Algorithms and Games

Michel de Rougemont

University Paris II & IRIF CNRS

Games

1. Games for Logic
   - Elementary equivalence
   - Logic and Complexity
   - Probabilistic method

2. Strategic games

3. Large Language models for Games
   - Transformers
   - Learning strategies and Chain of thought
   - Probing

1. Games for Logic

- Classes of Finite structures:
  - Words $\{U_n = (\{1,2,\ldots n\}, P, <)\colon\ n = 1,2 \ldots\}$
  - Graphs $\{G_n = (\{1,2,\ldots n\},\ E)\colon\ n = 1,2 \ldots\}$
  - Graphs with 2 distinguished elements

$$\{G\_n = (\{1,2,\ldots n\}, E, s\ t)\colon\ n = 1,2 \ldots\}$$

- Graph property P : degree 2, Connexity,….
  - Definable in some Logic L : $\forall n\ \ G_n \vDash\ \varphi(\quad) \leftrightarrow G_n \vDash\ P$

- Logical formulas
  - First order Logic $\exists x \forall y$ E(x,y)
  - Monadic Second order Logic, $\sum_1^1$ formulas

$$\exists U\ \ \exists x\ U(x) \cap [\ \forall y\ \mathrm{U}(y) \ \to \mathrm{E}(x,y)\ ]$$

1. Games for Logic

Elementary Equivalence: $U \approx V$ if $U \models \varphi \leftrightarrow V \models \varphi$

Ehrenfeucht-Fraisse games (1950's):

- 2 Players: Spoiler and Duplicator
- r-pebbles, winning condition for Duplicator: partial r-isomorphism

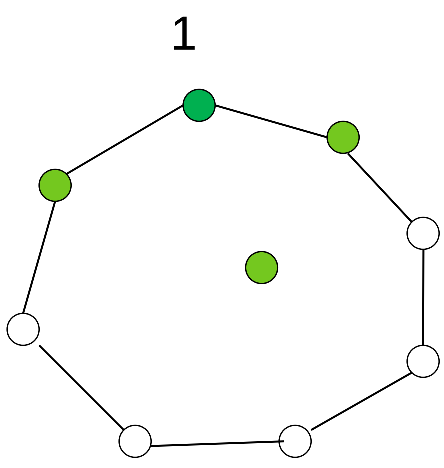
- P is First-order definable if there exists r, such that for all G_0 in P and G_1 in not P, Spoiler has a winning strategy.

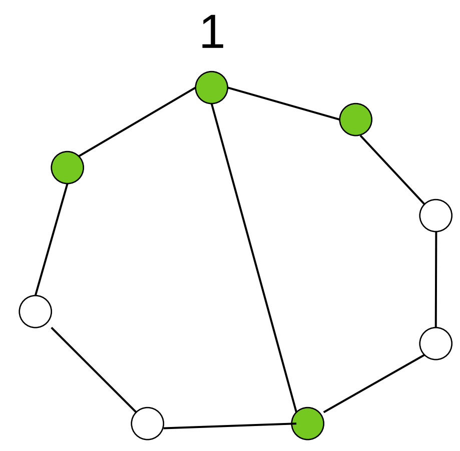
- P is not F.O. definable if for all r, there exists G_0 in P and G_1 in not P, Duplicator has a winning strategy

Games for Logic

Ehrenfeucht-Fraisse games:

- P: graph of degree 2
- G_0 and G_1 : Spoiler, Duplicator place pebbles alternatively
- r=4
- Spoiler wins (partial isomorphism of the substructures)



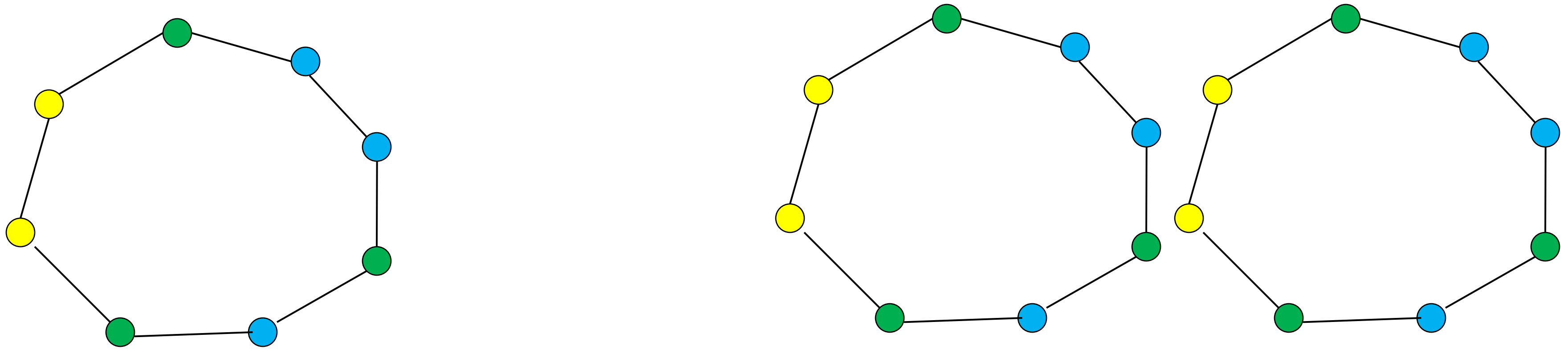


Games for Monadic 2nd order Logic

Monadic existential 2nd order games:

- Duplicator selects G_0 and G_1
- Spoiler colors G_0 with k colors (k=3), Duplicator colors G_1
- They play EF(r)

P: Connectivity



P: s-t Connexity

- Algorithms:
  - Easy on undirected graphs (Random walk starting with s: detects t with w.h.)
  - Hard on directed graphs

- Σ_1^1 definable on undirected graphs

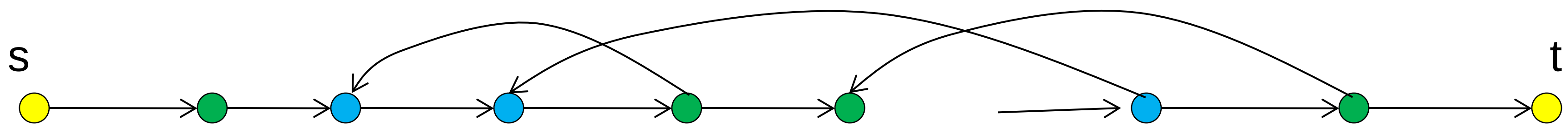
- Challenge: not Σ_1^1 definable on undirected graphs

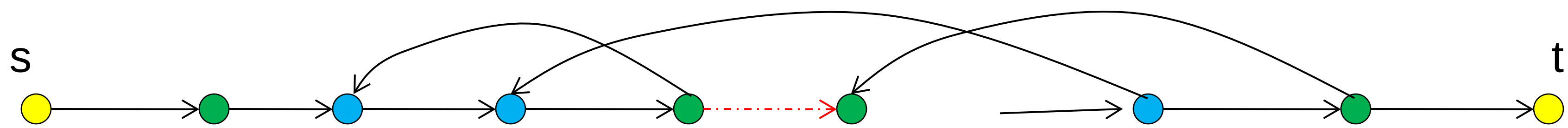
- Ajtai-Fagin games (1995):

Ajtai Fagin games for Monadic 2nd order Logic

- Duplicator selects G_0 and Spoiler colors G_0 with k colors
- Duplicator selects G_1 and colors G_1
- They play EF(r)

- Game easier to win for Duplicator
- Probabilistic method: back edges probabilistic





Probabilistic Method

- G_0 and G_1 are probabilistic graphs

$$Prob[Duplicator \quad wins\ ] > \delta > 0$$

- Conclusion 1: there exists G_0 and G_1 such that Duplicator wins the Ajtai-Fagin games.

- Conclusion 2: directed s-t Connexity not monadic

Σ_1^1 definable

- Important technique in TCS

Descriptive Complexity

- Complexity classes: L, NL, P, NP, coNP, ……#P
- Logics for each classes: NP = $\sum_1^1$
- Variations of the games for each complexity class
- Problem: worst-case complexity

Some NP complete problems are easy if the graphs follow some statistical hypothesis.

2 Strategic games

2 players, Utility matrices, mixed strategies σ, μ

Example: **stone, scissor, well** decisions for the 2 players

$$A=\begin{pmatrix} 0 & 1 & -1 \\ -1 & 0 & 1 \\ 1 & -1 & 0 \end{pmatrix} \qquad B=-A$$

$$Gain(I) = \sigma^t.A.\mu \qquad \mathrm{I} \quad Maximizes\ the\ Gain$$

$$Max_\sigma\ Min_\mu\ \ \sigma^t.A.\mu$$

Solution 1: Linear programs to find σ and μ (Equilibria)

Solution 2: Learn σ : $Best\ response\ to\ \ stat(II) \rightarrow Equilibria$

Fictitious player

Strategic games with N players

Utility tensors $A(x_1, \ldots. x_N)$

- Nash Equilibria (hard to compute)

- Algorithmic game theory
  - How do we learn « good strategies »

    (Fictitious player converges to Nash equibria on 0-1 games)
  - Non worst-case complexity
  - RL: Reinforcement learning
  - Given an equilibrium, what is the game?

Learning strategies for alternate games

Chess (joint work Luc Pommeret)

1. Learning phase: access 10^6 runs in format PGN (20 tokens)

   1. e4 e5 2. Nf3 Nc6 ...

2 Generation of the next token t_k according to $P(t_k \mid t_1 , t_2 ,\ldots.. t_{k-1})$ with a transformer

3. LLMs for Games

Key components of LLMs

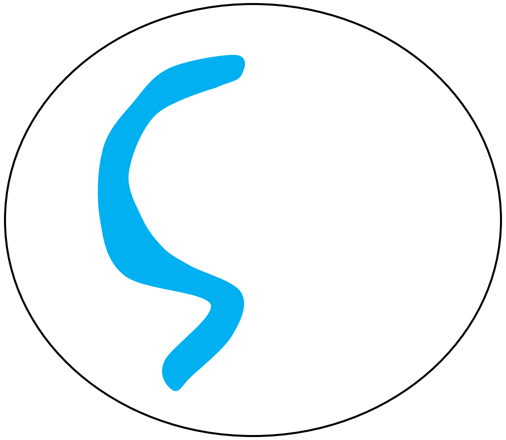
1. Tokens

2. Transformers

3. RLHF and Chains of thought

Transformers

1. Representation of a distribution on text decomposed by tokens



2. Generation of t_k according to $P(t_k \mid t_1, t_2, \ldots.. t_{k-1})$

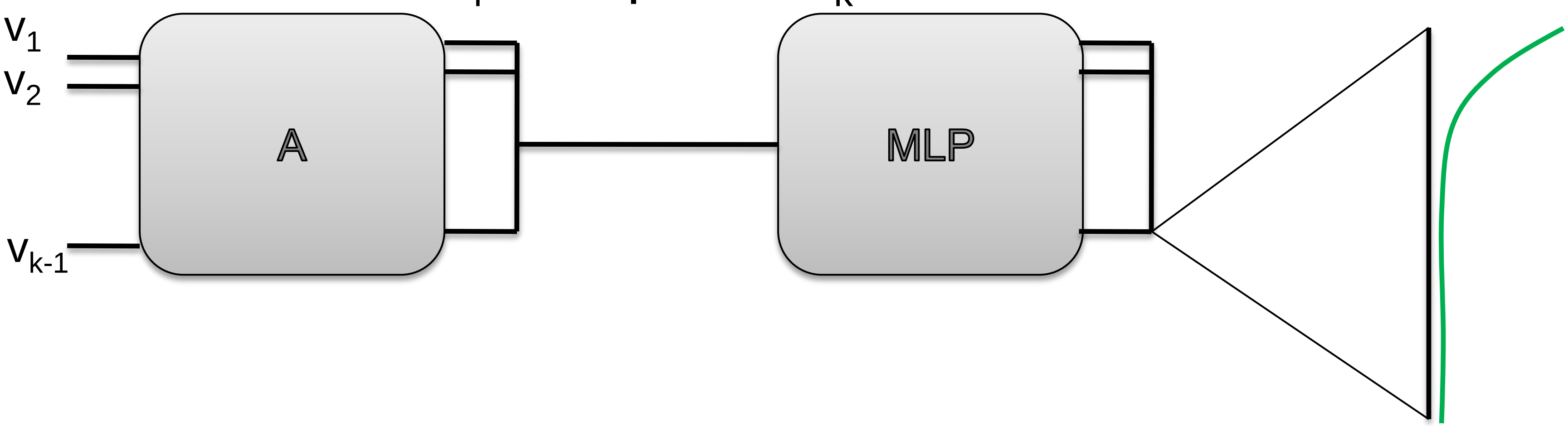
3. Attention + Multilayer Perceptron

Transformers

Each token t_i has an embedding $v_i \in R^d$ (d=768)

Attention Block A, Perceptron Block MLP

Transform the v_i and predict t_k



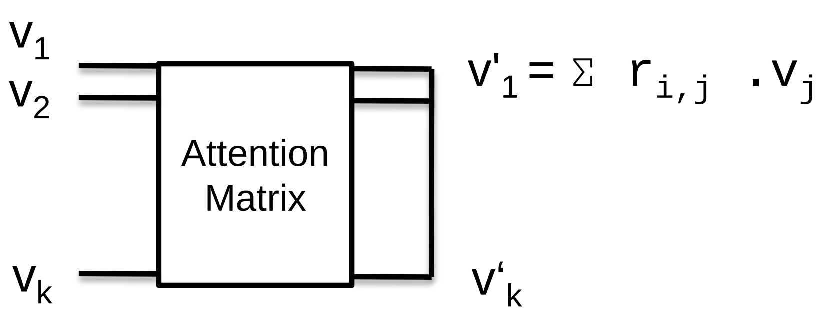
Transformers: input t_1 , t_2 ,….. t_{k-1}

output: P (t_k | t_1 , t_2 ,….. t_{k-1})

Attention: Q,K,V (d,d) matrices

$$v'_i = \sum_{j=1}^{k-1} r_{i,j}.V.v_j$$

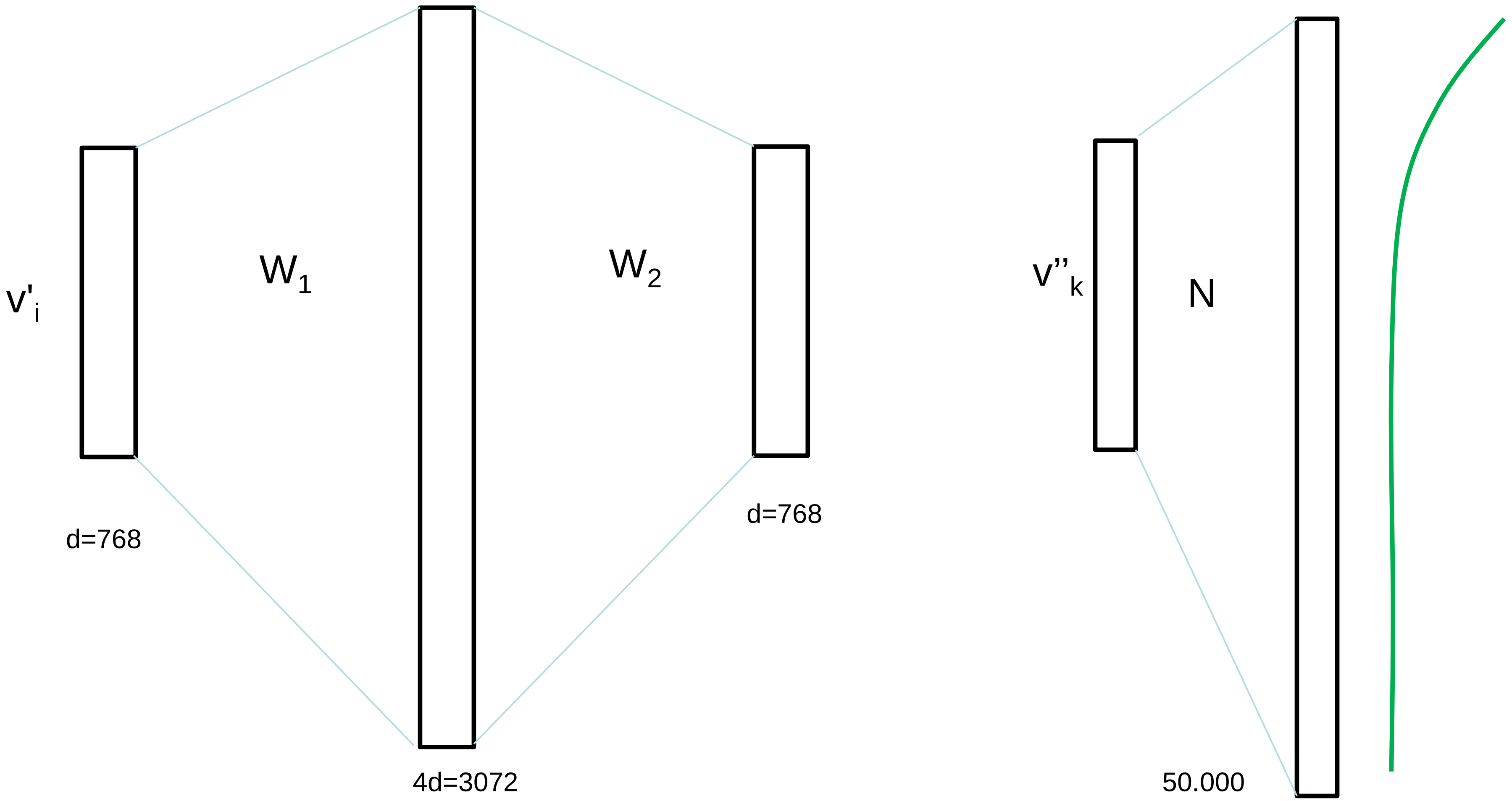
$$r_{i,j} = Softmax((K.v_i)^t, Q.v_j)$$



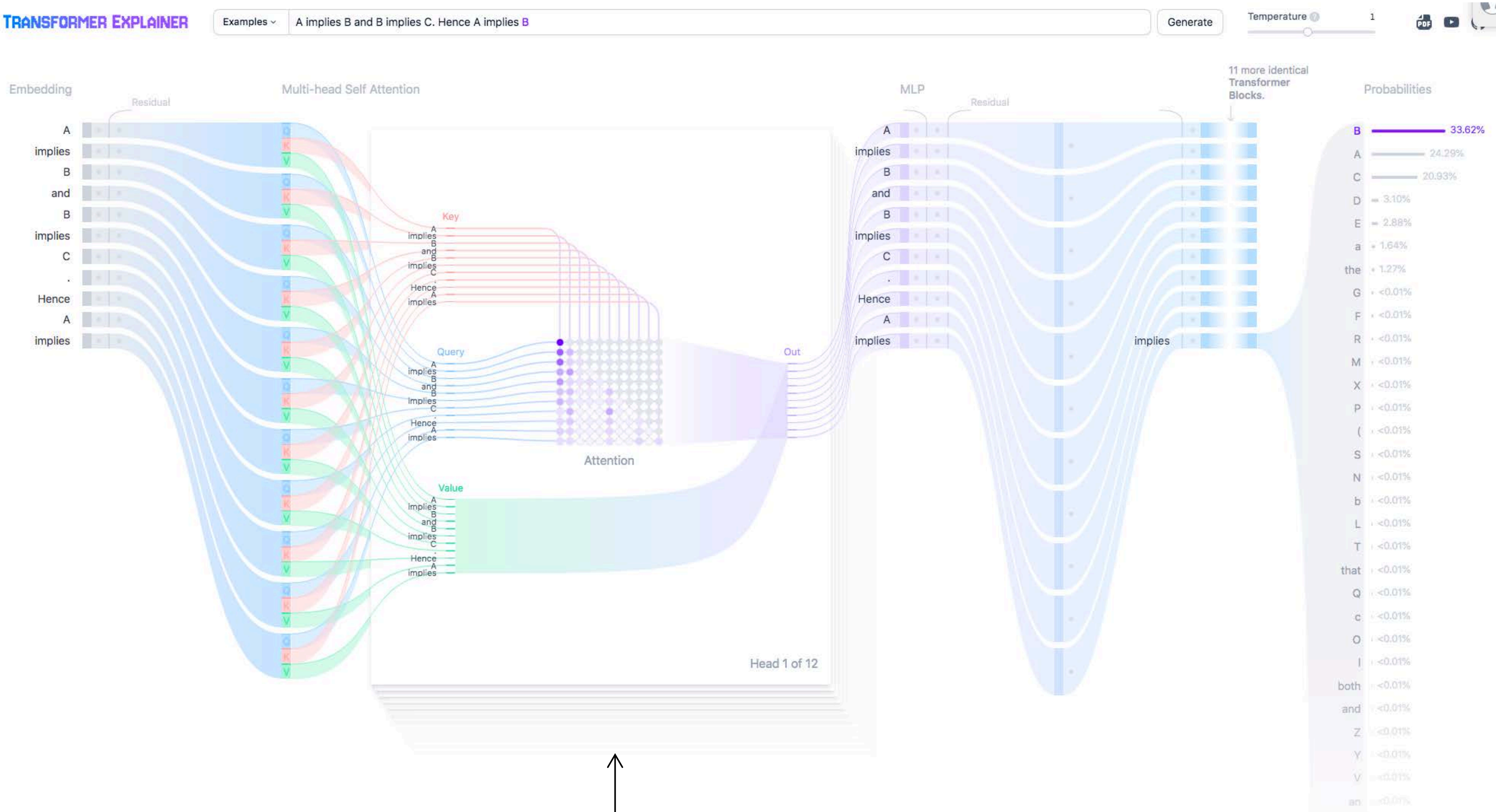
MLP

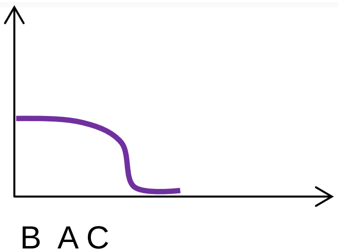
https://poloclub.github.io/transformer-explainer/

Perceptron Block MLB: W_1,W_2,N



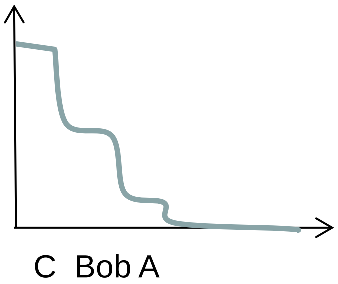
A implies B and B implies C. Hence A implies



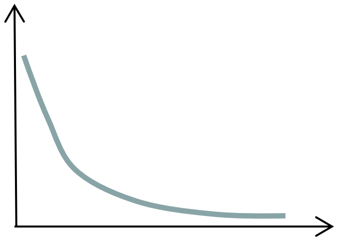


Deduction and Composition

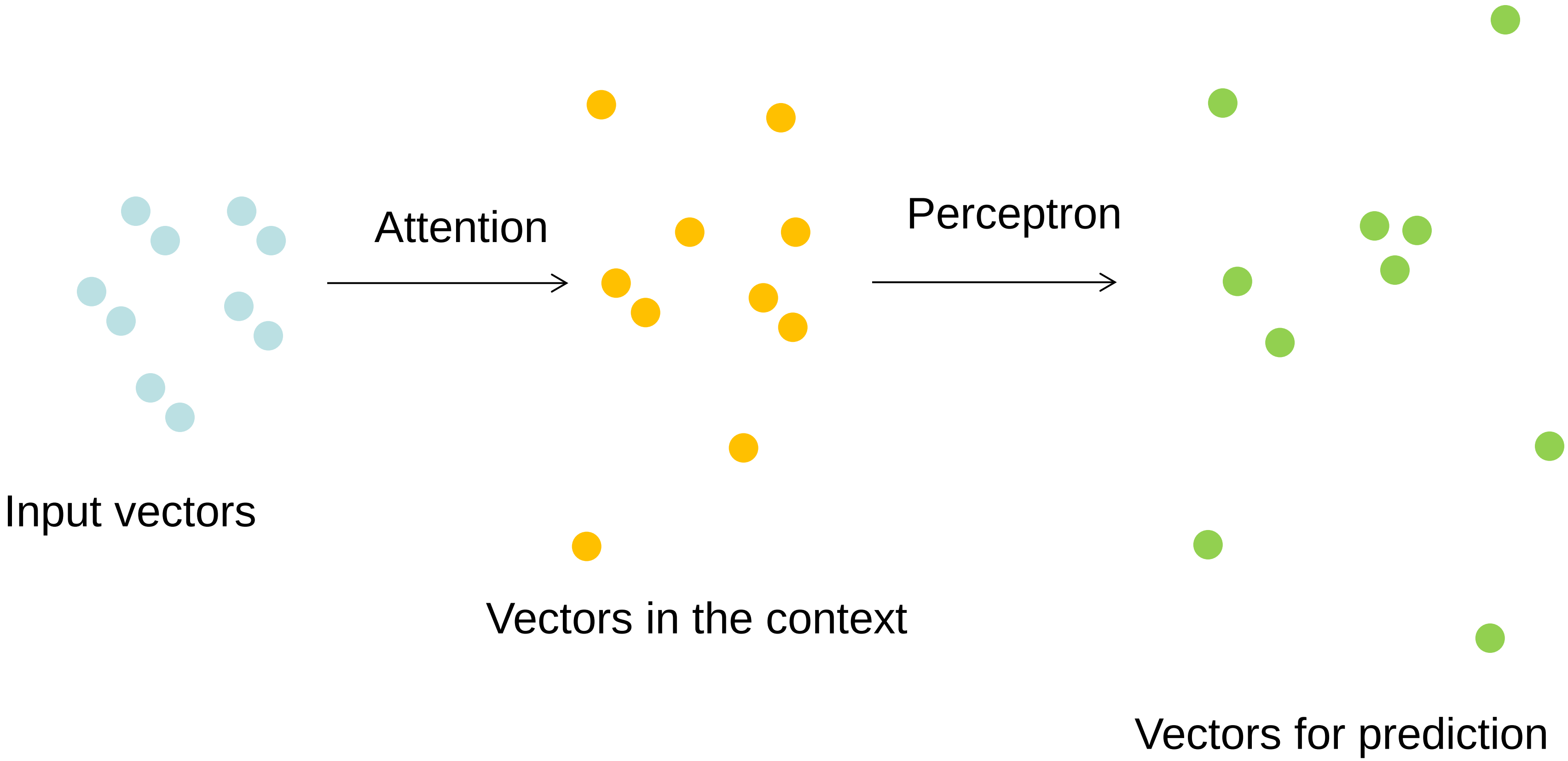
A implies Bob and Bob implies C. Hence A implies ?



f(1)=2, f(2)=3,g(1)=2, hence f(g(1))=?



Dynamics of the embeddings



Transport of a distribution

Composition by transformers: negative results

f(0)=3, f(1)=5, f(2)=7,…..

g(0)=7, g(1)=2, g(2)=3,….. What is f(g(1))=?

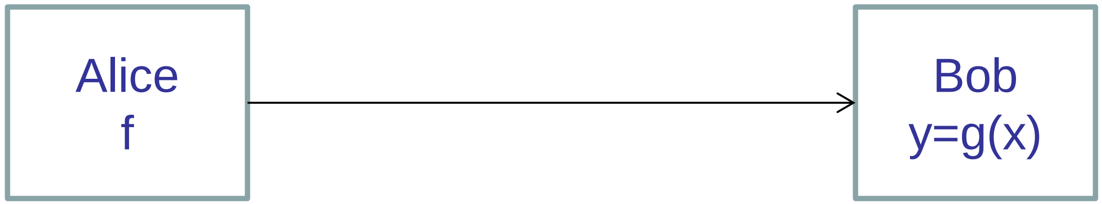
On the limitations of the Transformer Architecture,

Peng, Narayanan, Papadimitriou, Arxiv 2024

Composition by a Transformer is incorrect with high probability!!

Worst-case negative result.

Communication Complexity



One way complexity: Π is the length of Alice's message

Index problem: Bob must compute f(y)

Alice must send n.log n bits

Worst-case complexity, i.e. y is uniform on {1…n}

Theorem (PNP 2024): If Alice sends (n.log n – R) bits than the Composition is incorrect with probability R/ n.log n !!

Information Complexity

Alice
f

Bob
y=g(x)

Theorem: If Alice sends (n.log n – R) bits, than the Composition is incorrect with probability R/ n.log n !!

I(X; Y)= KL (P(X,Y) || P(X)*P(Y))
= H(X)+H(Y) – H(X,Y)= H(X) – H(X|Y)

I(Π ; f(i*) | i*) <Π/n= log n – R/n

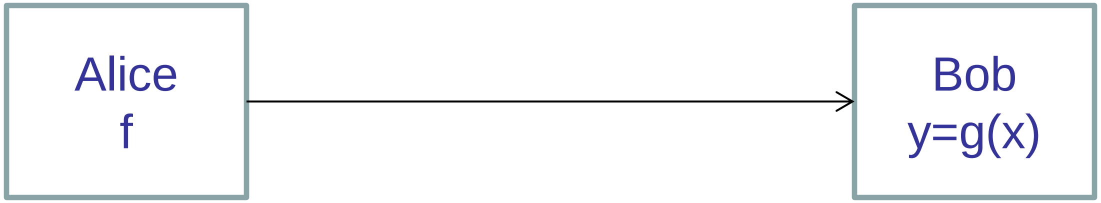
Fano Inequality: Assume we estimate X from Y, i.e. X'=F(Y)

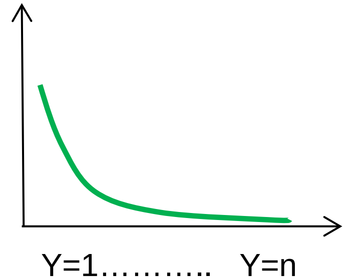
Error: $\mathrm{Prob}[\mathrm{X}' \neq X] = \delta$

Fano: H(δ)+ δ. $\mathrm{log\ n} > \mathrm{H(X\ |Y)}$

Conclusion: δ>R/3n.log n

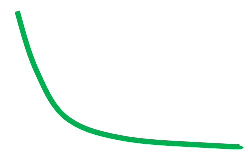
Communication Complexity: non-worst case





Assume y is non uniform and the distribution is public

Alice sends $f(i_1)$….$f(i_p)$, most likely y to Bob

Non worst-case complexity, i.e. y follows  on {1…n}

If Alice sends O(1) bits, the Composition is correct with high probability !!

Game strategy: Composition of moves

Chess: R(x,y) if $x \rightarrow y$ with a valid move

Goal: iterate R: $x \rightarrow y_1 \rightarrow y_2 \ldots. \rightarrow y_n$

Observation: R composes with high probability

The Transformer defines a distribution

$P(t_k \mid t_1, t_2, \ldots.. t_{k-1})$ $k=20$, $n=10^4$, size 10^{80}

Compressed to Q,K, V and W_1, W_2, N size c.10^6

Which classes of functions/relations compose?

Chain of Thought

A implies B and B implies C. Hence A implies?

ChatGPT 4.0 answers:

If A implies B, and B implies C, then logically A implies C.

This reasoning is based on the ***transitive property of implication*** *in propositional logic. Symbolically:*

$A \implies B \text{ and } B \implies C \;\Rightarrow\; A \implies C$

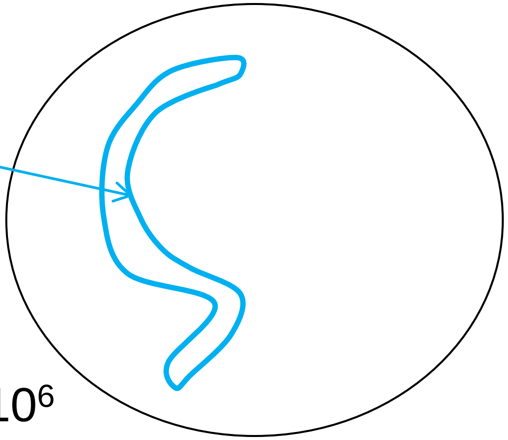
This means if A is true, C must also be true.

In general: chain of thought decomposes the answer.

Learn a distribution

Real Data are NOT worst-case

Distribution of Data $D(t_1, t_2, \ldots.. t_k)$



- Transformer compresses D

$D(t_k \mid t_1, t_2, \ldots.. t_{k-1})$ $k=20$, $n=10^4$, size 10^{80}

Compressed to Q,K, V and W_1, W_2, N size c.10^6

Logical interpretation

– 01- extension of ChatGPT, Chains of thought

– Deepmind mathematical engine

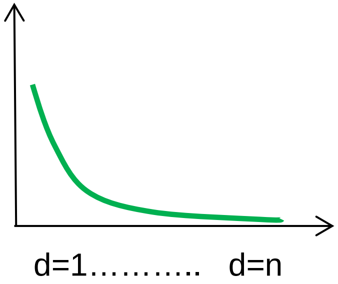
Algorithms for a property P on a Distribution D

A: Algorithm on D (Claire Mathieu, M dR : **Large very dense subgraphs in a stream of edges,** Network Science**, 2022)**

Ω is the probabilistic space of the algorithm

$$x \in P \;\rightarrow\; Prob_{\Omega}\,[A\ accepts] > 1-\delta$$
$$x\ not\ in\ P \;\rightarrow\; Prob_{D\,.\,\Omega}\,[A\ rejects] > 1-\delta$$

Example: Graphs with a power law Degree distribution.



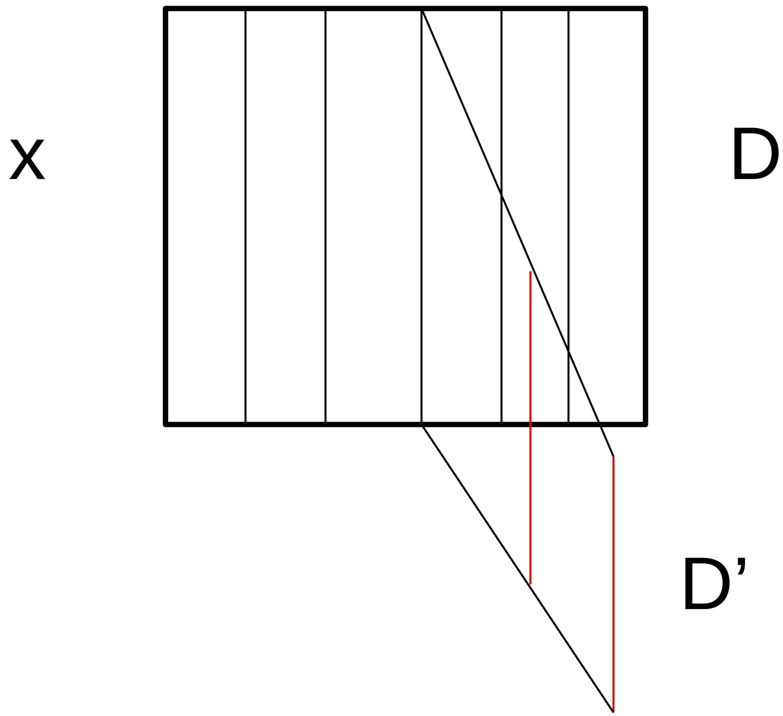
P: Maxclique (NP-hard) is easy.

Emergent skills (by probing), also defined for D

Probing (Hewitt , Manning 2028)

Can a Neural network for D can also predict D’

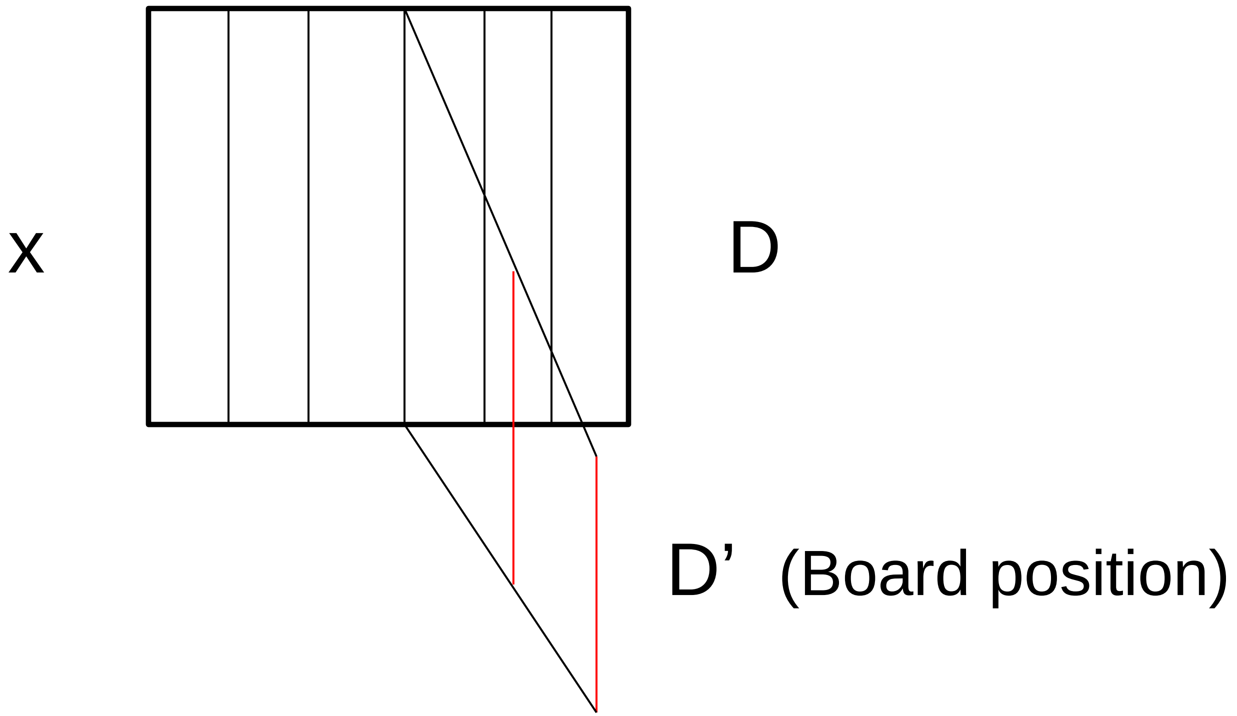
Can syntax can be inferred from a Transformer ?



Probing in Chess

Transformer learnt from the runs: can we probe

- The board position?
- The ratio White/Black figures ?
- The ELO number

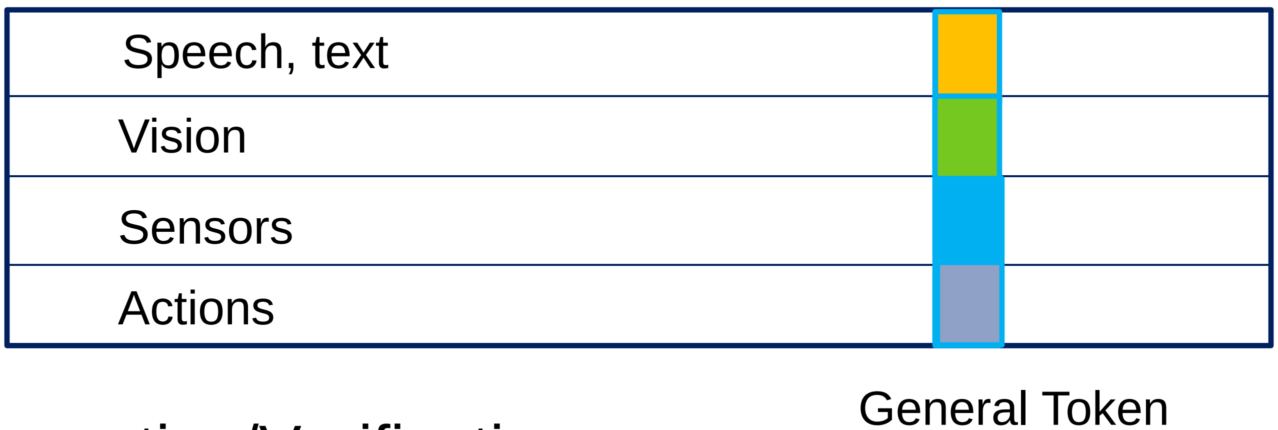


Extensions of LLMs

1. Speech acts: Question → Answer

   Question → Comment, Question

2. Multimodal data (Large Action Models in Robotics)



3. Generation/Verification
   – Formal verifier (Lean, Coq)
   – Probabilistic verifier

Conclusion

1. Descriptive Complexity = Worst case Complexity

2. Data as a distribution

   - Transfomers represent a distribution
   - Worst-case composition impossible
   - Relative composition possible

3. Logical interpretation of LLM's

References

1. Miklos Ajtai, Ronald Fagin, **Reachability is harder for directed than for undirected finite graphs**. Journal of Symbolic Logic, 55, pp. 113–150, 1992**.**
https://research.ibm.com/publications/reachability-is-harder-for-directed-than-for-undirected-finite-graphs

2. Julia Robinson, **An iterative method of solving a game**, Annals of Mathematics, 1951

   Fictitious play, converges slowly for 0-sum games

3. John Hewitt and Christopher D. Manning, **A Structural Probe for Finding Syntax in Word Representations,** 2018

   Relative computation with Neural Networks

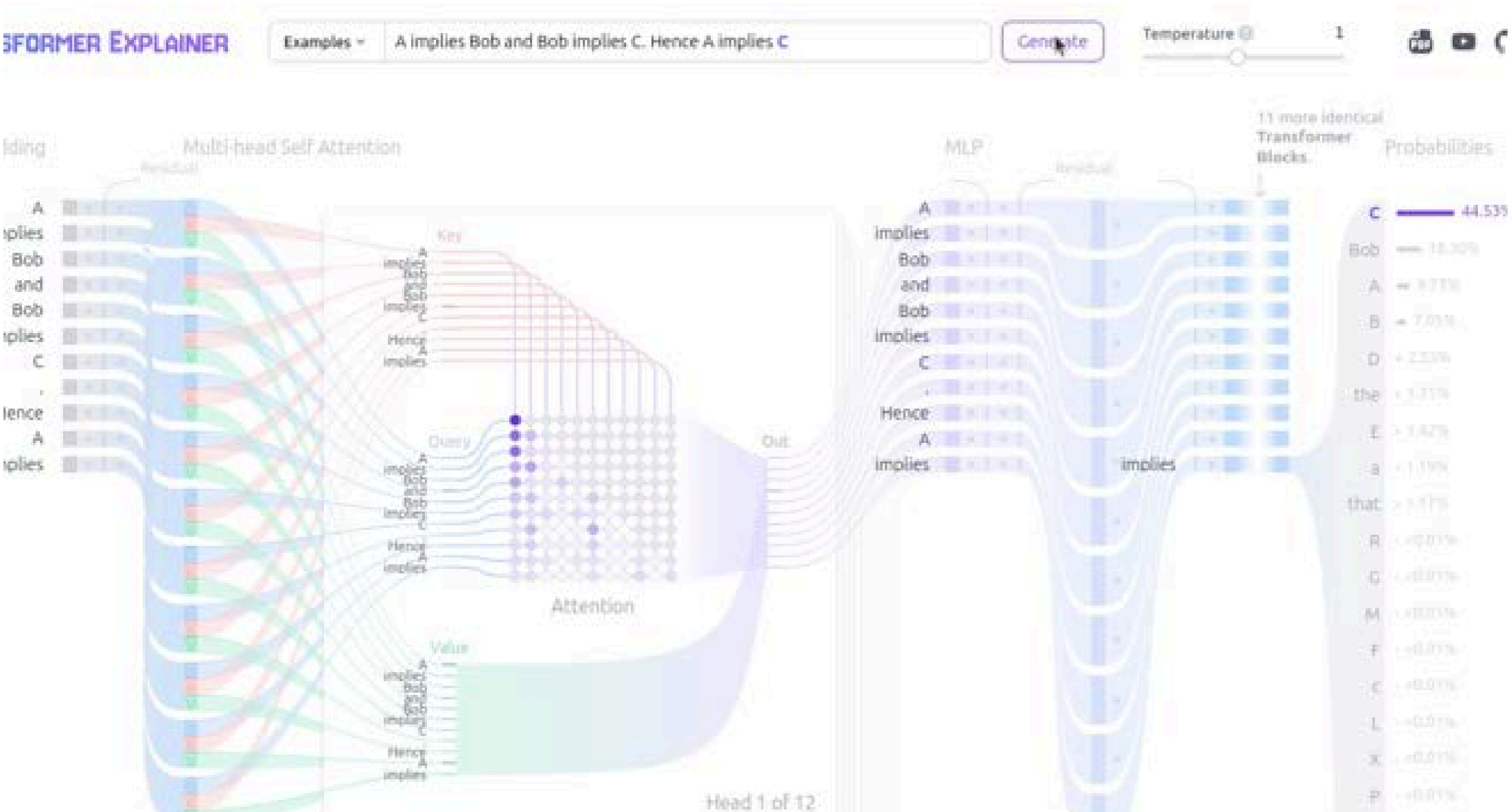
4. Claire Mathieu and Michel de Rougemont : **Large very dense subgraphs in a stream of edges,** Network Science**,** Cambridge University Press, **2022**

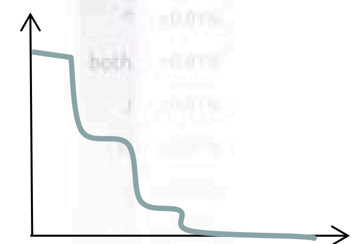
   Non worst-case algorithms on graphs

   https://hal.science/hal-03112040/document

5. Transformers, ChatGPT, 01: Sparks & Embers, Debate Jan. 2025
https://simons.berkeley.edu/news/sparks-vs-embers

A implies Bob and Bob implies C. Hence A implies ?





C Bob A

f(1)=2, f(2)=3,g(1)=2, hence f(g(1))=?