Item pool maintenance in computer adaptive tests: A network approach

Klint Kanopka

NYU | STEINHARDT

This is ongoing work!

- This is ongoing work:
  - Sophia Deng @ NYU Steinhardt
- There's a lot more than 12 minutes of stuff to discuss

Overview

- What are the fundamental issues in computer adaptive testing (CAT), and how do networks fit in?
- We had to write software, so we tried to make it useful and you have to hear about it
- Results from some network approaches
- Discussion

Fundamental CAT Issues

CAT issues fall into two categories

- Category 1: Item Calibration
  - Parameter drift
  - Recalibration
  - Item rotation
    - Vermeiren, et al. (2025)
    - Bolsinova, Gergely, and Brinkhuis (2025)
- Category 2: Item Usage
  - Item overexposure
    - Gorney and Reckase (2025)
  - Item compromise
    - Gorney, Lee, and Chen (2025)

How do we introduce networks to CAT?

- **Core idea:** If we treat the CAT item pool as a network where individual items are nodes, this opens a broader class of statistics and algorithms that can be used to monitor and interact with a CAT system
- We draw weighted edges in the graph between items that have had individuals co-respond to them
- For two items *i* and *j*, if N_{ij} individuals have responded to both items, the weight of the edge between nodes *i* and *j* is some function of N_{ij}

Why focus on item co-responses?

- Item pools are well calibrated when the pairwise relationships between item parameters are well known
- Odd exposure patterns can produce situations where the relationship between two items is subject to compounding error from intermediate links
- Functions like the inverse of the number of co-respondents can be used as edge weights to give a natural sense of distance between items that is related to the uncertainty in the relationship between their parameters

What does this buy us?

- Item selection algorithms can take network distance into account
- This allows for selection that optimizes for increased network density
  - Nodes become closer together on average
  - Improves item calibration by reducing uncertainty in differences between pairwise parameter estimates
  - Improves item exposure by naturally selecting items that are farther from the bulk of the network
  - As new items are introduced to the pool, also naturally focuses on building their pairwise relationships to other items
- **Augments, not replaces, current CAT research**

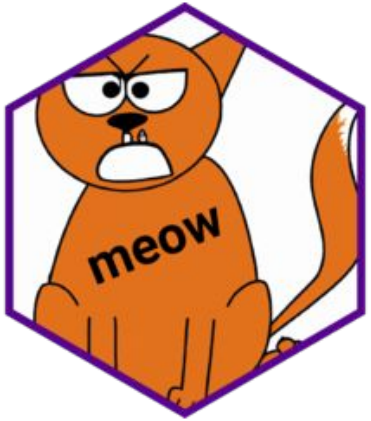
Constructing easy and reproducible CAT simulations

Doing CAT simulations

- Packages exist for conducting CAT simulations, like mirtCAT
    - Focused more on simulations with differences in operational conditions
    - Less useful for algorithmic innovations
- The papers referenced earlier all implement their own bespoke simulation frameworks
    - Comparing algorithmic innovations requires reimplementing someone else's code to work within your framework
    - This is annoying and makes thorough simulation studies with new methods less practical
- This sucks and makes innovation in the CAT space slower and harder to reproduce!

meow

- We needed our own bespoke simulation framework for adjacency matrices
- This was annoying, so we decided to invest time making it useful to other people
  - klintkanopka.com/meow
  - github.com/klintkanopka/meow
- meow is a modular and extensible CAT simulation framework with consistent and flexible APIs for different components!
- It's available for use now, feedback and contributions are very welcome!



NYU | STEINHARDT

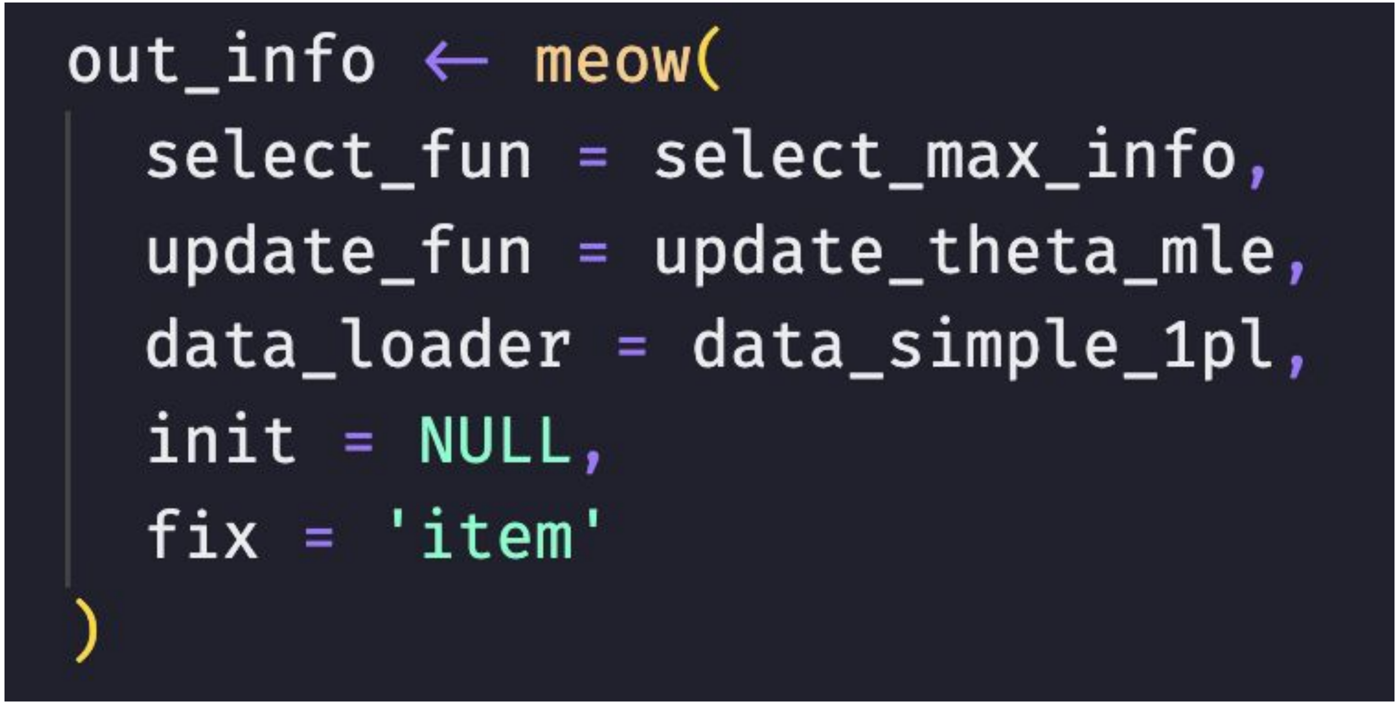
meow philosophy

- Each simulation has three parts:
    - A *data loader* function that determines person and item parameters and how responses are generated from them
    - An *item selection* function that implements how the next item for each respondent is selected and any necessary stopping rules
    - A *parameter update* function that determines if and how person and item parameters are updated after each item is delivered
- These chunks are designed to be interchangeable, so running fully crossed simulation designs becomes as simple as specifying the conditions you want to use
- Each full simulation is contained within one function call and outputs consistently formatted results

meow philosophy

- meow components are designed with a consistent API! We provide vignettes instructing users on how to implement their own, as well as use the package and generate visuals
- The entire package is open source, so we encourage authors to write data loaders, selection functions, and parameter update functions for their own algorithms and submit them via pull request for inclusion in the package
- This helps future authors include previous methods in their simulations with minimal friction!

Conducting a meow simulation



```
out_info ← meow(
    select_fun = select_max_info,
    update_fun = update_theta_mle,
    data_loader = data_simple_1pl,
    init = NULL,
    fix = 'item'
)
```

meow output

- Each simulation outputs a list with four objects:
    - $results is a dataframe with one row per simulation iteration that has current estimates and current bias for each person and item parameter
    - $adj_mats is a list of adjacency matrices, one per simulation iteration
    - $pers_tru and $item_tru are dataframes with one row per person/item and one column per parameter that contain the true values used in the simulation
- The $results dataframe is pre-formatted to be easily parsed and pivoted using dplyr tools for use with ggplot2
- The $adj_mats list is formatted to be easily integrated into dynamic network visualization tools

Results

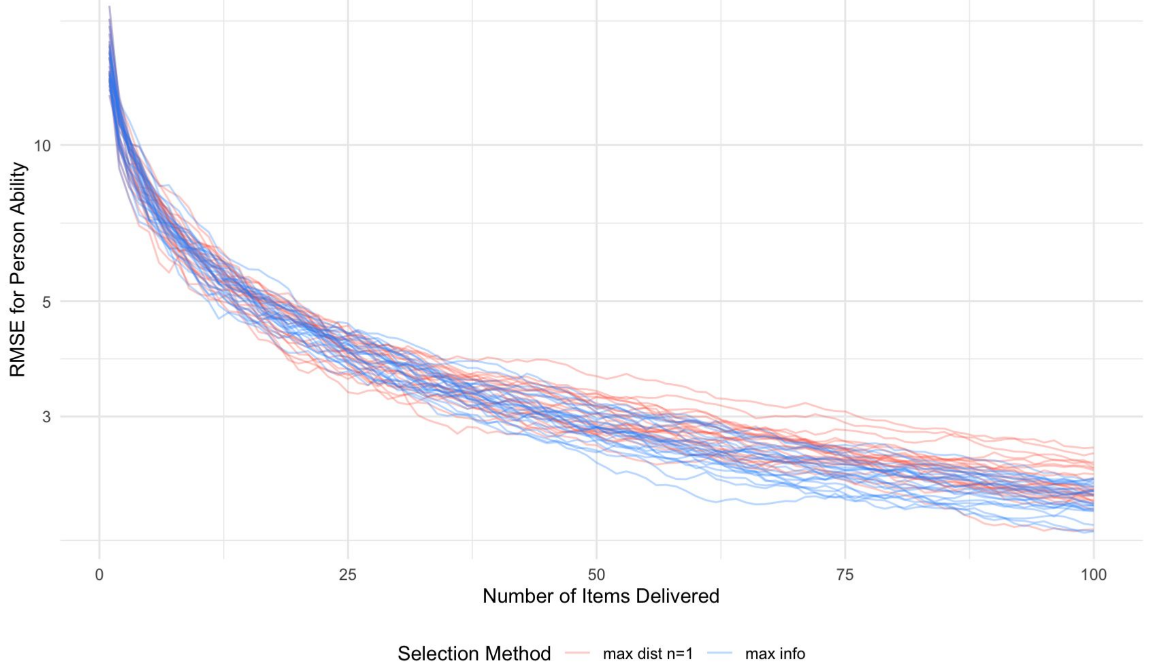
Simulation design

- Simulation conducted using meow to understand item selection patterns and potential negative effects on person parameter estimation efficiency
- Data loader is the built in data_simple_1pl():
    - 100 respondents with abilities drawn from N(0,1)
    - 200 items with difficulties drawn from N(0,1)
    - Responses generated according to a 1PL IRF
- Parameter update function is built in update_theta_mle()
    - Updates theta estimates using MLE
    - Treats item parameters as fixed and known

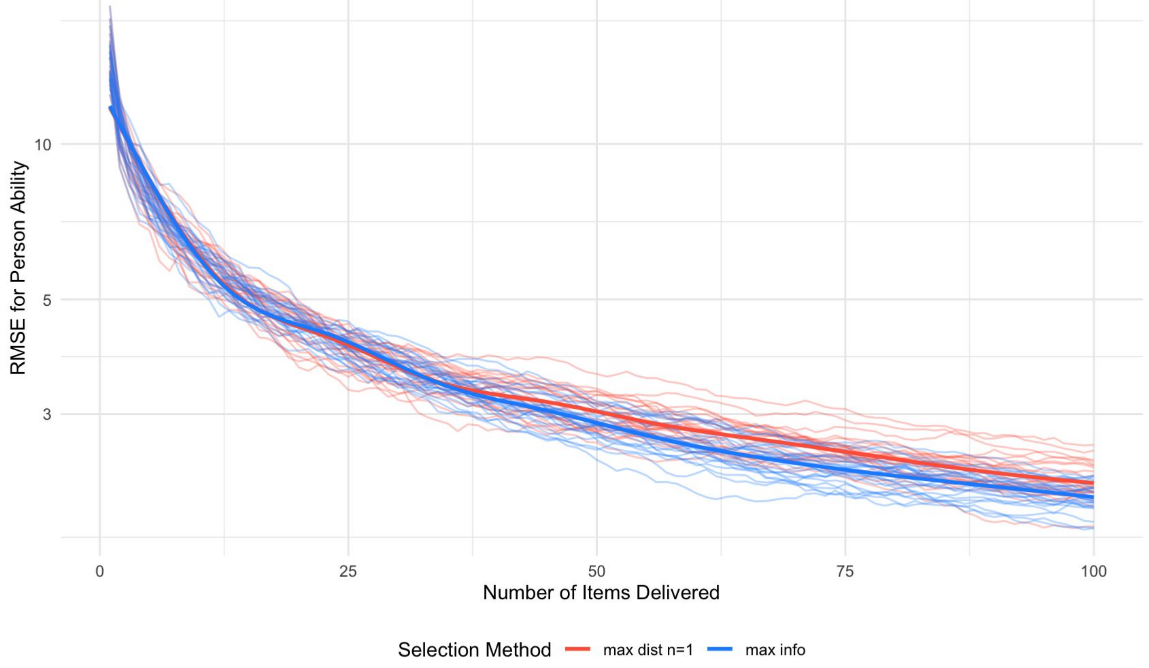
Simulation design

- Two item selection functions
- select_max_info() selects the next item to maximize information
- select_max_dist() selects the farthest item in the network from the items an individual has already responded to
  - Distances computed using Floyd-Warshall algorithm
  - Selects number of candidate items using the farthest 1, 3, 5, and 10 items
  - Ties broken using maximum information
- For each of the five conditions, 25 simulations were conducted with data consistent across conditions
- Each respondent starts with the same five items

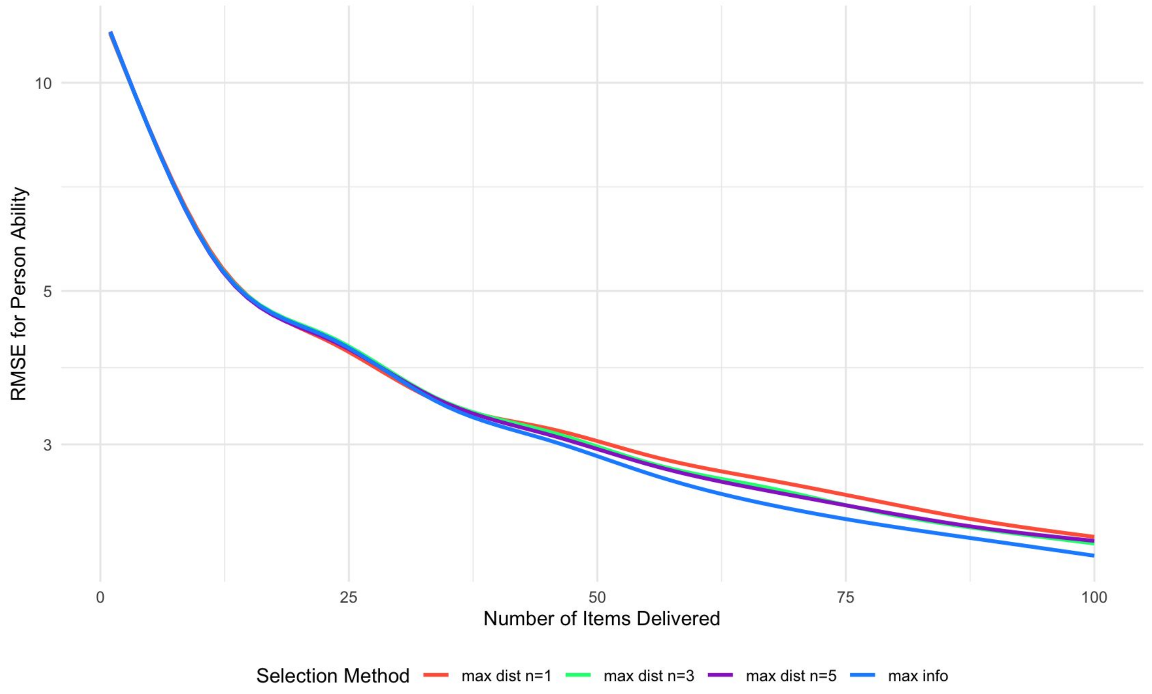
Max distance has higher RMSE



Max info pulls ahead after around 35 items



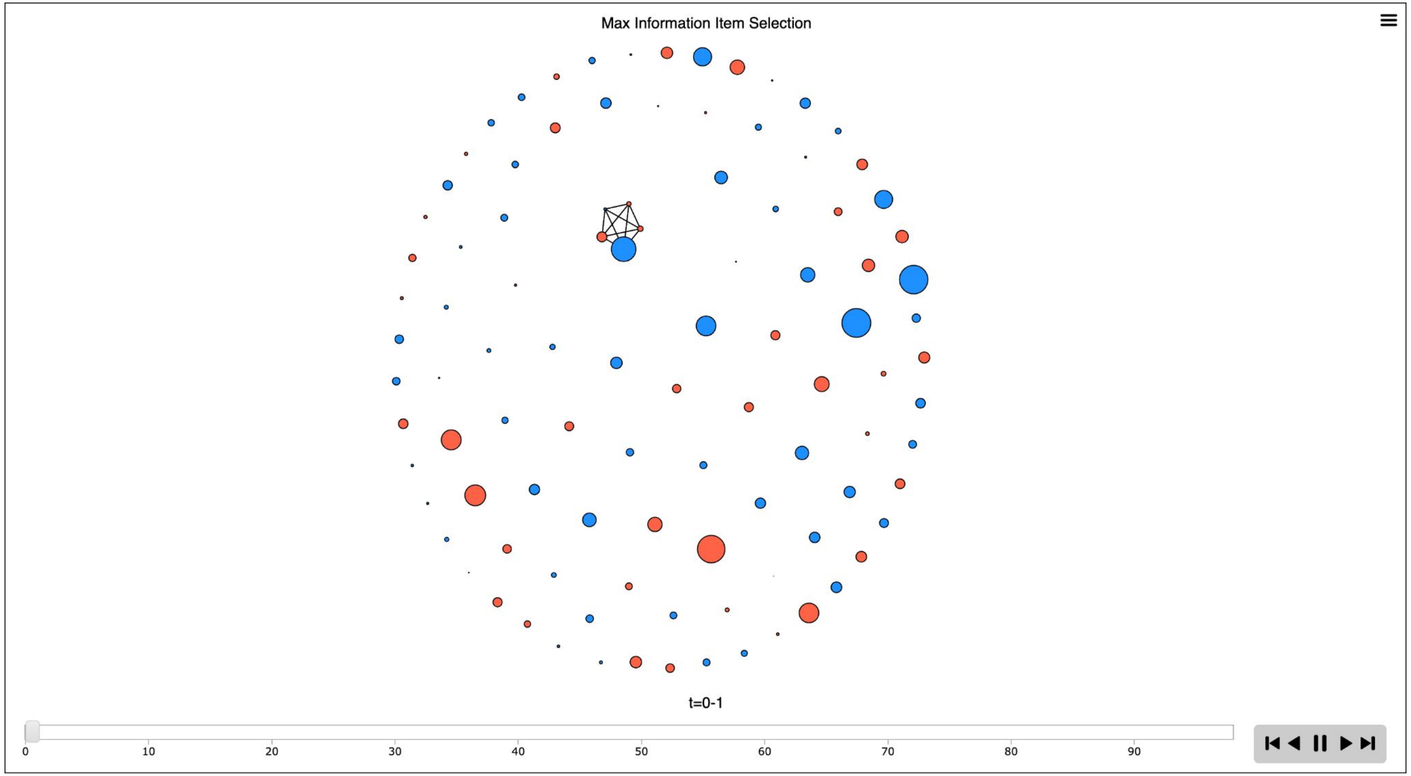
More candidate items closes the gap



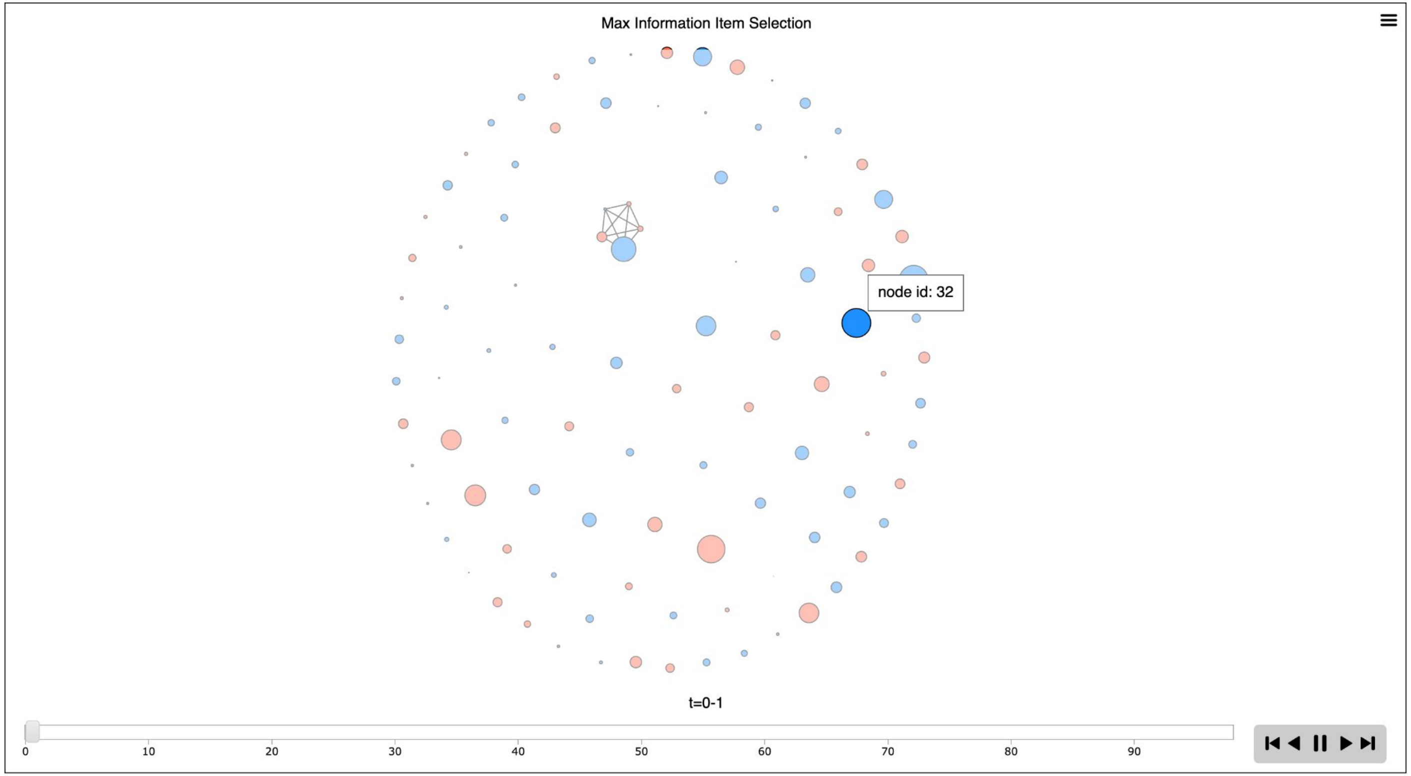
What about co-response patterns?

- Using statnet and ndtv, we construct dynamic visualizations of the item co-response network
- We're going to be looking at both item exposure and adjacency in the network
- This is normally a really interactive web visualization that is super helpful and takes 1-2 lines of code to build from standard meow ouput

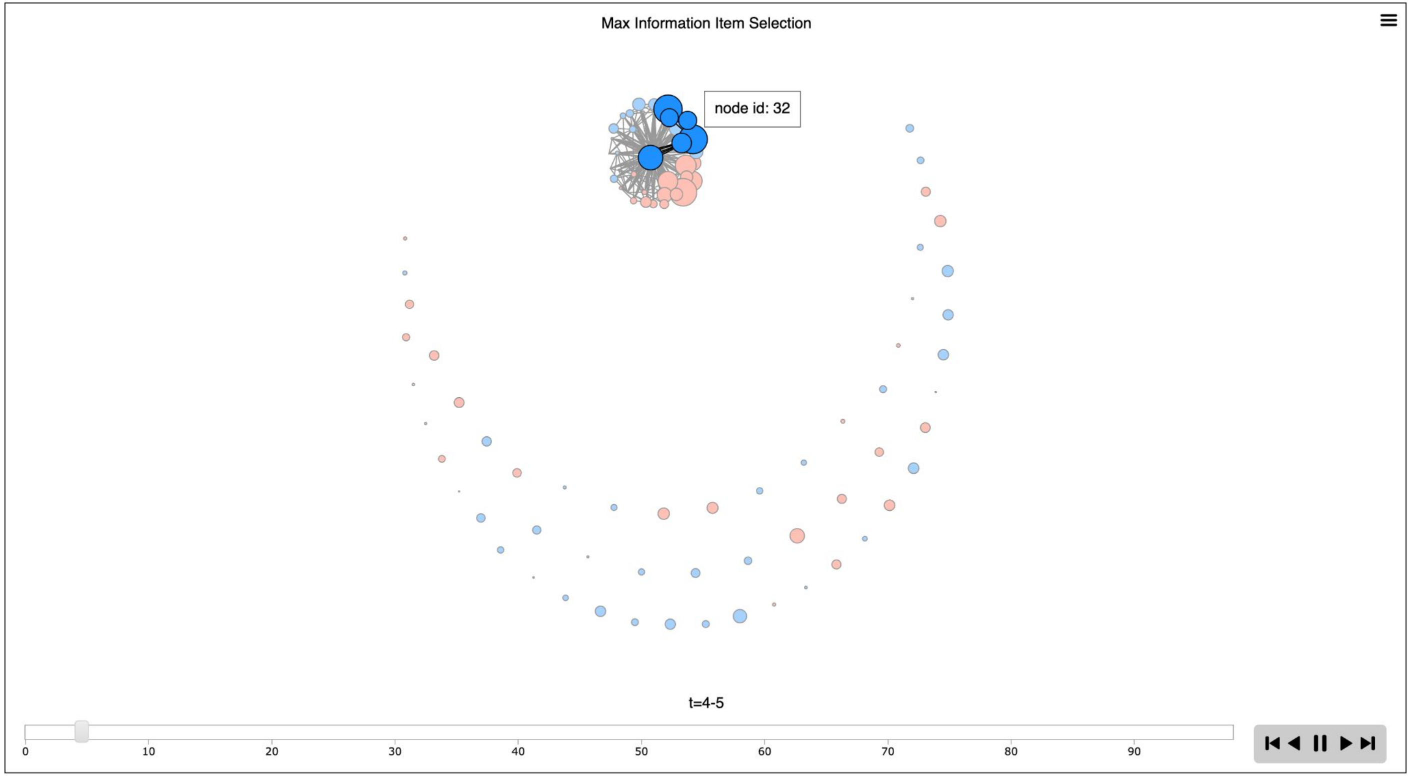
Max Info: t=1



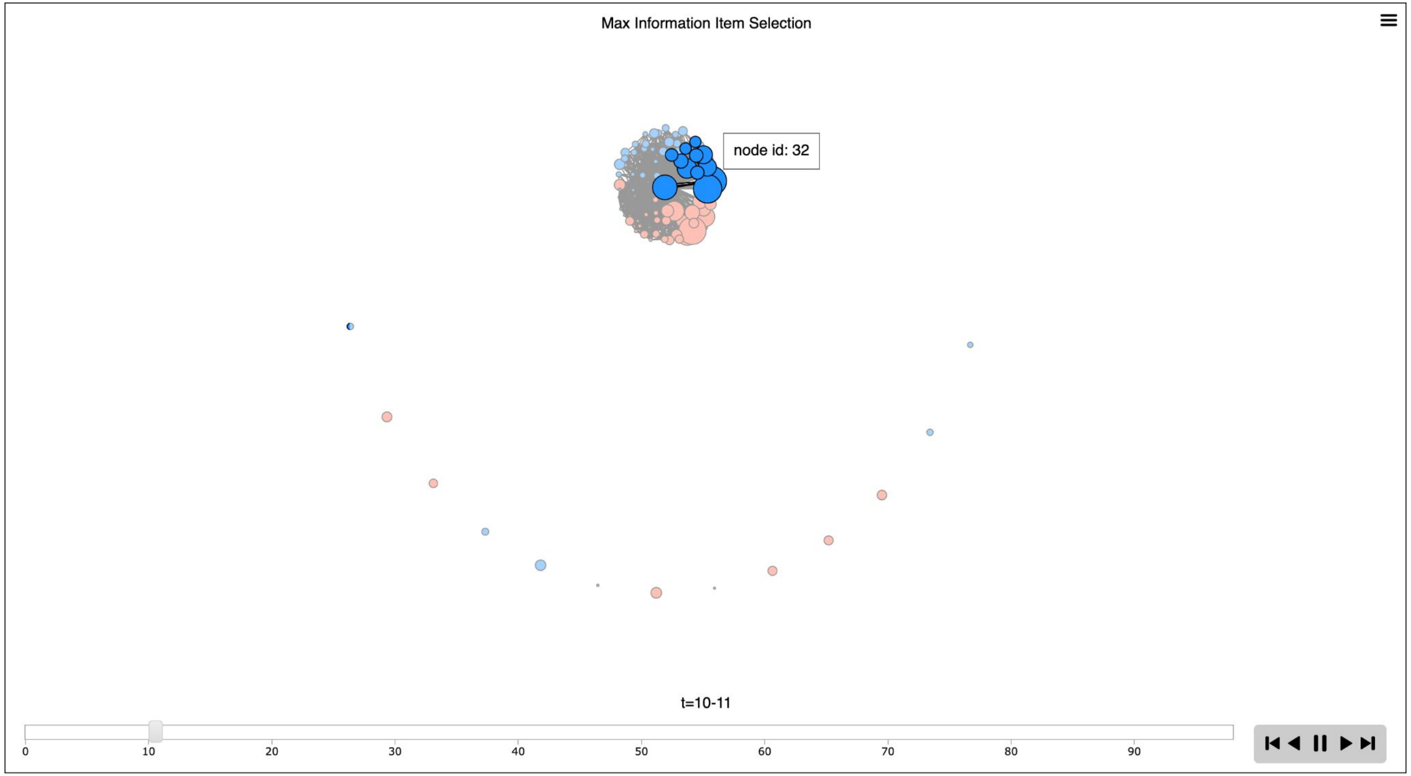
Max Info: t=1


Max Information Item Selection
node id: 32
t=0-1
0
10
20
30
40
50
60
70
80
90

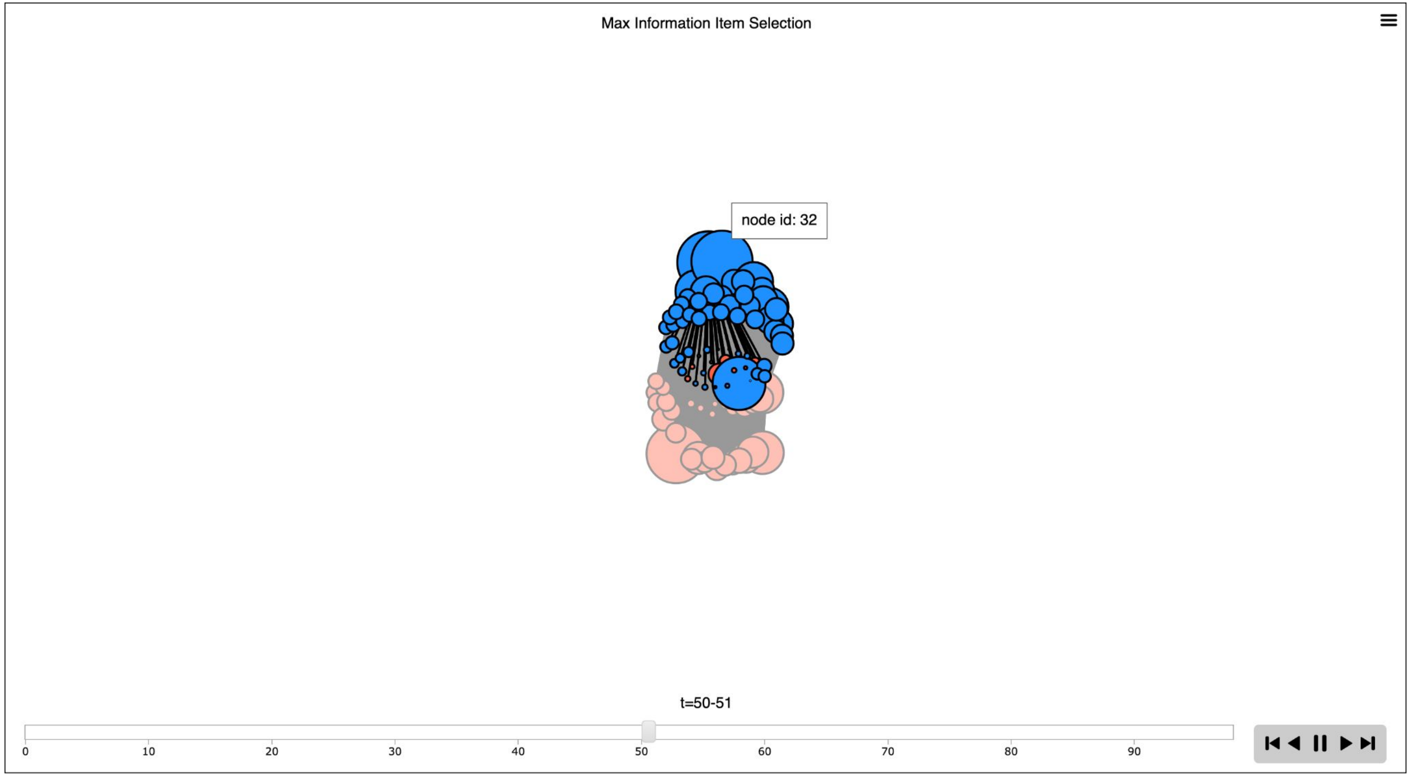
Max Info: t=5


Max Information Item Selection
node id: 32
t=4-5
0
10
20
30
40
50
60
70
80
90

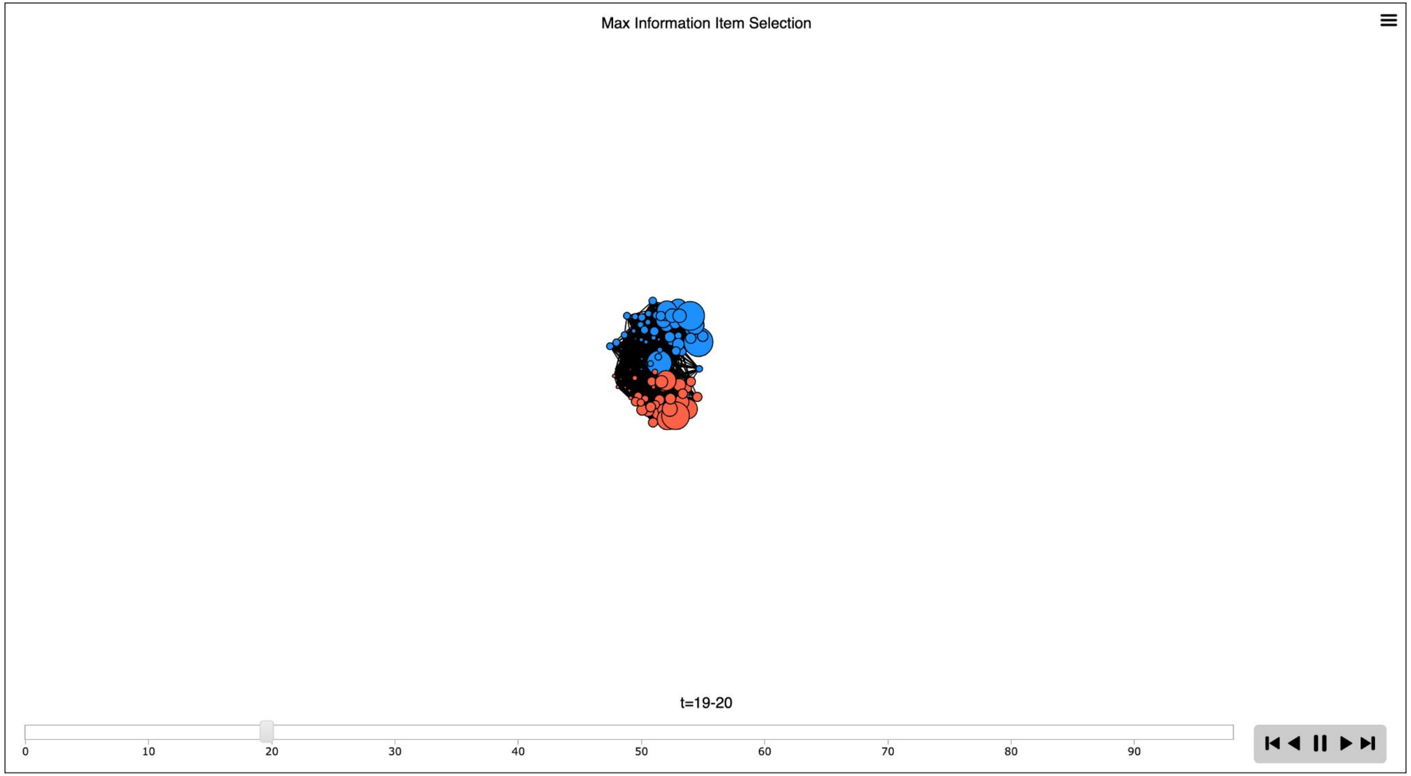
Max Info: t=11


Max Information Item Selection
node id: 32
t=10-11
0
10
20
30
40
50
60
70
80
90

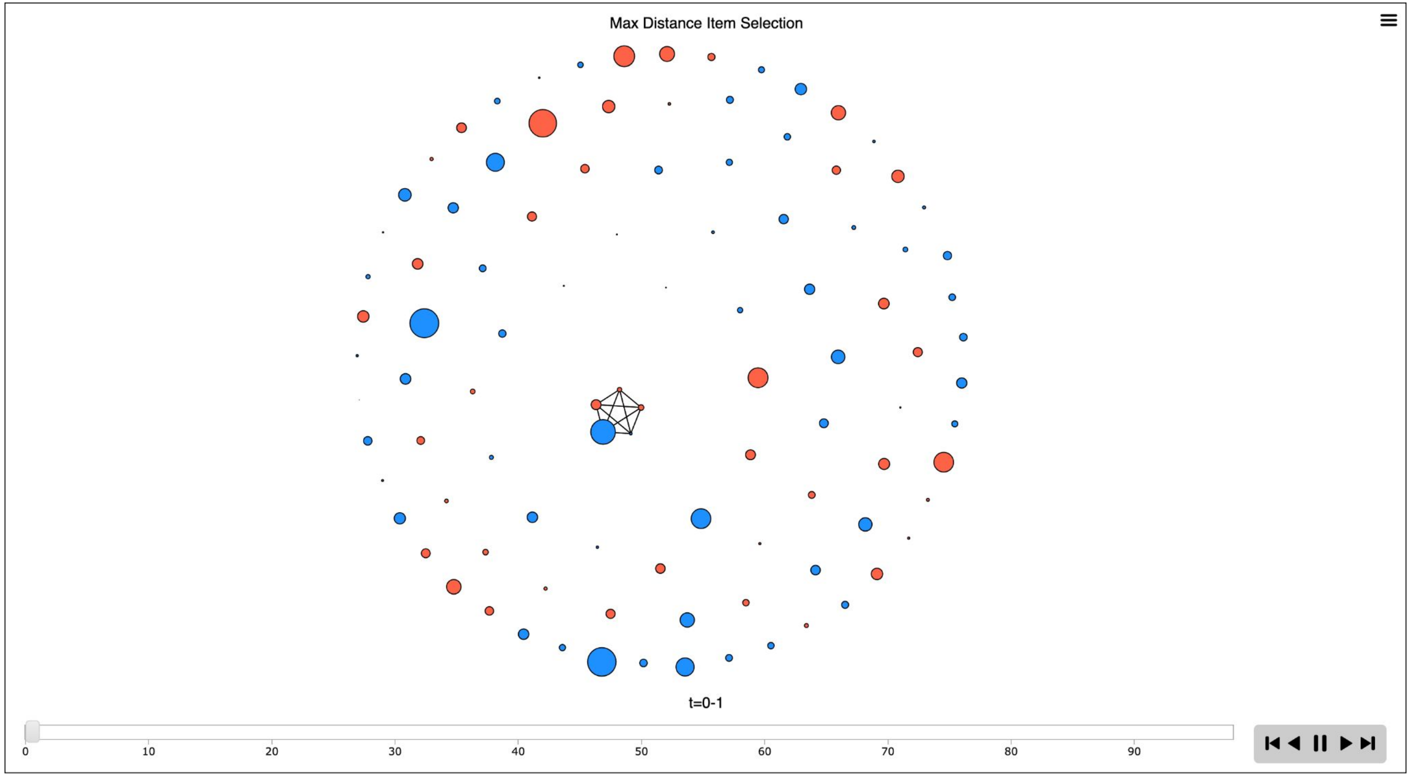
Max Info: t=51


Max Information Item Selection
node id: 32
t=50-51
0
10
20
30
40
50
60
70
80
90

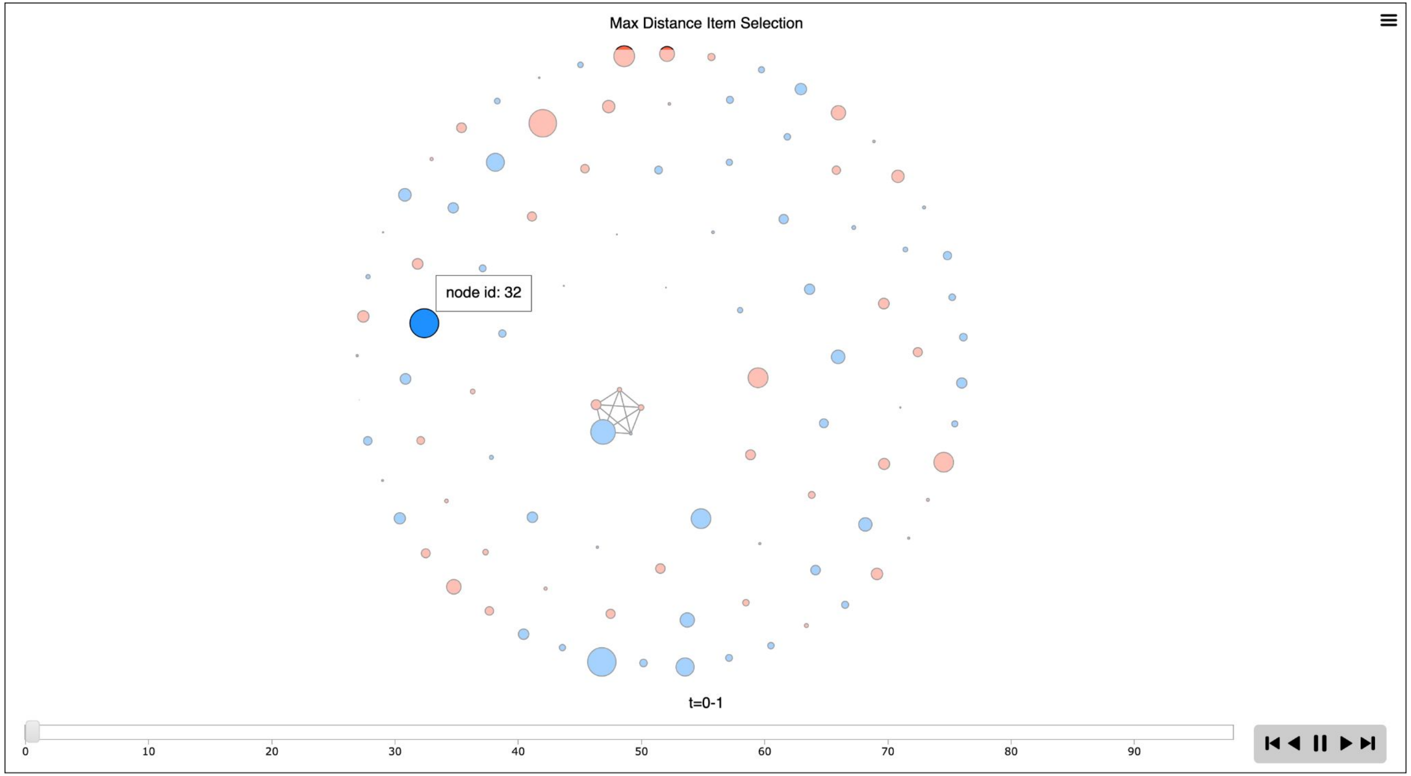
Max Info: All items exposed at t=20



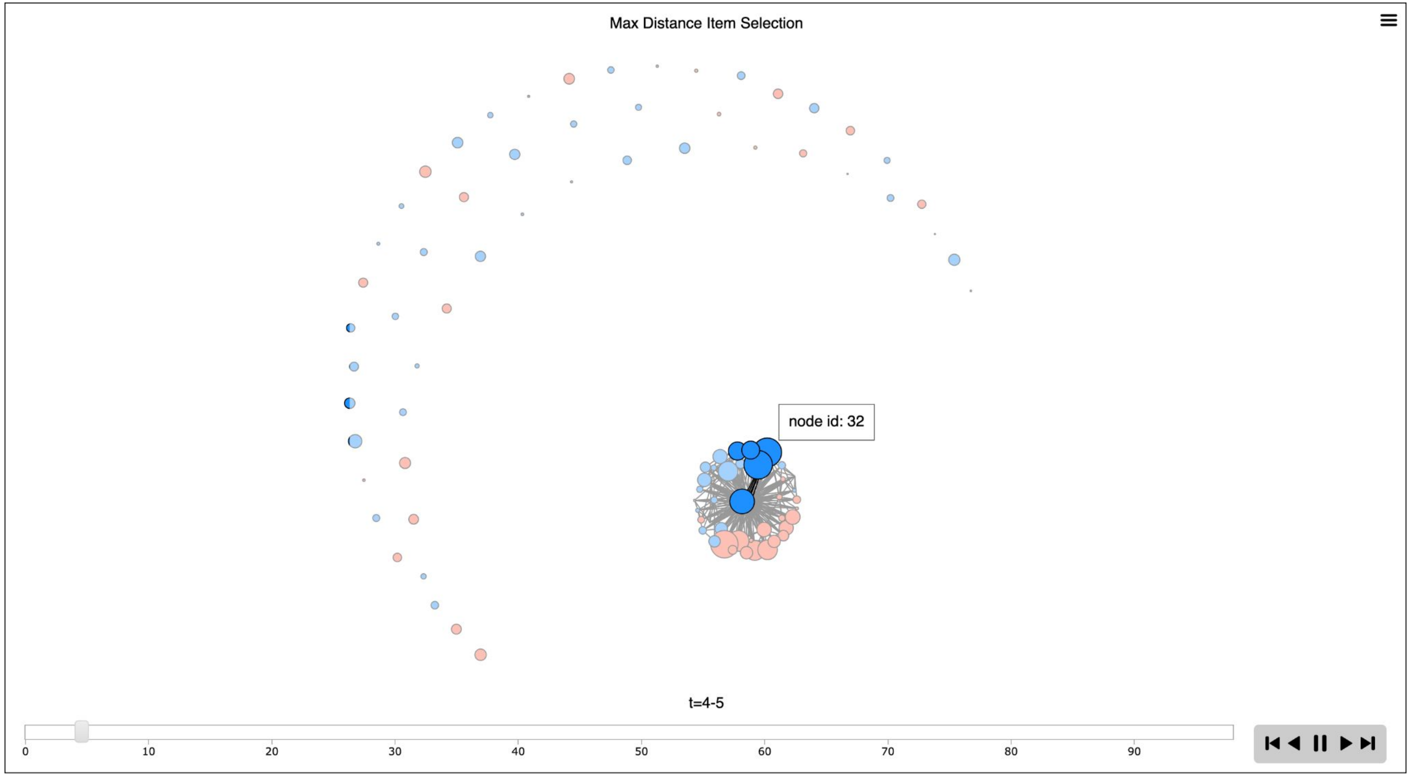
Max Dist: t=1



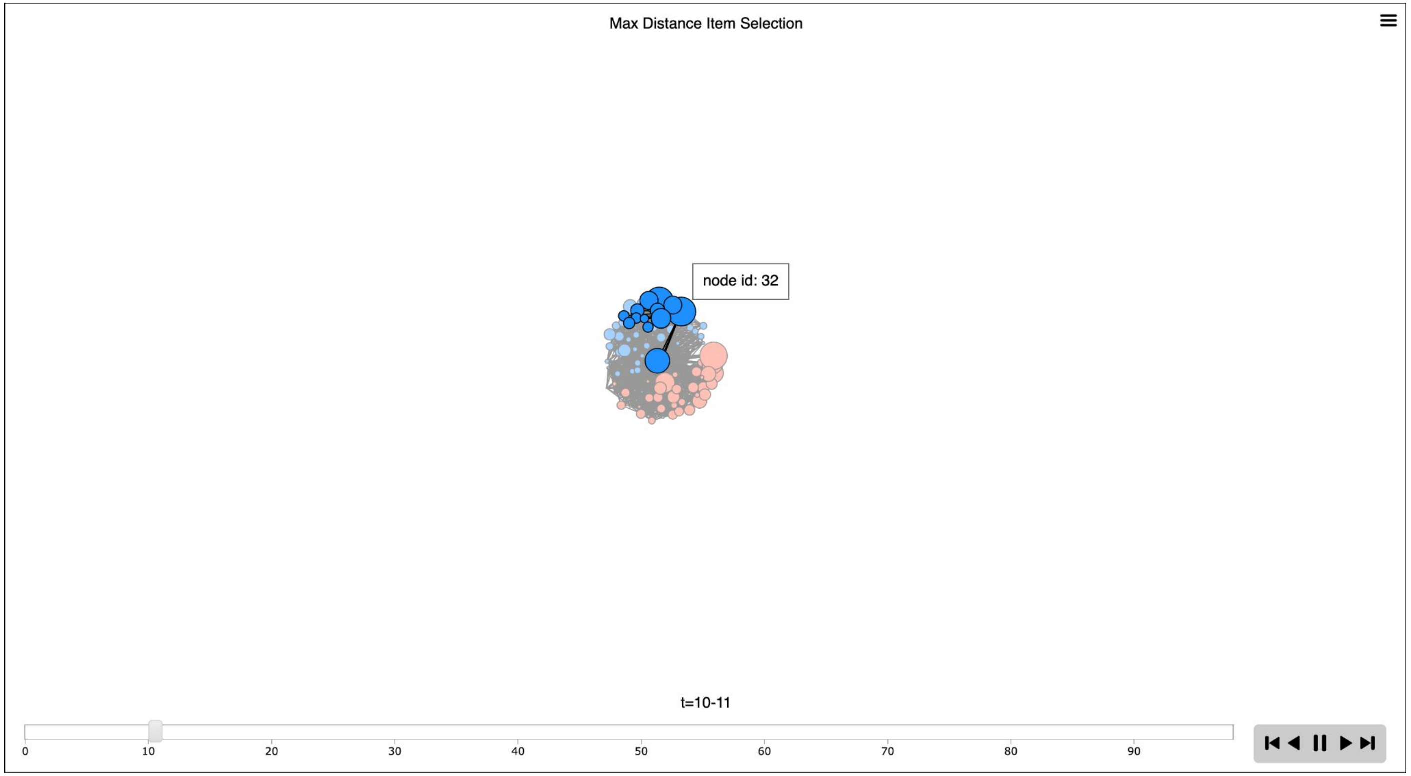
Max Dist: t=1


Max Distance Item Selection
node id: 32
t=0-1
0
10
20
30
40
50
60
70
80
90

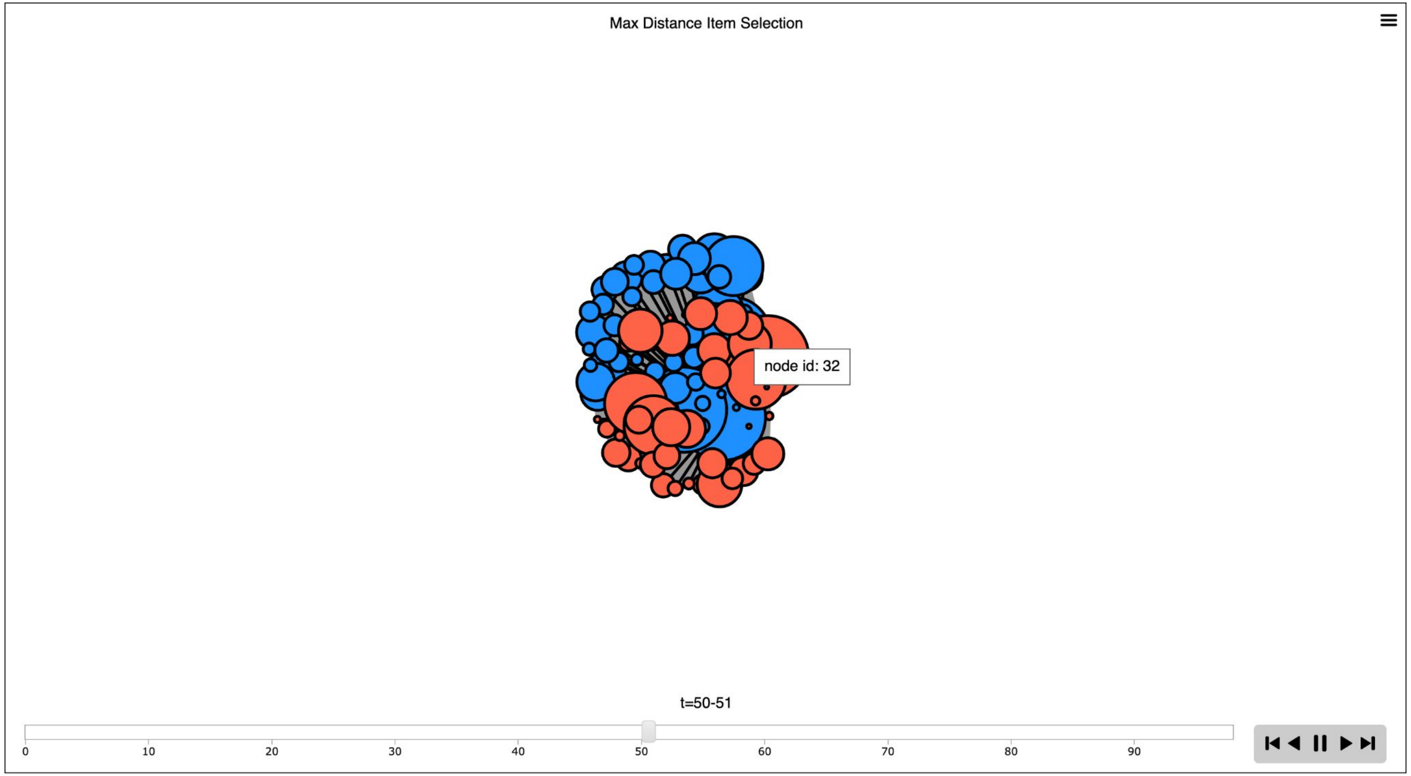
Max Dist: t=5


Max Distance Item Selection
node id: 32
t=4-5
0
10
20
30
40
50
60
70
80
90

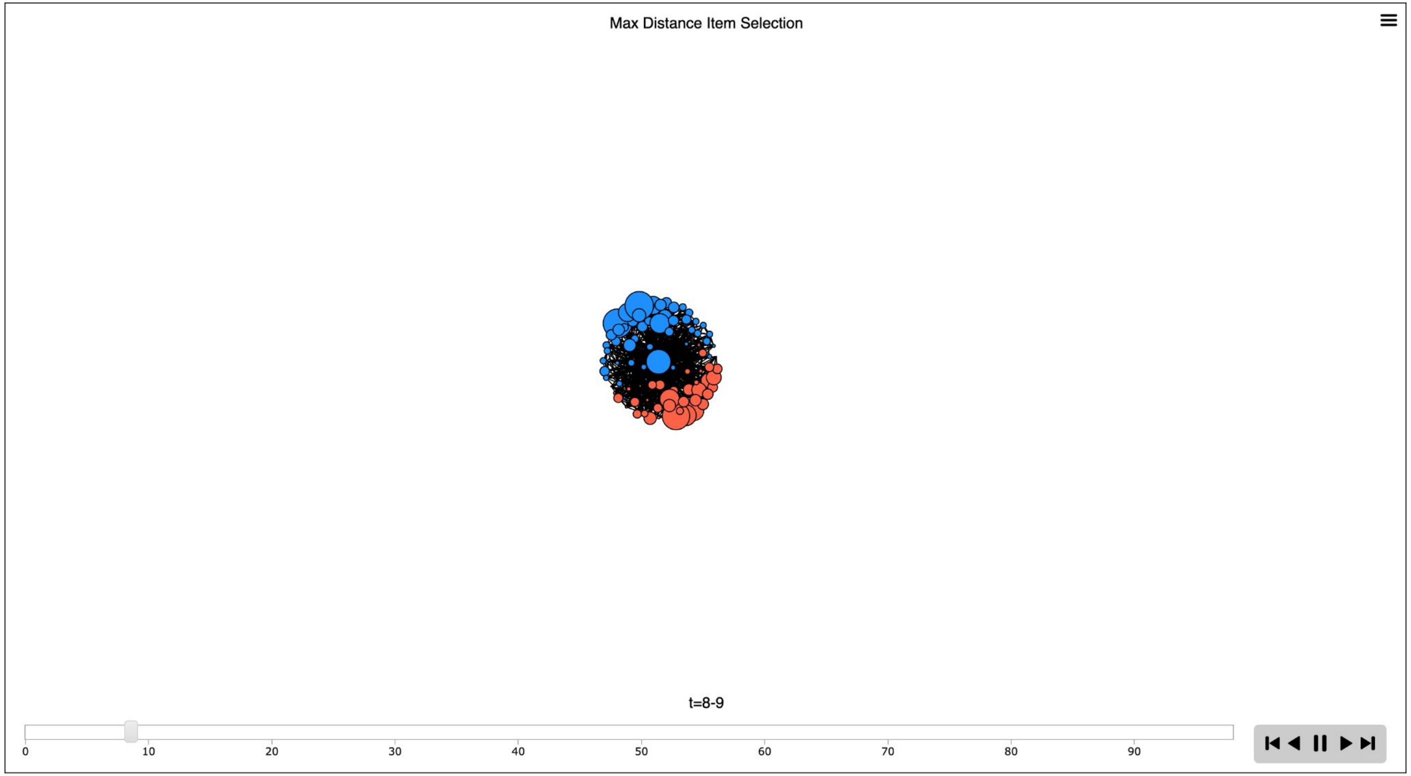
Max Dist: t=11


Max Distance Item Selection
node id: 32
t=10-11
0
10
20
30
40
50
60
70
80
90

Max Dist: t=51


Max Distance Item Selection
node id: 32
t=50-51
0
10
20
30
40
50
60
70
80
90

Max Dist: All items exposed at t=9



Discussion

Discussion

- Treating CAT item pools as networks allows for new tools to address item calibration and item usage
- Item selection that includes network distance metrics positively impact exposure patterns
  - Network density increases
  - More items are exposed
  - Network diameter, an upper bound on calibration accuracy, shrinks
- We are also preparing results from some network-augment parameter update algorithms! You'll see those in meow first!

Discussion

- The most exciting work in the CAT space is algorithmic advances that solve problems of item calibration and item usage within minimal tradeoffs in estimation efficiency
- meow provides an open source and easily extensible framework for conducting CAT simulations to compare algorithmic innovations
- You can use meow right now!

Thank you!

klint.kanopka@nyu.edu
klintkanopka.com
klint@bsky.social

References

- Bolsinova, M., Gergely, B., & Brinkhuis, M. J. (2025). Keeping Elo alive: Evaluating and improving measurement properties of learning systems based on Elo ratings. British Journal of Mathematical and Statistical Psychology.
- Gorney, K., Lee, C., & Chen, J. (2025). A Score-Based Method for Detecting Item Compromise and Preknowledge in Computerized Adaptive Testing. Journal of Computerized Adaptive Testing, 12(2).
- Gorney, K., & Reckase, M. D. (2025). Using Multiple Maximum Exposure Rates in Computerized Adaptive Testing. Journal of Educational Measurement.
- Vermeiren, H., Kruis, J., Bolsinova, M., van der Maas, H. L., & Hofman, A. D. (2025). Psychometrics of an Elo-based large-scale online learning system. Computers and Education: Artificial Intelligence, 8, 100376.