PRINCETON UNIVERSITY

Dynamic Graphs

SSMC ’23 | University of Kentucky

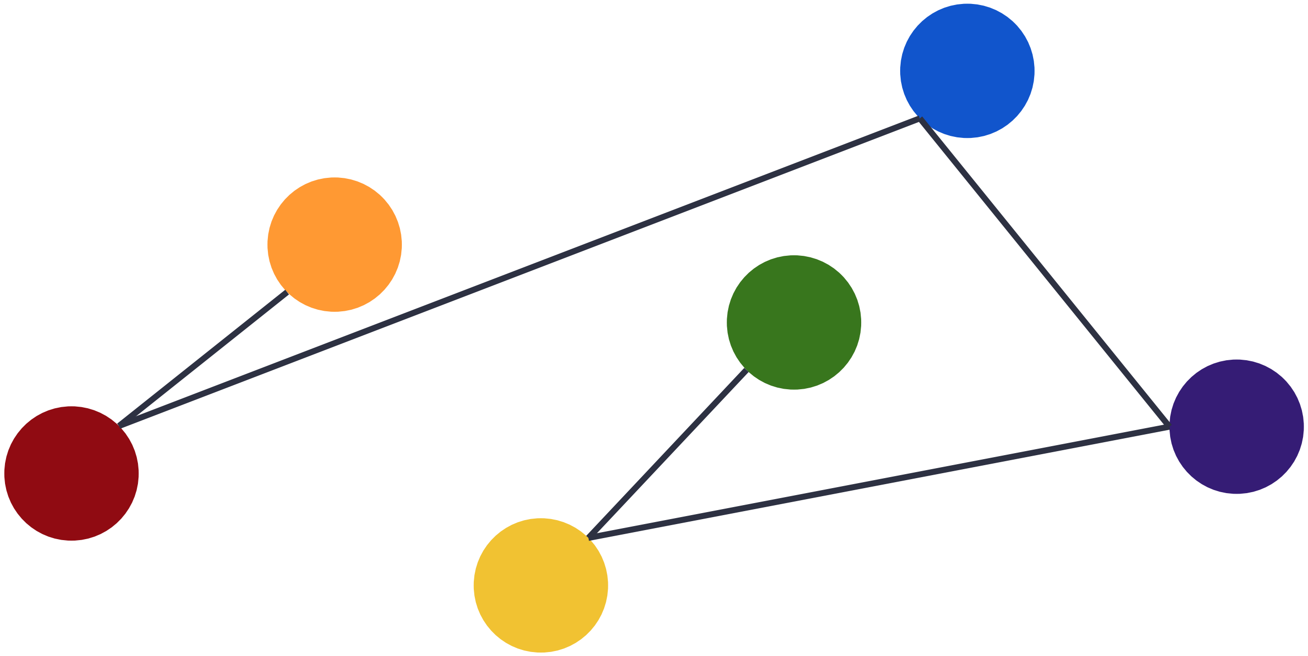
May 20, 2023 @ 3:15 PM

Dev Dabke

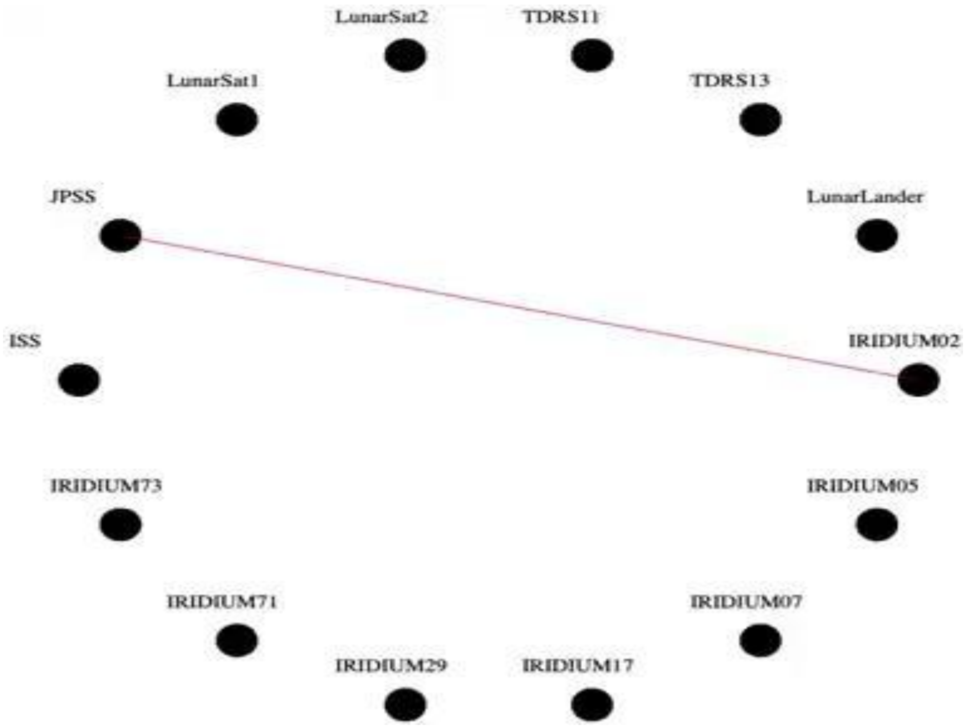
Overview

1. Brief mathematical introduction
2. Some observations and conjectures
3. An ML thing (if we have time)

🧮 Math Intro 🔢



A simple graph



A dynamic graph

Definition: Dynamic Graph

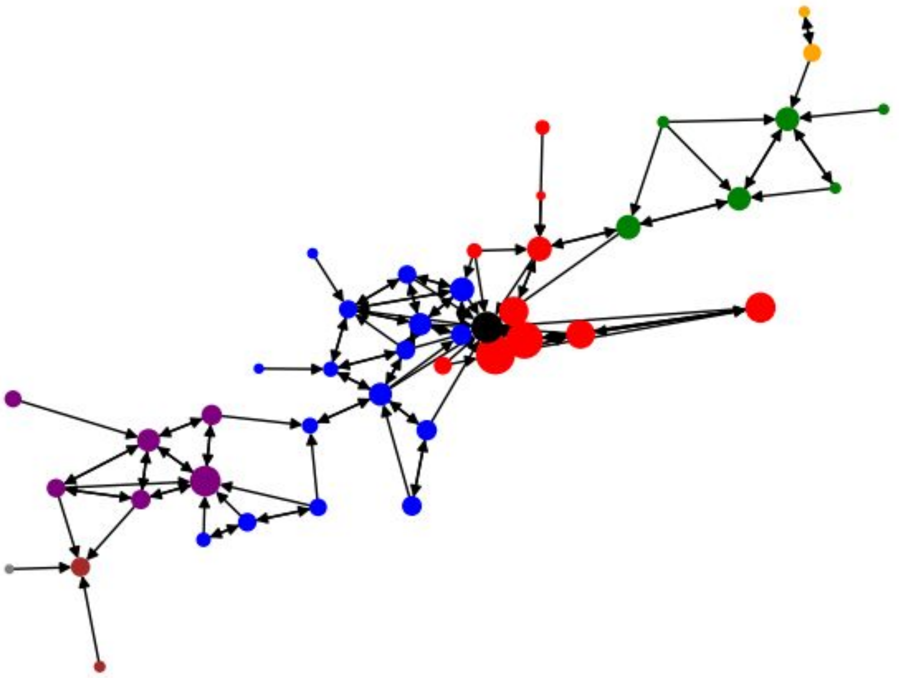
Dynamic graph $\mathcal{G} = (G_t)_{t \in \mathbb{T}}$ where

1. totally ordered indexing set $\mathbb{T}$
2. fixed finite vertex set V
3. edge set sequence $(E_t \subseteq V \times V)_{t \in \mathbb{T}}$
4. $G_t = (V, E_t)$

Dynamic Graphs

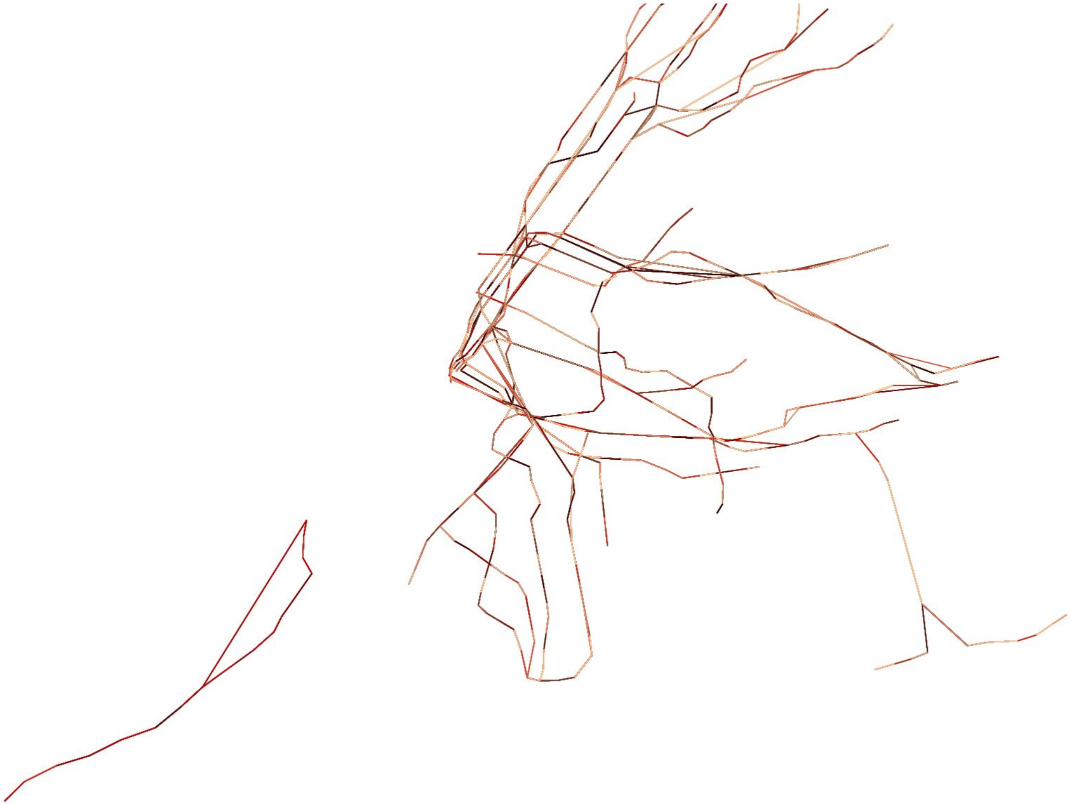
are **Everywhere**

- Commute networks
- Animal interaction networks
- Opinion dynamics
- Cell-cell signaling
- Social networks
- Satellite communication networks
- Basketball, sports
- Bird flocking



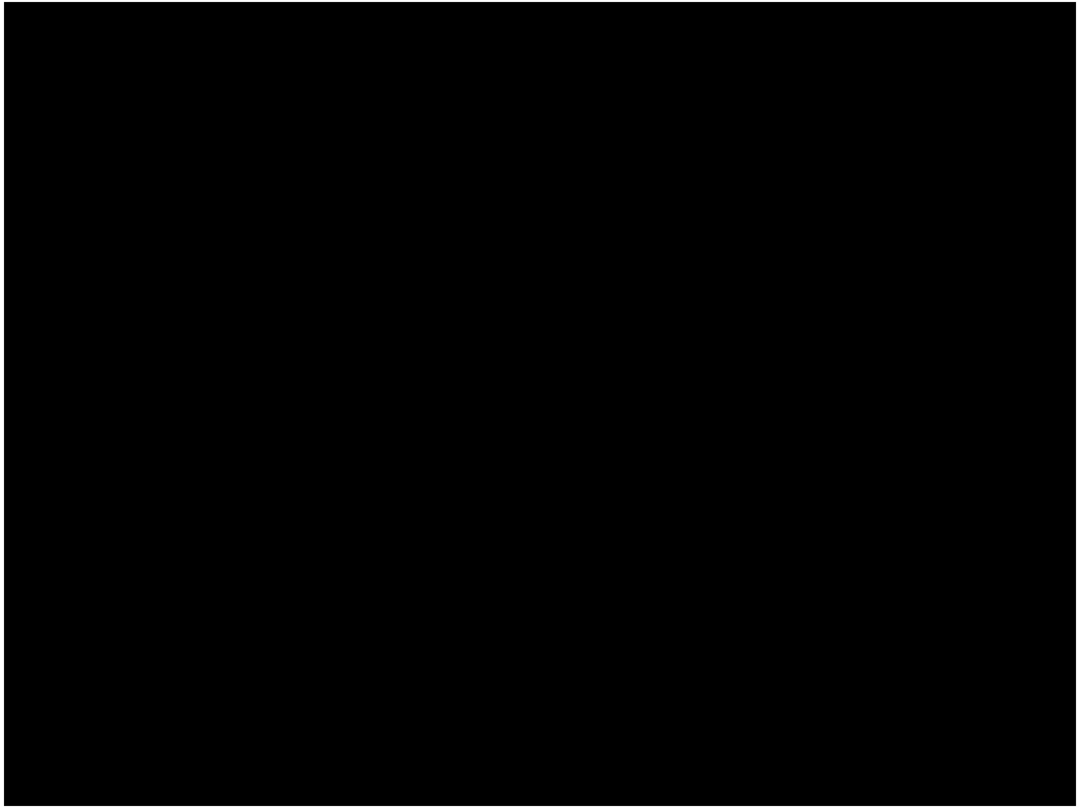
NYC Metro Area Commute Network

Dabke, Karntikoon, Aluru, Singh, Chazelle. *Network-augmented compartmental models to track asymp. disease spread*. (pre-print)



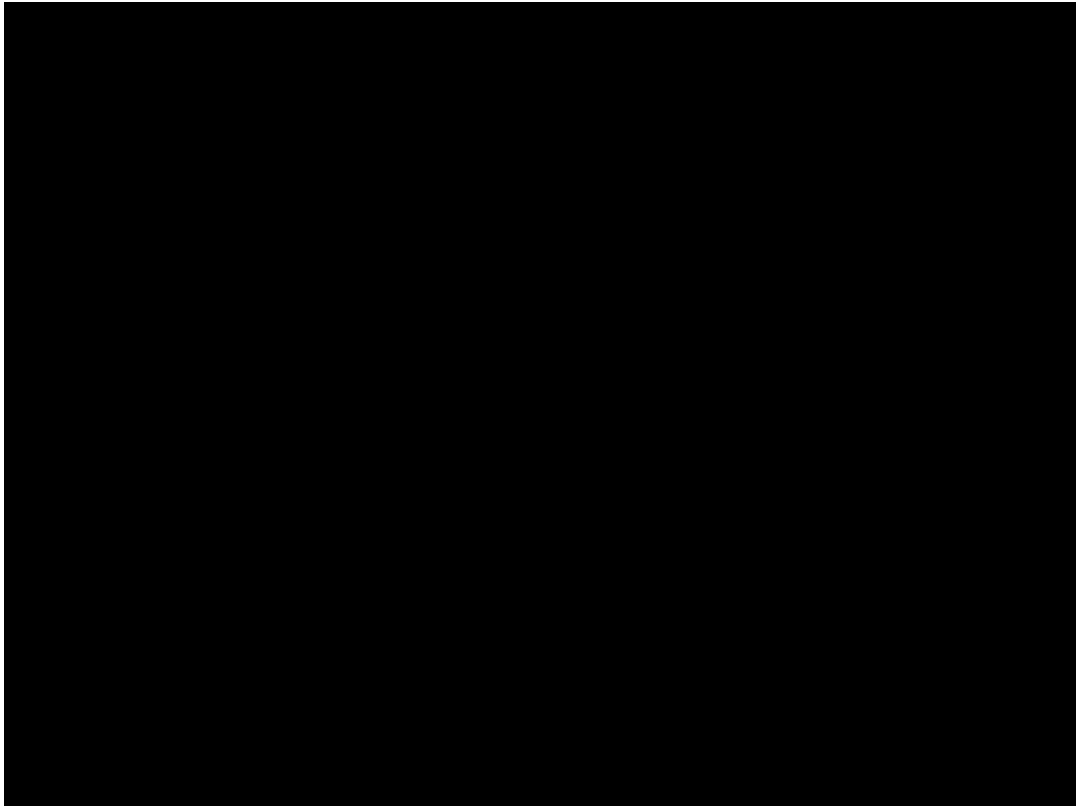
Transit Networks (GTFS)

Dabke, Green. *Analyzing transit networks with ideal routing machines*. (pre-print)

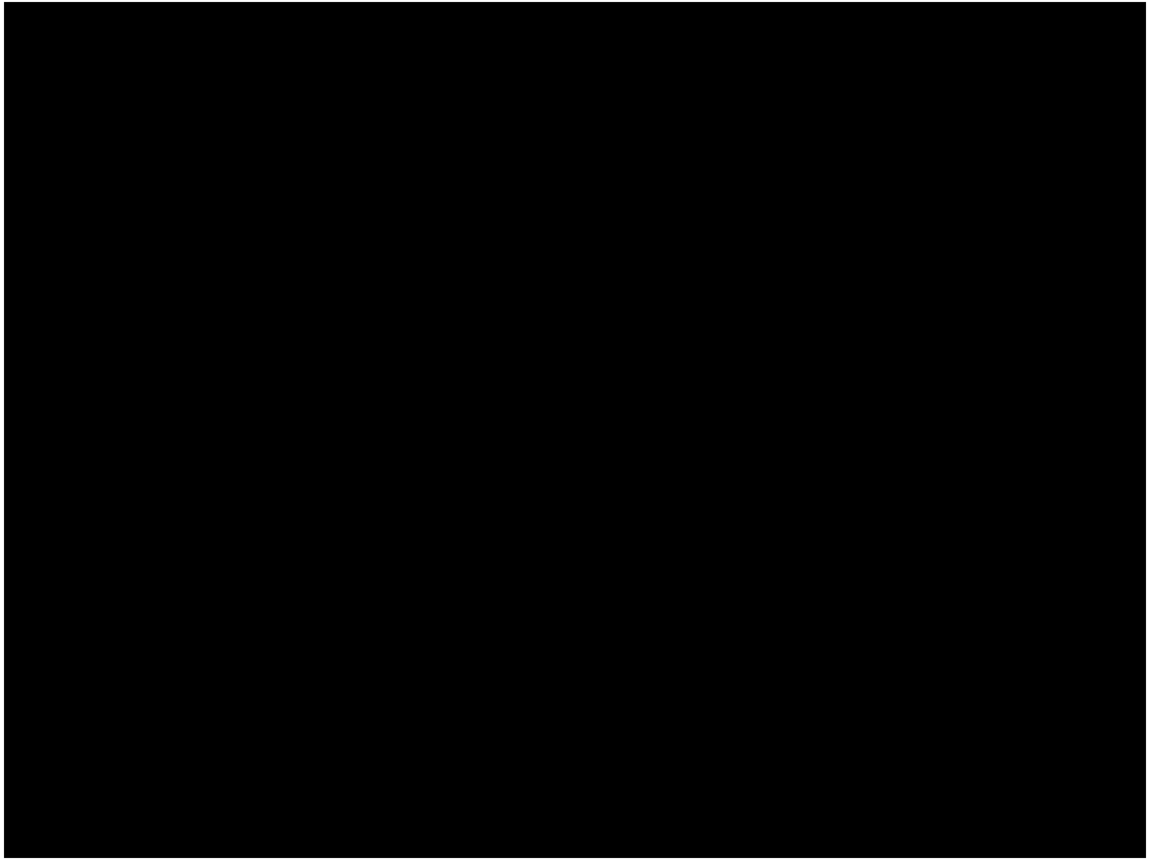


Animal herding networks

Dorabiala, Dabke, et al. *Spatiotemporal* k-*means*. (pre-print)

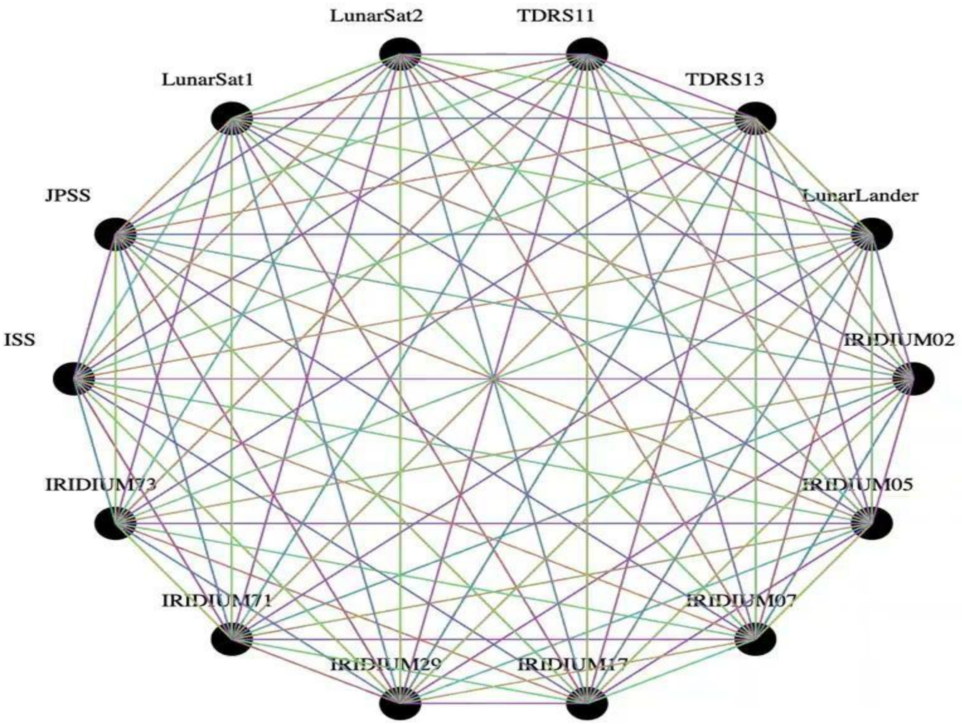


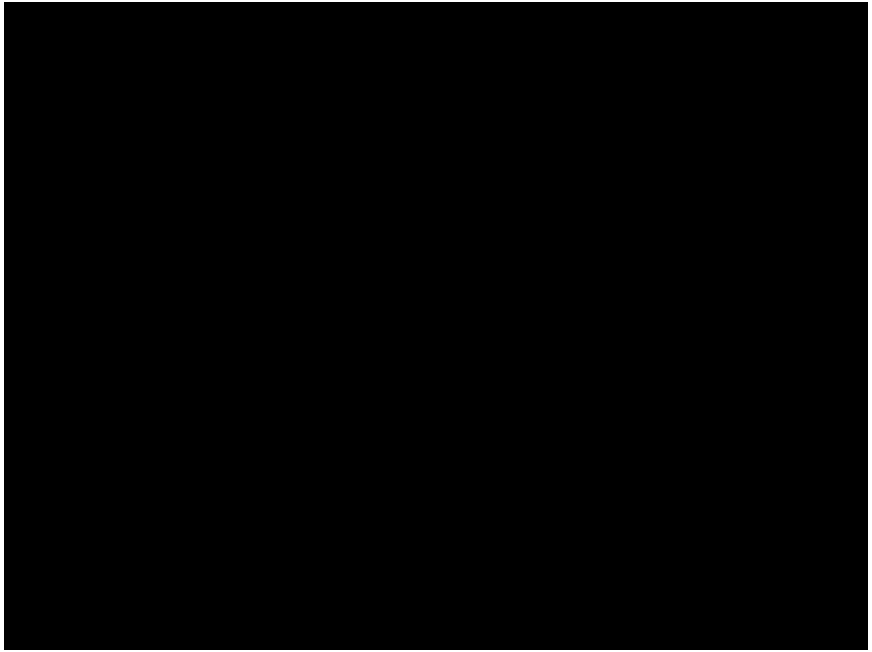
Embryos: spatial and chemical connectivity



Basketball

Dabke, Chazelle. *Extracting semantic information from dynamic graphs of geometric data*.
Dabke, Taylor. *Play classification in basketball networks*. (internal publication; pre-print)





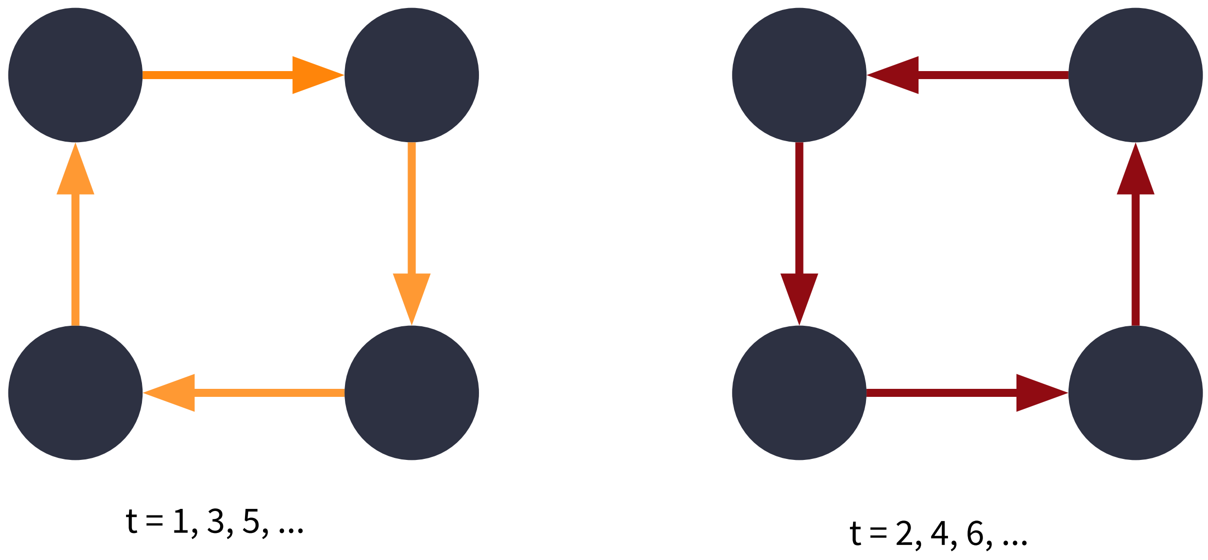
NASA: Satellites

Cleveland, Dabke, et al. *Introducing tropical geometric approaches to delay tolerant networking optimization.*
Hylton, Dabke, et al. *A survey of mathematical structures for lunar networks.*

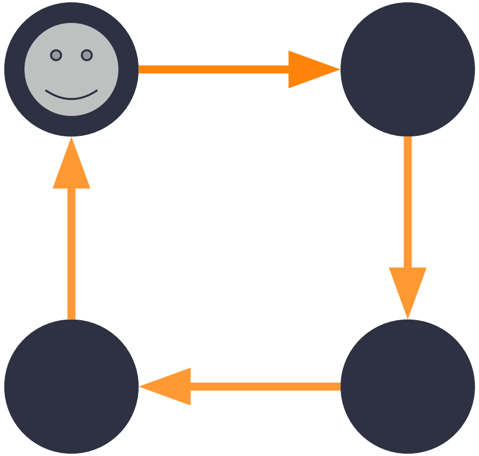
Some Problems

Problem

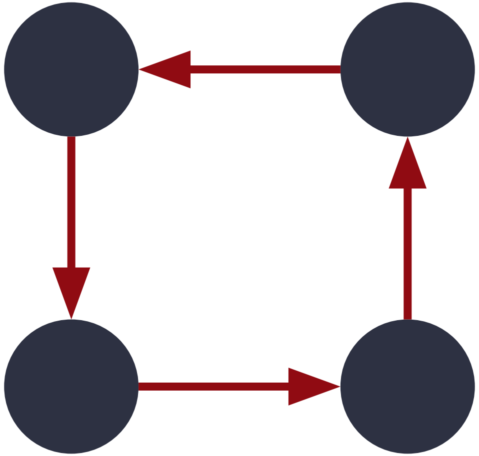
Local $\neq$ Global



The alternating cycle: a discrete-time sequence

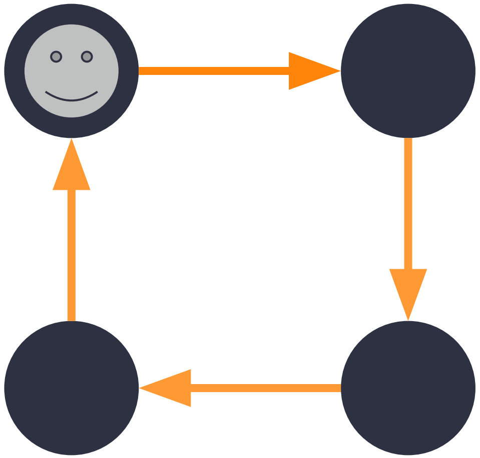


t = 1, 3, 5, ...

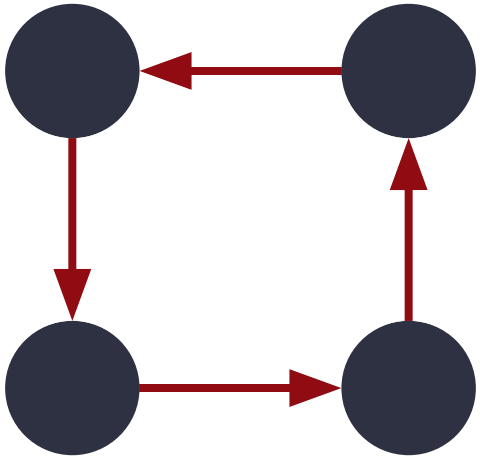


t = 2, 4, 6, ...

Dynamical system: move one edge at each time step

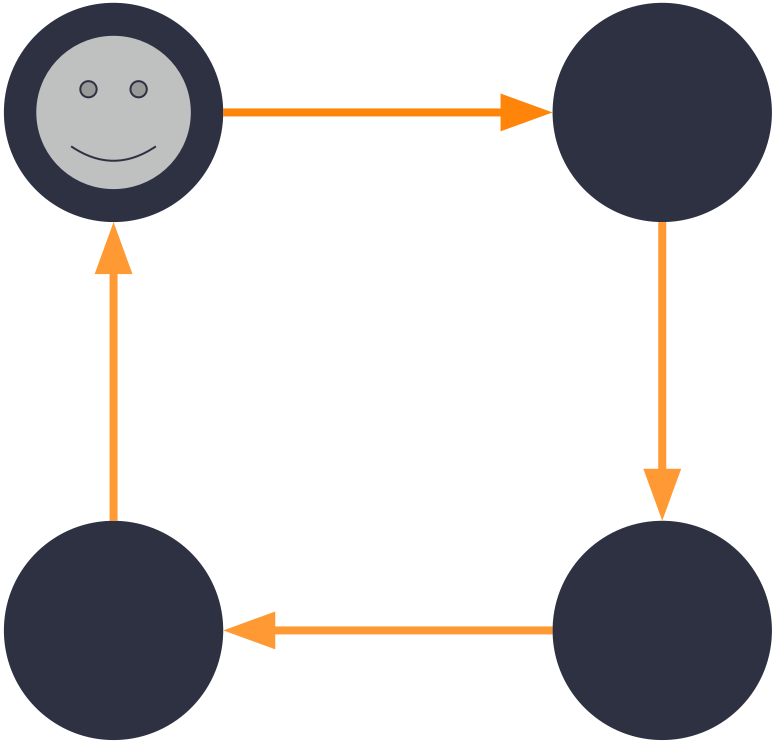


t = 1, 3, 5, ...

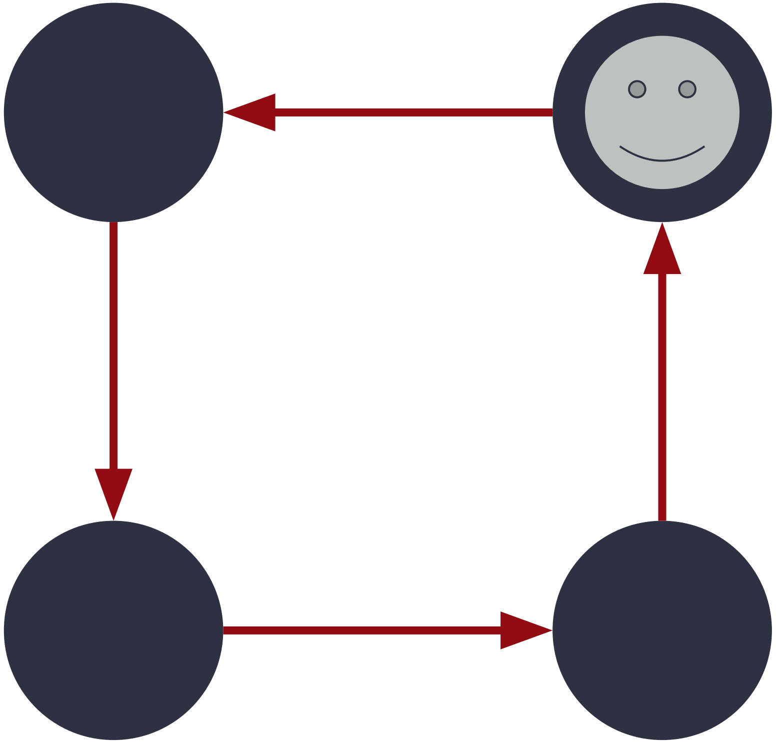


t = 2, 4, 6, ...

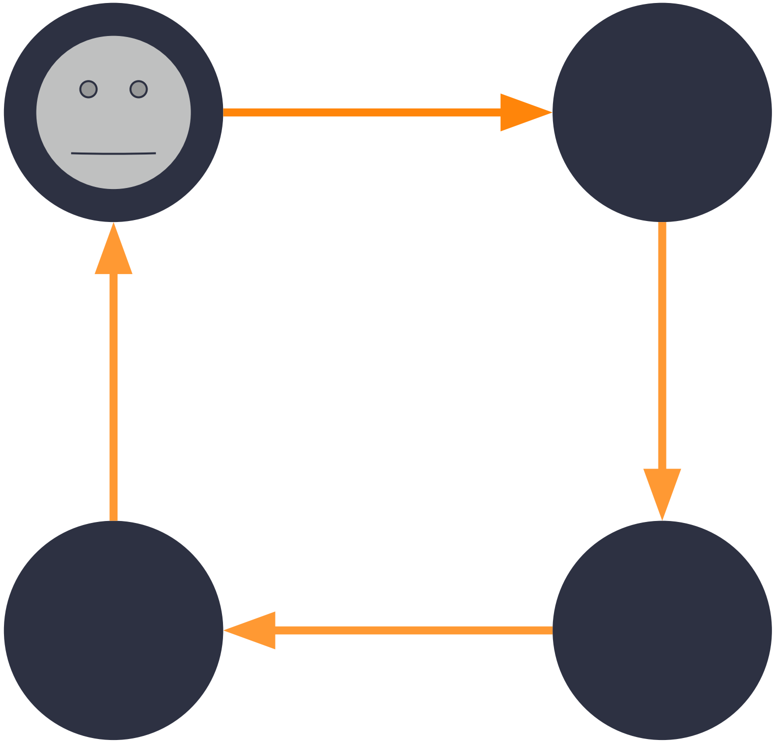
Fact: max diameter is number of vertices (if connected)



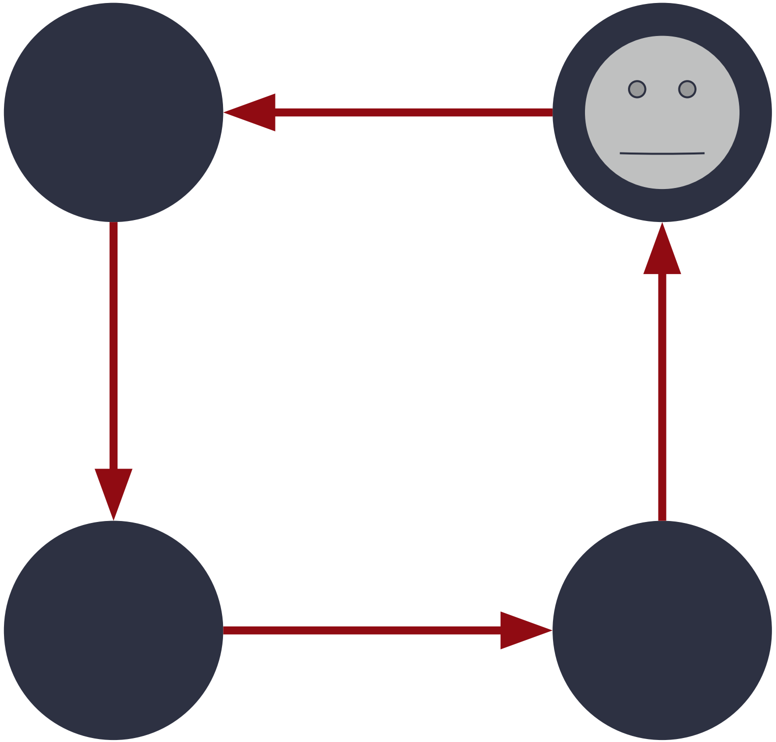
t = 1



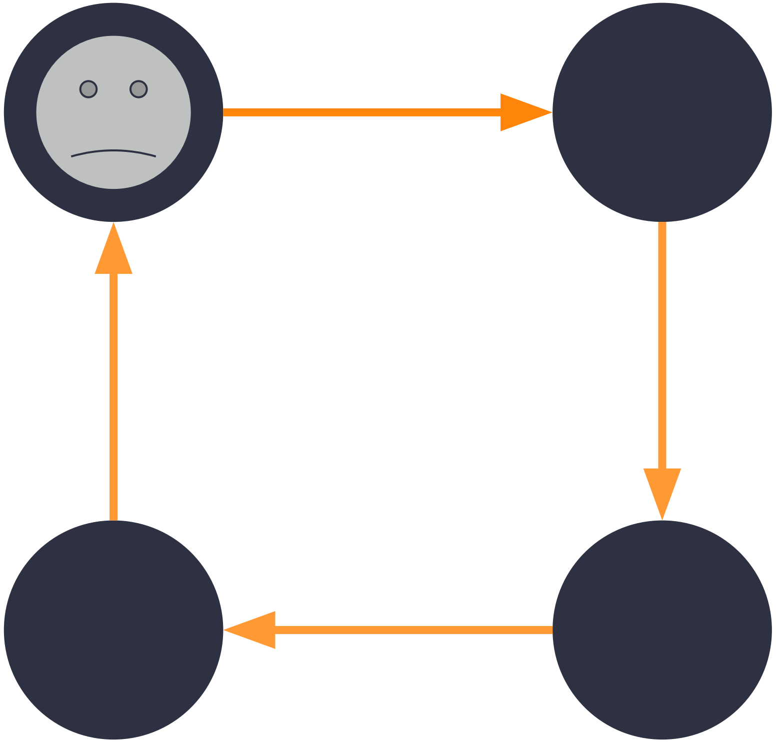
t = 2



t = 3

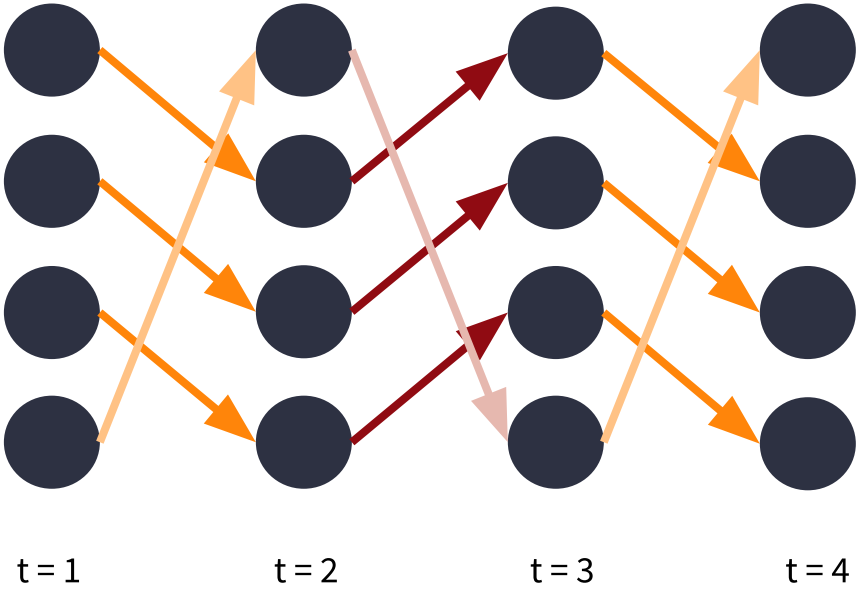


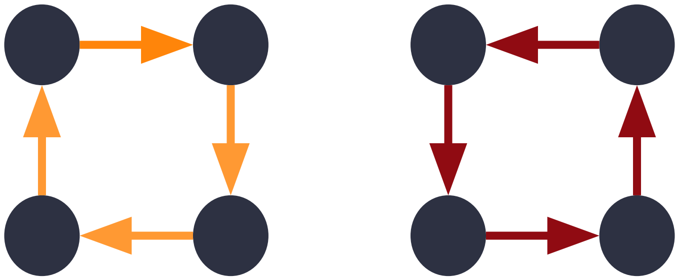
t = 4



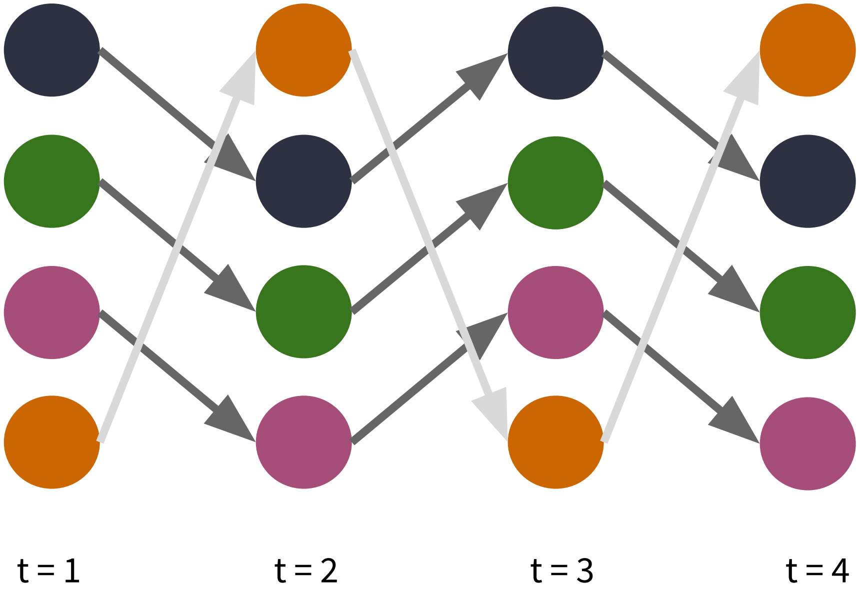
t = 5

Dynamically disconnected

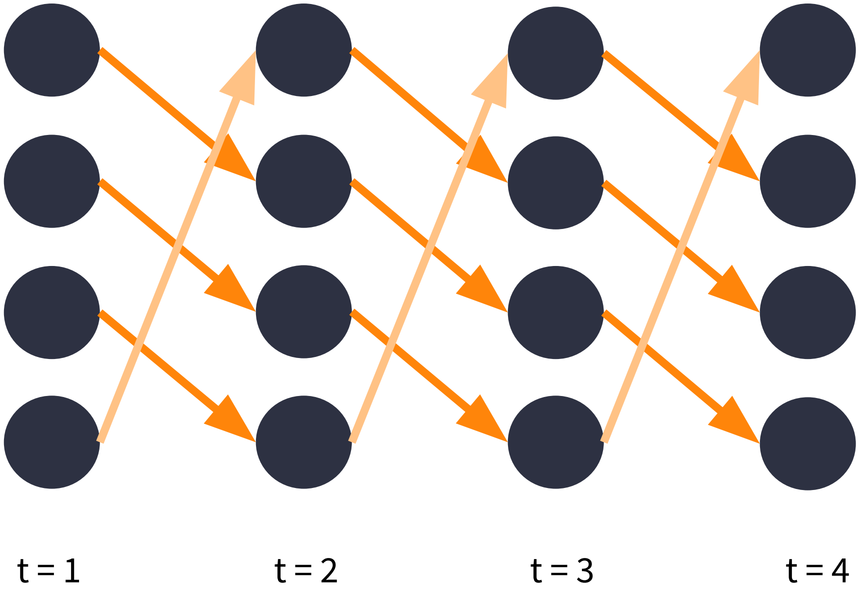




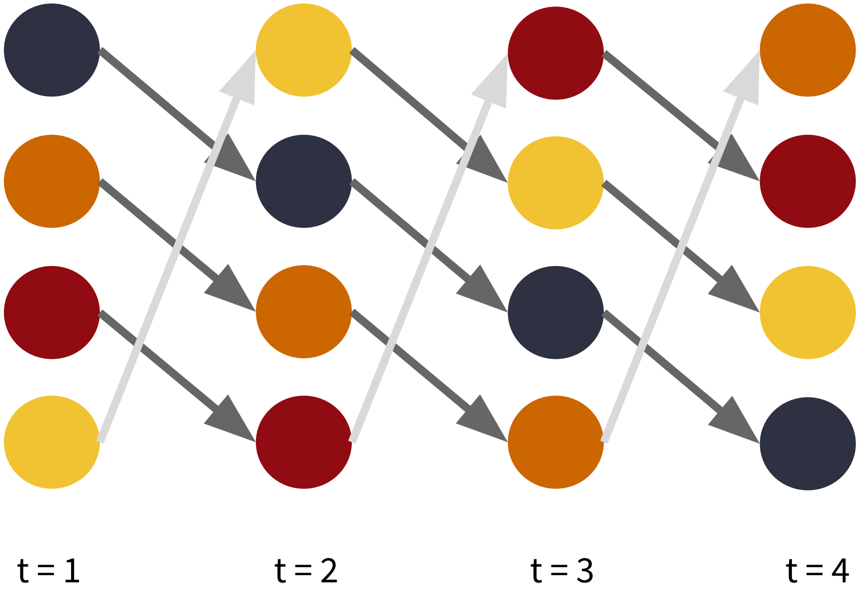
Idea: time-expanded graph



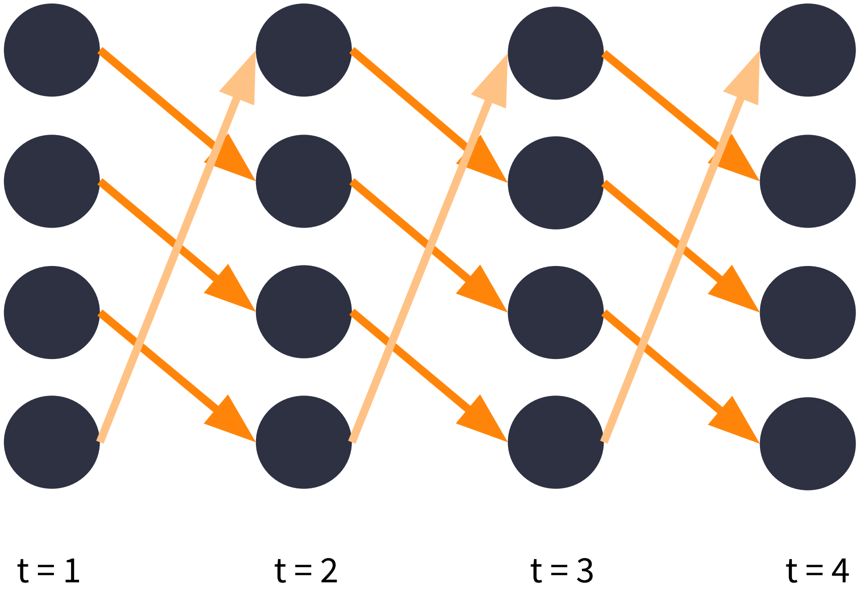
Observation: disconnected => dynamically disconnected?



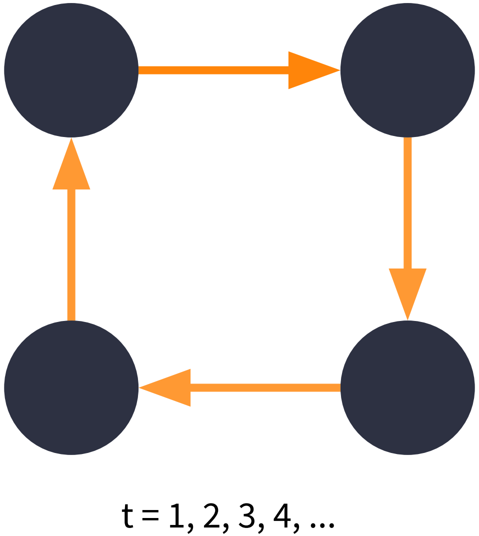
This one?

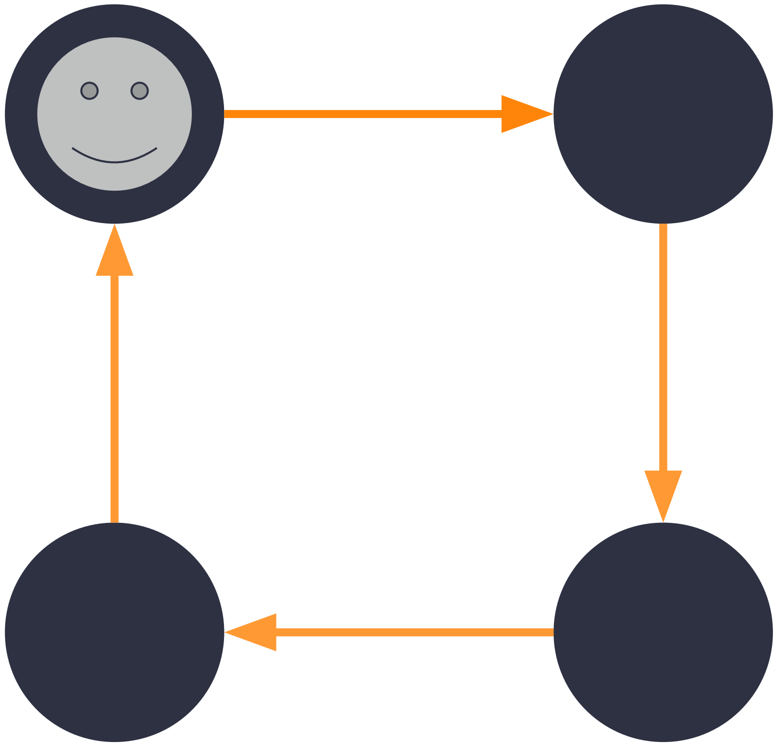


Disconnected again

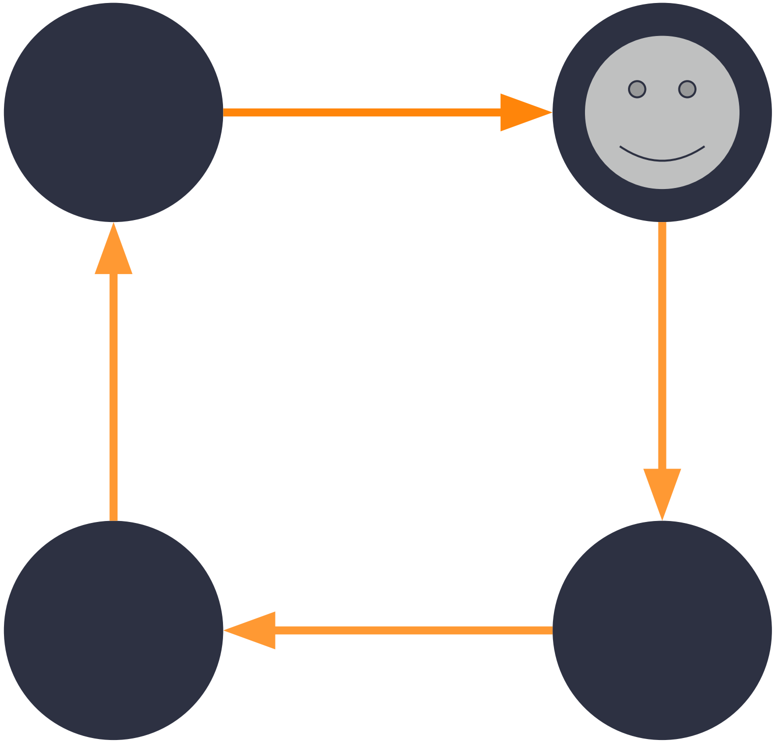


Time-expanded fixed cycle

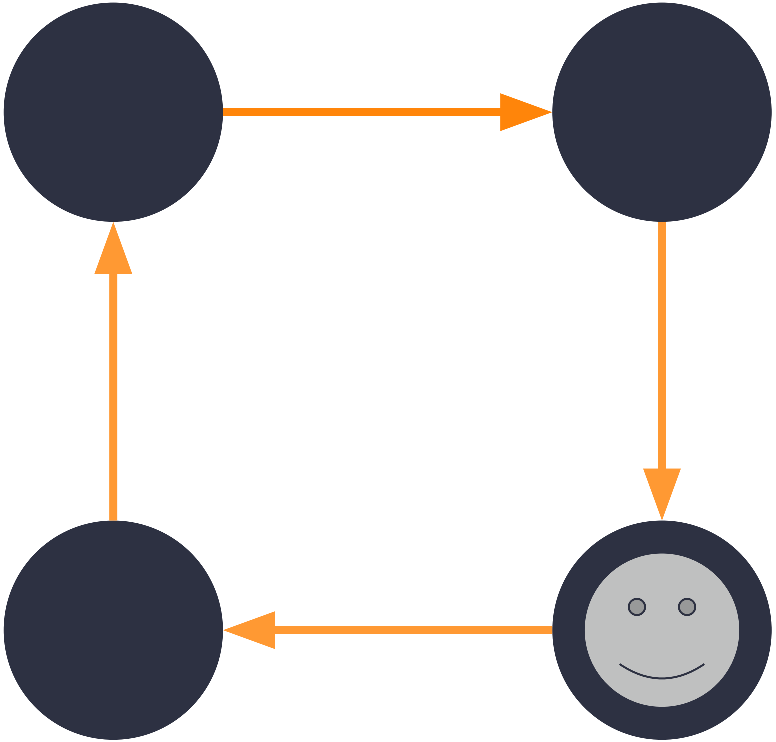




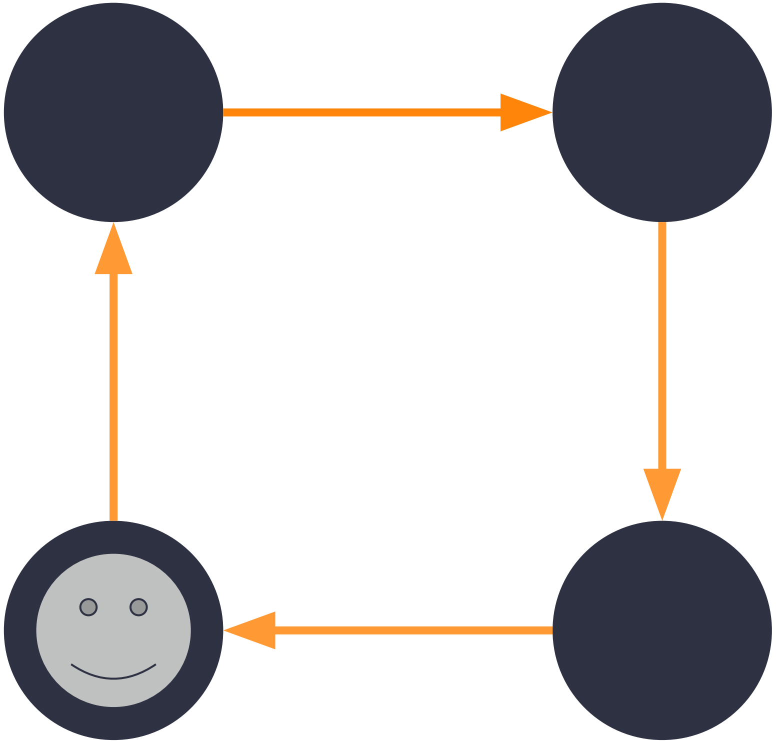
t = 1



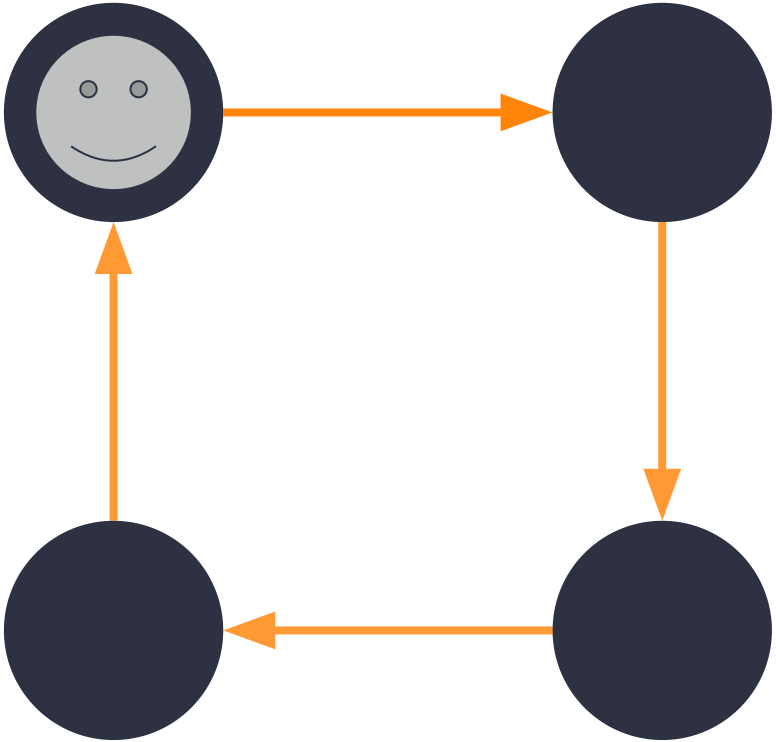
t = 2



t = 3



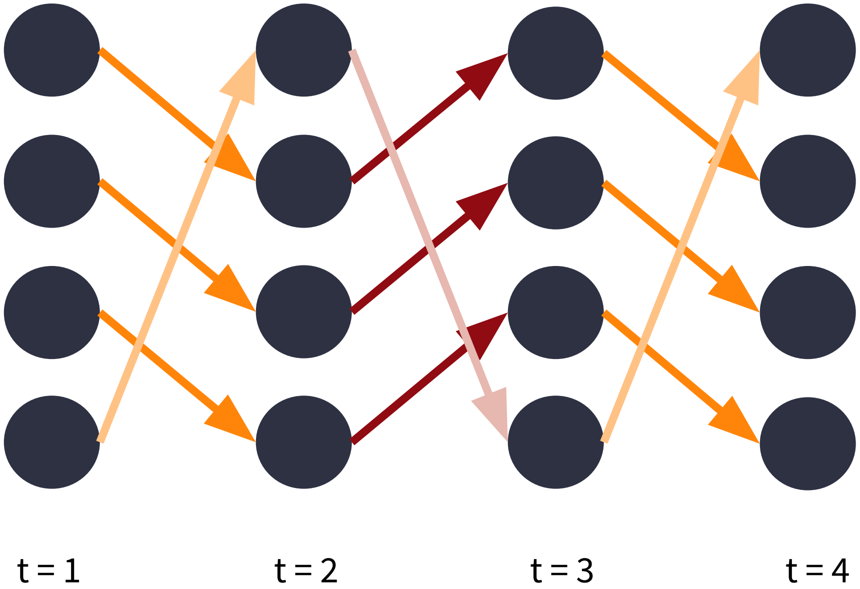
t = 4



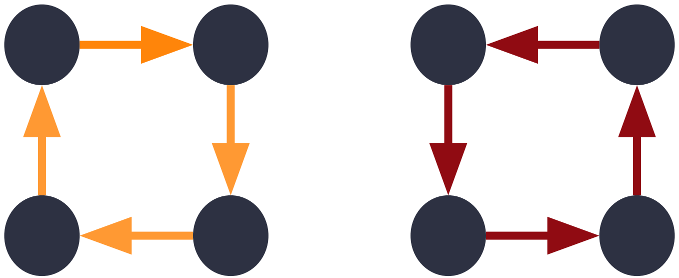
t = 5

Problem

$$\nu = n * t$$



too many nodes



Many Problems Don't "Just Work"

- Can we relate static and dynamic properties?
- How do we recover classical algorithms?
- Is there an efficient way to do all this?

🤓 Theory 💡

Two Orthogonal Dichotomies

Length

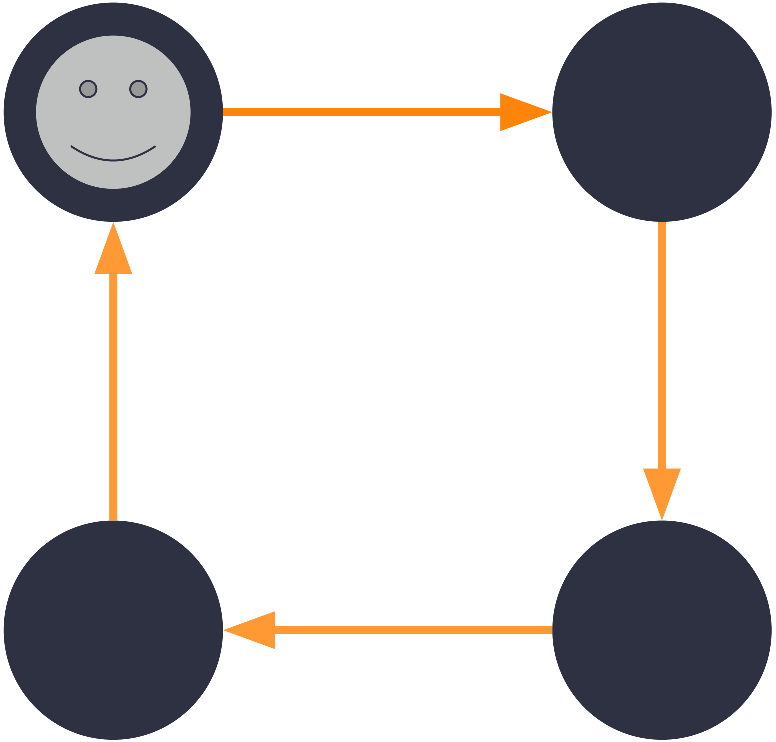
finite vs. infinite

Discretization

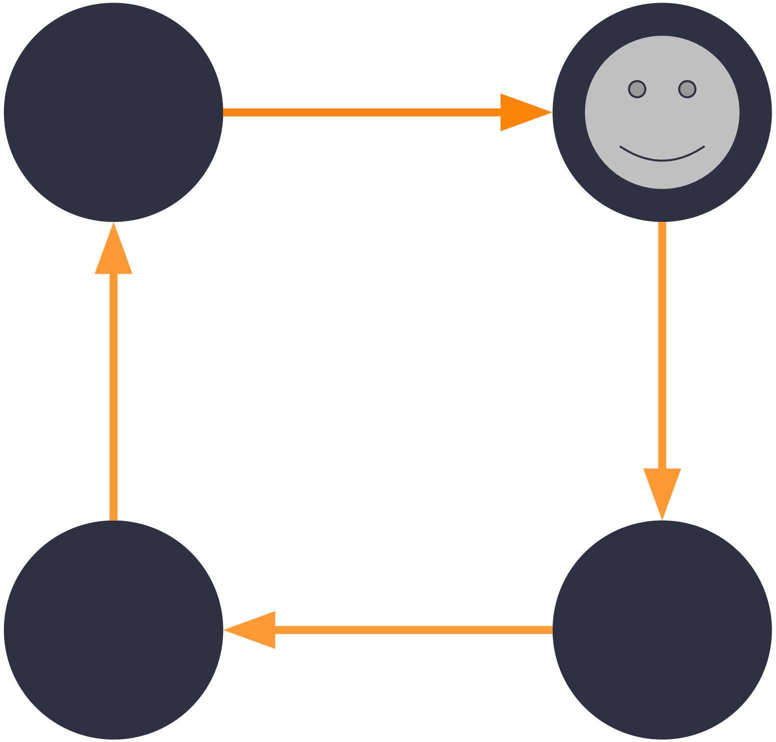
discrete-time vs. continuous-time

Dynamic Connectivity

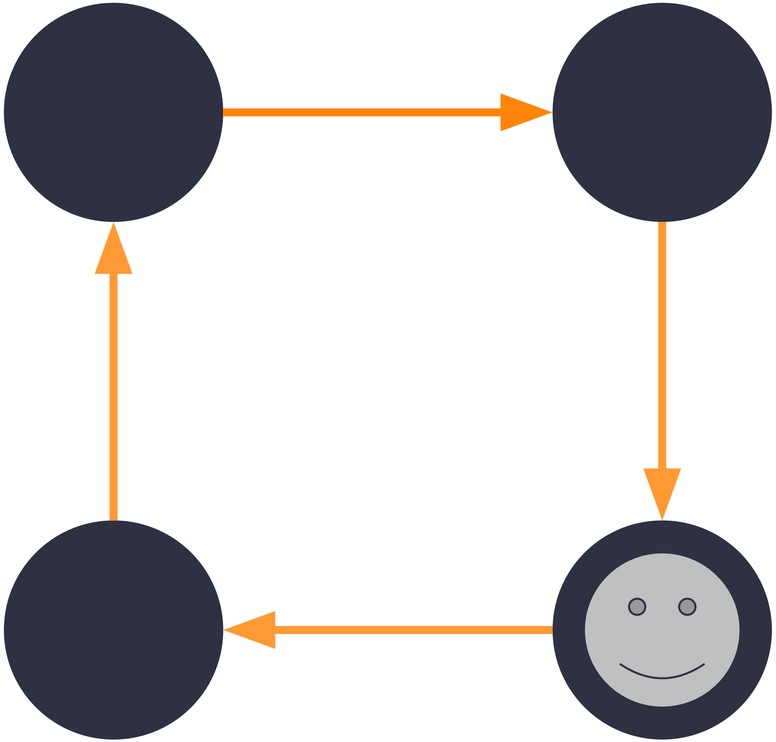
Journeys are Dynamic Paths



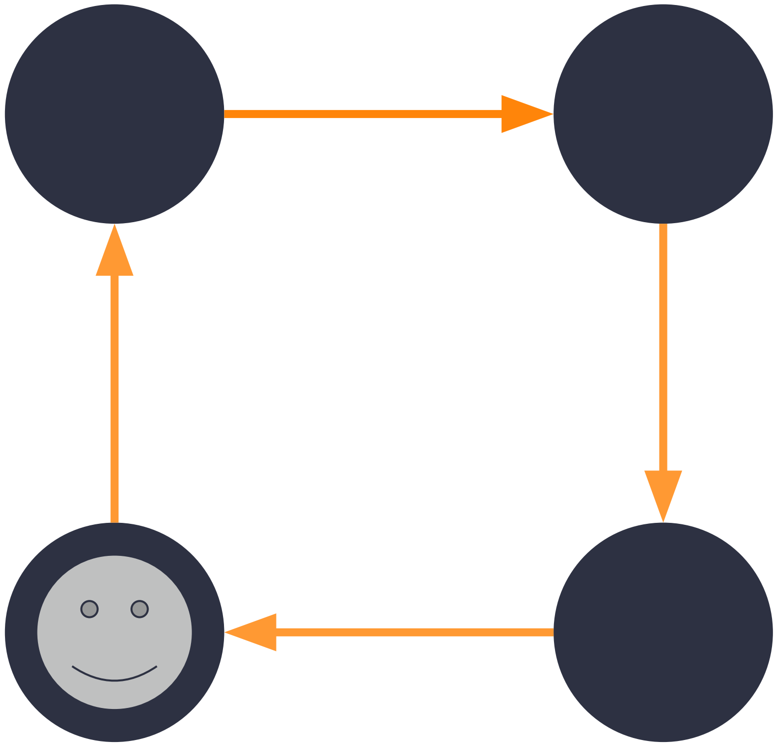
t = 1



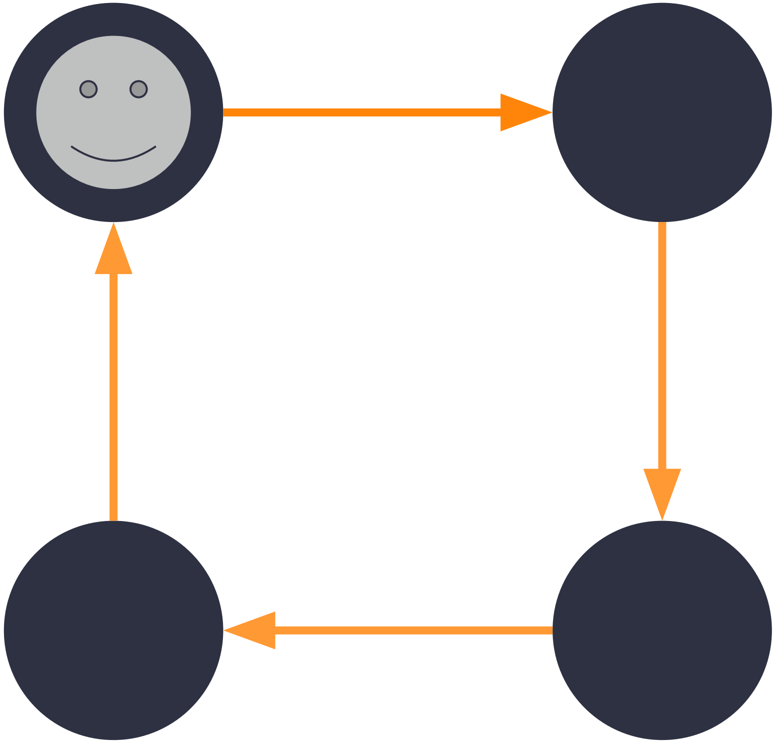
t = 2



t = 3



t = 4



t = 5

Definition: Dynamic Diameter

- discrete-time (infinite or finite)
- defined at each timestep
  - it's a sequence, not a number
- max of
  - shortest journey between all vertex pairs

Definition: Dynamically Connected

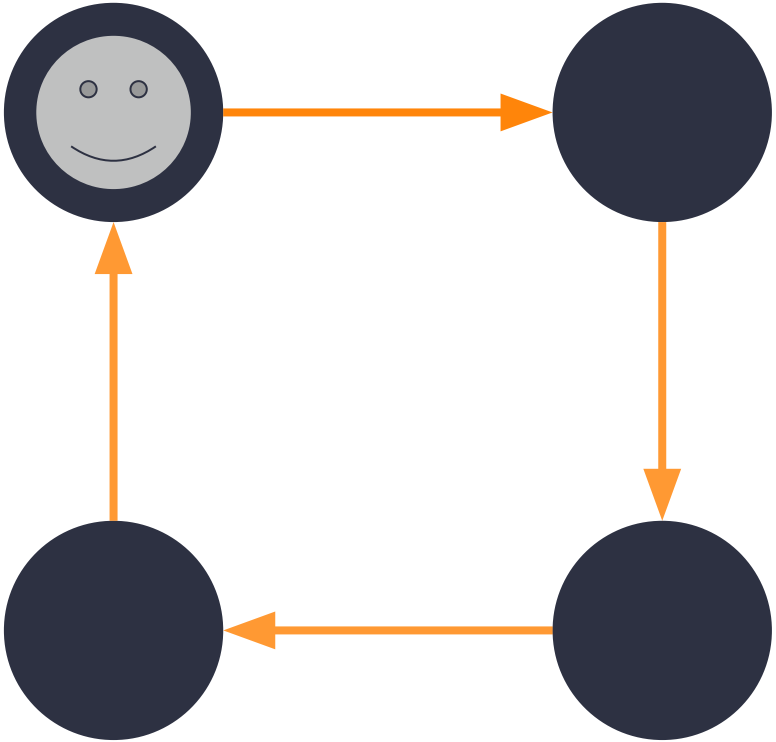
- *Connected* if diameter is always finite
- *Uniformly connected* if bounded

Proposition (We Didn’t Mess Up I)

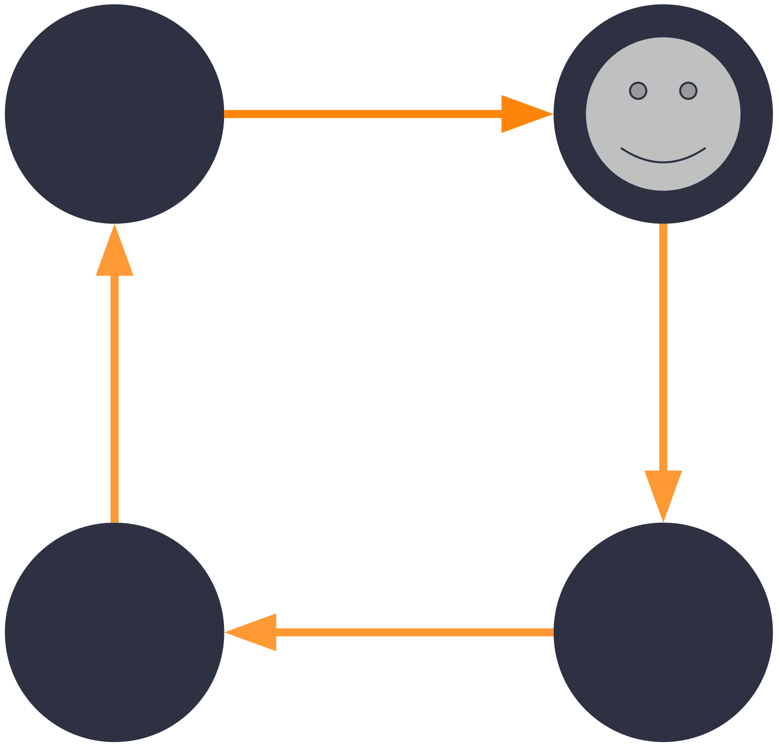
If at any time a vertex has no outbound edges, the graph is dynamically disconnected.

Proposition (We Didn’t Mess Up II)

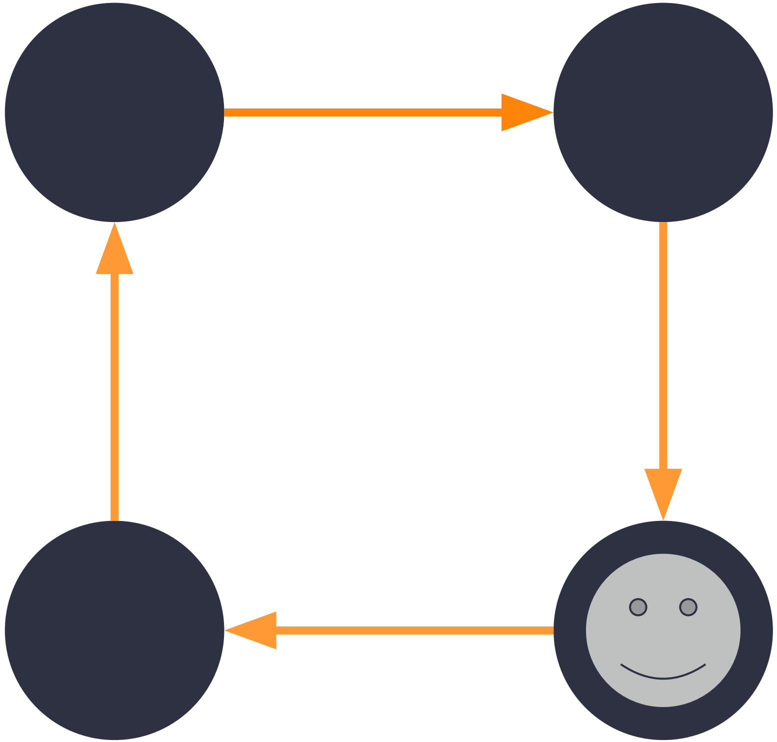
For a fixed dynamic sequence: diameter equals that of base graph.



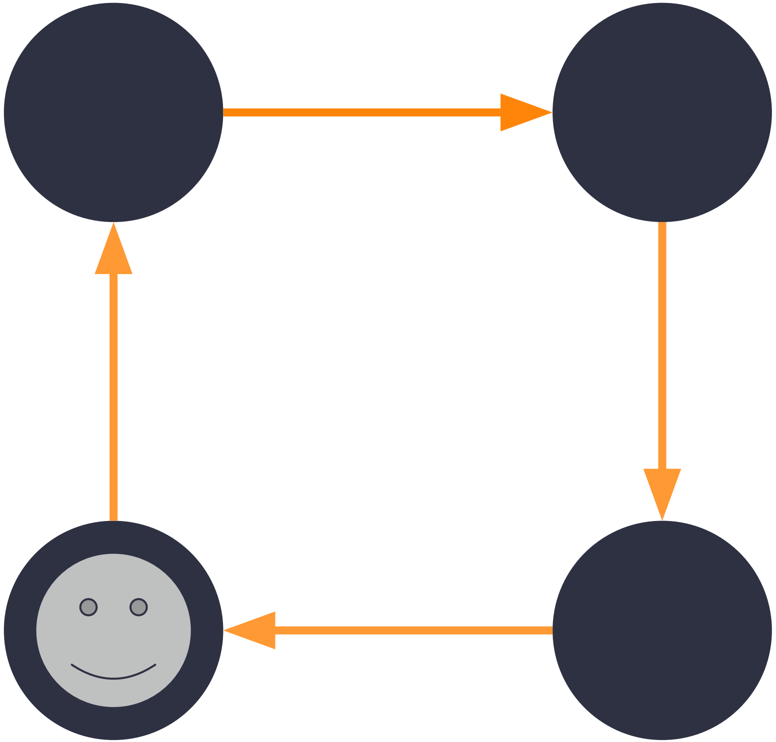
t = 1



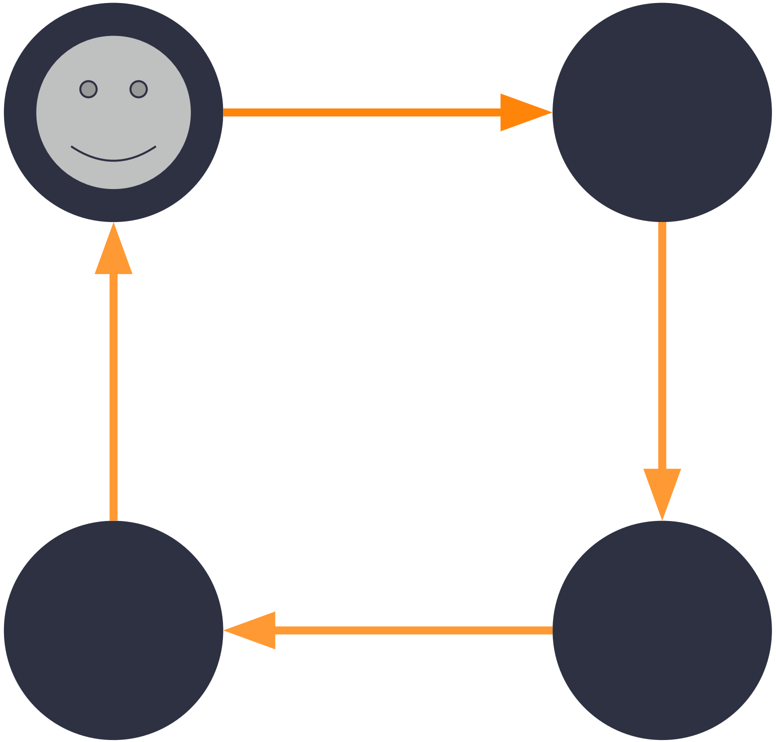
t = 2



t = 3



t = 4



t = 5

Question

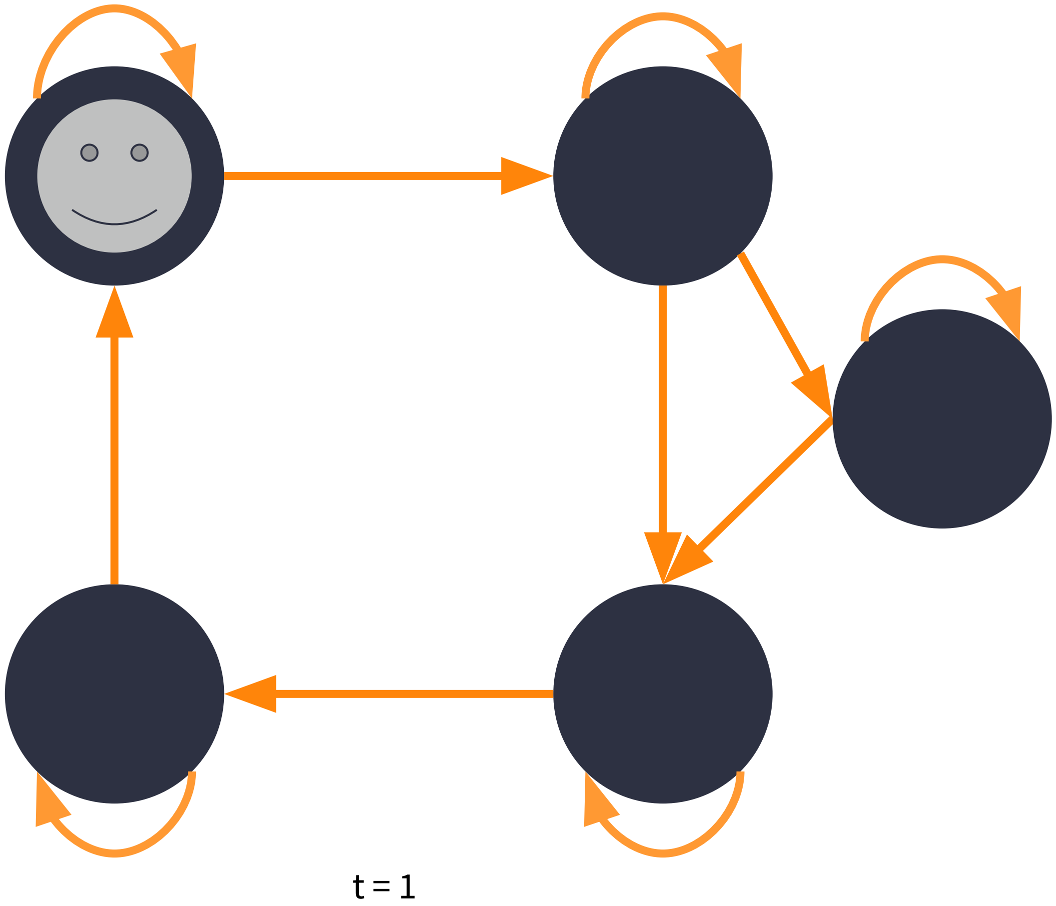
When does static imply dynamic?

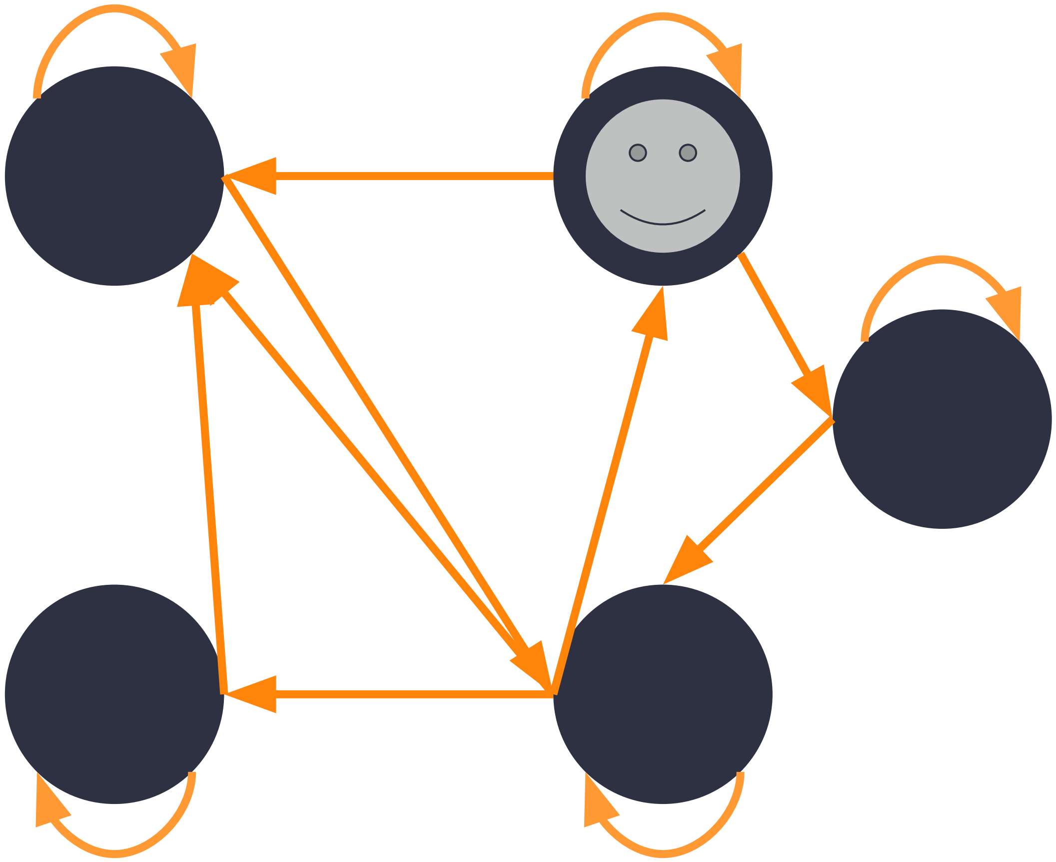
Result: Self-Loops are Sufficient

Static connectivity implies dynamic connectivity if self-loops.

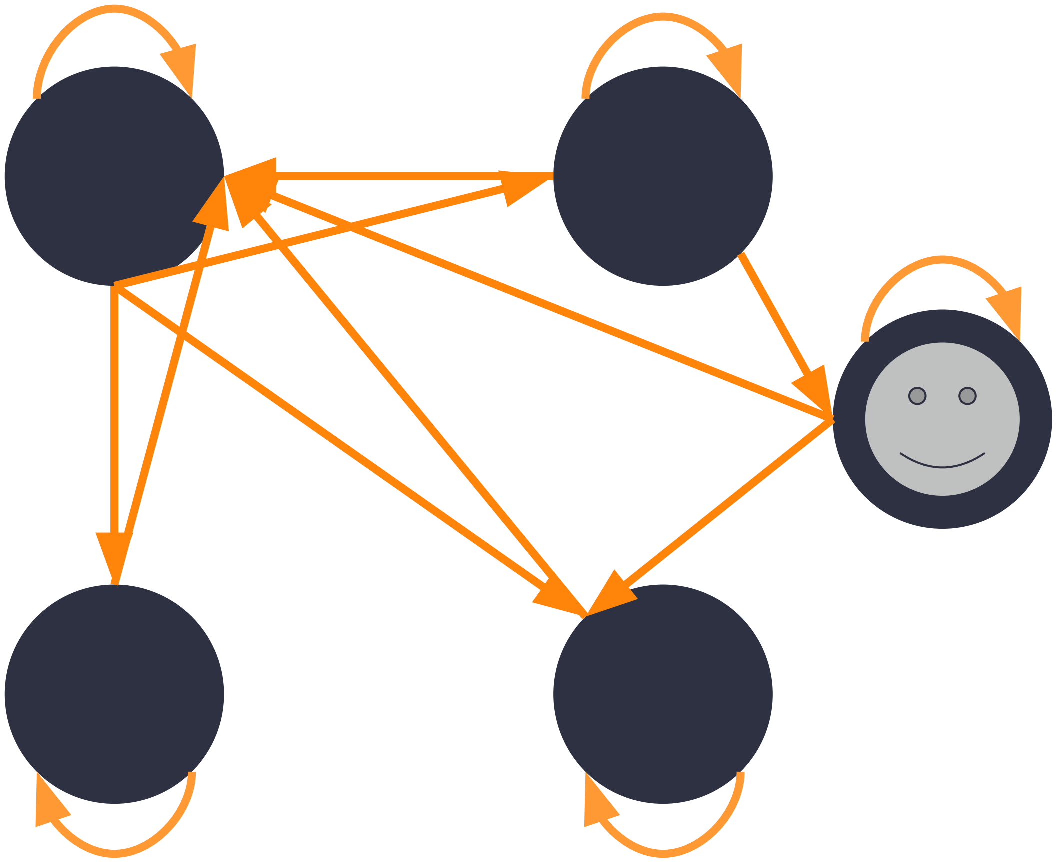
Other notes:

- Stronger (but more technical): weak monotonicity is sufficient.
- Uniform bound: number of vertices

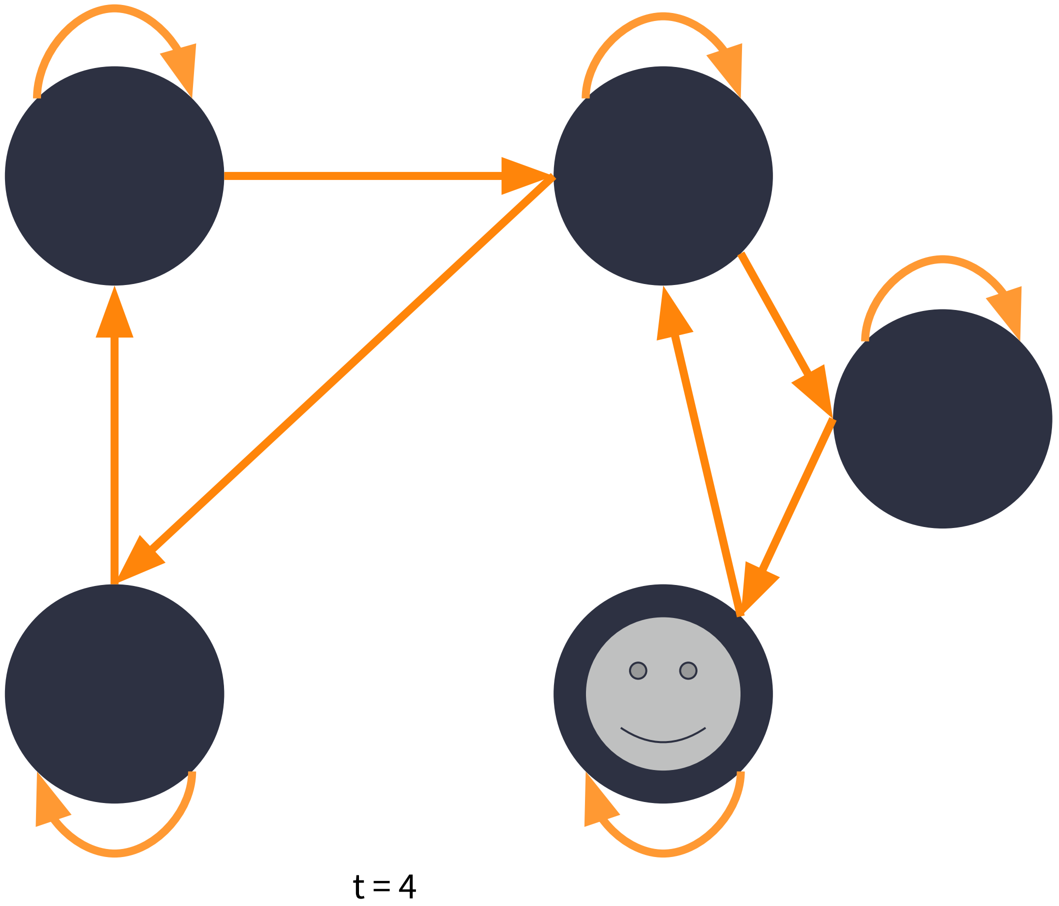

t = 1

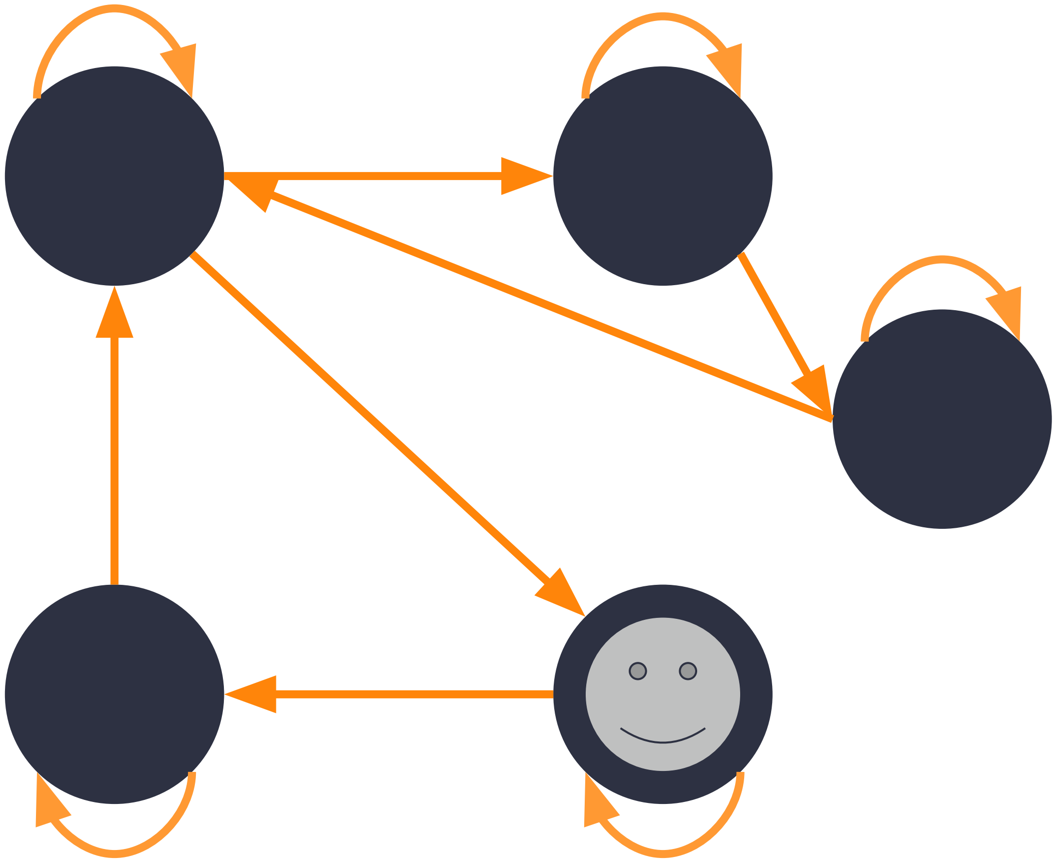


t = 2

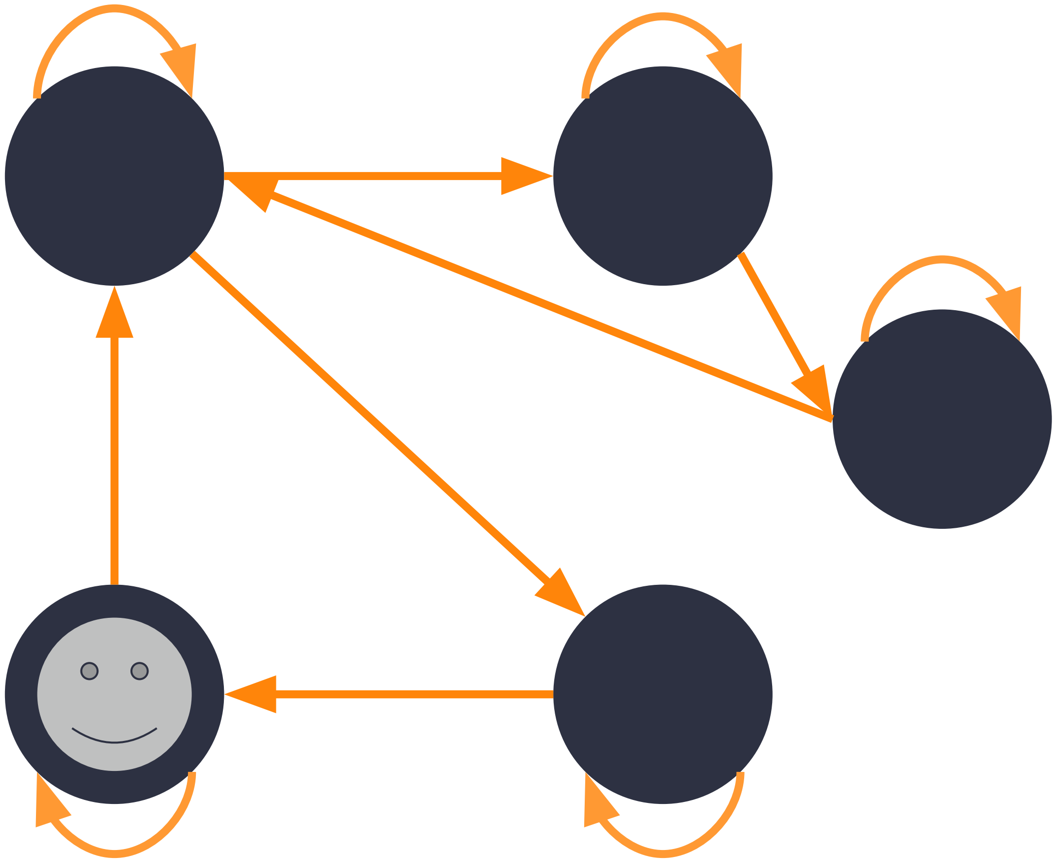


t = 3

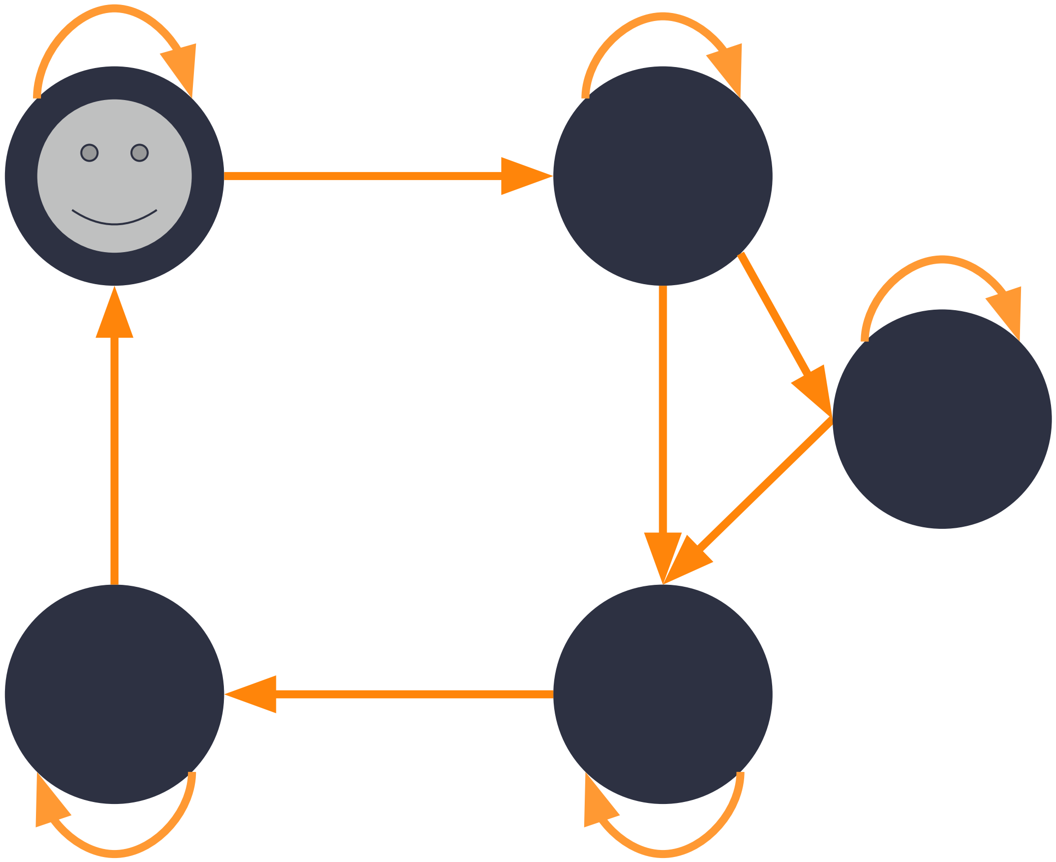

t = 4



t = 5

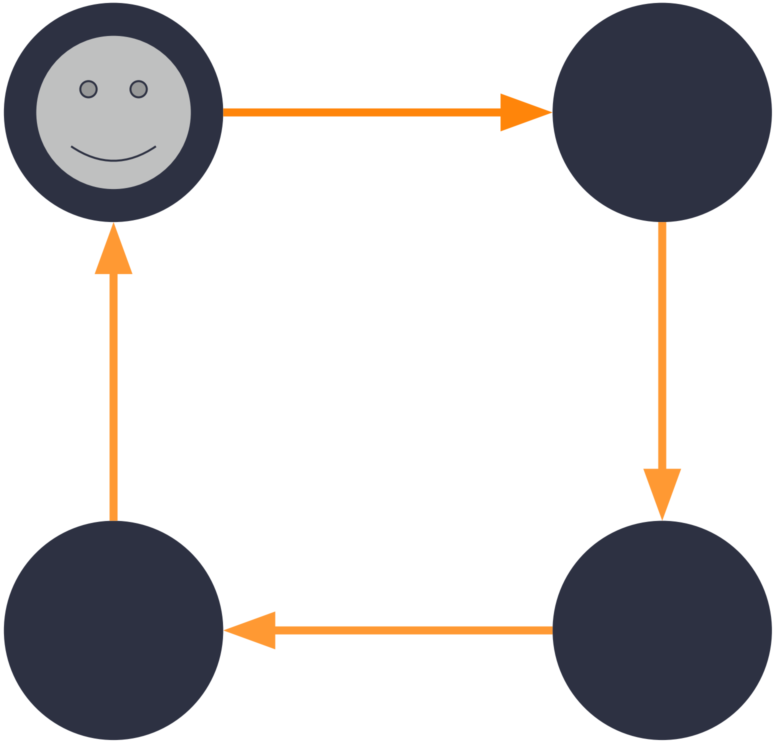


t = 6

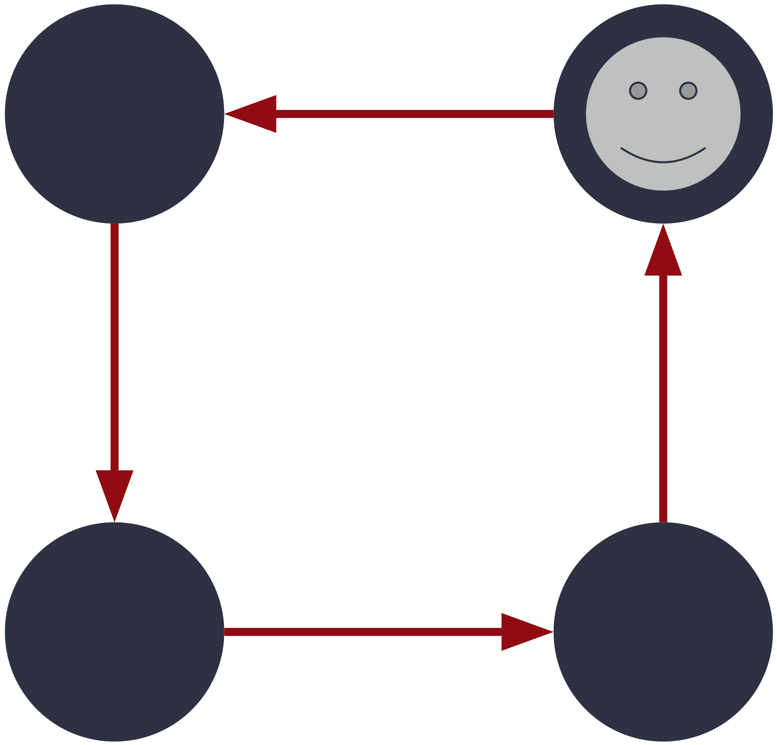


t = 7

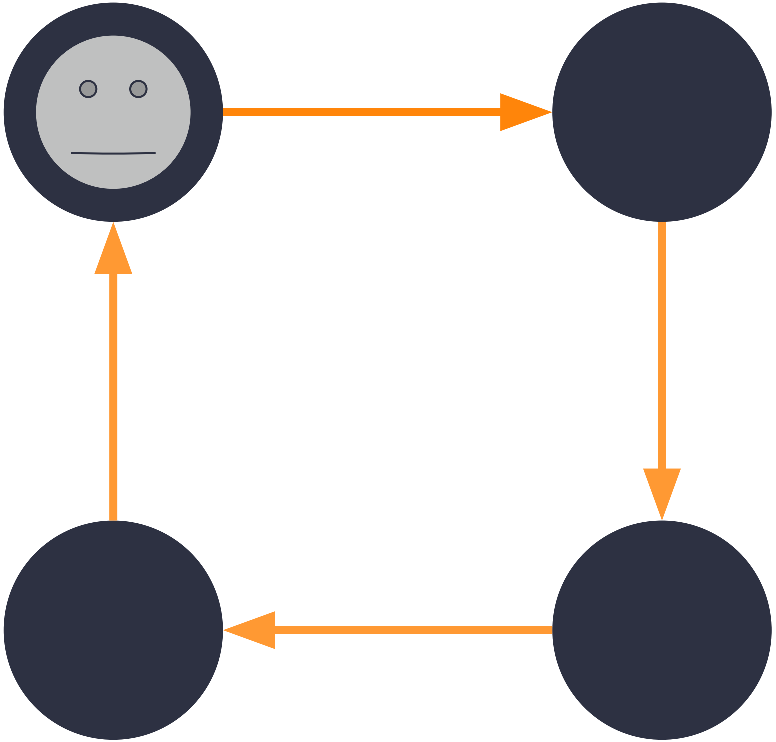
Recall



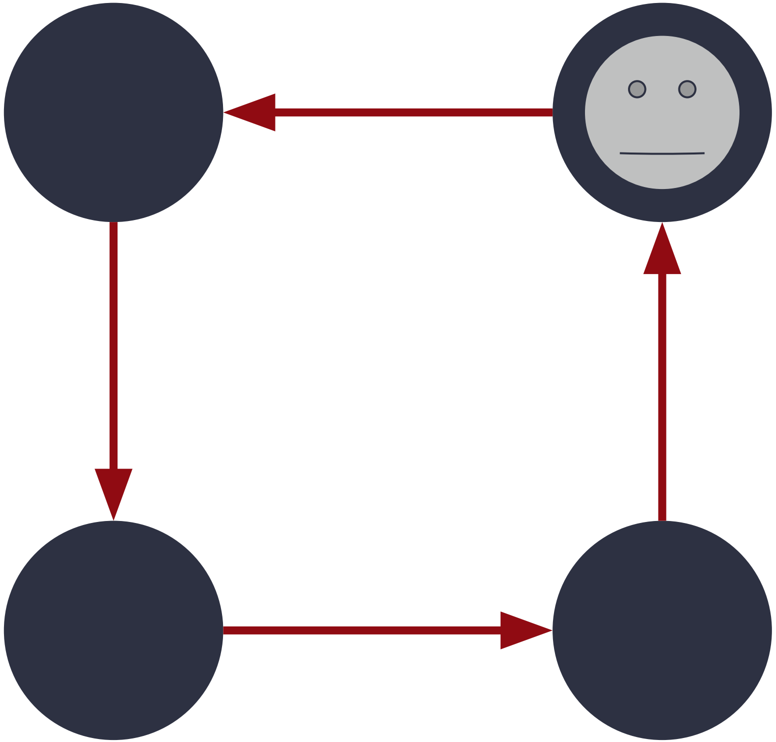
t = 1



t = 2



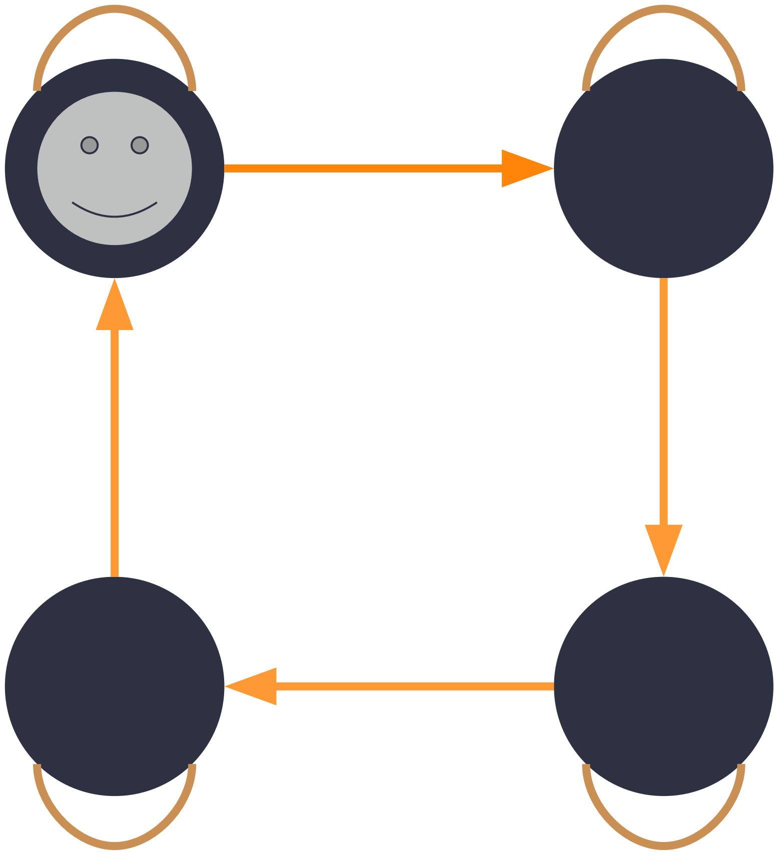
t = 3



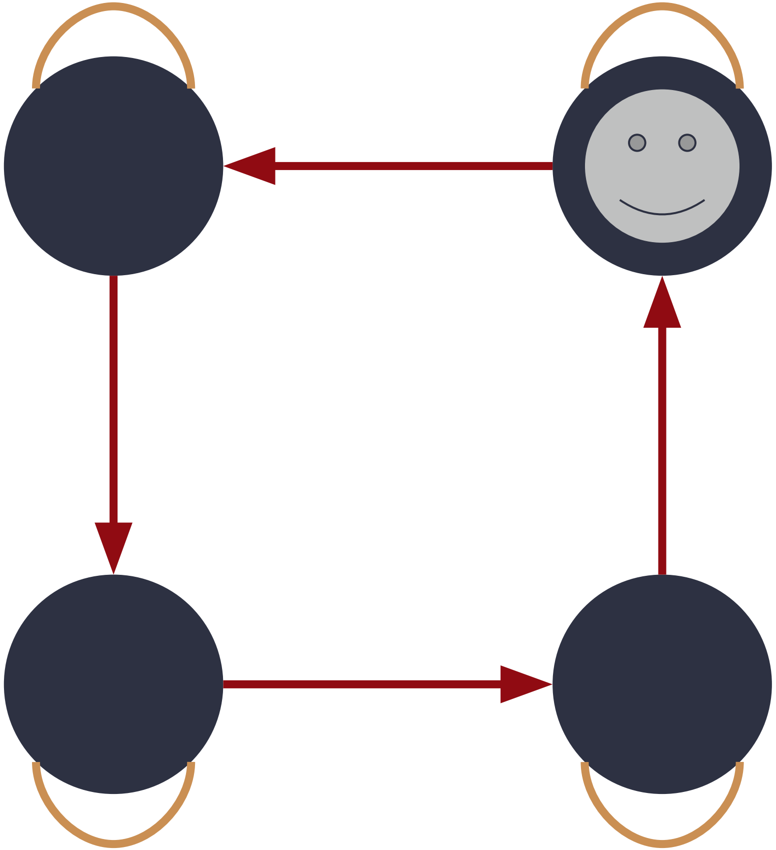
t = 4

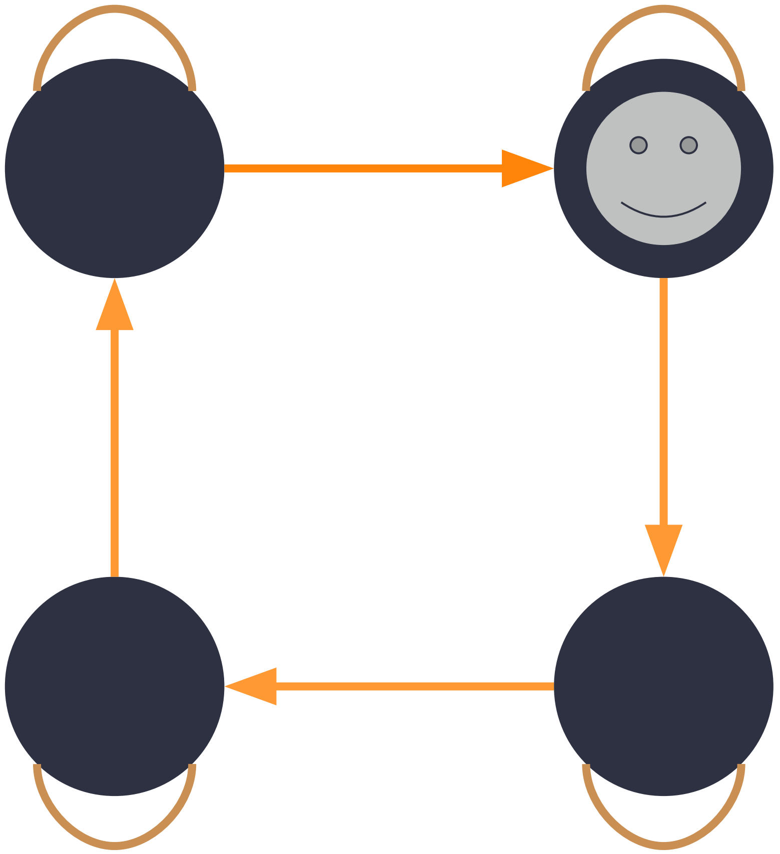
Dynamic diameter is ∞

Fixed with Self-Loops

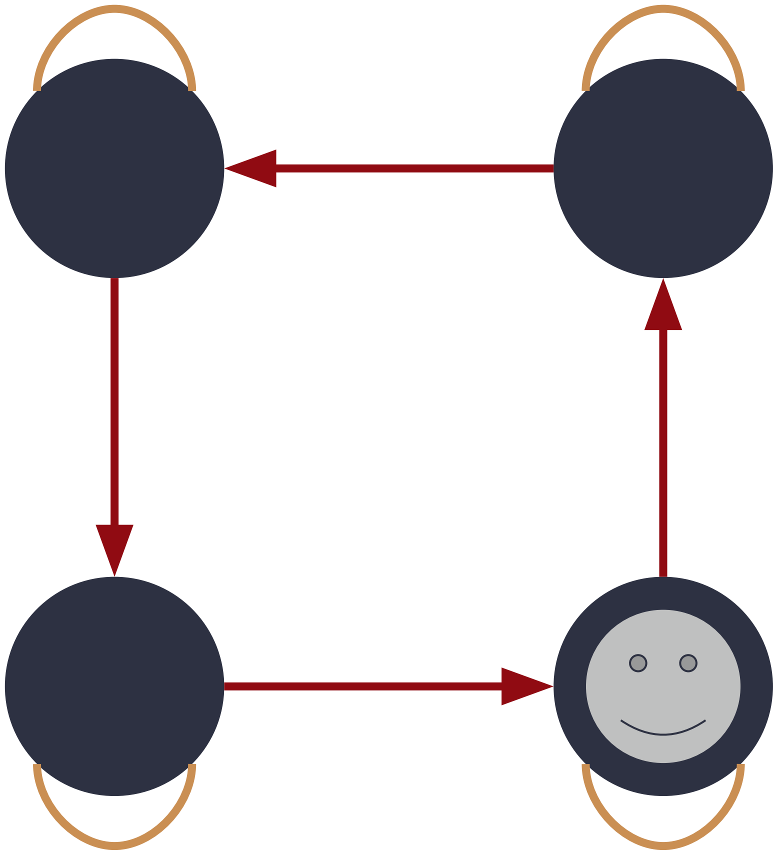


t = 1

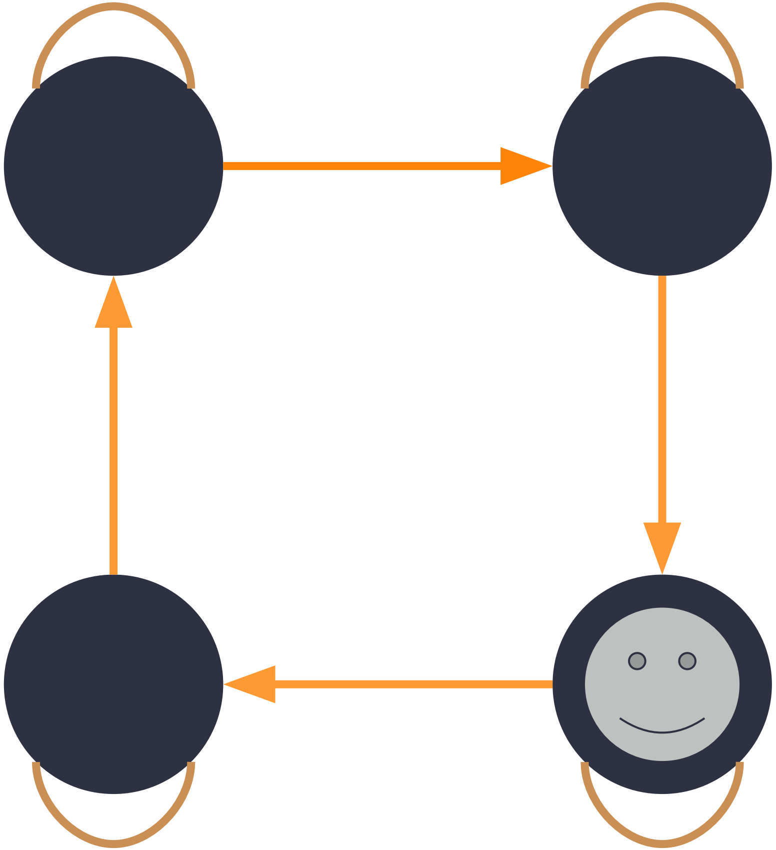

t = 2



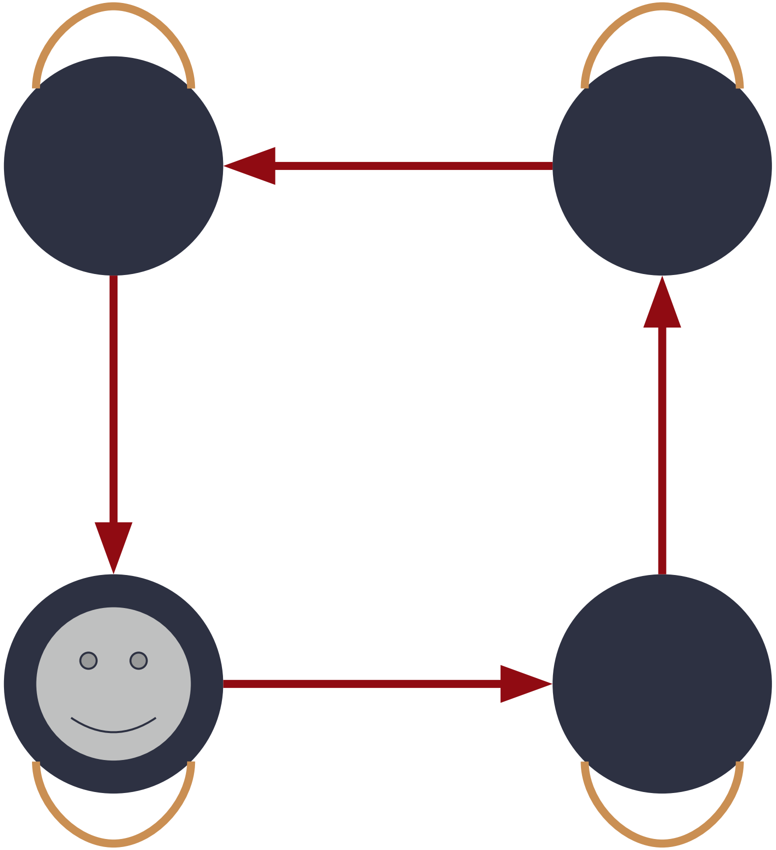
t = 3



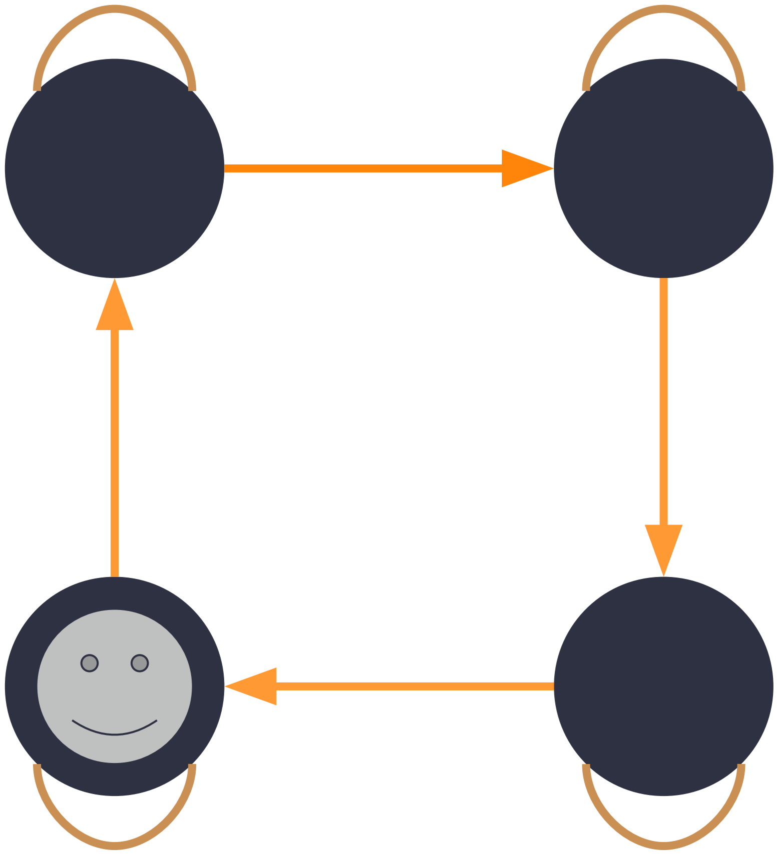
t = 4



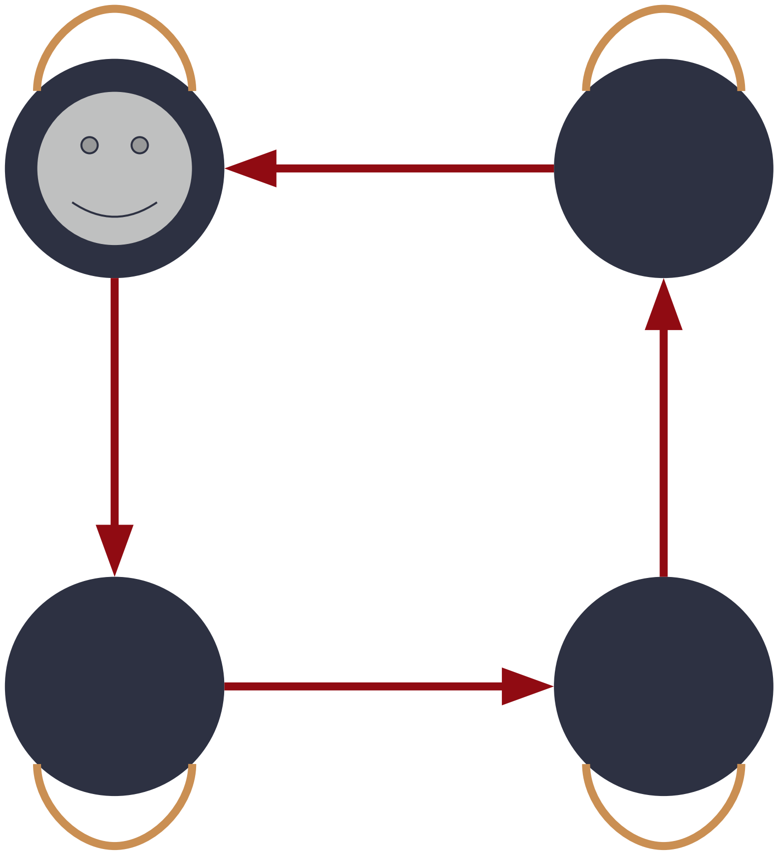
t = 5



t = 6



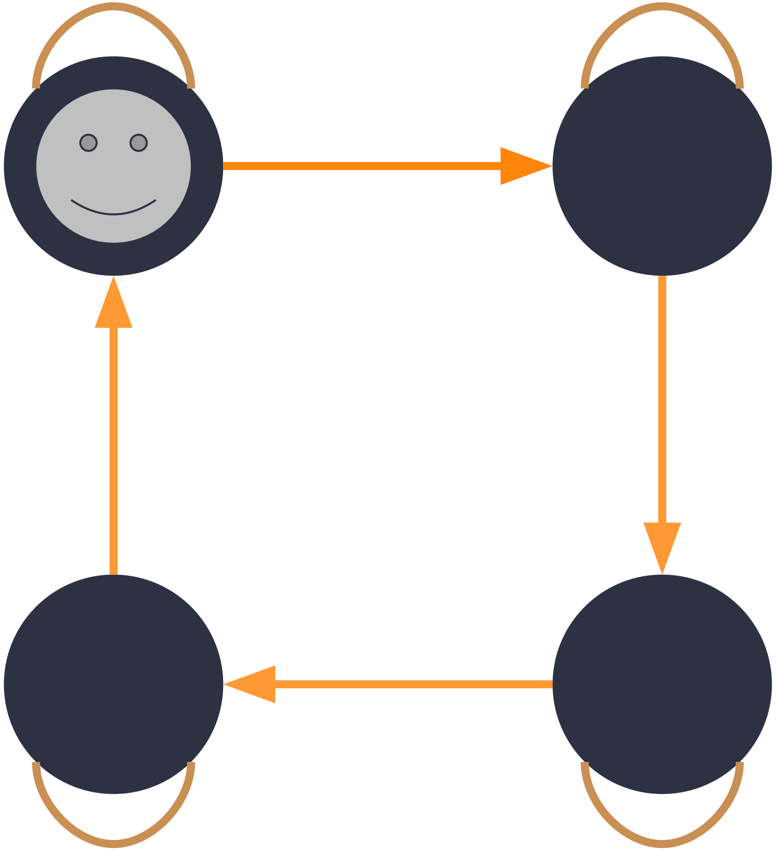
t = 7



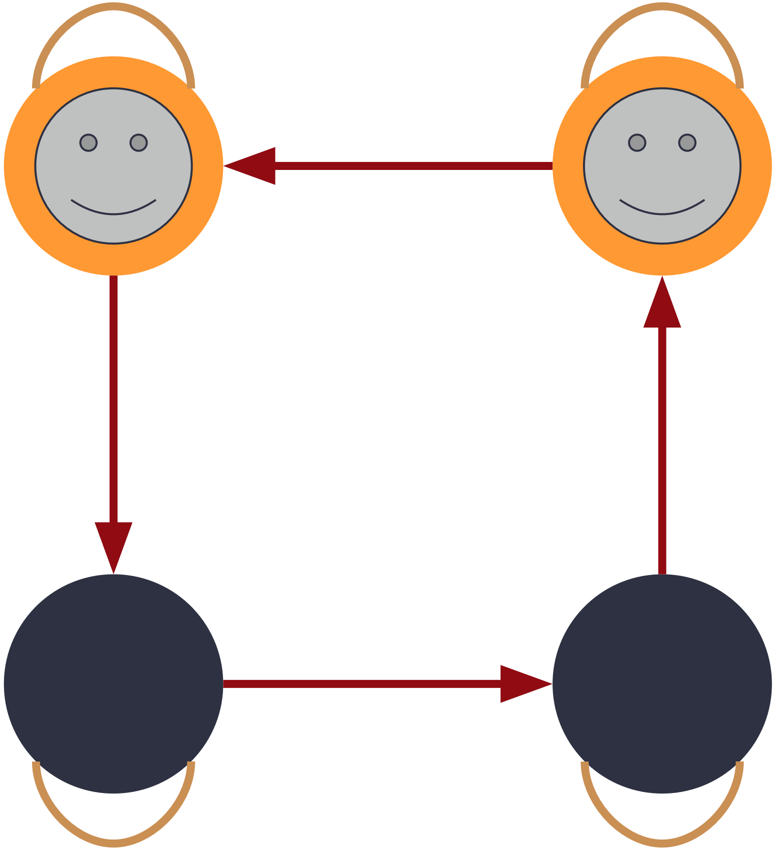
t = 8

Dynamic diameter is finite

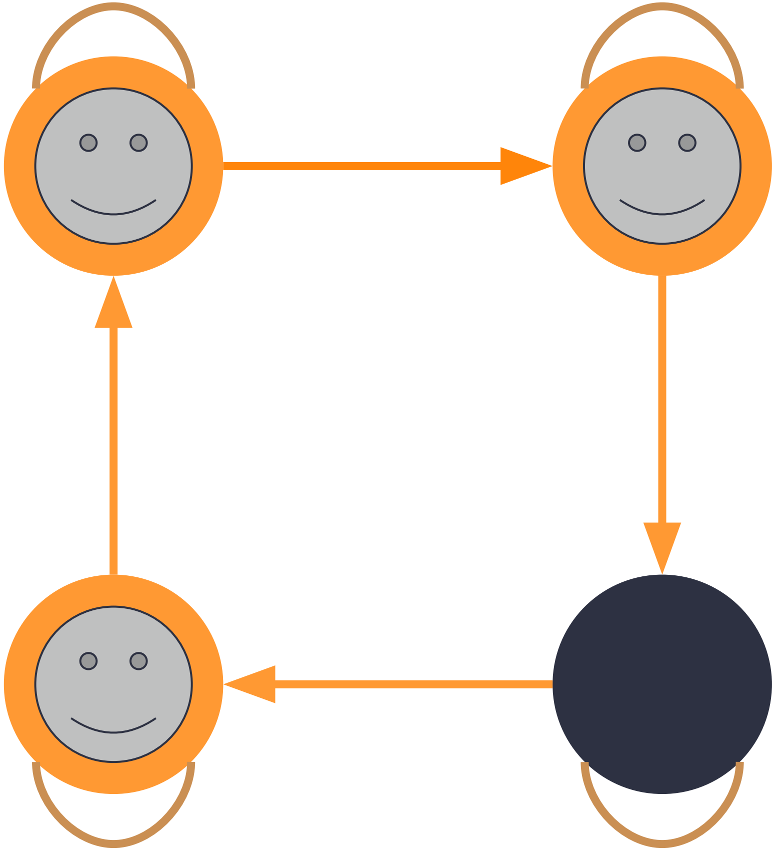
Bound Achieved



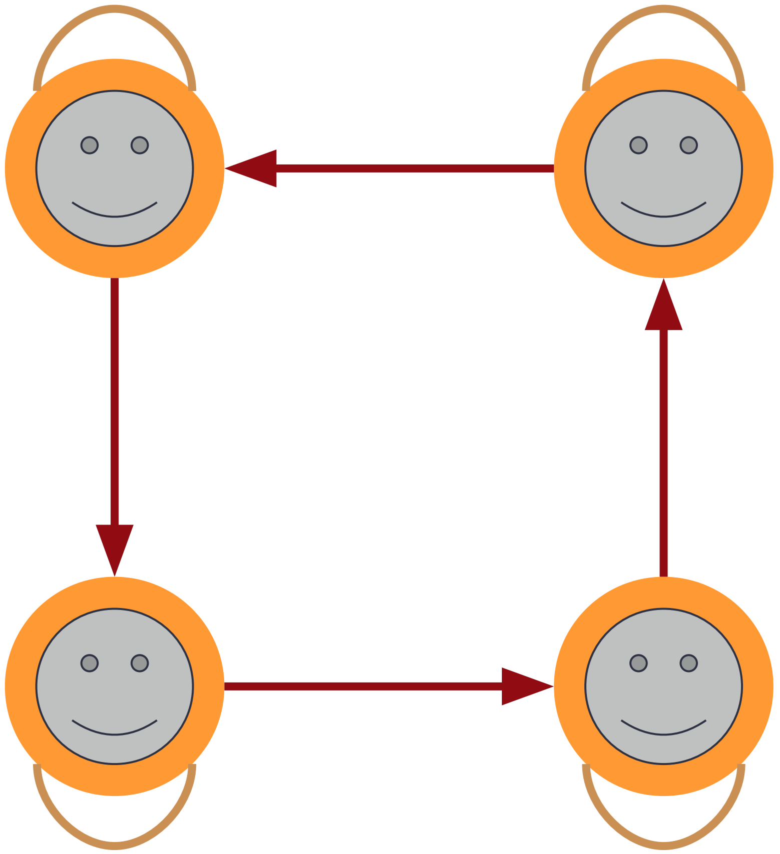
t = 1



t = 2



t = 3



t = 4

Observation

Finding Conditions is Difficult

Idea

Stochastic Case

Observation: Force Edges to Move

- A particle can get pathologically “stuck”
- Require edges to change around
    - Ensure that each possible edge appears infinitely often?

Model: Dynamic Erdős-Rényi

- Fix edge probability p $\in$ (0, 1)
- At every time step for every edge, flip a (biased) coin:
  - If heads, put the edge in
  - Otherwise, leave the edge out
- Note: edges across time are i.i.d. Bernoulli

Observation

Independence does not work

Proof: Independence => Disconnected

For each vertex at each timestep: probability of no outbound edges is $(1 - p)^n$, which is non-zero.

However ...

- Tweak: reflip all coins for a vertex if it has no outbound edges
    - Lose independence (a bit subtle)
- Based on simulations, conjecture: diameter is
    - constant if p constant
    - log n if p is (log n) / n

Observation

Self-loops are overpowered

Model: Dynamic Erdős-Rényi with Self-Loops

- Put in all self-loops
- Generate other edges (u, v) where $u \neq v$
  - Fix edge probability $p_{u,v} \in (0, 1)$
  - At every time step, flip a (biased) coin:
    - if heads, put in edge
    - otherwise, no edge

Proposition: Almost Surely Connected

1. Observation: every edge occurs infinitely often
2. By weak monotonicity, almost surely connected
3. Once connected, can never disconnect
4. Based on simulations, conjecture: diameter is
    a. constant if p constant
    b. log n if p is (log n) / n

Remaining Work

1. More rigorous treatment of non-self-loop case
2. Proof of proposed bounds
3. Additional models

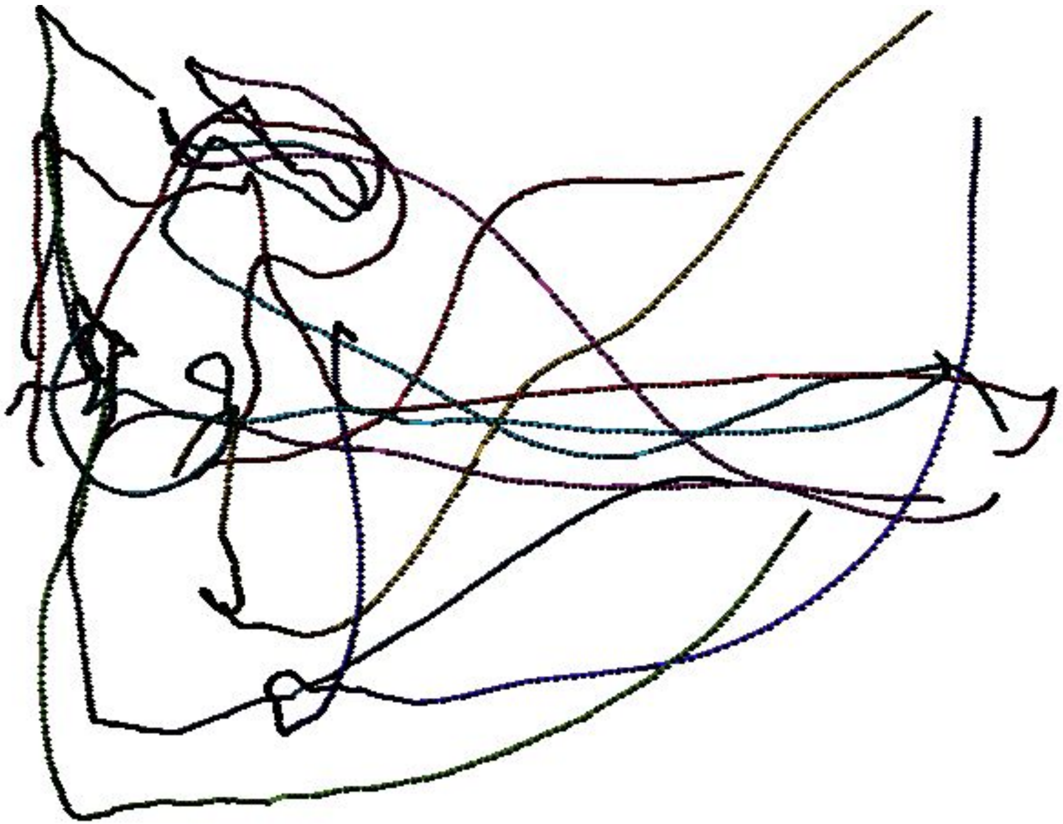
😎 Applications Abound 🧪

Dynamic Graph Projects

1. Viral spread across connected populations
   a. Rumors
   b. COVID-19
2. Basketball
   a. Using TDA
   b. Using ML
3. Space!
   a. Contact graph routing
   b. Tropical geometry
4. Others: animal clustering, transit, embryos, opinion dynamics

🤖 Machine Learning 🏀

Duke v. UNC (boooooo!)
Multi-agent system (invasion sport)



Raw Trajectory Data

Dataset

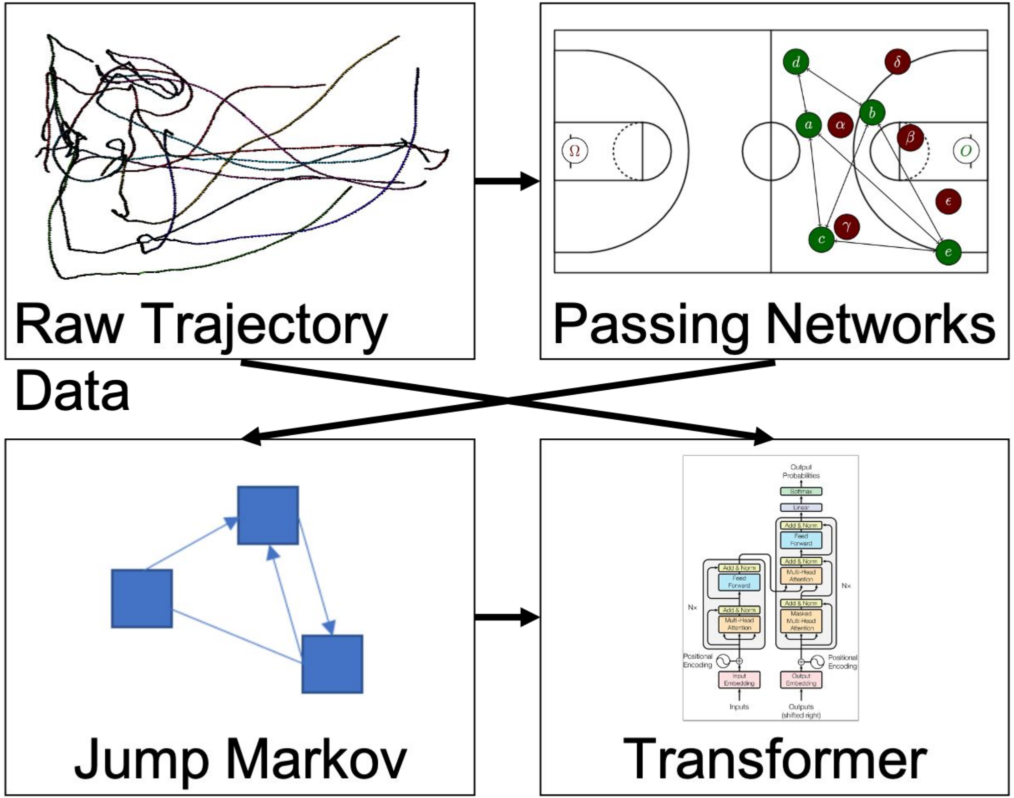
- (x, y)-coordinates of offense, defense, basketball
- 25 frames per second (40ms per frame)

Model Goals

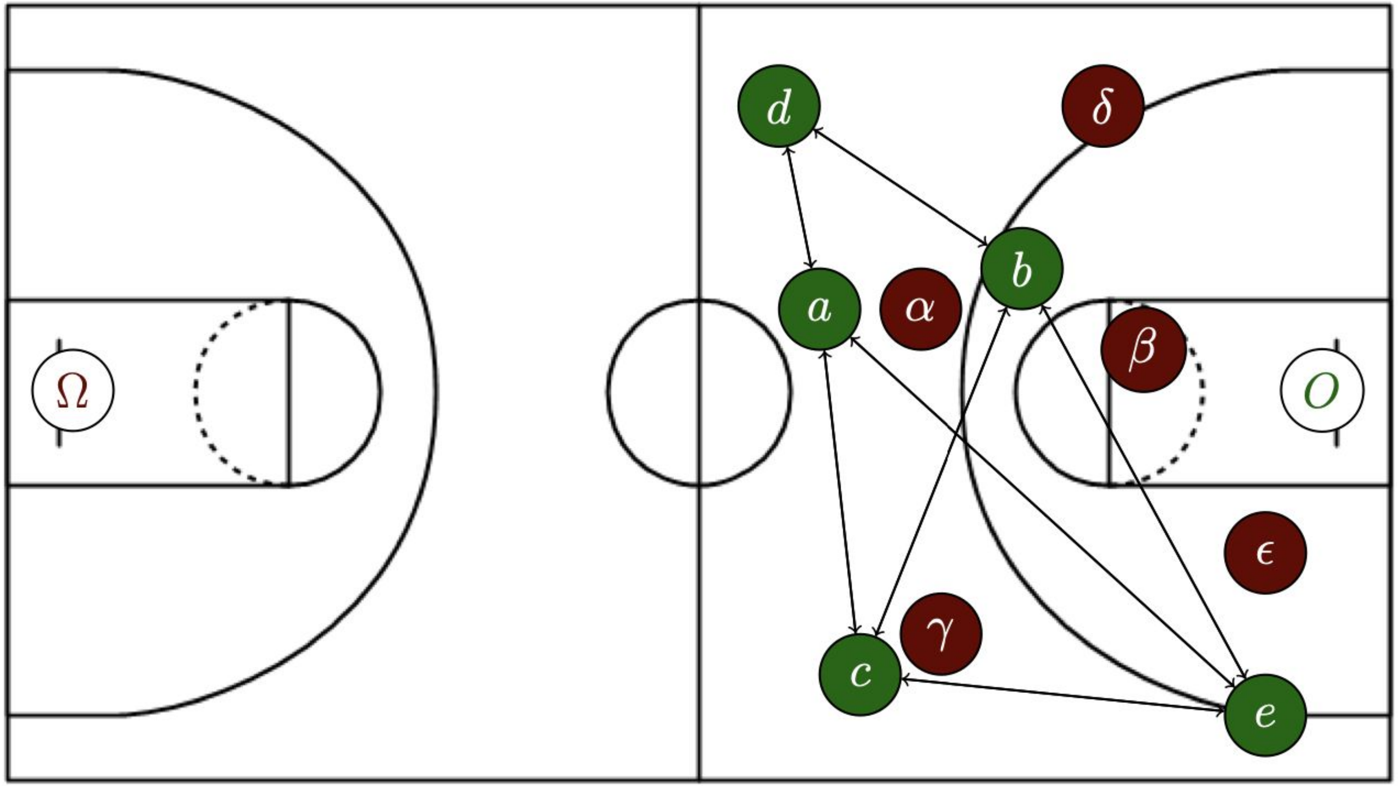
1. **Formation discovery**: a semantic understanding of the functional roles of players
2. **Compression and dimension reduction**: an efficient representation of a game, as player trajectory data is large and difficult to interpret
3. **Predictive power**: a mechanism for predicting trajectories of players
4. **Synthetic generation**: a tool for creating synthetic, but “realistic,” data

Prior Work

1. **Trajectory Prediction**: related to predictive power and synthetic generation
2. **Role Discovery**: related to formation discovery
3. **Network Analysis**: related to high compression and dimension reduction



Model Pipeline



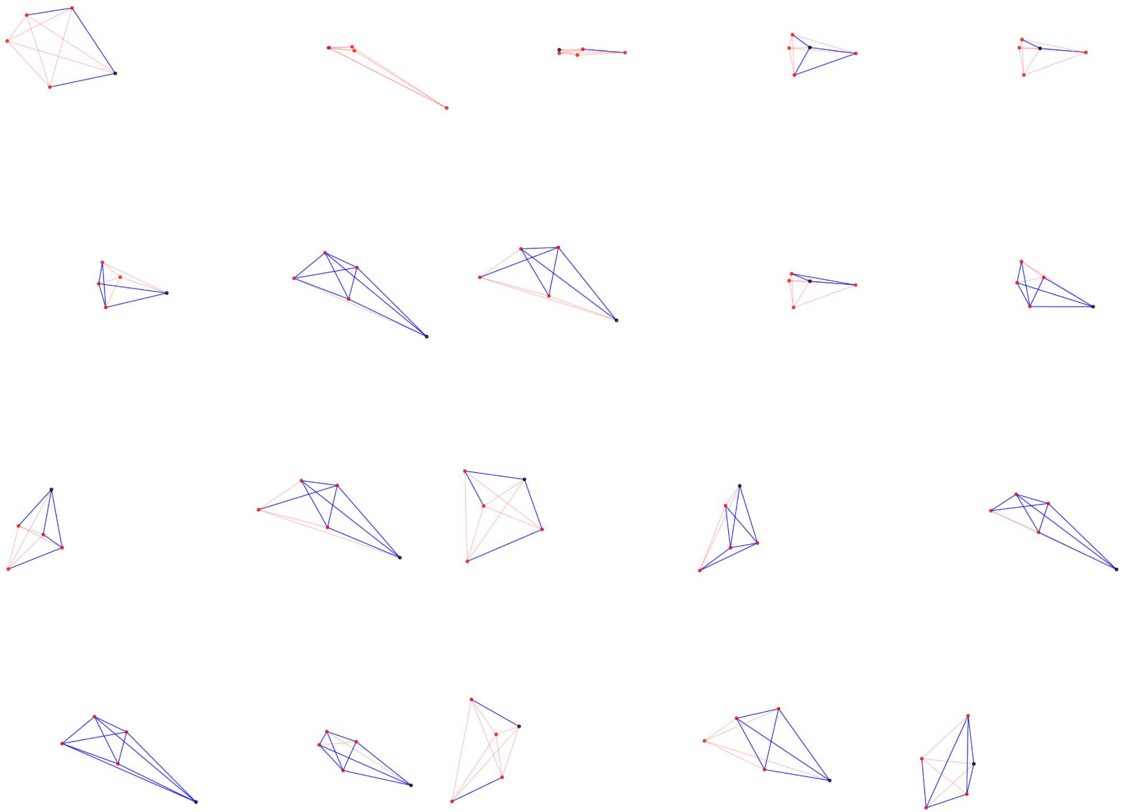
Step 1: Dynamic Passing Network

Observation

218 Graphs to Isomorphism

Step 2: Networks to Labels

1. Compute library of graphs seen in data
2. Assign each a unique label (frequency-based?)
3. Convert networks to labels



Passing Graphy Library

Interlude: Jump Markov

$$X(t) = E_{N(t)}$$

Interlude: Jump Markov

$$\boxed{X(t)} = E_{N(t)}$$

continuous-time *jump*
Markov process

Interlude: Jump Markov

$$X(t) = E_{\boxed{N(t)}}$$

Poisson counting process

Interlude: Jump Markov

discrete-time *Markov chain*

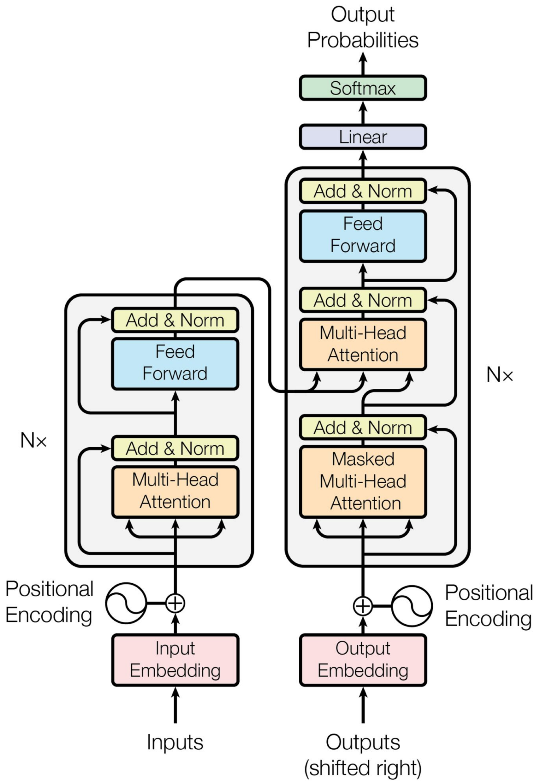
$$X(t) = \boxed{E}_{N(t)}$$

Interlude: Jump Markov

$$X(t) = E_{N(t)}$$

Idea: Semantic “Extraction”

1. Take library of passing graphs as tokens
2. Use NLP model to learn as a “language”
3. Good model: *Transformer*



Transformer Architecture

Experiment

- 40-10 prediction task
- Feed in graph data, along with base position data
- Predict trajectories
- Compare against true trajectories with MSE

66%

reduction in loss against benchmark
(40–10 trajectory prediction task)

🔮 Future Work 🔬

Theory

1. Further extensions of static properties and their relationship to their dynamic counterparts
2. Analysis of the stochastic setting
3. General framework for summarization
4. Clustering (spatiotemporal *k*-means)
5. Generalized optimal routing
6. Periodic systems

Applications

1. Animal herding behavior
2. General Transit Feed Specification (GTFS)
3. Twitter data
4. Additional satellite data

🙏 Acknowledgements □

🙋 Q&A 🙃