COT 2000
Foundations of Computing

Spring 2024

Lecture 21 – part 1

Lab 10 (Optional)
Homework 7 – 07/29/24
Exam 4 – 08/02/24

Lecture 21 – part 2

Review

Review

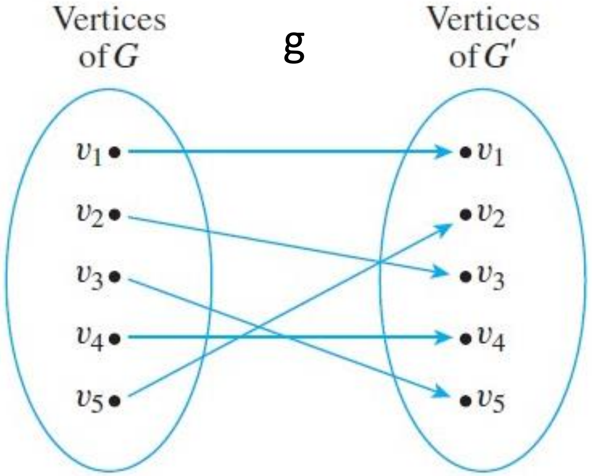
- Graph Isomorphisms
- Degree of a vertex, outdegree, indegree
- Isomorphic Invariants
- Graphic Sequence
- Data structures
  - Adjacency Matrix
  - Edge dictionary
  - Edge list

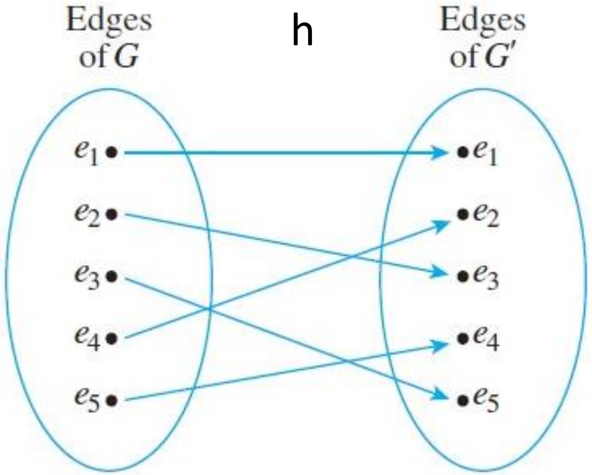
Graph Isomorphisms

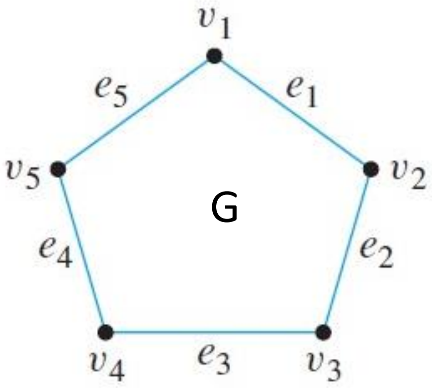
• Definition

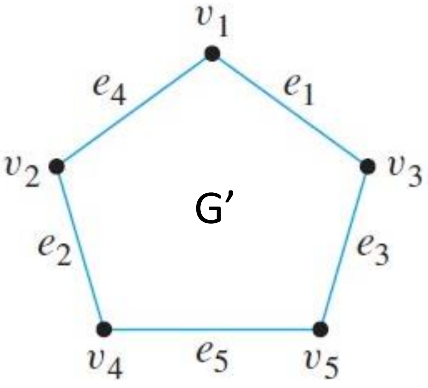
Let G and G' be graphs with vertex sets $V(G)$ and $V(G')$ and edge sets $E(G)$ and $E(G')$, respectively. **G is isomorphic to G'** if, and only if, there exist one-to-one correspondences $g: V(G) \to V(G')$ and $h: E(G) \to E(G')$ that preserve the edge-endpoint functions of G and G' in the sense that for all $v \in V(G)$ and $e \in E(G)$,

$$v \text{ is an endpoint of } e \quad \Leftrightarrow \quad g(v) \text{ is an endpoint of } h(e). \qquad 10.4.1$$



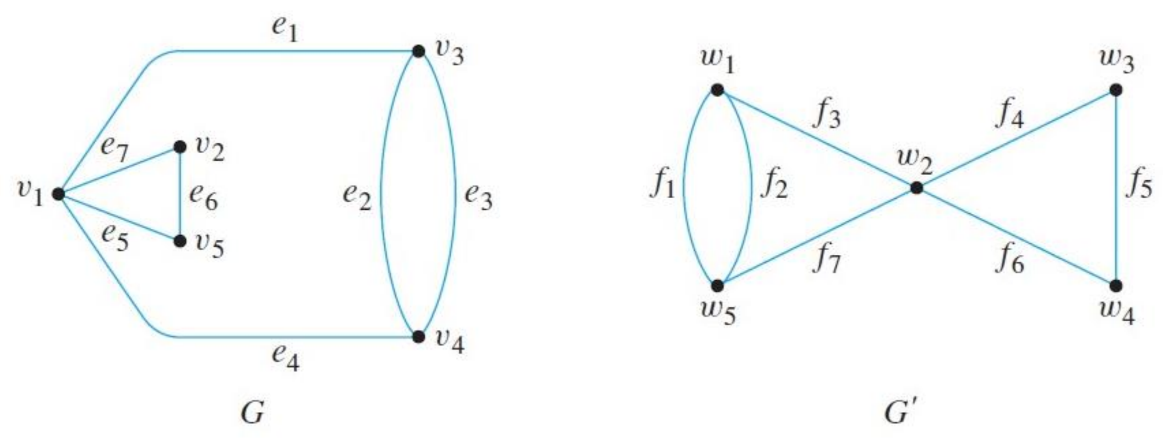






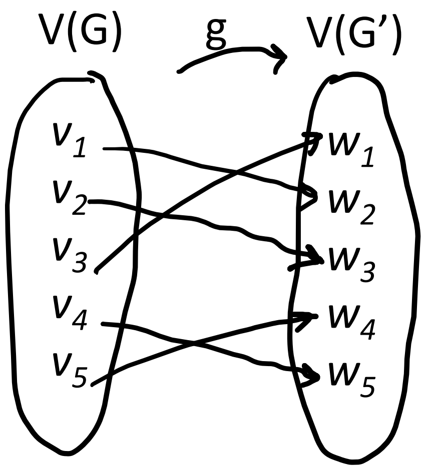
Note that these relabeling functions are one-to-one and onto.

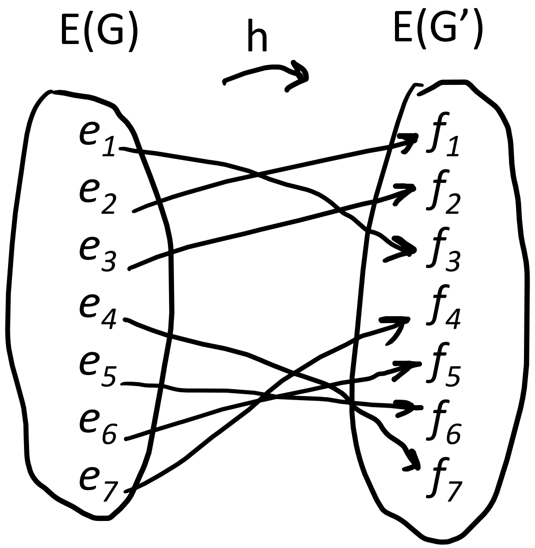
Exercise: Show that the following two graphs are isomorphic.



Solution:

To solve this problem, you must find functions **g:** V(G) → V(G’) and **h:** E(G) → E(G’)
such that for all **v** ∈ V(G) and **e** ∈ E(G), **v** is an endpoint of **e** if, and only if, **g(v)** is an endpoint of **h(e)**.





Conclusion:
There is isomorphism between *G* and *G’*.

Degree of a vertex

(a) Let v be a vertex of an **undirected graph**.

- The **degree** of v, denoted deg(v), is the number of edges that connect v to the other vertices in the graph.

(b) If v is a vertex of a **directed graph:**

- Then the **outdegree** of v, denoted outdeg(v), is the number of edges of the graph that initiate at v.
- The **indegree** of v, denoted indeg(v), is the number of edges that terminate at v.

Degree Sequence of a Graph: The degree sequence of a simple undirected graph is the non-increasing sequence of its vertex degrees

Isomorphic Invariant

A property that is preserved by graph isomorphism is called an **isomorphic invariant.**

Example:

- if you are given two graphs, one with 16 vertices and the other with 17, you can immediately conclude that the two are not isomorphic**.**

• Definition

A property P is called an **invariant for graph isomorphism** if, and only if, given any graphs G and G', if G has property P and G' is isomorphic to G, then G' has property P.

Theorem 10.4.2

Each of the following properties is an invariant for graph isomorphism, where n, m, and k are all nonnegative integers:

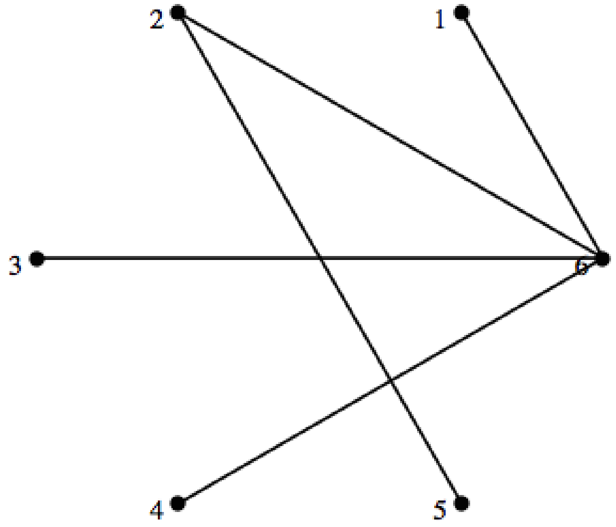
1. has n vertices;
2. has m edges;
3. has a vertex of degree k;
4. has m vertices of degree k;
5. has a circuit of length k;
6. has a simple circuit of length k;
7. has m simple circuits of length k;
8. is connected;
9. has an Euler circuit;
10. has a Hamiltonian circuit.

Traversals

Graphic Sequence

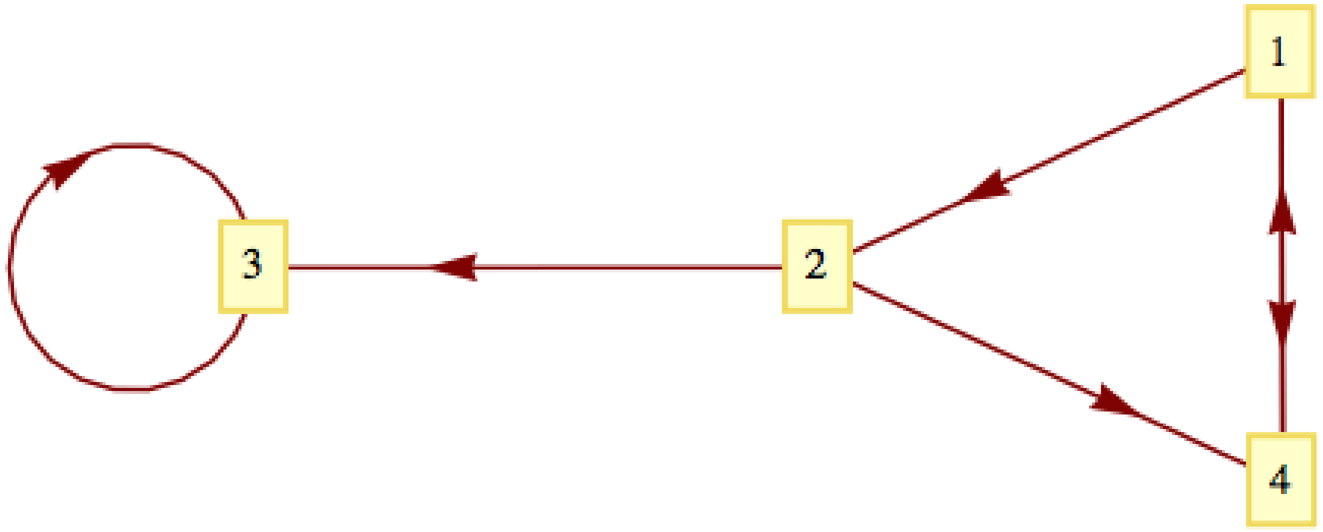
- A finite nonincreasing <u>sequence</u> of integers **d1, d2, . . . , dn** is **graphic** if there exists a simple undirected graph with n vertices having the sequence as its degree sequence.

Example: Is this sequence 4, 2, 1, 1, 1, 1 a graphic sequence ?



The sequence 4, 2, 1, 1, 1, 1 is **graphic** because the degrees of the graph in the figure match these numbers.
Note: There is no connection between the vertex number and its degree in this graph.

Example:



Adjacency Matrix

$$G = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

Edge Dictionary

G = { 1:[2,4], 2:[3,4], 3:[3], 4:[1] }

Edge List

G = [(1,2),(1,4),(2,3),(2,4),(3,3),(4,1)]

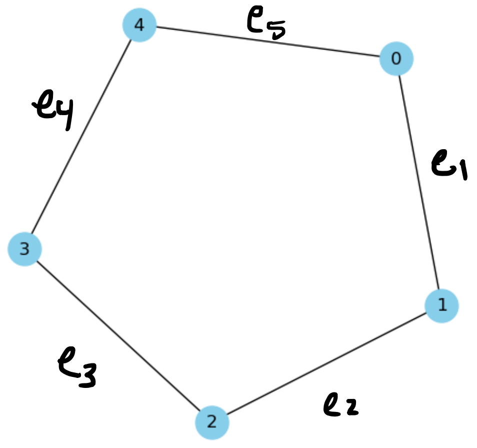
Exercise:

Directed graphs $G_1, \ldots, G_6$, each with vertex set {1, 2, 3, 4, 5} are represented by the matrices below. Which graphs are isomorphic to one another?

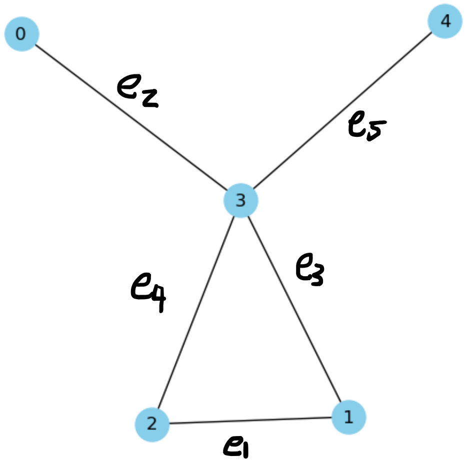
$$G_1 : \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix} \quad G_2 : \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad G_3 : \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$$G_4 : \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad G_5 : \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix} \quad G_6 : \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

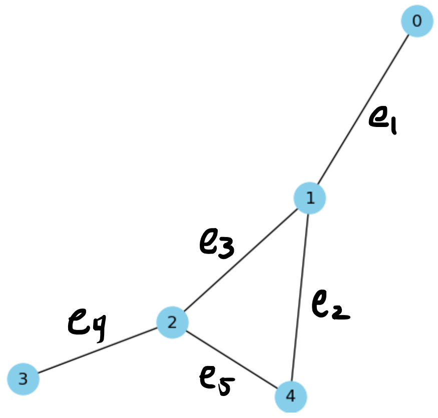
G1



G2



G3



G1

	v0	v1	v2	v3	v4
v0	0	1	0	0	0
v1	0	0	1	0	0
v2	0	0	0	1	0
v3	0	0	0	0	1
v4	1	0	0	0	0

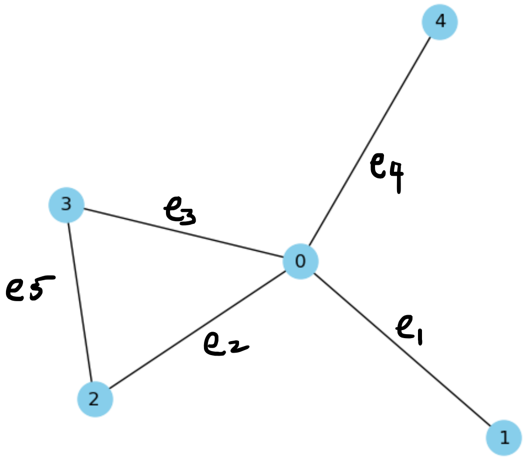
G2

	v0	v1	v2	v3	v4
v0	0	0	0	0	0
v1	0	0	1	0	0
v2	0	0	0	0	0
v3	1	1	1	0	1
v4	0	0	0	0	0

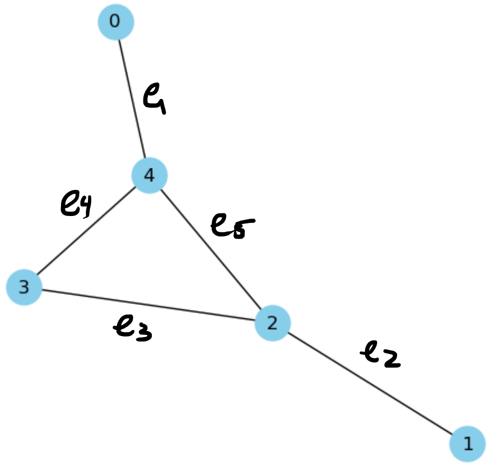
G3

	v0	v1	v2	v3	v4
v0	0	0	0	0	0
v1	1	0	0	0	1
v2	0	1	0	0	0
v3	0	0	1	0	0
v4	0	0	1	0	0

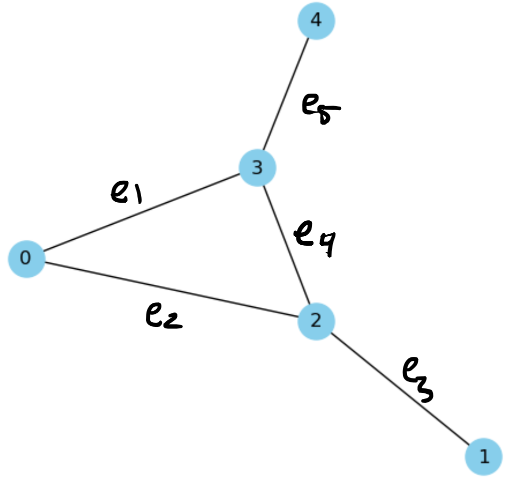
G4



G5



G6



G4

	v0	v1	v2	v3	v4
v0	0	1	1	1	1
v1	0	0	0	0	0
v2	0	0	0	0	0
v3	0	0	1	0	0
v4	0	0	0	0	0

G5

	v0	v1	v2	v3	v4
v0	0	0	0	0	1
v1	0	0	0	0	0
v2	0	1	0	1	0
v3	0	0	0	0	1
v4	0	0	1	0	0

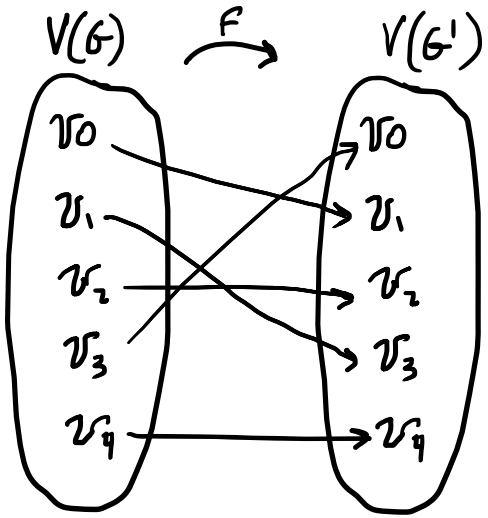
G6

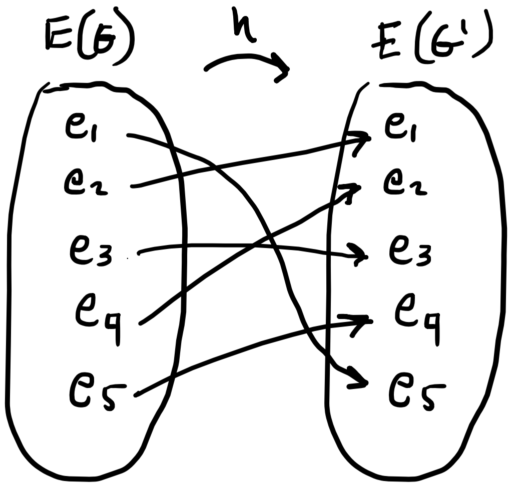
	v0	v1	v2	v3	v4
v0	0	0	0	1	0
v1	0	0	0	0	0
v2	1	1	0	0	0
v3	0	0	1	0	0
v4	0	0	0	1	0

Python code

```
{'G1 and G2': False,
 'G1 and G3': False,
 'G1 and G4': False,
 'G1 and G5': False,
 'G1 and G6': False,
 'G2 and G3': False,
 'G2 and G4': True,
 'G2 and G5': False,
 'G2 and G6': False,
 'G3 and G4': False,
 'G3 and G5': True,
 'G3 and G6': True,
 'G4 and G5': False,
 'G4 and G6': False,
 'G5 and G6': True}
```

✓





Lecture 21 – part 3

Special Topic (*)

The Power of Tsetlin Machines

*Note: This topic is not included on exam.

Traditional Neural Network

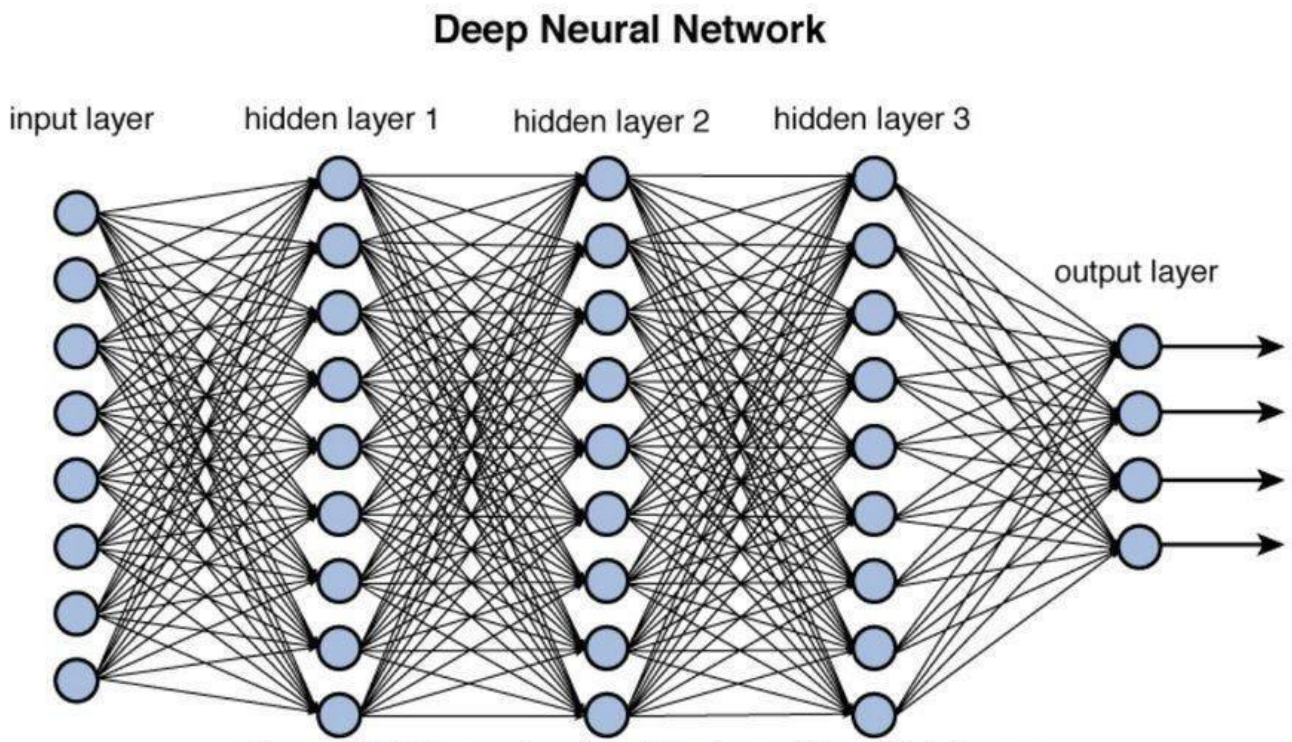


Figure 12.2 Deep network architecture with multiple layers.

What are Tsetlin Machines

Tsetlin Automata

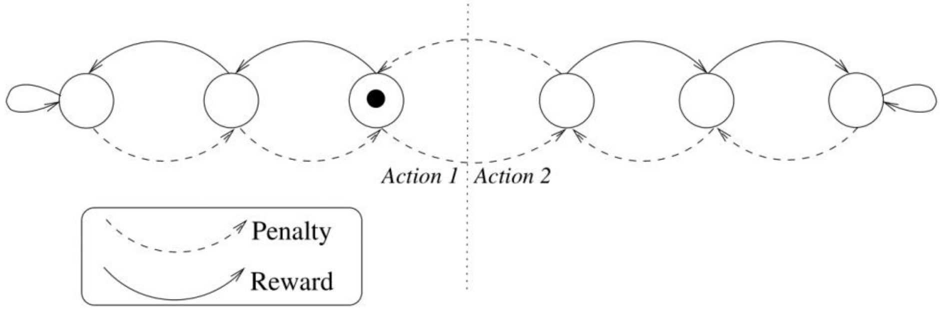


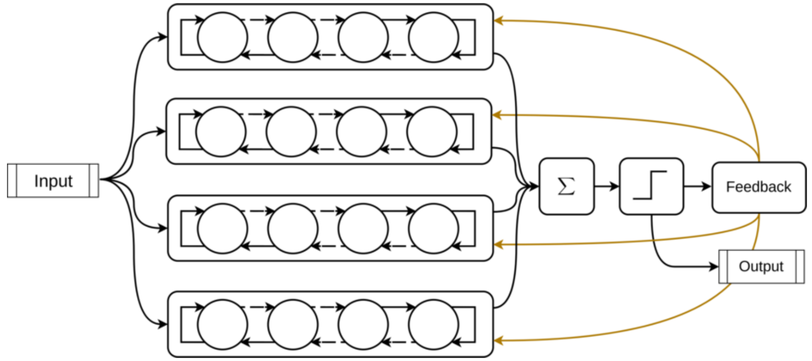
Image reproduced from https://arxiv.org/pdf/1804.01508.pdf

+

Propositional Logic

If condition then class

= Tsetlin Machine

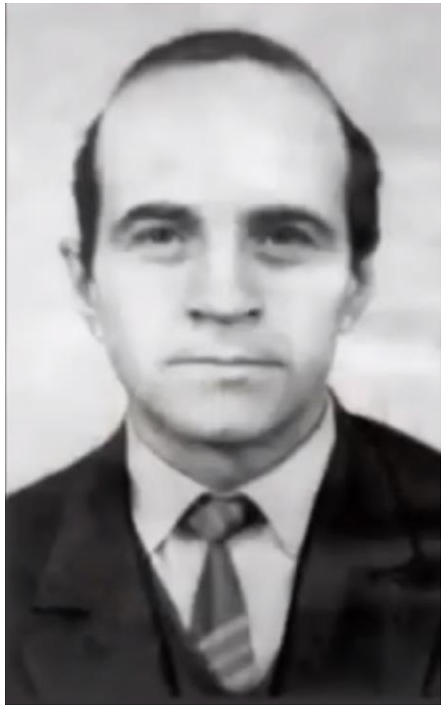


Advantages

- **Scalability:** The parallel nature of TMs allows them to be effectively scaled for large datasets and complex problems.
- **Transparency and Interpretability:** The learned rules are highly interpretable. Each clause in the model can be understood as a human-readable rule, making it easier to analyze and trust the decisions made by the model.
- **Highly efficient**, both in terms of computational resources and power consumption, making them suitable for edge computing devices.

Tsetlin Machines have been applied for classification, regression, and even image recognition, demonstrating their versatility and potential as a complement or alternative to traditional machine learning models.

Michael Lvovitch Tsetli (1924 – 1966)
Invented the Tsetlin automaton (1961)



Professor Ole-Christoffer Granmo
Created the Tsetlin machine (2018)

UiA University of Agder
CAIR CENTRE FOR ARTIFICIAL INTELLIGENCE RESEARCH

University of Agder, Norway

M. L. Tsetlin, “On behaviour of finite automata in random medium,” Avtomat. i Telemekh, vol. 22, no. 10, pp. 1345–1354, 1961

Granmo, Ole-Christoffer. "The Tsetlin Machine--A Game Theoretic Bandit Driven Approach to Optimal Pattern Recognition with Propositional Logic." arXiv preprint arXiv:1804.01508 (2018).

Extremely simple dataset

Input Features

Vehicles dataset						
Datapoint	**Four Wheels**	**Transport People**	**Wings**	**Yellow**	**Blue**	**Vehicle**
1	Yes	Yes	No	No	Yes	Car
2	Yes	Yes	No	Yes	No	Car
3	Yes	Yes	No	Yes	No	Car
4	Yes	Yes	Yes	No	Yes	Plane
5	Yes	No	Yes	Yes	No	Plane
6	No	Yes	Yes	No	Yes	Plane

Output Classes

Rule Examples

If condition **then class**

The **condition** part is a placeholder for a Boolean expression that describes a **pattern** in the data, to be learnt by the Tsetlin machine.

Rule 1: **If** (four wheels **and** Transport People) **then car**

Rule 2: **If** (wings) **then plane**

Rule n:

Binarization

Literals: Input features and their negations

Dataset											
#	Four W	T.People	Wings	Yellow	Blue	**not** Four W	**not** T. People	**not** wings	**not** yellow	**not** blue	Class
1	1	1	0	0	1	0	0	1	1	0	car
2	1	1	0	1	0	0	0	1	0	1	car
3	1	1	0	1	0	0	0	1	0	1	car
4	1	1	1	0	1	0	0	0	1	0	plane
5	1	0	1	1	0	0	1	0	0	1	plane
6	0	1	1	0	1	1	0	0	1	0	plane

Rule Memory

A Tsetlin machine simulates forgetting and memorization.
Literals under position five(5) **do not participate** on the rule condition.

Maximally Memorized	10										
	9										
	8										
	7	Four W									
Memorized	6		T. People						**not** wings		
Forgotten	5				Yellow						
	4					Blue					**not** blue
	3						**not** Four W	**not** T. People		**not** yellow	
	2			Wings							
Maximally Forgotten	1										

Tsetlin Automata (columns)

Updated rule: If (Four W and T. People and not wings) **then** car

Rule Learning Algorithm

Initialization:

Set all literals to position 5

Step 1: Rule Evaluation:

Observe object and evaluate condition by assessing object literals.

Step 2: Recognize Feedback (Type 1a)

<u>If rule condition is true:</u>

Memorize the object true literals by incrementing position for p<Memorization_value.

Forget the object false literals by decrementing position for p<Forget_value.

Step 3: Erase Feedback (Type 1b)

<u>If rule condition is false</u>

Forget all literals (True or False) by decrementing position for p<Forget_value.

Step 4: Reject Feedback (Type 2) – (For a different class)

<u>If rule condition is true:</u>

Increment all forgotten literals (pos <=5) that are false for the object (no randomization).

<u>Else:</u> do nothing

Initialization

All literals initialized at memory position 5

Maximally Memorized	10										
	9										
	8										
	7										
Memorized	6										
Forgotten	5	Four W	T. People	Wings	Yellow	Blue	**not** Four W	**not** T. People	**not** wings	**not** yellow	**not** blue
	4										
	3										
	2										
Maximally Forgotten	1										

Updated rule : **If** () **then** car
Condition : (<empty>) = True (By default)

Datapoint 1:

Step 1. Rule evaluation: Condition: (empty) = **True** **Before**

Maximally Memorized	10										
	9										
	8										
	7										
Memorized	6										
Forgotten	5	Four W	T. People	Wings	Yellow	Blue	**not** Four W	**not** T. People	**not** wings	**not** yellow	**not** blue
	4										
	3										
	2										
Maximally Forgotten	1										

Step 2: Recognize feedback (Type 1a)

Car										
#	Four W	T.People	Wings	Yellow	Blue	**not** Four W	**not** T. People	**not** wings	**not** yellow	**not** blue
1	**1**	**1**	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**0**
Prob.	0.03	0.02	0.50	0.60	0.01	0.80	0.70	0.30	0.20	0.50
Skip Increment if p>0.1	+1	+1			+1			skip	skip	
Skip decrement if p>0.9			-1	-1		-1	-1			-1

Datapoint 1:

After

Maximally Memorized	10										
	9										
	8										
	7										
Memorized	6	Four W	T. People			Blue					
Forgotten	5								**not** wings	**not** yellow	
	4			Wings	Yellow		**not** Four W	**not** T. People			**not** blue
	3										
	2										
Maximally Forgotten	1										

Updated rule : **If** (Four W and T.People and Blue) **then** car

Datapoint 2:

Step 1. Rule evaluation: (Four W and T.People and Blue) = (1 and 1 and 0) = **False** **Before**

Maximally Memorized	10										
	9										
	8										
	7										
Memorized	6	Four W	T. People			Blue					
Forgotten	5								**not** wings	**not** yellow	
	4			Wings	Yellow		**not** Four W	**not** T. People			**not** blue
	3										
	2										
Maximally Forgotten	1										

Step 3: Erase feedback (Type 1b)

Car										
#	Four W	T.People	Wings	Yellow	Blue	**not** Four W	**not** T. People	**not** wings	**not** yellow	**not** blue
1	**1**	**1**	**0**	**1**	**0**	**0**	**0**	**1**	**0**	**0**
Prob.	0.95	0.1	0.2	0.25	0.3	0.5	0.6	0.95	0.7	0.8
Skip decrement if p>0.9	Skip	-1	-1	-1	-1	-1	-1	skip	-1	-1

Datapoint 2:

After

Maximally Memorized	10										
	9										
	8										
	7										
Memorized	6	Four W									
Forgotten	5		T. People			Blue			**not** wings		
	4									**not** yellow	
	3			Wings	Yellow		**not** Four W	**not** T. People			**not** blue
	2										
Maximally Forgotten	1										

Updated rule : **If** (Four W) **then** car

Datapoint 3:

Step 1. Rule evaluation: (Four W) = (1) = **True**

Before

Maximally Memorized	10										
	9										
	8										
	7										
Memorized	6	Four W									
Forgotten	5		T. People			Blue			**not** wings		
	4									**not** yellow	
	3			Wings	Yellow		**not** Four W	**not** T. People			**not** blue
	2										
Maximally Forgotten	1										

Step 3: Recognize feedback (Type 1a)

Car										
#	Four W	T.People	Wings	Yellow	Blue	**not** Four W	**not** T. People	**not** wings	**not** yellow	**not** blue
1	**1**	**1**	**0**	**1**	**0**	**0**	**0**	**1**	**0**	**1**
Prob.	0.05	0.01	0.5	0.03	0.6	0.7	0.8	0.95	0.85	0.95
Skip Increment if p>0.1	+1	+1		+1				skip		skip
Skip decrement if p>0.9			-1		-1	-1	-1		-1	

Datapoint 3:

After

Maximally Memorized	10										
	9										
	8										
	7	Four W									
Memorized	6		T.People								
Forgotten	5								**not** wings		
	4				Yellow	Blue					
	3									**not** yellow	**not** blue
	2			Wings			**not** Four W	**not** T. People			
Maximally Forgotten	1										

Updated rule : **If** (Four W and T.People) **then** car

Datapoint 4:

Step 1. Rule evaluation: (Four W and T.People) = (1 and 1) = **True** **Before**

Maximally Memorized	10										
	9										
	8										
	7	Four W									
Memorized	6		T.People								
Forgotten	5								**not** wings		
	4				Yellow	Blue					
	3									**not** yellow	**not** blue
	2			Wings			**not** Four W	**not** T. People			
Maximally Forgotten	1										

Step 3: Reject feedback (Type 2)

Plane										
#	Four W	T.People	Wings	Yellow	Blue	**not** Four W	**not** T. People	**not** wings	**not** yellow	**not** blue
1	**1**	**1**	**1**	**0**	**1**	**0**	**0**	**0**	**1**	**0**
Prob.	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	Skip	Skip	Skip	+1	Skip	+1	+1	+1	Skip	+1

Datapoint 4:

After

Maximally Memorized	10										
	9										
	8										
	7	Four W									
Memorized	6		T.People						**not** wings		
Forgotten	5				Yellow						
	4					Blue					**not** blue
	3						**not** Four W	**not** T. People		**not** yellow	
	2			Wings							
Maximally Forgotten	1										

Updated rule : **If** (Four W and T.People and not wings) **then** car

Datapoint 5:

Step 1. Rule evaluation: (Four W and T.People and not wings) = (1 and 0 and 0) = **False** **Before**

Maximally Memorized	10										
	9										
	8										
	7	Four W									
Memorized	6		T.People						**not** wings		
Forgotten	5				Yellow						
	4					Blue					**not** blue
	3						**not** Four W	**not** T. People		**not** yellow	
	2			Wings							
Maximally Forgotten	1										

Step 3: Reject feedback (Type 2)

Plane										
#	Four W	T.People	Wings	Yellow	Blue	**not** Four W	**not** T. People	**not** wings	**not** yellow	**not** blue
1	**1**	**0**	**1**	**1**	**0**	**0**	**1**	**0**	**0**	**1**
Prob.	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	Skip	Skip	Skip	Skip	Skip	Skip	Skip	Skip	Skip	Skip

Datapoint 5:

After

Maximally Memorized	10										
	9										
	8										
	7	Four W									
Memorized	6		T.People						**not** wings		
Forgotten	5				Yellow						
	4					Blue					**not** blue
	3						**not** Four W	**not** T. People		**not** yellow	
	2			Wings							
Maximally Forgotten	1										

Updated rule : **If** (Four W and T.People and not wings) **then** car

Datapoint 6:

Step 1. Rule evaluation: (Four W and T.People and not wings) = (0 and 1 and 0) = **False** **Before**

Maximally Memorized	10										
	9										
	8										
	7	Four W									
Memorized	6		T.People						**not** wings		
Forgotten	5				Yellow						
	4					Blue					**not** blue
	3						**not** Four W	**not** T. People		**not** yellow	
	2			Wings							
Maximally Forgotten	1										

Step 3: Reject feedback (Type 2)

Plane										
#	Four W	T.People	Wings	Yellow	Blue	**not** Four W	**not** T. People	**not** wings	**not** yellow	**not** blue
1	**0**	**1**	**1**	**0**	**1**	**1**	**0**	**0**	**1**	**0**
Prob.	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	Skip	Skip	Skip	Skip	Skip	Skip	Skip	Skip	Skip	Skip

Datapoint 6:

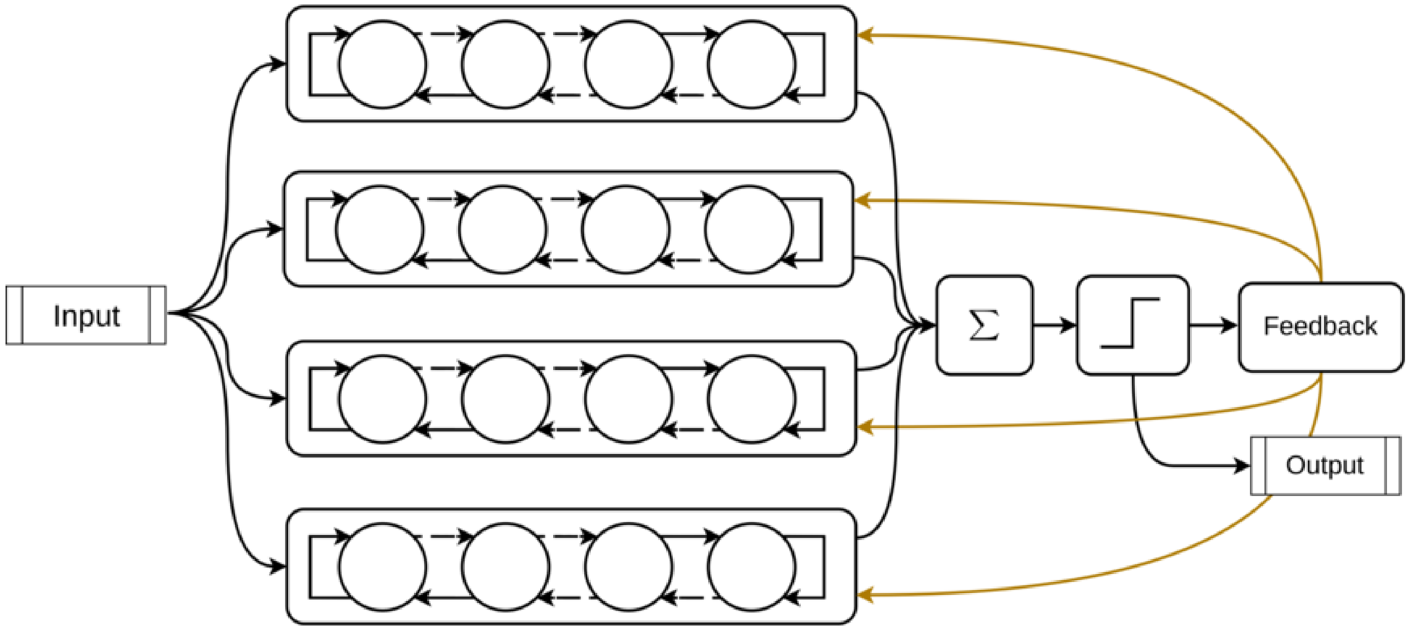
After

Maximally Memorized	10										
	9										
	8										
	7	Four W									
Memorized	6		T.People						**not** wings		
Forgotten	5				Yellow						
	4					Blue					**not** blue
	3						**not** Four W	**not** T. People		**not** yellow	
	2			Wings							
Maximally Forgotten	1										

Updated rule : **If** (Four W and T.People and not wings) **then** car

End of epoch

Rule coordination



Resources

- Professor Ole-Christoffer Granmo publications: Google Scholar Page
- Book (WIP): https://tsetlinmachine.org/
- Code and Datasets on GitHub: https://github.com/cair/TsetlinMachine
- International Symposium on the Tsetlin Machine (ISTM): https://istm.no/
- Python Package: https://pypi.org/project/pyTsetlinMachine/