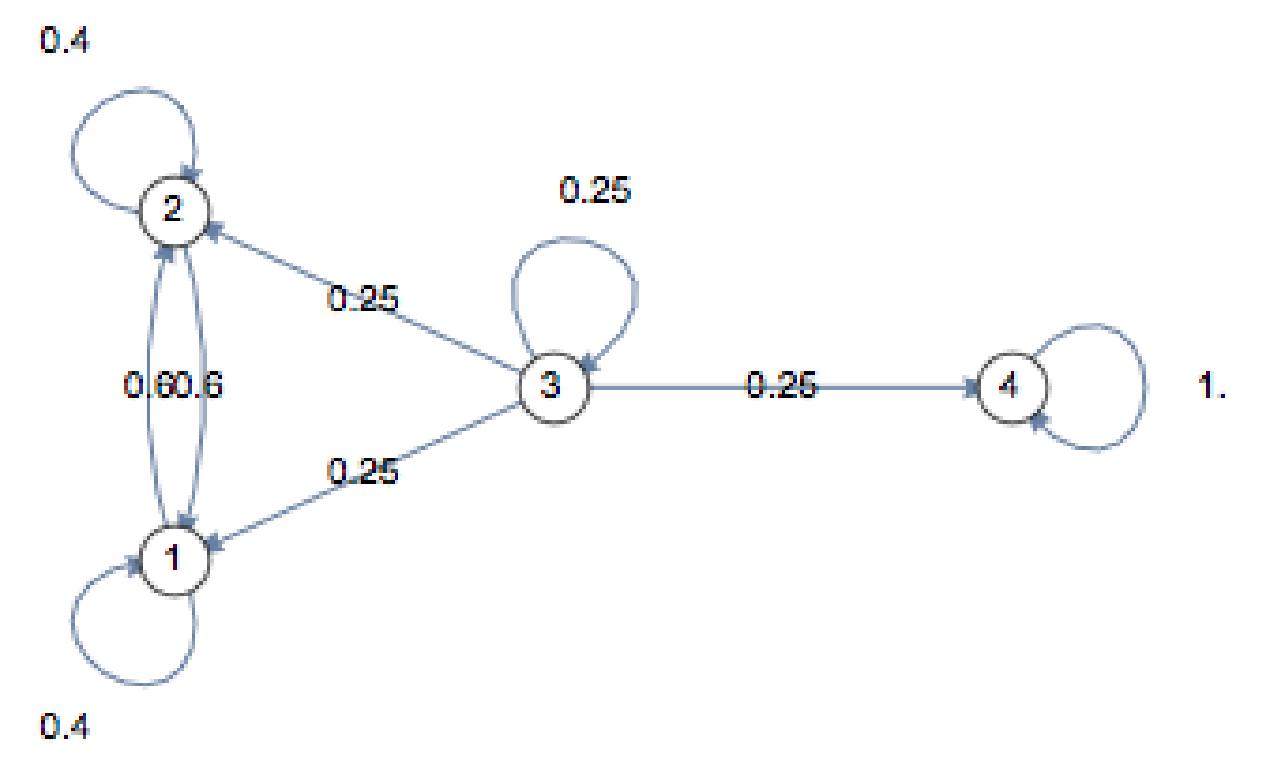
- It is based on the **conditional probability** $P(x_t \mid x_{t-1})$ – This is the probability of a given transition
- Example $P(x_t \mid x_{t-1}=A4)$?

MARKOV MODEL – SOME DEFINITIONS

- A **Markov propriety** is a stochastic process where the conditional probability distribution of future states of the process (conditional on both past and present states) depends only upon the present state (**context**)
- A **Markov chain** is a discrete-time random process with the Markov property
- A discrete--time random process means a system which is in a certain state at each step, with the state changing randomly between steps
- The changes of the state are **transition** with a given **transition probability**



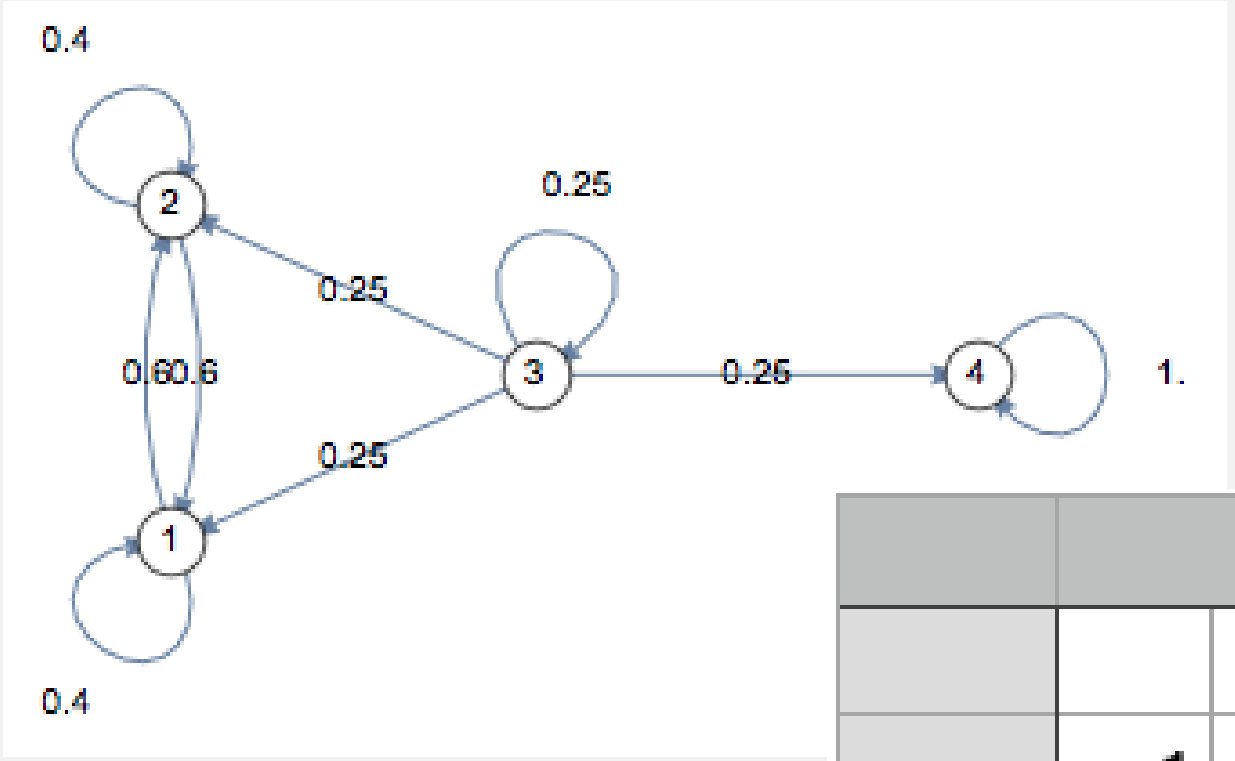
MARKOV MODEL – SOME DEFINITIONS



- A Markov Chain can be represented by a transitional network
- Each state represents a **discrete event**
- Each state provide it self as output
- **Transition** between states are modelled though links
- The **transition probability** is a conditional probability
  - Es: P(S1| S5)

- The generated sequence is the set of states visited during the process:
- Es: 1-2-2-2-3-1-1-1-2-3-4-4-4-4

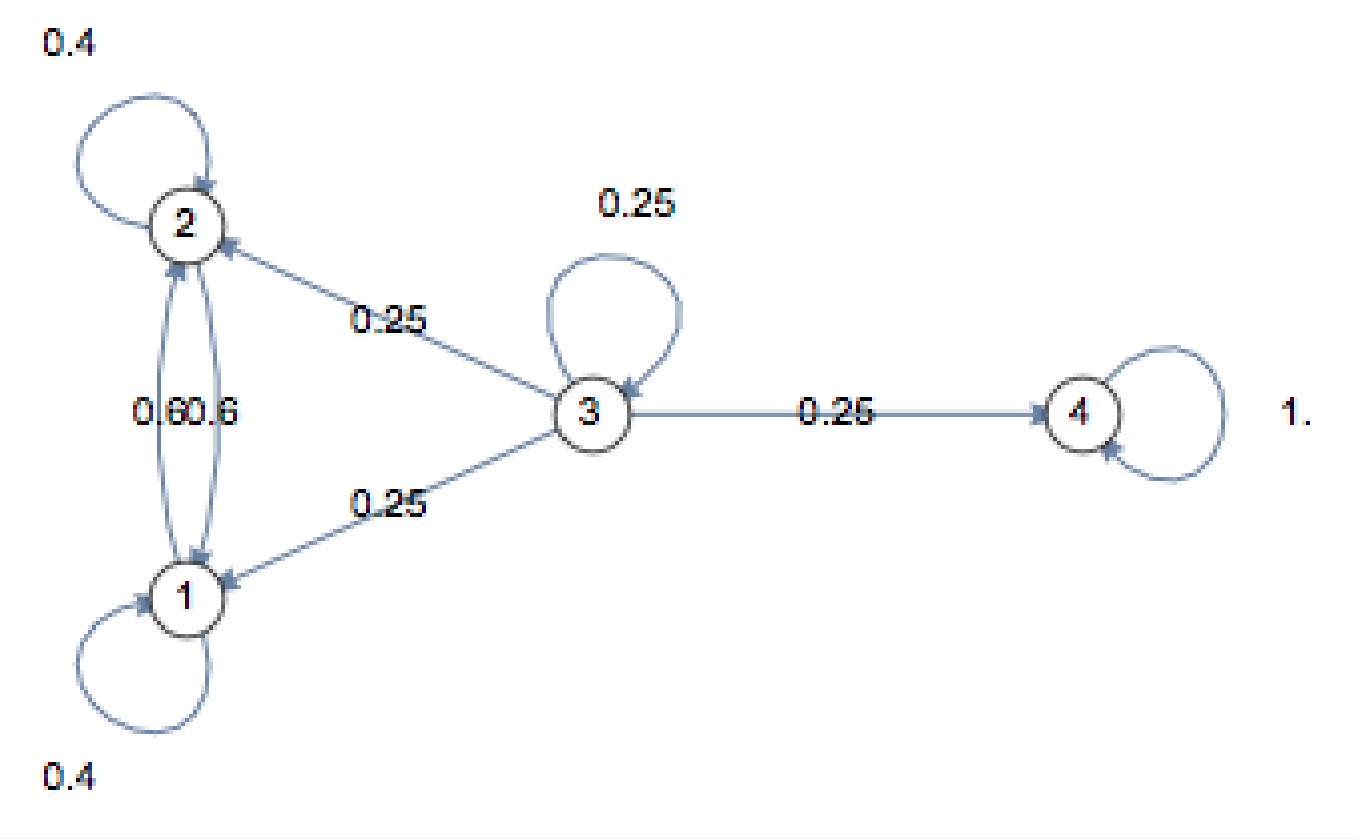
MARKOV MODEL – TRANSITION MATRIX



- All the states, all the transitions and all the transition probabilities can be collected in a **transition probability matrix**
- Each line sums to 1

		OUTPUT			
		1	**2**	**3**	**4**
INPUT	**1**	0.4	0.6		
	2	0.6	0.4		
	3	0.25	0.25	0.25	0.25
	4	1			

MARKOV MODEL – AN EXAMPLE



- In order to use a Markov chain to model an automatic music generation system:
  - Need of model the set of states
    - Example: **a state for each note in the scale**
  - Need to define the set of transitions and transition probabilities
    - 1 = C
    - 2 = D
    - 3 = E
    - 4 = F

MARKOV MODEL – AN EXAMPLE

- All the states, all the transitions and all the transition probabilities can be collected in a **transition probability matrix**
- Each line sums to 1
- Harry F. Olson (1976) analysed 11 monophonic melodies by Stephen Foster

Note	B2	C4 #	D4	E4	F4#	G4	G4 #	A4	B4	C5#	D5	E5
B2			1									
C#			1									
D	1/16	1/16	2/16	5/16	3/16	1/16		1/16		1/16	1/16	
E		1/16	3/8	3/16	1/4			1/16			1/16	
F#			1/8	1/4	5/16	1/8		1/8	1/16			
G					1/4	3/16		3/8	3/16			
G#								1				
A			1/16		5/16	1/16	1/16	4/16	3/16		1/16	
B4								9/13	2/13		2/13	
C5#									1/2		1/2	
D5								1/4	7/16	3/16	1/16	1/16
E5								6/16		10/16		

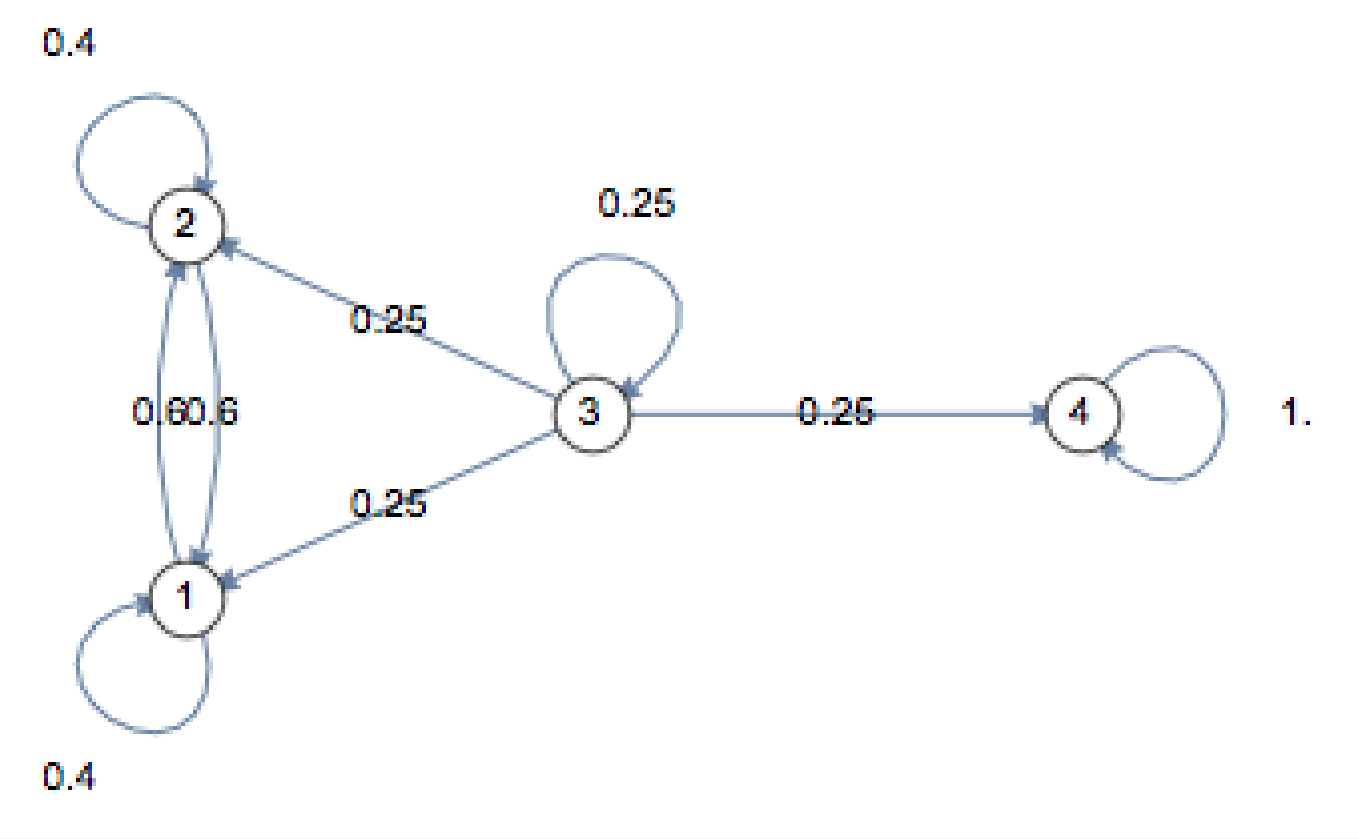
MUSIC THEORY - REVIEW

- Given a melody **C Eb D F Eb G F A G Bb A C Bb D C**

- Let's consider a melody as a pattern on scale
- Pattern can be encoded dependently to the scale

- Pattern mapping on the **Chromatic scale** (distances in halftones - C Db D Eb E F Gb G Ab A Bb B):
  - +3, -1, +3, -2, +4, -2, +4, -2, +3, -1, +3, -2, +4, -2
- Pattern mapping on the **C Dorian scale** (C D Eb F G A Bb):
  - +2, -1, +2, -1, +2, -1, +2, -1, +2, -1, +2, -1, +2, -1

MARKOV CHAIN– AN EXAMPLE

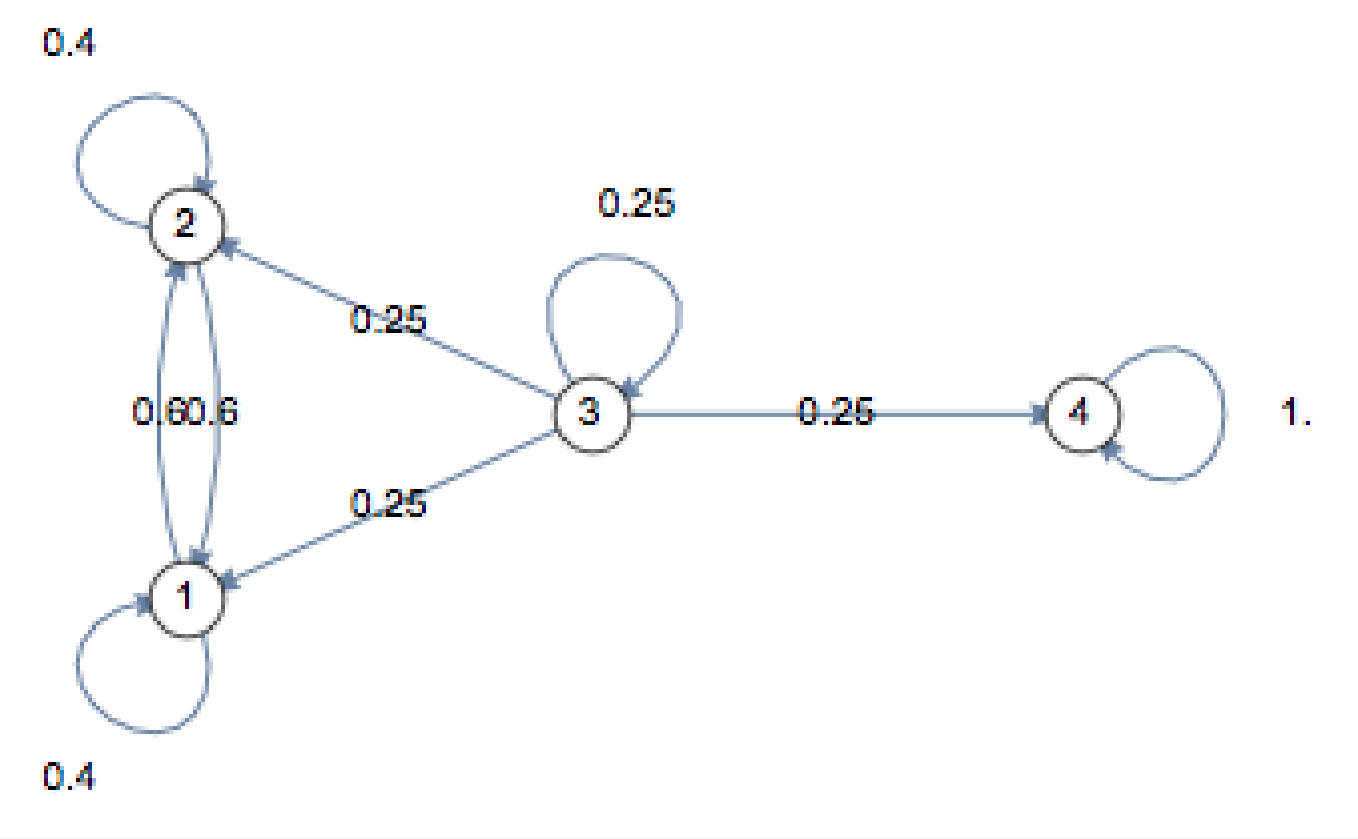


- Using Markov Chain e can model the Pattern
- It independent from the scale (we can change the scale without changing the model)
- A seed note is needed

- 1 = +3
- 2 = -1
- 3 = +1
- 4 = -2

MARKOV CHAIN– AN EXAMPLE

- … and we can model the note duration as well



- 1 = whole note
- 2 = quarter note
- 3 = Eighth note
- 4 = Sixteenth note

MARKOV CHAIN– MUSIC - ILLIAC SUITE

- The first experiment in using Markov Chain in music is the **Illiac Suite for String Quartet** (1957) by two Americans, the composer **Lejaren Hiller** and the mathematician **Leonard Isaacson**
- The composition has four movements. In the 4th movement Markov model has been applied
- The composition is for four voices (String Quartet)
- Each voice is generated independently
- The rhythm is fixed to eighth notes
- The transition probability matrix is built by hand and changes along the composition
- The pause is also included as a note

https://www.youtube.com/watch?v=n0njBFLQSk8

MARKOV CHAIN– ORDERS

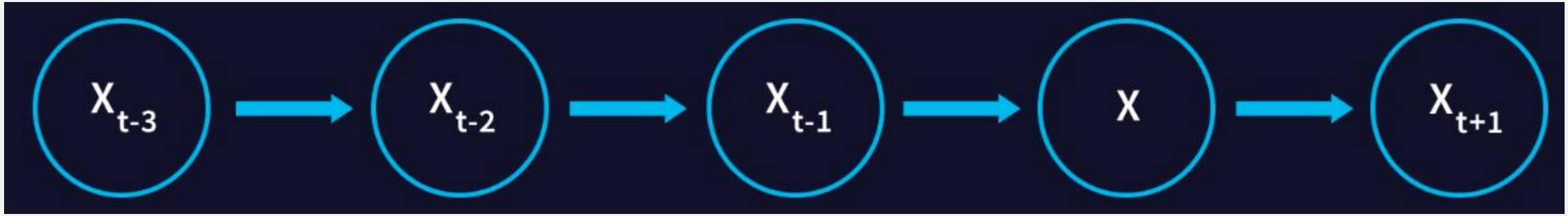
- Markov Chain is based on the ideas that the current state is dependent on a context (previous state and transition probability)
- A monophonic composition is a ”random walk” that follows the transition probability matric

- In music composition generally a given note in the sequence in not only dependent on the previous note, but more likely is dependent by a set of previous note

$$P(X_t = S_j \mid X_{t-1} = S_i, X_{t-2} = S_k, \ldots)$$

- With this model is possible to capture compositional patterns

MARKOV CHAIN– ORDERS



- Order 0: probability distribution of the events, $\mathbf{P(x_t)}$
- Order 1: $\mathbf{P(x_t | x_{t-1})}$
- Order 2: $\mathbf{P(x_t | x_{t-1}, x_{t-2})}$
- Order d: $\mathbf{P(x_t | x_{t-1}, x_{t-2} \ldots, x_{t-d})}$

MARKOV MODEL – ORDERS

- Transition probability matrix should consider all the possible sequence of events
- The length of the sequence is proportional to the order of the chain
- Tendency to have a sparse matrix

	A	B	C
(A,B)	0	1/2	1/2
(B, C)	1/2	1/2	0
(C, A)	0	1	0
(B, B)	0	0	1

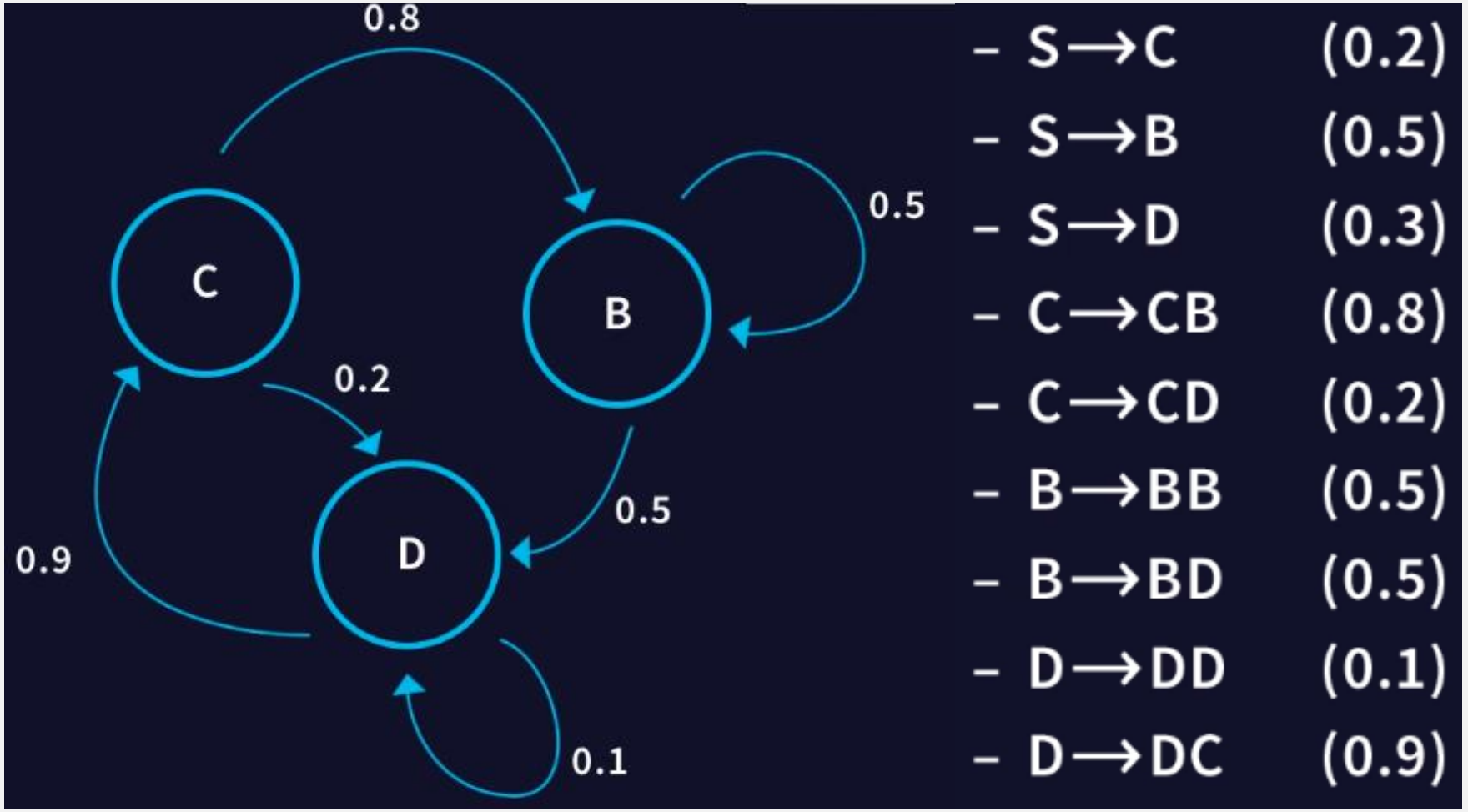
MARKOV CHAIN– LEARNING ALGORITHM

- Transition probability matrix can be composed by hand
  - Following some composition rule (similar to grammar)
  - Following some typical composition rule by a specific composer (**style imitation**)

- Transition probability matrix can be learnt
  - By analysing transition frequencies between notes in a large corpus of melodies
  - By analysing transition frequencies between notes in a large corpus of melodies by a specific composer (**style imitation**)
  - By analysing transition frequencies between notes in a real-time composition (**continuation**)

MARKOV CHAIN– A GRAMMAR APPROACH

Note	C	B	D
C		0.8	0.2
B		0.5	0.5
D	0.9	0.1	



- A Markov model can be represented by a Grammar model
- A transition probability matrix can be represented by transitional networks

MARKOV CHAIN– TEXT COMPOSITION

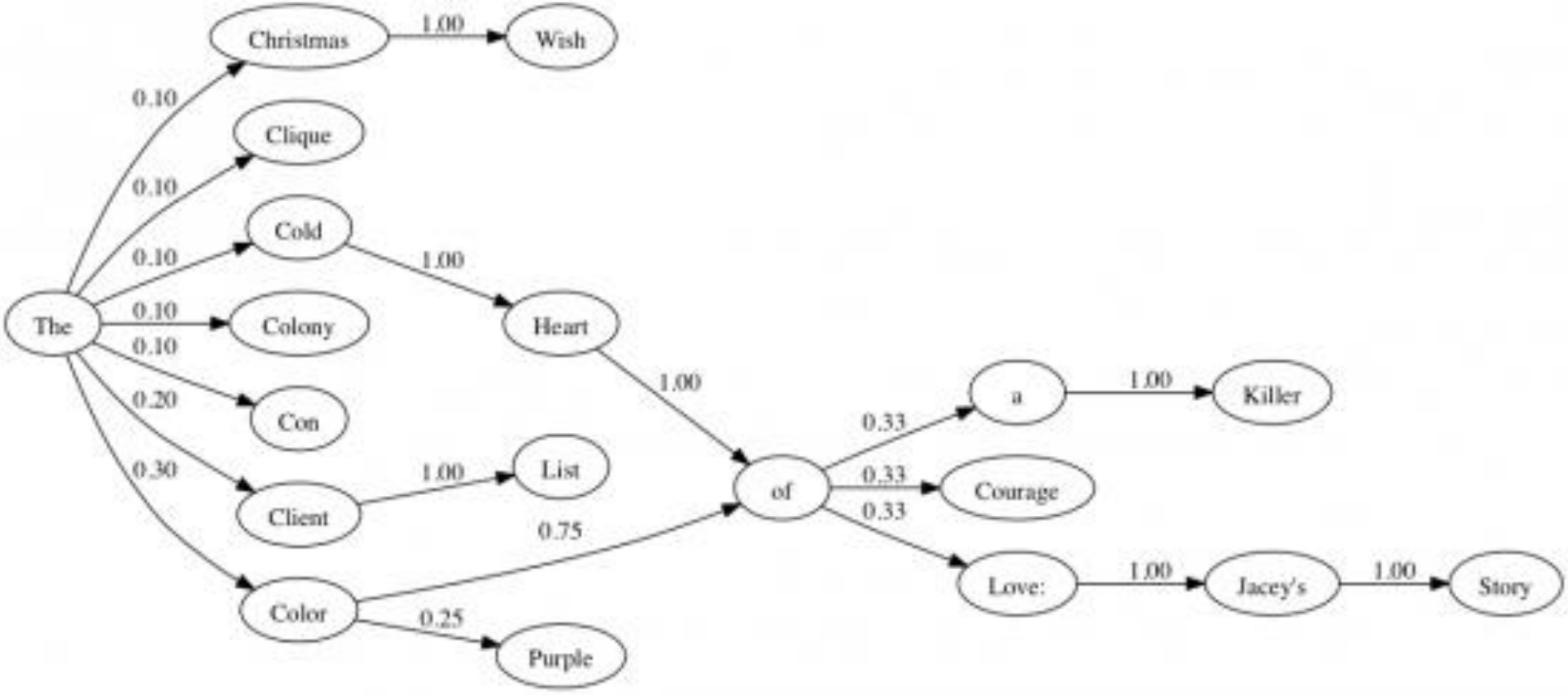
- Corpus of data
  - *Today it is hot and sunny*
  - *Today it is cold and cloudy*
  - *Today it is cold and snowing*

- The sentence starts with *Today*
- P(*it* | *Today*) = 1
- P(*is* | *it*) = 1
- P(*hot* | *it*) = 1/3
- P(*cold* | *it*) = 2/3

- With Markov chain Oder 1 we can have sentences like
  - *Today it is hot and snowing*

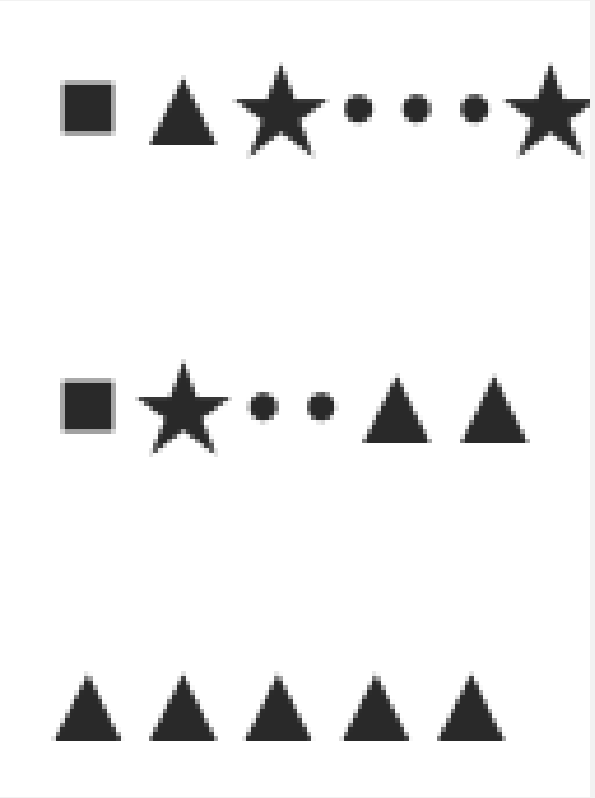
Perhaps higher order would be needed

MARKOV CHAIN– TEXT COMPOSITION



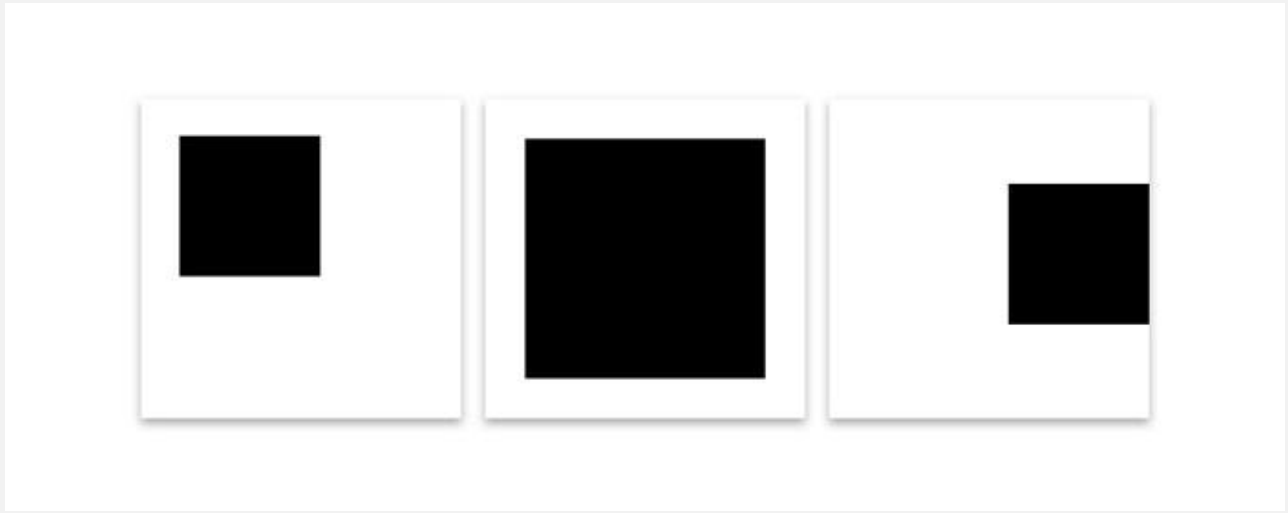
- https://projects.haykranen.nl/markov/demo/

MARKOV CHAIN– GRAPHIC



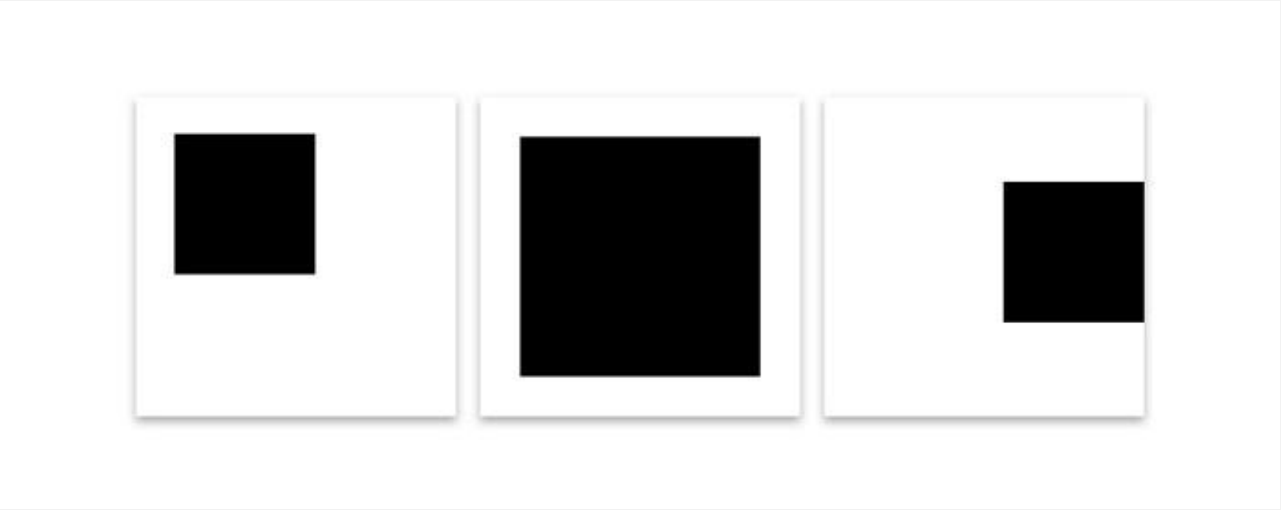
- We can consider a graphic texture like a sentence
- Each symbol is comparable to word, so each symbol corresponds to a state

MARKOV CHAIN– GRAPHIC



- More interesting is to generate new pictures in the same style of the corpus
- Let's consider a corpus of pictures
- Each of these images can be treated as a "sentence".
- **The concept of order stays the same, in that we say that the colour of the next pixel in the image (the next "word" in the "sentence") is determined by however many pixels came before that**.

MARKOV CHAIN– GRAPHIC



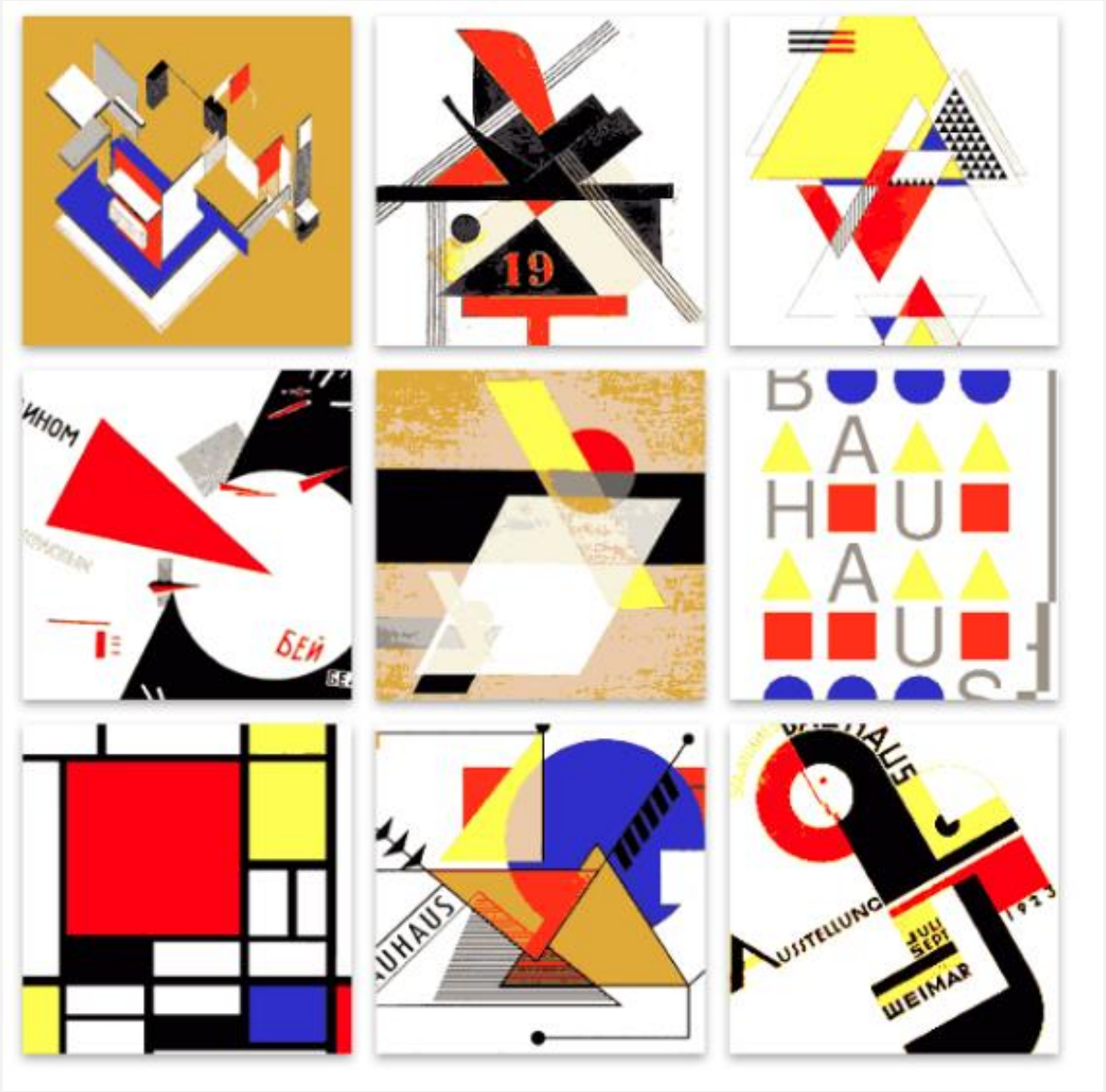
- Pictures are composed by patterns –> we need high order chains to "recreate those pattern"
- Give 200x200 images, let's take Markov Chain of order 200.
- Let's take the first line of pixel randomly and than generates the rest by using the generative chain

MARKOV CHAIN– GRAPHIC

- With a very limited corpus of data (3 pictures) we obtained interesting results

MARKOV CHAIN– GRAPHIC

- Bauhaus-style picture generation
- Size: 200x200 pixels



MARKOV CHAIN– GRAPHIC

- Result with Markov Chain of order 120

- Result with Markov Chain of order 90

- Result with Markov Chain of order 1

MARKOV MODEL – MUSIC COMPOSITION

- Idea
  - Take a corpus of sequences of chords used in beatles' songs

    ['F', 'Em7', 'A7', 'Dm', 'Dm7', 'Bb', 'C7', 'F', 'C', 'Dm7',...]

Goal

- Generate new chord sequences through Markov Chains

MARKOV CHAIN – MUSIC COMPOSITION

Learning process:

- Consider the chords sequence

  ['F', 'Em7', 'A7', 'Dm', 'Dm7', 'Bb', 'C7', 'F', 'C', 'Dm7',...]

- We can make **bigrams** out of it

  ['F Em7', 'Em7 A7', 'A7 Dm', 'Dm Dm7', 'Dm7 Bb', 'Bb C7', ...]

- Example: suppose that we want to start with chord F, we have 18 bigrams starting with it

  ['F Em7', 'F C', 'F F', 'F Em7', 'F C', 'F A7sus4', 'F A7sus4', ...]

- The bigram is equivalent to consider an order 1 in the Markov Model

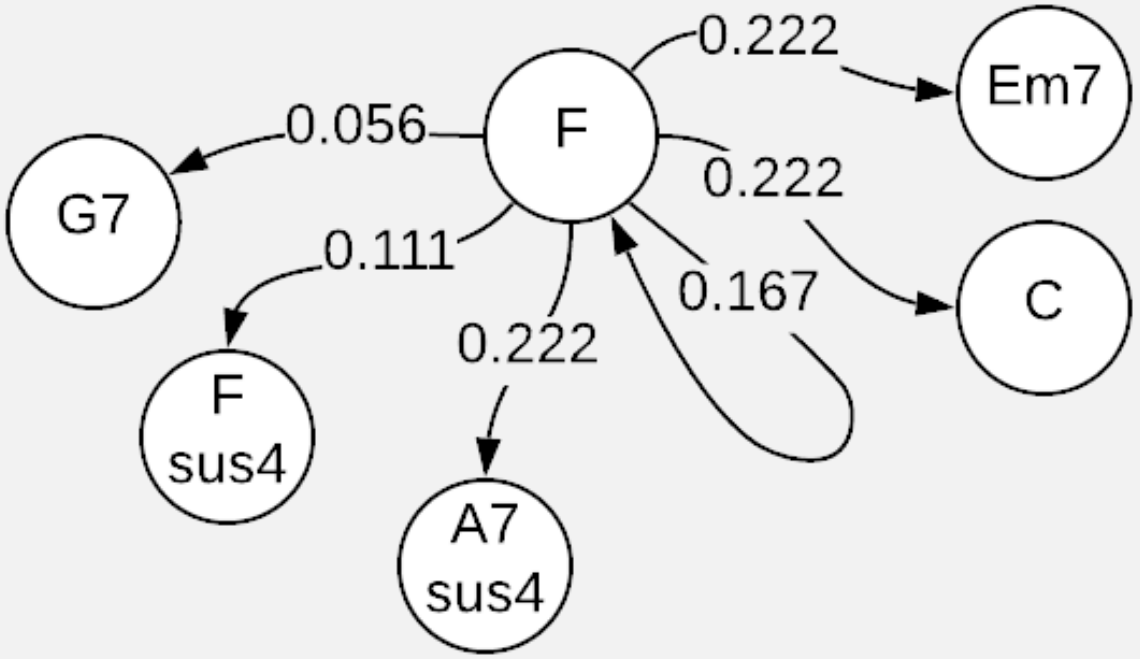
MARKOV CHAIN FOR MUSIC GENERATION

- we compute the frequency of each unique bigram appearing in the sequence we get

  'F Em7': 4, 'F C': 4, 'F F': 3, 'F A7sus4': 4, 'F Fsus4': 2, 'F G7': 1

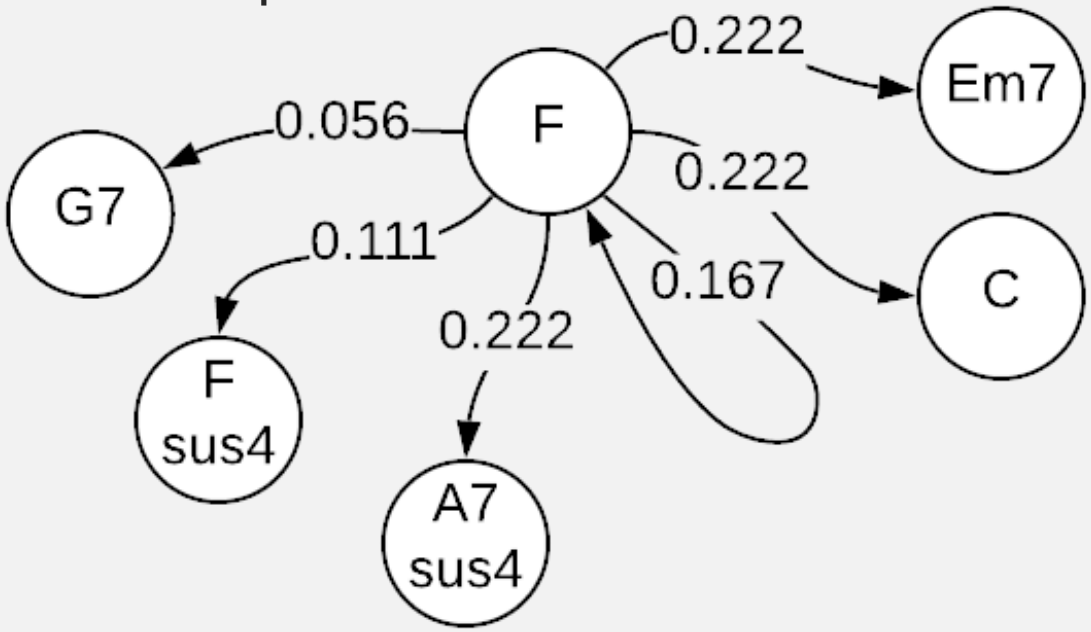
- If we normalize it, we can get the probability of each bigram, which can be interpreted as a Markov Chain graph

'F Em7': 0.222,
'F C': 0.222,
'F F': 0.167,
'F A7sus4': 0.222,
'F Fsus4': 0.111,
'F G7': 0.056



MARKOV CHAIN FOR MUSIC GENERATION

- Each node of this graph represents possible states that our sequence can achieve
- In this specific case: chords following *F*
- We can build a graph for each chord



Starting from a chord we can generate sequences by selecting the appropriate bigram graphs

e.g.

'Bb','Dm','C','Bb','C7','F','Em7','A7','Dm','Dm7','Bb','Dm','Gm6'

Markov Models of order n correspond to a set of n-grams

main.py

def predict_next_state(chord:str, data:l Until

Labs > Lab5_Cognitive_Agents > Exercise II > main.py > ...

```python
import numpy as np
import pandas as pd
from collections import Counter
import time
import argparse
from pythonosc import udp_client

#np.random.seed(42)


chords_midi_dict={
    'F':[5,9,12],
    'Em7':[4,9,11,14],
```

PROBLEMS 1 TERMINAL ... 2: Python

```
(cpac_keras) MacBook-Pro-di-luca-2:cpac_course_2020 lucacomanducci$
```

Lab5* Python 3.7.5 64-bit ('cpac_keras': conda) 0 1 LF Python

synth.scd

```
}).add;    // see below for more on .add
)


// Play chord

(
p=Pbind(
\instrument, \harpsi,
    \note, Pseq([[-7, 3, 7, 10]],1),
\dur, 1,
\legato, 0.4,
//\strum, 0.1 // try 0, 0.1, 0.2, etc
 ).play;
)


// OSC

(
x = OSCFunc( { | msg, time, addr, port |
    var
chord,note1,note2,note3,note4,pyFreq,pyAmp,pyDet
une,pyLfo;

    // Handle end of sound
    if (msg[1] =='stop'){
        h.free
    }
    {
    // handle class A message (freq and
amplitude)
        if (msg[1]=='chord3'){
            // Parse message
            note1 = msg[2].asFloat;
            note2  = msg[3].asFloat;
            note3  = msg[4].asFloat;
            chord=[note1,note2, note3];
            chord.postln();
        };
        if (msg[1]=='chord4'){
            // Parse message
            note1 = msg[2].asFloat;
            note2  = msg[3].asFloat;
            note3  = msg[4].asFloat;
            note4  = msg[5].asFloat;
            chord=[note1,note2, note3,note4];
```

Post window Auto Scroll

```
[ 0.0, 4.0, 7.0 ]
[ 5.0, 9.0, 12.0 ]
[ 2.0, 5.0, 9.0 ]
[ 10.0, 14.0, 17.0 ]
[ 2.0, 5.0, 9.0 ]
-> an EventStreamPlayer
-> an EventStreamPlayer
-> an EventStreamPlayer
-> an EventStreamPlayer
WARNING: keyword arg 'note' not fou
-> an EventStreamPlayer
-> an EventStreamPlayer
[ 10.0, 14.0, 17.0 ]
[ 0.0, 4.0, 7.0 ]
[ 5.0, 9.0, 12.0 ]
[ 2.0, 5.0, 9.0 ]
[ 10.0, 14.0, 17.0 ]
[ 5.0, 9.0, 12.0 ]
[ 2.0, 5.0, 9.0 ]
[ 0.0, 4.0, 7.0 ]
[ 5.0, 9.0, 12.0 ]
[ 5.0, 10.0, 12.0 ]
[ 5.0, 9.0, 12.0 ]
[ 2.0, 5.0, 9.0 ]
[ 10.0, 14.0, 17.0 ]
[ 2.0, 5.0, 9.0 ]
[ 10.0, 14.0, 17.0 ]
[ 5.0, 9.0, 12.0 ]
[ 10.0, 14.0, 17.0 ]
[ 5.0, 9.0, 12.0 ]
[ 5.0, 9.0, 12.0 ]
[ 5.0, 10.0, 12.0 ]
[ 5.0, 9.0, 12.0 ]
[ 2.0, 5.0, 9.0 ]
[ 10.0, 14.0, 17.0 ]
[ 2.0, 5.0, 9.0 ]
[ 0.0, 4.0, 7.0 ]
[ 5.0, 9.0, 12.0 ]
[ 2.0, 5.0, 9.0 ]
[ 0.0, 4.0, 7.0 ]
[ 5.0, 9.0, 12.0 ]
[ 2.0, 5.0, 9.0 ]
[ 10.0, 14.0, 17.0 ]
[ 5.0, 9.0, 12.0 ]
```

Interpreter: Active Server: 0.01% 0.03% 0u 0s 2g 109d 0.0dB

MARKOV MODEL – SOME CONSIDERATIONS

- Markov chains were much used in early automatic music composition
- **CONS**:
- Markov chains created by hands
  - Set of rules to limited
  - Expensive as human effort
- Markov chains generated from a corpus of pre-existing compositions
  - Captured just local statistical similarities
  - Low order Markov chains produced strange, unmusical compositions that wandered aimlessly
  - High order ones essentially rehashed musical segments from the corpus and were also very computationally expensive to train
  - A fixed order is not able to capture the inner structure of the composition

MARKOV MODEL – SOME CONSIDERATIONS

- Lately, Markov chains started to be used:
  - to provide musical ideas to composers, even if the probabilities are specified by hand instead of generated from a corpus
  - with a pre-processing stage on the corpus and with post-processing stage with constraints
  - with other methods
    - Verbeurgt et al. [2004] used Markov chains to generate a basic pattern for the melody, which was then refined with an artificial neural network.
    - Lo and Lucas (2006) trained Markov chains with classic music pieces, but, instead of generating compositions with them, used them as fitness evaluators in an evolutionary algorithm to evolve melodies encoded as sequences of pitches
    - Pachet et al. (2011) proposed a framework to combine Markovian generation of music with rules (constraints) to produce better results

MARKOV MODEL – SOME CONSIDERATIONS

- Markov chains remained a feasible option for restricted problems
  - real-time performances, as jazz improvisation
  - style imitation on automatic music composition
  - musicological studies on pattern analysis on music composition (study the repeated pattern in a specific composer)

- All these areas are restricted, the corpus of data to analyse is limited
- A good result can be achieved with even with a not generalized transition probability matrix

MARKOV MODEL – JAZZ CONTINUATOR

- Improvisation is not "freely playing whatever comes "
- The soloist is rarely playing completely free.
- An improvisational soloist is always following a complicated set of rules and being creative within the context of those rules.
- Freedom in jazz improvisation comes from understanding **structure** and attaining from **improvisational resources**

Patterns

MARKOV MODEL – – JAZZ CONTINUATOR

- Composers tend to compose a melody using a sequence of sub-structures (part of the melody) each with a certain semantic
  - An example is the chords progression
- The length of each sub-structure can vary along the piece

- During improvisation often musicians develop repetitive sequence of notes called pattern

- In order to capture the semantic, the nuances and the intention of the composer or the improviser, a good model should be able to capture an entire sub-structures or patterns
- A n-order Markov Chain is effective in capturing only sub-structures or pattern of length n

MARKOV MODEL – VARIABLE LENGTH

- However, a problem still remains -> capture the inner structure of a piece that, generally, has not a fixed length
- **Variable Length Markov Chains** attempt to overcome the issue

- In a Markov Chain model each random variable in a sequence with a Markov property depends on a fixed number of random variables.
- In the VLM model this number may vary depending on the specific observed realization: **context,** which is

$$x_1^n = x_1 x_2 \ldots x_n$$

n realizations of a random variable (sequence of notes)

MARKOV MODEL – VARIABLE LENGTH

- Review:
  - The learning phase generates $P(x_i|S)$ for a sequence S (context) of events of a certain length
  - The $P(x_i|S)$ is then used in the generative phase
  - This model provides a probability assignment for each state in the sequence given its past states

- In Markov Chain S has a fixed length
- In VLMC the length of S can vary along the corpus

MARKOV MODEL – VARIABLE LENGTH

- VLMD are models that predict the conditional probability distribution at any level of depth
- Three components:
  - **Counting**: build up the transition table
  - **Smoothing**: deals with "unseen" sequences and on when there is only one continuation by putting together continuations of lower levels
  - **Variable length modelling:** instead of maintaining a representation for each length create a more efficient, effective and unified model

MARKOV MODEL – VARIABLE LENGTH

- **Counting**
  - The transition probability matrix is built by counting the number of occurrences of symbols σ appearing after the context S in the training sequence.
  - All the sequences S of variable length should be stored in the probability matrix
  - -> The training and the generative phases can be not computationally affordable

  - Most of the VLMM take advantage of a variable length model to provide a mode efficient organization of the transition probability matrix
  - Many methods:
    - **Probabilistic Suffix Ttree (PTS)**
    - Context Tree Weighting (CTW),
    - Prediction by Partial Match (PPM)

MARKOV MODEL – JAZZ CONTINUATOR

- A **Jazz continuator** learns from the soloist its styles (patterns and improvisational resources) and continues from where the soloist stop or play with him/her a "face-to-face improvisation challenge"
- It is important to encode musical information in the model

https://www.youtube.com/watch?v=TT7R4aPtAf0

MATERIALS

- **References**

- Curtis Roads – The Computer music Tutorial, 1996 – Chapter 19
- Jose David Fernandez, Francisco Vico, AI Methods in Algorithmic Composition: A Comprehensive Survey, Journal of Artificial Intelligence Research 48 (2013)
- Chien-Hung Liu and Chuan-Kang Ting, Computational Intelligence in Music Composition: A Survey, IEEE Transactions on Emerging Topics in Computational Intelligence, December 2016,
- S. Bollini, The Virtual Jazz Player: real-time modelling of improvisational styles using Variable-Length Markov Models incorporating musicological rules, Master Thesis Politecnico di Milano, 2011
- A. Tolver, An introduction to Markov Chain, Lecture notes for stochastic process, Department of Mathematical Sciences, 2016

MATERIALS

- **Further readings**
- Verbeurgt, K., Fayer, M., & Dinolfo, M. (2004). A hybrid Neural-Markov approach for learning to compose music by example. In Proceedings of the Canadian Conference on Advances in Artificial Intelligence, pp. 480–484.
- Lo, M., & Lucas, S. M. (2006). Evolving musical sequences with N-Gram based trainable fitness functions. In Proceedings of the IEEE Conference on Evolutionary Computation
- Pachet, F., Roy, P., & Barbieri, G. (2011). Finite-length Markov processes with constraints. In Proceedings of the International Joint Conference on Artificial Intelligence.
- M. Machler, Peter Bühlmann, Variable Length Markov Chains: Methodology, Computing and Software - Journal of Computational and Graphical Statistics · January 2012