December, 2025

ATHENS course: **MOB_0AT09_TP**

Emergence in Complex Systems

Micro-study

teaching.dessalles.fr/ECS

This document contains students' contributions written during the Athens course *Emergence in Complex Systems* taught in November 2025.

We are very thankful to students for having be active participants, and also for the quality of their work.

Ada Diaconescu & Jean-Louis Dessalles

Contents

Christel	**Al Hage**	Modeling Ant Foraging Behavior	**37**
Amran	**Ameen**	Ant optimization of the Paris metro system	**15**
Mathieu	**Antonopoulos**	Cocktail Party improvements : make the party more fun	**25**
Pablo	**Bertaud--Velten**	Cocktail Party improvements : make the party more fun	**25**
Fares	**Boudelaa**		**89**
Riccardo	**Capellupo**	Ant optimization of the Paris metro system	**15**
Ana Margarida	**Cardoso Almeida**	Schelling Segregation with probabilistic multi-variable tolerance	**55**
Antoine	**Chedid**	Agent-Based Market Simulation	**5**
Hongxu	**Chen**	Modeling Ant Foraging Behavior	**37**
Christelle	**Clervilsson**	Modeling Ant Foraging Behavior	**37**
Arthur	**Coltro de Andrade**	Repeated Games In Small World Networks	**45**
Doga Selin	**Damar**	Modeling a more complex ant behavior	**75**
Nikola	**Dobricic**	Decentralizing Navigation	**81**
Yangtao	**Fang**	Social bubbles	**63**
Diego	**Fleury Corréa de Moraes**	Repeated Games In Small World Networks	**45**
Andrea	**Gasparini**	Agent-Based Market Simulation	**5**
Aghiles	**Gasselin**	Extention of segregation with city constraints	**31**
Vasil	**Georgiev**	Schelling Segregation with probabilistic multi-variable tolerance	**55**
Séthy	**Herlidou**	Schelling Segregation with probabilistic multi-variable tolerance	**55**
Mikhail	**Kataevskii**	Extention of segregation with city constraints	**31**
Tia	**Manoukian**	Decentralizing Navigation	**81**
Lucas	**Martim**	Repeated Games In Small World Networks	**45**
Alessa	**Mayer**	Modeling Ant Foraging Behavior	**37**
Margherita	**Necchi**	Decentralizing Navigation	**81**
Tymon	**Orlowski**	Extention of segregation with city constraints	**31**
Domenico Armando	**Palumbo**	Agent-Based Market Simulation	**5**
Sam	**Pegeot**	Social bubbles	**63**
Xinyi	**Pu**	Extention of segregation with city constraints	**31**

Josef	**Rabmer**	Schelling Segregation with probabilistic multi-variable tolerance	**55**
Andrea	**Scalmato**	Agent-Based Market Simulation	**5**
Guilherme	**Soeiro de Carvalho Caporali**	Repeated Games In Small World Networks	**45**
Frantisek	**Spacek**	Social bubbles	**63**
Stepan	**Svirin**	Cocktail Party improvements : make the party more fun	**25**
Jan	**Svoboda**	Social bubbles	**63**
Miguel	**Veganzones Parellada**	Ant optimization of the Paris metro system	**15**
Dennis	**Waniek**	Modeling a more complex ant behavior	**75**
Ada	**Yetis**	Decentralizing Navigation	**81**
Emine Goksu	**Yildiz**	Modeling a more complex ant behavior	**75**
Yufei	**Zhou**	Cocktail Party improvements : make the party more fun	**25**
Hester	**Zoet**	Ant optimization of the Paris metro system	**15**

November, 2025

ATHENS course : **TP-09**

Emergence in Complex Systems

Micro-study

teaching.dessalles.fr/ECS

Name: **Antoine Chedid; Andrea Gasparini; Domenico Armando Palumbo; Andrea Scalmato**

Agent-Based Market Simulation

Abstract

This paper investigates the emergence of complex market phenomena in an agent-based simulation framework. Through explicit peer-to-peer transactions between autonomous agents following simple local rules, we demonstrate how macroscopic market properties—including price stability, market bubbles, and crashes—emerge from the collective interactions of heterogeneous trading strategies. Our model reveals that market efficiency, volatility patterns, and dynamic equilibria are not imposed top-down but rather emerge from the composition and proportions of different agent types within the population. These findings provide insights into how simple behavioral rules at the individual level can generate the complex and often unpredictable dynamics observed in real financial markets.

Problem Statement

Financial markets are one of the most complex systems in modern society, involving diverse agents whose behaviors range from perfectly rational to purely speculative. This heterogeneity makes predicting market dynamics particularly challenging.

Despite this complexity, certain macroscopic phenomena recur consistently across different markets and time periods. Market bubbles—situations where asset prices deviate significantly from their fundamental intrinsic values—are an example of such emergent behaviors. These bubbles are not orchestrated by central authorities but arise from the collective actions of individual market participants, each following their own decision rules.

This study seeks to identify the key factors and agent behaviors that influence the emergence of these macroscopic market phenomena.

Specifically, we investigate:

1. How do different trading strategies interact to produce stable or unstable market dynamics?
2. What agent compositions lead to efficient markets versus bubble formation?
3. Can we identify critical thresholds where market behavior undergoes qualitative changes?

Methodology

To study emergence phenomena in financial markets, we developed an agent-based model that captures essential market dynamics while maintaining analytical tractability. Our simulation implements a simplified market environment focused on a single asset, with explicit peer-to-peer transaction matching between agents.

Agent Types and Behavioral Rules

The model incorporates three distinct agents, each implementing a different trading philosophy:

- **Fundamental Traders**: These value-driven agents embody rational economic behavior. They have a belief about the asset's intrinsic fair value and trade to drive market prices toward this perceived fundamental value. When they observe the market price below their fair value estimate, they buy (identifying undervaluation); when prices exceed fair value, they sell (identifying overvaluation). This creates stabilizing pressure towards equilibrium prices.
- **Momentum (Follower) Traders**: These trend-following agents amplify existing price movements through positive feedback mechanisms. They analyze recent price history over a lookback window; when prices show upward momentum, they buy (expecting continuation); when momentum turns negative, they sell. This behavior can destabilize markets by reinforcing price trends, potentially creating bubbles during upswings and crashes during downturns.
- **Noise Traders**: These agents introduce stochastic elements that simulate market frictions and unpredictable trading activity. With fixed probability, they randomly decide to buy or sell regardless of price levels, fair values, or trends. While their individual impact is limited, collectively they prevent perfect market efficiency and model the influence of uninformed or irrational market participants.

Market Mechanics

Each agent is initialized with endowments of both cash (liquidity) and shares (stock holdings), creating realistic budget constraints. At each simulation time step, agents independently evaluate market conditions and decide whether to buy, sell, or hold. Critically, trades are only executed when a willing buyer is matched with a willing seller—there is no implicit market maker pro- viding infinite liquidity. This peer-to-peer matching mechanism ensures that price discovery emerges naturally from supply-demand dynamics rather than being imposed externally.

The current market price is determined by the most recent successfully executed transaction, reflecting real-time supply-demand equilibrium. This mechanism allows prices to respond dynamically to shifts in agent behavior and composition.

Implementation Details

Agent Decision Algorithms

- **Fundamental Traders**: Each agent's fair value belief is sampled from a Gaussian distribution centered on the true fundamental value with a standard deviation of 10%. They compute a mispricing ratio as *(fair_value - market_price) / fair_*value. When this ratio exceeds a positive threshold θ, they interpret the asset as undervalued and buy; when less than θ, they sell (overvalued). The threshold parameter controls their trading aggressiveness.
- **Momentum (Follower) Traders**: They calculate momentum by averaging market prices over a lookback window of recent time steps. This moving average captures trend direction. If momentum exceeds a threshold (indicating strong upward trend), they buy; if momentum falls below the negative threshold (downward trend), they sell. The lookback window length determines their responsiveness to price changes.
- **Noise Traders**: With probability p at each time step, they enter a trade. Upon entry, they randomly select buy or sell with equal 50% probability, independent of any market signals. The trade probability p controls overall noise level in the system.

Experimental Scenarios

We conducted experiments under two primary scenarios:

- **Efficient Market Scenario**: A population of 80 fundamental traders and 20 noise traders, designed to test whether rational agents can maintain market efficiency. We also examined this scenario under exogenous shocks, shifting the true fair value mid-simulation to observe adaptation dynamics.
- **Heterogeneous Market Scenario**: Mixed populations with varying proportions of all three agent types, enabling investigation of how agent composition influences emergent market phenomena such as bubbles, crashes, and dynamic equilibria.

Throughout all experiments we systematically varied key parameters including agent proportions, decision thresholds, and trade fractions to identify conditions that determine the emergence of different macroscopic market behaviors.

Results and Analysis

Efficient Market

In our first experiment, we simulated a market populated of 80 fundamental traders and 20 noise traders, with the fundamental value set at 100. This configuration was designed to test the hypothesis that rational agents can collectively drive a market toward efficiency.

Figure 1 demonstrates an emergent phenomenon: after an initial transient period of approximately 25 time steps, the market price stabilizes near the fundamental value. This convergence occurs without central coordination: it emerges purely from individual agents following their local trading rules. The slight oscillations around the fair value are attributable to noise trader activity, which prevents perfect price stability but maintains prices within a narrow band around the fundamental value.

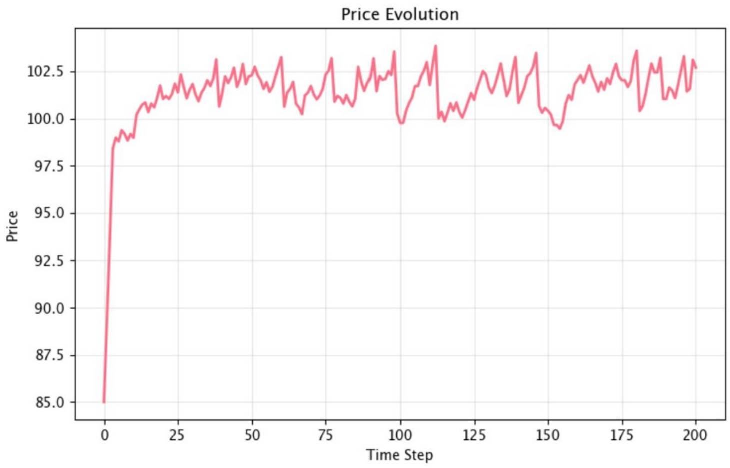


Figure 1: Price evolution in a market with 80 fundamental traders and 20 noise traders (fair value = 100). The market price (red line) converges to the fundamental value (100) after an initial transient period. This demonstrates emergent market efficiency through rational agent behavior.

KEY INSIGHT: Price stabilization emerges spontaneously when rational agents dominate the market, demonstrating how local optimization behaviors can produce a global system-level efficiency.

Market Adaptation to Fundamental Value Shifts

To test the market's adaptive capacity, we introduced an exogenous shift at time t = 250 by abruptly shifting the fundamental value from 100 to 120. This simulates real-world events such as a company launching a breakthrough product or positive regulatory changes that enhance asset value.

Figure 2 reveals the market's remarkable self-correcting properties. Following the value shift, agents whose fair value beliefs were distributed around the old equilibrium suddenly perceive massive undervaluation. This triggers intense buying activity, driving prices upward. After a transition period, the market establishes a new equilibrium near the updated fundamental value of 120. The speed and smoothness of this transition demonstrate the system's robustness to external shocks.

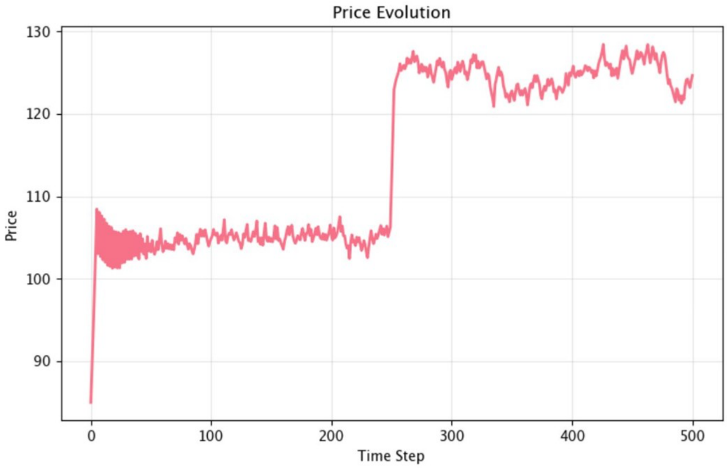


Figure 2: Market response to fundamental value shift at t = 250 (from 100 to 120). The market price initially remains near the old equilibrium but rapidly adjusts once agents detect the mispricing. This demonstrates emergent adaptive behavior in response to structural changes.

KEY INSIGHT: Markets composed primarily of rational traders exhibit self-correcting dynamics, automatically realigning to new fundamental values after transient adjustment periods.

Bubble Formation and the Role of Momentum Traders

Having established baseline efficient market behavior, we next investigated how introducing trend followers (also called momentum traders) alters market dynamics. Momentum traders' positive feedback mechanism—buying when prices rise and selling when they fall—has the potential to destabilize markets by amplifying price movements beyond fundamental values, causing a snowball effect.

Critical Transitions in Agent Composition

Our experiments revealed three distinct regimes of market behavior dependent on momentum trader proportion:

- Low Momentum Regime (<10% momentum traders): The market behaves similarly to the pure fundamental trader case, with prices converging to and oscillating around the fair value. Momentum traders are too few to overcome fundamental traders' stabilizing influence.
- Intermediate Regime (10-30% momentum traders): A fascinating emergent phenomenon appears: the market establishes a dynamic equilibrium at a price slightly offset from the fundamental value. Small perturbations trigger oscillations that neither explode into bubbles nor decay to fundamental value. The system exhibits bounded non-periodic behavior around this shifted equilibrium.
- High Momentum Regime (>30% momentum traders): Full bubble-crash cycles emerge. Prices undergo explosive growth far above fundamental value, followed by rapid collapses. Trading volume spikes dramatically during crashes, reflecting panic selling.

Figures 3 and 4 illustrate the contrast between the intermediate and high momentum regimes. With 20% momentum traders (Figure 3), we observe a dynamic equilibrium state with controlled oscillations around a price level slightly above the fair value. With 40% momentum traders (Figure 4), the system produces dramatic boom-bust cycles with peak prices before crashing back down.

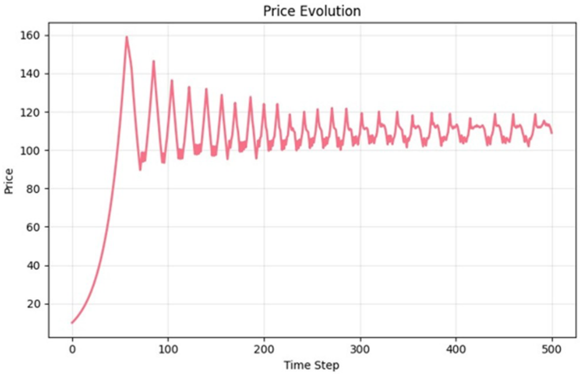


Figure 3: Market dynamics with 20% momentum traders, 60% fundamental traders, and 20% noise traders (fair value = 100). The system shows a dynamic equilibrium with controlled oscillations around a price level moderately above the fair value. This regime represents a critical transition zone between efficient markets and bubble formation.

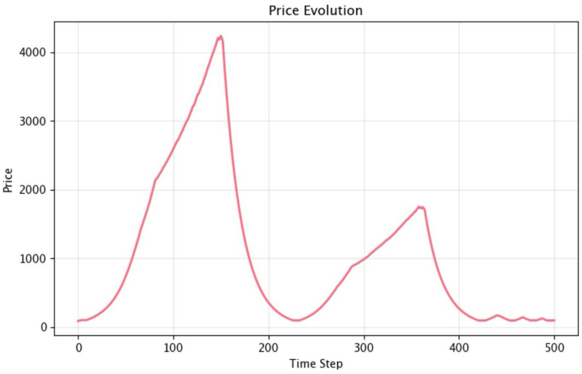


Figure 4: Bubble market dynamics with 40% momentum traders, 40% fundamental traders, and 20% noise traders (fair value = 100). The market undergoes repeated boom-bust patterns Each cycle exhibits: (1) gradual price inflation as momentum builds, (2) rapid acceleration near the peak, (3) quick crash when momentum reverses, and (4) brief stabilization before the next cycle begins.

Markets exhibit qualitatively different emergent behaviors based on agent composition. Small changes in momentum trader proportion can trigger phase transitions from stable efficiency to dynamic equilibria to extreme bubble-crash cycles. By adjusting population parameters, we can control both the amplitude and frequency of market instabilities.

Mechanistic Analysis: Trading Activity During Bubbles

To understand the dynamics underlying bubble formation, we decomposed trading activity by agent type during the bubble-crash cycle shown in Figure 4. Figures 5 and 6 reveal the complementary behaviors driving market instability.

During the initial bubble inflation phase (time steps 0-150), momentum traders (green) exhibit intense buying activity as prices rise and positive momentum builds. Simultaneously, fundamental traders (pink) increasingly sell, recognizing that prices have deviated far above the fair value. At the bubble peak, momentum traders dominate the buy side while fundamental traders dominate the sell side. The crash initiates when momentum traders exhaust their buying power or momentum indicators turn negative. This triggers a cascade: momentum traders reverse to selling (first green peak in Figure 5), amplifying the price decline. As prices collapse below the fair value, fundamental traders switch from selling to buying (first pink peak in Figure 6), eventually stabilizing the market before the cycle repeats.

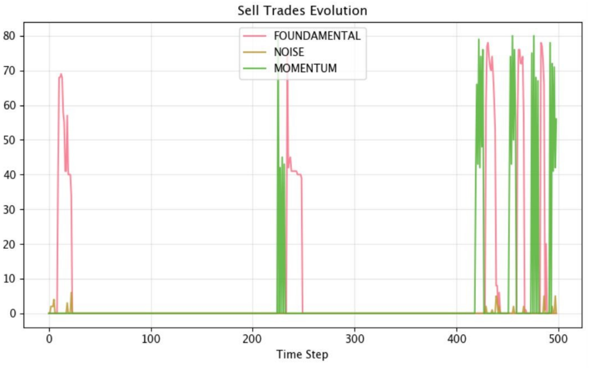


Figure 5: Selling activity decomposed by agent type during the bubble scenario (corresponding to Figure 4). Pink bars show fundamental trader selling, which peaks early in the bubble as they recognize overvaluation. Green bars show momentum trader selling, which spikes dramatically during the crash as negative momentum triggers mass exits. Orange line indicates noise trader activity, which remains relatively constant throughout. The timing mismatch between fundamental and momentum selling drives the boom-bust asymmetry.

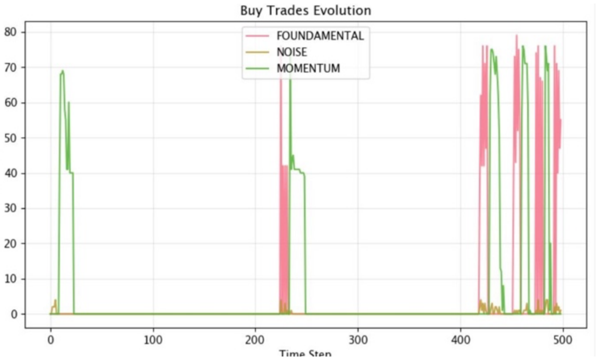


Figure 6: Buying activity decomposed by agent type during the bubble scenario. Green bars show momentum trader buying, which dominates during bubble inflation as positive feedback drives continued purchases. Pink bars show fundamental trader buying, which is suppressed during the bubble (as prices are overvalued) but spikes after the crash when prices fall below fair value. This pattern reveals how opposing strategies create cyclical market dynamics: momentum traders fuel bubbles while fundamental traders provide post-crash support. Orange line indicates noise trader activity, which remains relatively constant throughout.

Comparative Performance of Trading Strategies

Beyond understanding emergent market dynamics, our model enables quantitative comparison of different trading strategies. Figure 7 shows the final wealth distribution after 500 time steps in the high-momentum bubble scenario (40% momentum traders, Figure 4).

Remarkably, momentum traders perform worse than even random noise traders in this scenario. While momentum traders successfully ride the bubble upward, they fail to exit before crashes, typically selling after prices have already collapsed. This "buy high, sell low" pattern—the opposite of profitable trading—results from their backward-looking momentum indicators, which identify trends only after they've already begun. They amplify bubbles but capture none of the profits, leaving them worse off than agents trading randomly.

In contrast, fundamental traders achieve the highest final wealth. By consistently buying below fair value and selling above it, they profit both from their stabilizing trades during bubble formation and from buying opportunities created by crashes. This demonstrates that value-based strategies can outperform in volatile markets, even when they fail to prevent bubbles.

KEY INSIGHT: Agents who contribute most to market instability (momentum traders) suffer the worst financial outcomes. This demonstrates a disconnect between market impact and individual profit.

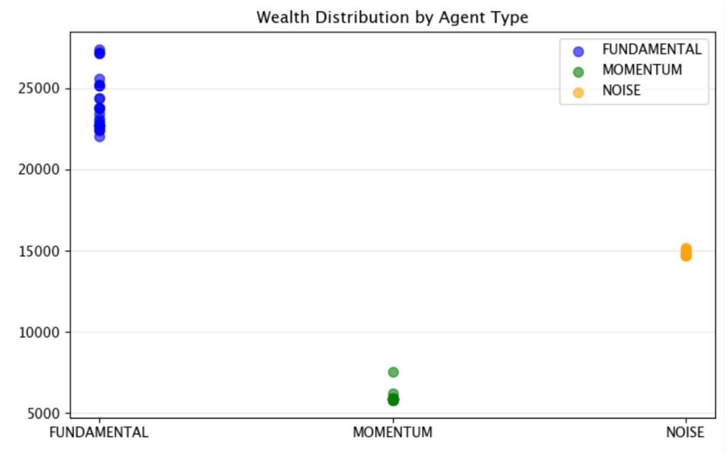


Figure 7: Final wealth distribution by agent type after 500 time steps in the bubble scenario (all agents started with initial wealth of 10000). Fundamental traders (blue) achieve the highest median wealth. Momentum traders (green) show the lowest median wealth, with many agents suffering significant losses. Noise traders (orange) perform intermediately, with outcomes centered near initial wealth. This illustrates that in heterogenous markets, with several trend followers, value-based strategies dominate trend-following approaches while contributing less to bubble formation.

Future Work

While our model successfully demonstrates key emergence phenomena, several extensions could enhance realism and uncover additional emergent behaviors:

Heterogeneous Beliefs

Currently, fundamental traders' fair value beliefs cluster tightly around the true fundamental value (±10% standard deviation). Introducing greater heterogeneity—where agents hold radically different valuation estimates—would create richer price discovery dynamics. This could reveal how disagreement among 'rational agents generates trading volume and volatility even in the absence of momentum traders or noise.

Intra-Timestep Price Dynamics

Our current model updates prices only once per time step, based on a single representative transaction. Real markets feature continuous price discovery with each transaction potentially moving prices. Implementing dynamic intra-timestep pricing—where early transactions within a time step influence later ones—would better capture phenomena like flash crashes, liquidity cascades, and order book dynamics.

Adaptive Strategies and Learning

Currently, agents follow fixed strategies throughout the simulation. Introducing adaptive agents who learn from experience—adjusting their thresholds, lookback windows, or even switching strategies based on past performance—could generate evolutionary market dynamics. This would address the question of whether certain strategies survive in the long run and whether markets evolve toward or away from efficiency over extended time horizons.

Conclusions

This study demonstrates that complex market phenomena—including efficiency, bubbles, crashes, and dynamic equilibria—emerge naturally from interactions between agents following simple local rules. No central coordination or top-down market maker is required; these properties arise organically from the composition and proportions of different trading strategies within the population.

Our results reveal critical phase transitions where small changes in agent composition produce qualitative shifts in market behavior. Markets dominated by rational fundamental traders converge to efficiency; moderate momentum trader presence creates dynamic equilibria offset from fundamental value; and high momentum trader proportions generate severe boom-bust cycles. These findings suggest that market stability critically depends on the diversity and balance of trading strategies.

Finally, our model shows that agents who contribute the most to market instability often suffer the worst financial outcomes. This emergent property has implications for market design, regulation, and our understanding of financial crises as self-organized phenomena arising from decentralized interactions rather than coordinated manipulation.

November, 2025

ATHENS course : **MOB_0AT09_TP**

Emergence in Complex Systems

Micro-study

teaching.dessalles.fr/ECS

Name: **Amran Ameen, Riccardo Capellupo, Miguel Veganzones and Hester Zoet**

AntNet application on Métro de Paris

Abstract

This project aims to find an alternative connection between stations for optimizing for pasenger throughput of the Métro de Paris. This will be done through the bio-inspired path-finding AntNet algorithm (a decentralized, agent-based path-finding model) as opposed to more traditional methods based in the Centrality Analysis metrics. The goal is to determine if AntNet's local optimization result aligns with the static structural importance derived from established network science methods and its resemblance with the connections of the real Métro de Paris network.

Introduction

The work of AntNet by Schoonderwoerd et al. in 1997 [1] presents a dynamic routing algorithm based on the principles of Ant System introduced by Appleby and Steward in 1994 [2]. Unlike the original model, which simulated ants traveling randomly to find the closest food source, AntNet employs simple localized agents ("ants") to modify the routing tables of every node within a network.

The AntNet algorithm

The core principle of AntNet involves agents traveling asynchronously through the network. An agent's age upon arrival at a node reflects the traversal time of the path it took from its origin. The algorithm calculates an optimal path as follows: agents (ants) are initialized and launched randomly. When they reach a node, they update the local routing table by considering their source as the destination for future traffic. The update rule is probabilistic and age-dependent: "If you are going to my source, go first to the node I am coming from (if I am 'young' enough)" or "Avoid this path (if I am 'old')". The influence of an agent diminishes with age and agents are artificially aged at nodes that are experiencing a heavy overload. Agents move probabilistically and update the routing tables locally at each node they visit, effectively creating a well distributed, but also very adaptive system for finding an optimal path.

The concept of emergence, or emergent behavior, is fundamental to this approach. The AntNet can be characterized as an emergent behavior, as the global traffic patterns and near-optimal routing structures arise spontaneously from the simple, local interactions of many independent agents. Crucially, no central controller dictates the optimal path or network state; instead, efficient flows emerge from the decentralized agent behavior, incorporating historical path costs (like crossing times) and local routing table updates (analogous to pheromone updates).

We hypothesize that AntNet can be successfully applied for optimization purposes beyond traditional network routing, such as evaluating the structural efficiency of a big city's transportation system, such as the metro system of Paris. In this project, the group proposes to test a new approach to a first iteration design of the metro system of Paris *Métro de Paris* using the introduced AntNet algorithm which represents the emergent behavior of the ants in following pheromones. The current Metro system evolved slowly over decades, with significant expansion coinciding with events such as the 19th-century World Exhibitions [3], and is now primarily updated and assessed using centrality analysis (to be explained later).

The results of the algorithm are not an immediate representation of the optimum infrastructure when taking into account the socio-technical parameters that play a significant role in the design process of such a complex network. These variables are not taken into account in the model. However, the results highlight alternative configurations that emerge purely from decentralized optimization dynamics. The aim is to explore whether the emergent patterns generated by AntNet can reveal new, potentially more efficient structural arrangements for the Paris metro network that may not have arisen through the gradual, historically driven development of the existing system. In this way, AntNet serves as a tool not for imitation but for discovery, enabling the identification of novel routing structures that emerge from bottom-up agent interactions rather than top-down design decisions.

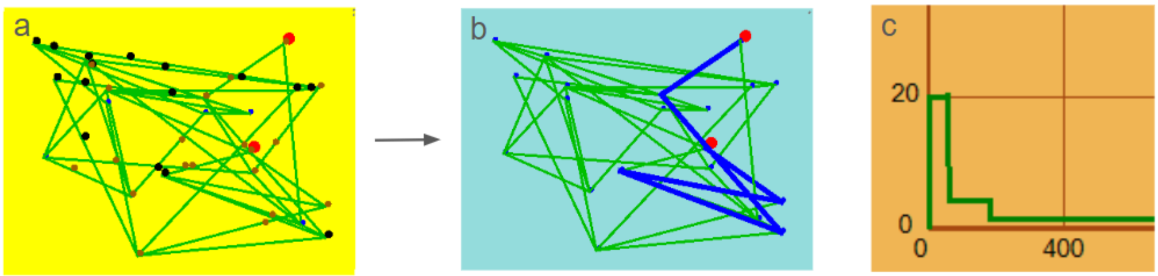


Figure 1: AntNet as an optimization technique. a) The initial AntNet, with ants initialized randomly. b) The ants will create an optimal path between the two destination points (the blue line represents the most optimal path found thus far). c) A visualization of the length of the optimal route found over time, where the step function plateaus at the optimum path length.

Centrality Analysis in Urban Planning

Centrality analysis is a quantitative method from network science that models a transportation system as a mathematical graph [4], where stations are nodes (or vertices) and the lines connecting them are links (or edges). This analysis is crucial for urban planners, as it mathematically determines the relative structural importance of each station, independent of simple passenger entry/exit counts. By correlating these structural metrics with real-world passenger flow, planners can identify bottlenecks, strategically place transfer points, and anticipate demand on future lines to maximize network efficiency.

The most common metrics used are Degree Centrality (number of connecting lines), Closeness Centrality (proximity to the network centre), and Betweenness Centrality (vitality as a transfer hub) [5]. For urban rail networks like *Métro de Paris*, a variation of Betweenness Centrality is often the most informative. This metric calculates how many times a particular station lies on the shortest path between every other pair of stations in the network. A high Betweenness score reveals a critical transfer hub and structural choke point (e.g., Châtelet–Les Halles), which guides optimization suggestions such as increasing capacity or the planning of relief lines (e.g., Metro Line 14).

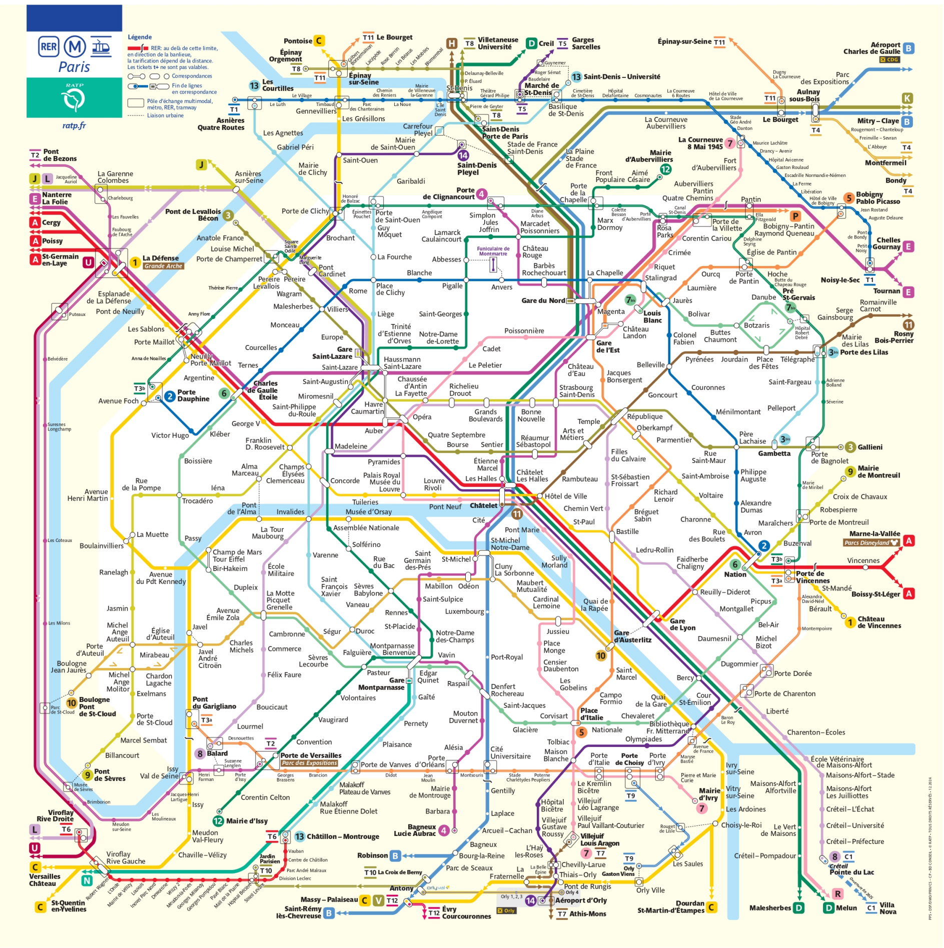


Figure 2: Metro Map of Paris [1]

[1] Cited on 20th November 2025, URL: https://www.ratp.fr/en/plan-metro

Project Goal

In this project, we aim to implement and run the AntNet algorithm on the Paris Métro station structure, assuming ideal connections and randomly generated traffic for each station. From the results, we are particularly interested in identifying agreements and discrepancies between our generated alternative and the existing metro network. It is not expected to obtain a replicate version of the current infrastructure; rather, it is a visual, heuristic comparison that gives a first-order sense of the model's output. For example, observing whether the model does or does not produce a highly connected grid of stations across the city. This is appropriate because the design of the current metro system is not solely based on mathematical or computational models, but also on historical, socio-technical, and practical considerations.

Method

The approach of our project is as follows.

1. Coordinates of the metro stations

Data for metro stations of the Métro de Paris' network was found publicly available in csv format [6].

2. Symmetric kNN graph as a search space

To reduce the dimensionality of the search space, we will restrict the possible connections between nodes on the graph to the k nearest neighbors. This represents the fact that some metro stations do not make sense in a physically connected graph. Our approach is to naively prune all but the k closest connections, but a more sophisticated approach would be required for a rigorous study.

The search space defined by the resulting sparse graph will most likely not contain the optimal solution of the densely connected graph, but most good solutions will lie in the subspace of this graph. This makes the search more efficient in three ways. First, the search space is effectively denser in appropriate solutions. Less edges per node reduces the computational cost of each iteration. Additionally, pruning edges increases the convergence rate of the algorithm since the it samples fewer combinations.

Additionally, we enforced each edge to be bidirectional, which increases the search space and simplifies ensuring the graph is fully connected.In this particular case, $k = 4$ was chosen for being the smallest number for which the grah was fully connected.

3. Modeling metro stations

Differences in traffic volume of the metro stations was modeled by placing different ammounts of food in each one. Food depletes as ants reach it and it recovers over time. This will lead to areas with high traffic, where food is split across many ants, and regions with less traffic, where less food is exploited by a smaller population of ants.

Ants travel around the map in the search for food, thus a station with more traffic should expect more traffic of ants. Food was distributed with a random field on the metro stations modeled with a 2D gaussian distribution centered in the middle of the map. This gives a higher importance to the stations in the city center to model that these locations are more likely to be the destination of a trip since passengers from all other areas are expected to commute to the city center. In practice, some additional restrictions should be added such that the commute from and to the city center and the inter-center traffic rates match real data. Additionally, some locations such as airports would need special consideration which we spare in this project.

4. Colonies

Ant colonies are placed on metro stations in the outskirts of the map. These model ants that want to move form the outskirts to the center, and is another simplification we do. This ensures that low-food stations in the outskirts are reached and incorporates information from the real metro system layout into our simulation. We chose 12 colonies as the starting point of this study, although this number should at least match the number of terminal stations of the current layout for better results.

The 12 colonies have different pheromones which attract their own and repel other ants. This will ensure that each colony tends to cover one section of the metro system.

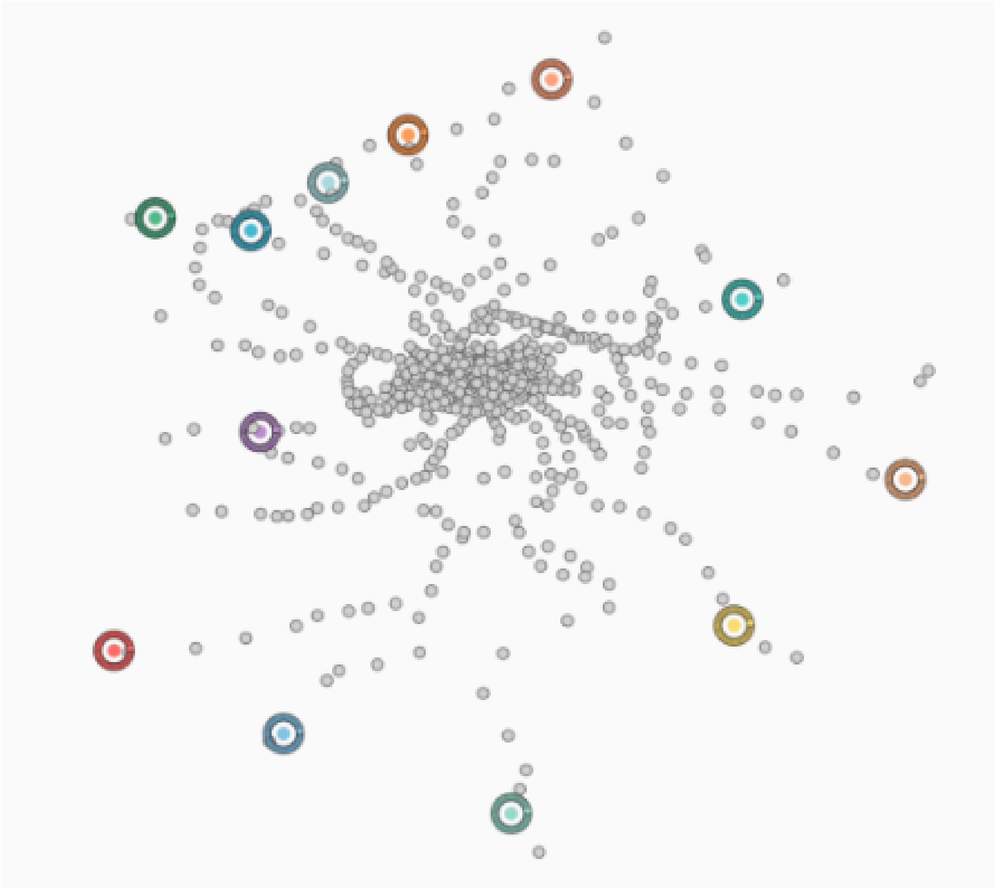


Figure 3: The metro map of Paris (without any ant movement). The grey nodes represent the metro station of Paris, where the absolute coordinates are used from [6], the colored points represent the initial placement of the ant colonies, where we chose to use 12 colonies in order to obtain 12 metro line suggestions.

5. Ant movement

Ants move randomly from station to station searching for food. Initially, ants are initialized in the outskirts of the map, then they are incentivized to reach the city center and travel around until their expected food consumption reaches an equilibrium.

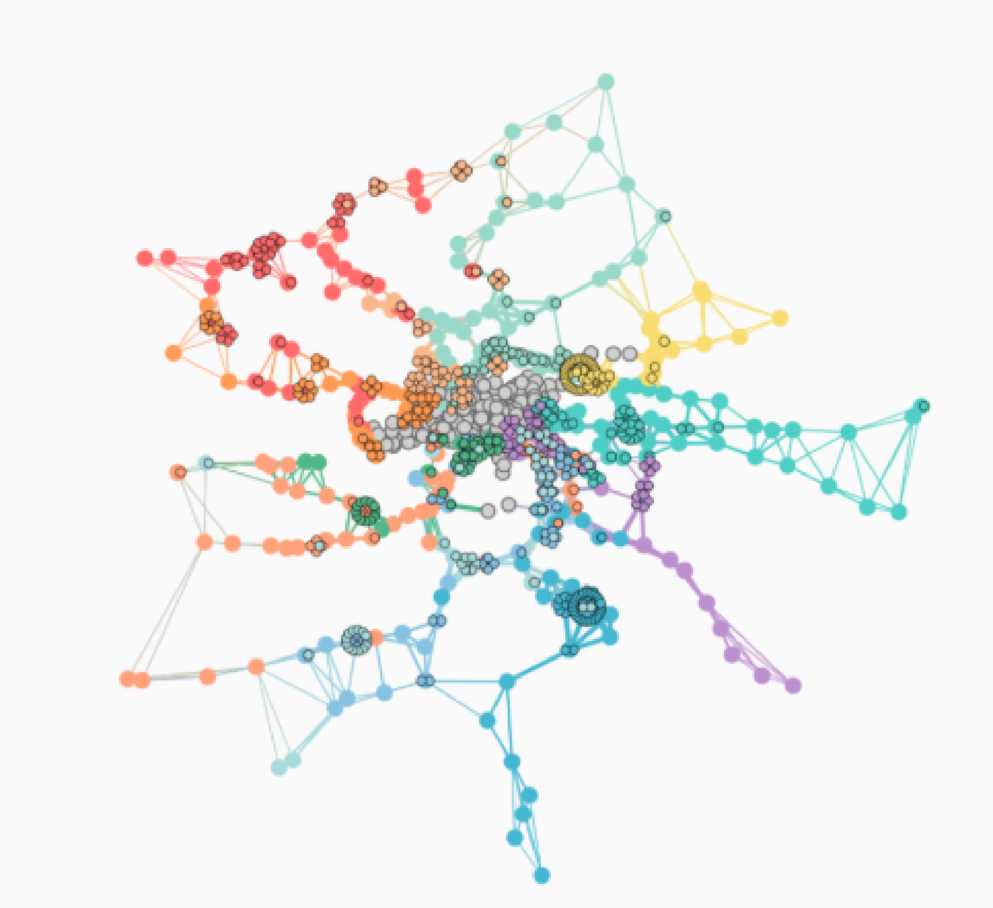


Figure 4: The resulting AntNet, showing the bottlenecks in our version of the metro network reconstruction.

Results

Figure 4 presents the final configuration of the AntNet after the simulation concluded. The network's edges are color-coded to indicate the colony dominance, where an edge's color corresponds to the colony that has deposited the highest concentration of pheromones along that path.

We observed a general tendency for colonies to establish relatively independent paths leading toward the central region of the graph. However, some degree of colony mixing is evident. For instance, the orange colony shows noticeable overlap with the trails established by the red and blue colonies, suggesting shared usage or a shifting dominance on those common edges.

A key observation is the varying degrees of success in path establishment. Certain initiated colonies, such as the green colony, failed to form a distinct or robust trail. This colony's intended path appears to have been dominated by its neighboring orange colony. Our primary hypothesis for this outcome is a difference in the foraging pace or efficiency among colonies. The more dominant colony likely achieved a higher initial pheromone concentration on its preferred path faster, perhaps due to quicker satisfaction with the food sources (nodes) along its route, thereby out-competing the less successful colonies through a positive feedback loop.

The resulting AntNet exhibits significant deviations from the expected ideal-case scenario, particularly when compared to a functional transport network like the Métro de Paris. Specifically, two major shortcomings were noted:

- **Incomplete Node Coverage (Consumption):** Contrary to the goal of connecting all service points, the AntNet simulation did not result in all nodes being 'consumed' by the ants. This indicates an inefficient foraging strategy where potential food sources were ignored, likely because the total volume of food and its generation were more than the ants could consume, leading to a halt of the exploration. This could also happen in our setup due to inefficient connections that are always avoided in favor of more efficient connections.
- **Suboptimal Path Topologies:** The resulting AntNet displays structural flaws, notably the presence of interconnected nodes forming short-circuits and instances of backward steps within a foraging pattern. A real-world metro system aims for a primarily unidirectional flow from a point of origin to a destination, minimizing loops and retracing of steps, which was not observed here. The desired network property of traveling from one side of the city to the other was also not met.

This latter deviation can be directly attributed to the design of the simulation's food distribution model. The use of a Gaussian distribution concentrated the simulated 'food' (or high-utility nodes) disproportionately in the centre of the graph. This strong central attractor pulled all colony paths inward, preventing the formation of expansive, city-spanning lines necessary for efficient urban transit.

Discussion

Our primary research goal was to see if the AntNet simulation could autonomously generate a network pattern similar to the existing Métro de Paris (as shown in Figure 2), was ultimately not achieved, based on the visualization in Figure 4. This lack of alignment is not entirely unexpected and can be explained by both the model's design constraints and the inherent limitations of the AntNet use:

1. **Impact of Gaussian Food Distribution:** As noted in the results, the Gaussian distribution of food sources served as a critical constraint. By prioritizing the graph's center, the model inherently steered all foraging agents toward a centralized cluster, failing to generate the necessary linear extensions to the outskirts of Paris that characterize a functioning metropolitan transport system.
2. **Exclusion of Essential Planning Factors:** The very nature of our naive ant foraging behavior, which does not account for many relevant parameters of the model, leads to nonsensical results. The resultant network, optimized purely for throughput, fails to incorporate travel convenience, and geometrical implications of the design, such that turns are slower than straight lines

Furthermore, the current metro network is a product of long-term, incremental development, a process shaped by historical precedents—such as the World Exhibition that spurred its initial construction—and gradual expansion. The AntNet simulation, on the other hand, is a one-shot optimization algorithm that lacks the temporal depth and adaptive capacity of real-world urban growth.

Therefore, our final conclusion is that the AntNet simulation served as a valuable heuristic test to explore how a decentralized, stigmergic intelligence model would approach a complex network like the Métro de Paris. That being said, the model is not powerful enough to work on any real-world graph. Accurate restrictions for the graph would be needed to obtian meaningful solutions, exhaustive and accurate cost functions would be needed to obtain solutions which are also good in practice. Given these components are in place, this algorithm is easily capable of finding them efficiently.

Bibliography

1. Schoonderwoerd, R., Holland, O. E., Bruten, J. L., & Rothkrantz, L. J. M. (1997). Ant-Based Load Balancing in Telecommunications Networks. Adaptive Behavior, 5(2), 169–207. https://doi.org/10.1177/105971239700500203
2. Applyby, S., & Steward, S. (1994). Mobile Software Agents for Control in Telecommunication Networks. BT Technology Journal, 12(2), 104-113.
3. Bobrick, B. (1947-), Bobrick, B. (1981). Labyrinths of iron: A history of the world's subways. Newsweek Books. New York.
4. Derrible, S. (2012). Network centrality of metro systems. PloS One, 7(7), e40575. https://doi.org/10.1371/journal.pone.0040575
5. Platform, A. (2020, September 25). Centrality Analysis. Medium:
https://axuplatform.medium.com/centrality-analysis-832ae76a6eca

November, 2025

ATHENS course : **MOB_0AT09_TP**

Emergence in Complex Systems

Micro-study

teaching.dessalles.fr/ECS

Names:

- **Stepan Svirin**
- **Yufei Zhou**
- **Mathieu Antonopoulos**
- **Pablo Bertaud-Velten**

Cocktail Party: Make the party more fun

Abstract

The original simulation has agents speaking over themselves, eventually reaching points where everyone is screaming at the top of their lungs. We extended the standard "Cocktail Party" simulation by introducing agent mobility and more complex behavior. Enabling movement allowed agents to relocate to quiet zones, stabilize the system at the populations maximum vocal capacity. Then, we added walls and points of interest, which created "quiet islands" inside the rooms. We also introduced a drunkenness mechanism, meaning individuals gradually become louder over time and eventually, this rising drunkenness overwhelms any spatial organization, causing the map to become saturated with noise. Finally, to avoid unrealistic noise saturation, we implemented a physiological voice fatigue mechanism, that acts as a negative feedback loop, forcing agents to remain silent after shouting for too long.

Problem

In the standard "Cocktail Party" simulation [1] everyone eventually ends up shouting at maximum volume. Our study looks at whether we can keep the noise manageable by introducing real-life features like allowing agents to move, adding walls, and making voices get tired. We also test if these rules still work when people get drunk and try to speak louder.

Method

Drunkenness

To make the simulation more dynamic and counterbalance the effects of agent movement, we introduced a new parameter, BeerPrice, which controls how quickly an individuals SNR increases over time. This drunkenness mechanism works by incrementally raising each agents SNR at every iteration, with the growth rate determined by the cost of beer: the cheaper the beer, the faster agents get louder. In our implementation, SNR increases linearly between 0 and 0.1 per iteration.

The resulting behavior is that, even after the global sound level initially stabilizes due to agents relocating, the continued rise in SNR eventually pushes the noise level upward again. This renewed increase prompts further movement, creating new waves of reorganization.

This hypothesis is grounded in personal observations, and while it aligns with common sense, these behaviors would still need to be empirically validated.

WNM Model: Walls, Noise Propagation, and Movement

To increase the realism of the simulation, we changed the environment from a simple open torus to a bounded 2D grid with physical structures. We implemented a `WallLayout` helper class to generate static walls, what allowed us to partition the space into two distinct zones: a "Bar" and a "Quiet Room". We precomputed an adjacency graph that maps valid transitions between cells. This graph enforces topological constraints, and ensures that neither agents nor sound waves can pass through walls or corners.

We also had to change the acoustic model to respect new geometry. Sound now spreads locally from each speaking agent via a Breadth-First Search (BFS) restricted by the wall layout. The source intensity is modulated by the agent's dynamic drunkenness level, described previously. This approach allows walls to act as sound barriers, create acoustic shadows and ensure that the enclosed rooms remain acoustically distinct from the open floor.

Finally, agents now use pathfinding algorithms to traverse corridors and doorways physically. We implemented a priority-based decision tree for movement: agents generally drift towards the Bar based on an attraction parameter, but will actively scatter to neighboring cells if local density exceeds a `CrowdThreshold`. Furthermore, we added a "patience" system: if an agent's local noise exceeds their `ComfortNoise` threshold for too long, they become tired and navigate specifically toward the Quiet Room or a calculated low-noise area.

Voice Fatigue Model

The base model allows agents to shout indefinitely, which quickly drives the average voice level close to its maximum and removes interesting temporal dynamics. To correct this, we introduced voice fatigue as an intrinsic constraint on how long an individual can maintain a high voice level.

Each agent is assigned a scalar `Energy` variable, bounded between 0 and `EnergyMax`, together with a Boolean `Resting` state that indicates whether the agent is currently forced to stay silent. Voice fatigue is governed by five parameters: `EnergyMax` sets both the initial and maximal energy available for speaking; `FatigueThreshold` specifies the voice level above which speaking starts to consume energy; `FatigueLoss` defines how much energy is lost per time step when speaking above that threshold; `RestGain` controls how fast energy is recovered while resting; and `RestMinToSpeak` gives the minimal energy required for an agent to resume speaking.

At each iteration, if an agent is not resting and its current `VoiceLevel` is higher than `FatigueThreshold`, its `Energy` is decreased by `FatigueLoss`. When `Energy` reaches zero, it is clipped at this value and the agent switches to the resting state. While resting, the agent no longer contributes to the surrounding noise and its `Energy` increases at rate `RestGain` until it reaches `EnergyMax`. As soon as `Energy` exceeds `RestMinToSpeak`, the resting state is cleared and the agent becomes able to speak again.

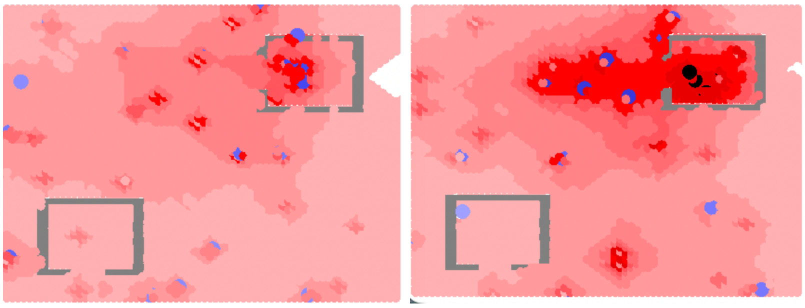
Voice updates are then gated by this resting state. If `Resting` is true, the agent is forced to remain silent and its `VoiceLevel` is set to zero; otherwise, the original voice update rule from the base model is applied unchanged. The fatigue mechanism is therefore orthogonal to the spatial extensions and to drunkenness: it can be combined with movement, walls and evolving SNR without modifying their logic, while preventing permanent saturation of the global voice level.

Results

WNM Model: Experimental Results

In this section, we present the simulation results to explore how the addition of walls, improved sound propagation, and advanced agent behavior influence the average noise level and the overall state of the party. The heatmaps in this section display two distinct rooms: the Quiet Room (where agents prefer to speak more quietly) is located in the bottom-left corner, and the Bar (a point of interest) is located in the top-right corner.

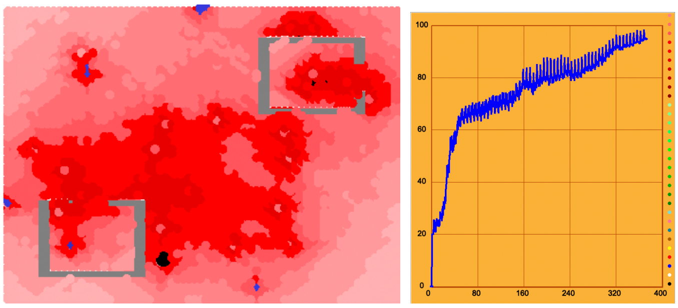
Figure 1 visualises how the noise heatmap progresses during the party. Subfigure (a) illustrates the initial state, while (b) shows the party in progress. We can observe that as agents become more intoxicated, the Bar attracts a higher density of people and the noise level increases, while some agents prefer to rest away from the Bar or seek refuge in the Quiet Room.



(a) Initial state of the party (b) Party state after 100 steps

Figure 1: Progression of the simulation environment. (a) shows the low-noise initial state, while (b) illustrates the noise accumulation and crowding around the Bar as the simulation progresses.

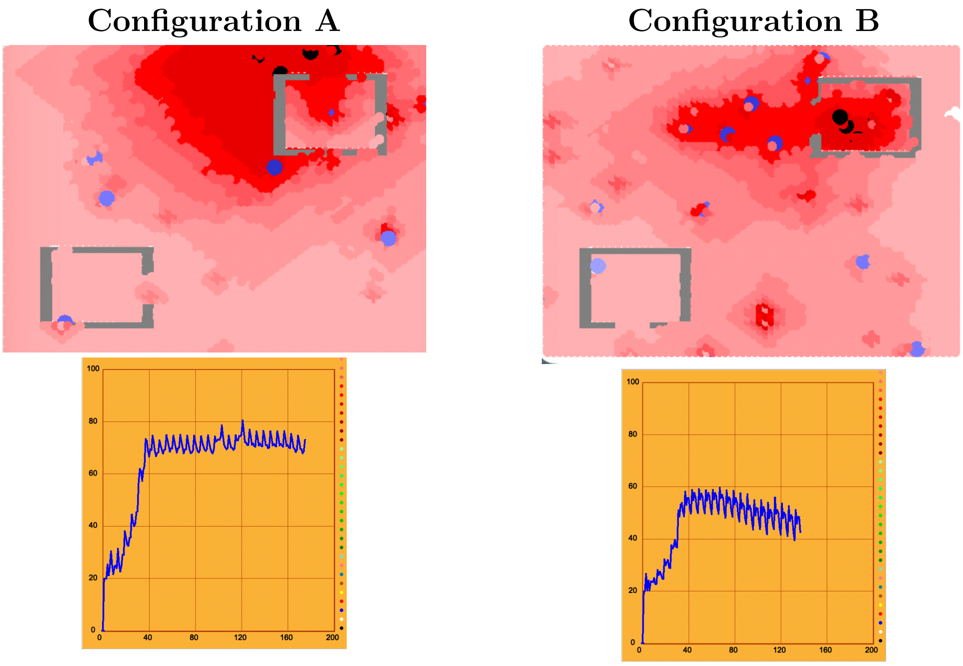
Figure 2 illustrates the scenario with extremely cheap beer, which increases the Bar's attractiveness. Surprisingly, we observe that as the party progresses, agents become less concentrated around the Bar and disperse throughout the available space. Driven by the effects of drunkenness, the average noise level rises continuously until it reaches its maximum.



(a) Heatmap of the party (b) Noise Plot

Figure 2: Visual analysis of the "Drunk Case" scenario, showing spatial noise distribution (left) and temporal noise evolution (right).

Figure 3 provides a comparison of heatmaps and noise levels between two different room configurations, while keeping other model parameters constant. It is evident that the positioning of rooms and their entrances directly affects the average noise level.



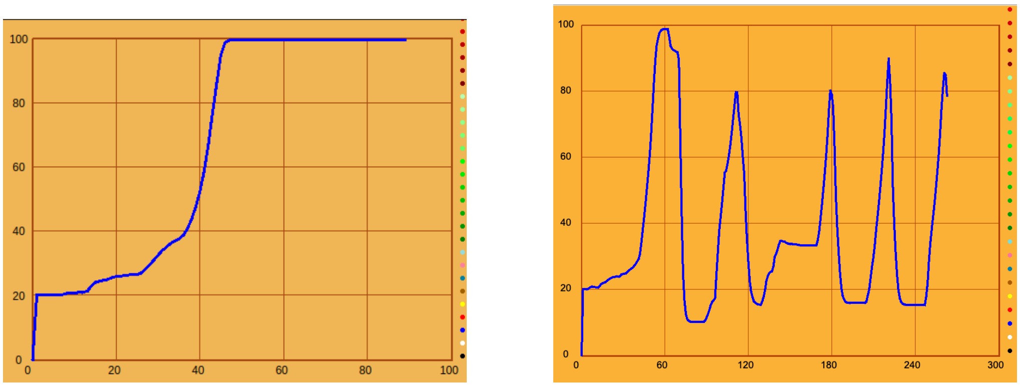
Configuration A **Configuration B**

Figure 3: Comparative analysis. Configuration A shows significant noise level, while B shows small and even decreasing average noise.

Voice Fatigue: Experimental Results

We evaluated the effect of voice fatigue on the global dynamics by comparing the base model with the fatigue-extended version. In the base case, where agents can speak loudly at every time step, the average voice level rises quickly and then remains close to the maximum for the rest of the simulation. Once the noise has saturated, there is little variation left for the system to exhibit.

With fatigue enabled, the trajectory of the average voice level changes qualitatively. As shown in Figure 4, periods of intense speaking drain the agents' energy and progressively push many of them into rest. When a large fraction of the population is forced to remain silent, the global noise level drops. After a sufficient recovery period, agents regain enough energy to speak again, creating a new phase of rising noise.



(a) Average voice in the base model (b) Average voice with fatigue

Figure 4: Effect of voice fatigue on the average voice level. Without fatigue, the average voice saturates near its maximum. With fatigue, alternating phases of loud activity and collective silence produce oscillations in the global noise.

Instead of converging to a flat high plateau, the system shows self-organized oscillations between noisy and quiet phases. These fluctuations are driven purely by the internal state of agents and do not require any change in the external environment, which makes fatigue a simple but effective way to restore richer temporal patterns in the party dynamics.

Discussion

WNM Model

The experiments demonstrate that this model can effectively simulate the evolution of the party. As shown in Figure 1, agents successfully navigate toward the Bar over time, clustering there and driving up noise levels in a manner similar to the real-world dynamics.

Figure 3 shows that spatial layout is a decisive factor in noise control. Inefficient designs, such as Configuration A, force agents into bottlenecks, this overcrowding forces them to raise their voices to be heard. This means, optimizing room and doorway placement is important for improving circulation and reducing local density, which lowers the overall noise level. This framework could serve as a design tool for architects and event planners, allowing them to simulate crowd flow and mitigate acoustic saturation in venues.

The dispersion seen in Figure 2 comes from a self-limiting feedback loop. As cheap beer increases voice levels, the noise at the Bar quickly exceeds agents' comfort thresholds. This triggers avoidance behaviors like `seek_quiet` and `escape_crowd`, forcing agents to retreat to quieter areas. Thus, a paradox emerges: while the alcohol attracts the crowd, the resulting noise pollution drives them away.

Voice Fatigue

The fatigue mechanism complements the spatial and behavioral extensions by introducing an individual constraint on vocal effort. Figure 4 shows that limiting vocal energy is sufficient to prevent the trivial outcome where everyone shouts at the maximum level for the entire simulation.

From a dynamical perspective, fatigue creates a simple negative feedback. When many agents speak loudly at the same time, they collectively deplete their energy, which pushes a growing fraction of them into enforced silence. This drop in active speakers reduces the global noise level and allows energy to recover, until agents can rejoin the conversation. As a result, the system alternates between noisy, highly engaged phases and quieter intervals where a large part of the population is resting.

This mechanism captures a realistic aspect of human behavior: people cannot shout indefinitely and must occasionally pause, even in a very loud environment. In our model, this constraint restores nontrivial temporal patterns in the average voice level and interacts naturally with movement and drunkenness. For instance, a loud period near the Bar can be followed by a collective rest phase, during which agents may relocate to quieter regions, before noise rises again when they start speaking. Voice fatigue therefore provides a minimal microscopic rule that helps keep the party from collapsing into a permanently saturated state.

Conclusion

Our results show that the shape of the room matters. Walls can create quiet spots, but if the layout is bad, people crowd together and have to shout to be heard. We also found that when agents get "drunk," they become too loud for the room layout to handle, forcing everyone to spread out to find peace.

By implementing "Voice Fatigue" and forcing agents to rest after shouting for too long, we stopped the constant screaming. Instead of staying at the maximum level forever, the noise now goes up and down naturally, making the simulation closer to real life.

Bibliography

References

[1] Jean-Louis Dessalles. Evolife. https://evolife.telecom-paris.fr/. Accessed: 24 November 2025.

November, 2025

ATHENS course : **MOB_0AT09_TP**

Emergence in Complex Systems

Micro-study

teaching.dessalles.fr/ECS

Name: **Aghiles Gasselin, Mikhail Kataevskii, Tymon Orłowski, Xinyi Pu**

Applying Schelling segregation model to city maps with budget and family incentives

Abstract

This project extends the Schelling segregation model by integrating real-world factors such as neighbourhood borders, housing prices, agent budgets, and inheritance, and by applying the model to real city maps like New York and Paris. Through simulations, we show how simple individual preferences can generate complex and often unexpected segregation patterns, including price-driven clustering and partial family-based grouping.

1 Problem

The problem this project addresses is understanding how simple individual preferences and constraints can unexpectedly generate large-scale residential segregation in real cities and involving economic factors such as budgets, house prices, and inheritance, as well as geographical structure and social preferences like wanting to live near family, all of which are not well understood in terms of their combined effect on segregation. The project therefore aims to explain why and how these different factors interact to produce the complex, often undesirable patterns of segregation observed in cities like New York and Paris.

2 Method

2.1 Integrating real city maps

We created a new type of agent that corresponds to a map border. This agent is represented by a black dot that never moves. These borders are ignored during satisfaction calculations.

Additionally, we implemented a loader function that parses city map images and populated black pixels with border agents. The remaining cases are pupulated randomly by other agents, like in the starting scenario. We also adjusted the grid position so that each agent corresponds to exactly one pixel of the background. This way we could run the simulation on top of real city maps. We used the cities of New York and Paris as examples.

2.2 Integrating Housing Prices and Budgets

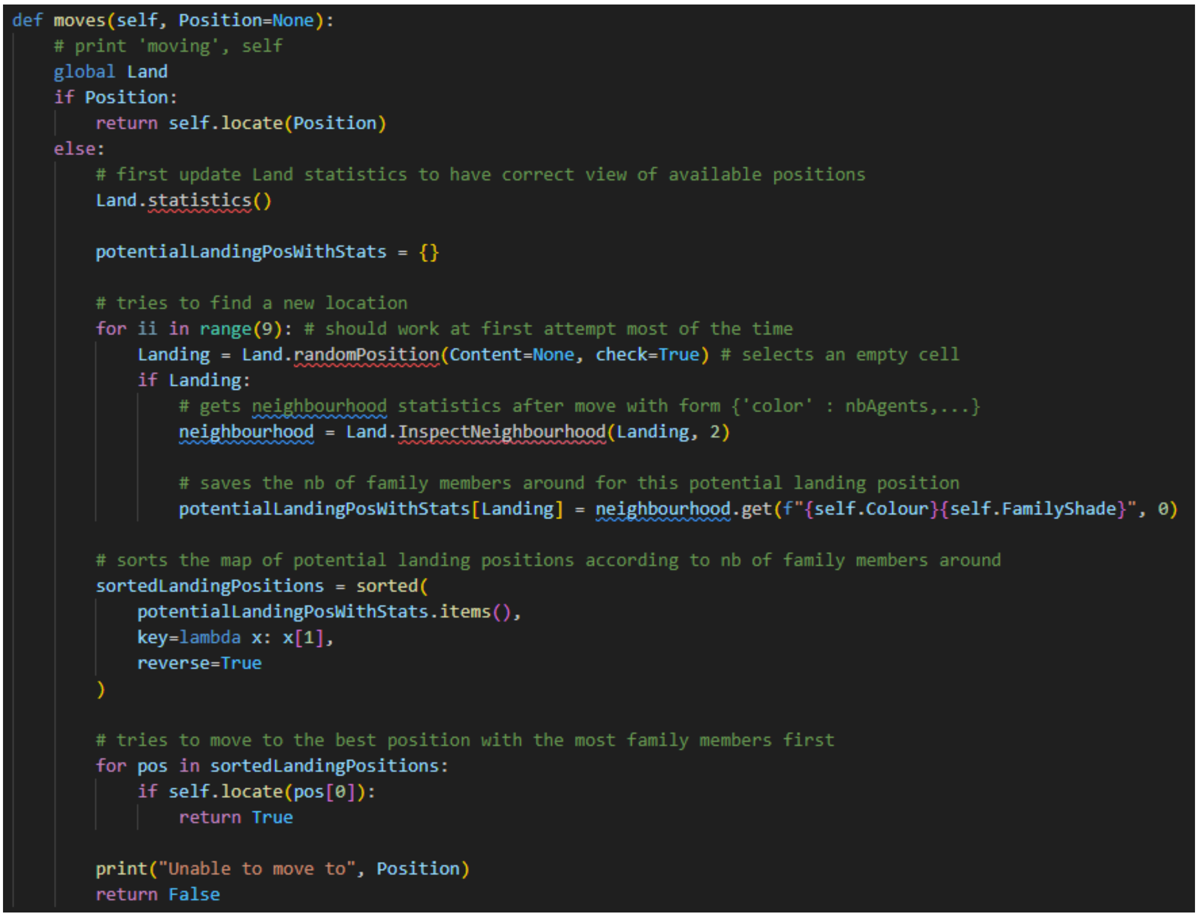
Each cell is assigned a house price (low, medium, high), and agents have corresponding budgets. Agents may only move to affordable cells. Their satisfaction score combines colour similarity and neighbourhood wealth using weighted components. Agents relocate only when the new position improves their satisfaction, creating positive feedback. The satisfaction score for an empty cell is computed as:

$$\text{Score} = a \cdot \text{color\_diff} + b \cdot \text{budget\_diff}$$

The weights a and b control how much agents care about neighborhood colour composition versus neighborhood wealth. A lower score indicates higher satisfaction, and if Score $<$ Tolerance, the agent remains in that cell permanently. In the model test, we varied the parameters (a, b) and the distribution of house prices. In the base configuration, the grid is divided into three vertical regions: left third (low price), middle third (medium price), and right third (high price). This creates controlled economic gradients to observe price-driven segregation.

2.3 How agents are moving closer to their family

To isolate the effect of family proximity on agent movement, we modified only the movement function while keeping the original satisfaction rule unchanged. This ensures that agents still decide whether they are unhappy exactly as in the base model, but once unhappy, they now choose where to move using a family-oriented heuristic. As shown in the code, when an agent is unhappy, it samples several random empty cells and computes, for each, the number of nearby family members of the same colour and family shade (the shade is used to distinguish between different families). These potential landing positions are stored and sorted in decreasing order of family presence. The agent then attempts to relocate to the position with the highest number of nearby relatives. This mechanism encourages agents to drift toward areas where small pockets of family members already exist, without forcing global cohesion or modifying satisfaction criteria.



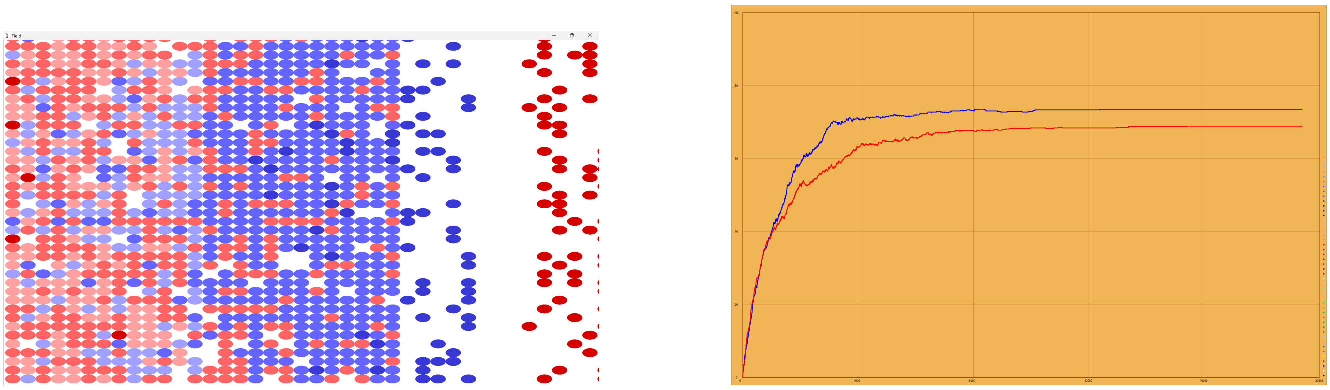
```
def moves(self, Position=None):
    # print 'moving', self
    global Land
    if Position:
        return self.locate(Position)
    else:
        # first update Land statistics to have correct view of available positions
        Land.statistics()

        potentialLandingPosWithStats = {}

        # tries to find a new location
        for ii in range(9): # should work at first attempt most of the time
            Landing = Land.randomPosition(Content=None, check=True) # selects an empty cell
            if Landing:
                # gets neighbourhood statistics after move with form {'color' : nbAgents,...}
                neighbourhood = Land.InspectNeighbourhood(Landing, 2)

                # saves the nb of family members around for this potential landing position
                potentialLandingPosWithStats[Landing] = neighbourhood.get(f"{self.Colour}{self.FamilyShade}", 0)

        # sorts the map of potential landing positions according to nb of family members around
        sortedLandingPositions = sorted(
            potentialLandingPosWithStats.items(),
            key=lambda x: x[1],
            reverse=True
        )

        # tries to move to the best position with the most family members first
        for pos in sortedLandingPositions:
            if self.locate(pos[0]):
                return True

        print("Unable to move to", Position)
        return False
```

Figure 1: Family-oriented movement algorithm used to rank and choose landing positions.

3 Results

3.1 Effects of Economic Preferences

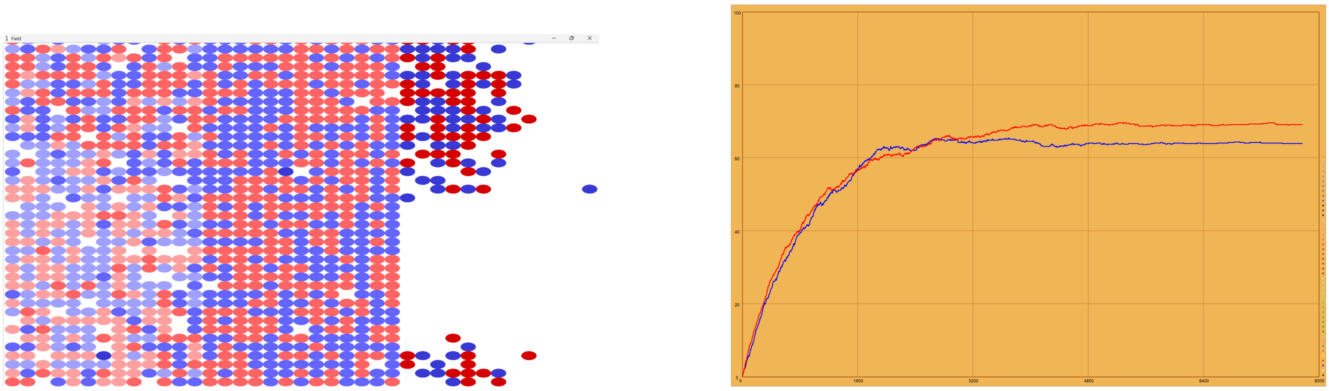
When colour is weighted strongly, simulations produce clear colour segregation. High-budget agents occupy all price regions, while low-budget agents remain restricted to cheap areas. When wealth is weighted more heavily, budget-based clustering dominates and colour clusters become fragmented. The respective simulation results are shown in Figure 2 and Figure 3.



(a) Segregation patterns

(b) Corresponding satisfaction curves

Figure 2: Segregation and satisfaction dynamics for parameters a = 0.9, b = 0.1.



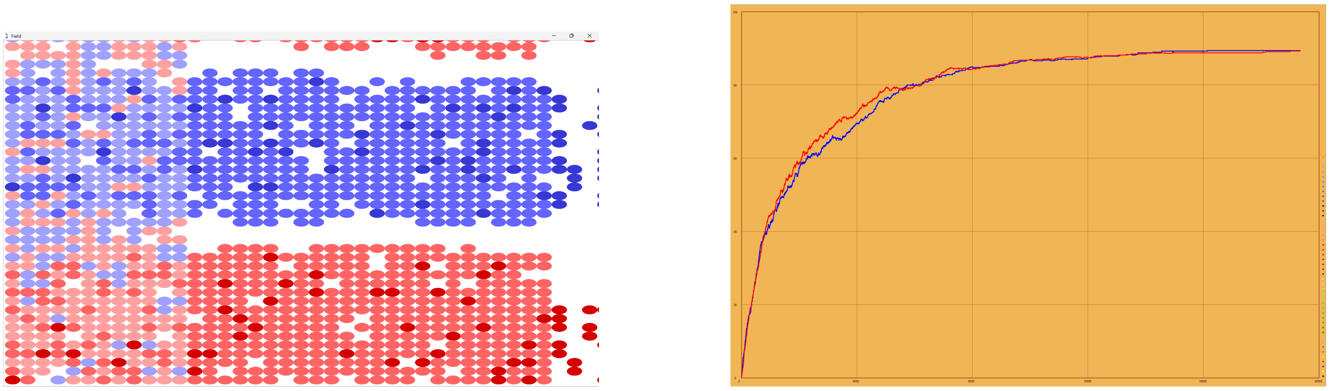
(a) Segregation patterns

(b) Corresponding satisfaction curves

Figure 3: Segregation and satisfaction dynamics for parameters a = 0.1, b = 0.9.

3.2 House Price Distribution Matching Budget Demand

We simulate a scenario where agents are mainly influenced by neighbors' colours. As shown in Figure 4, when the map's price distribution matches population budgets, global satisfaction increases and segregation becomes more pronounced, with strong clustering driven by affordability and local similarity. Satisfaction curves exhibit strong positive feedback.



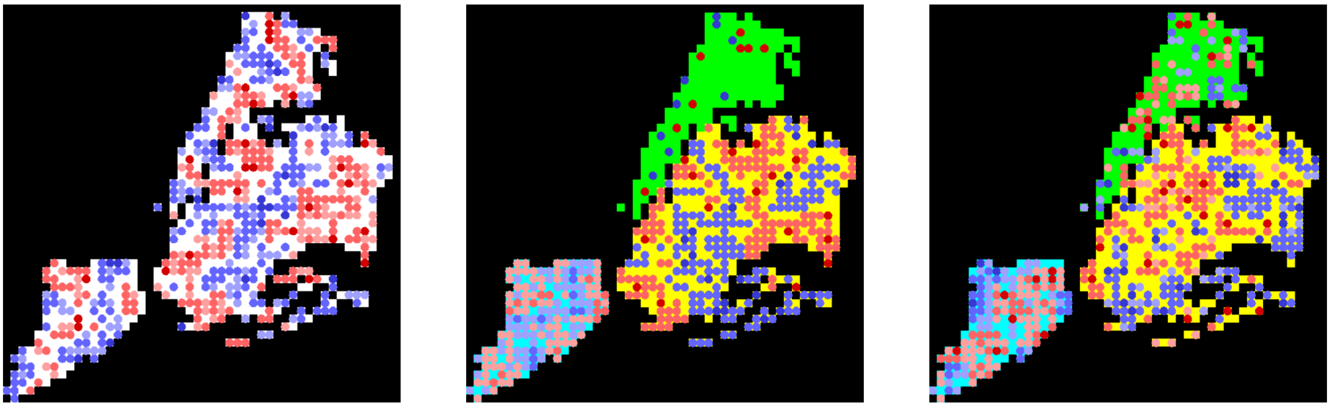
(a) Segregation patterns

(b) Corresponding satisfaction curves

Figure 4: Segregation and satisfaction dynamics with matching house prices to market demand.

3.3 Using Real City Maps

We apply our model to New York City. This city contains 3 price regions that are clearly separated by water. Note that there are still some bridges between the expensive and normal areas, while the cheap area is located entirely on a separate island. When we add price index to our model, we observe that segregation generally decreases in cheap and expensive areas (5b). In cheap areas, this happens because people are eager to get a cheap house, even if it is surrounded by other agents. In expensive areas, this happens because fewer people can move in, so rich agents can spread out without forming groups. In the inheritance scenario, agents get their first house for free (5c). Note that in this case some poor agents get stuck in their expensive neighborhoods because all cheap houses are gone. This phenomenon contributes to creating a more diverse landscape.



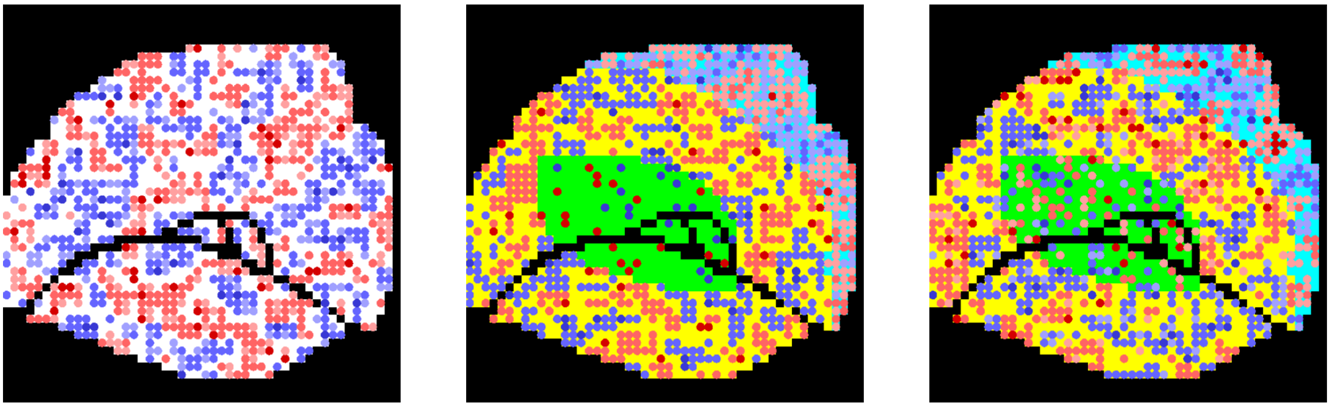
(a) New York without price index

(b) Price index without inheritance

(c) Price index + inheritance

Figure 5: Applying the price model to New York City

The same phenomenon can be observed with the map of Paris, where all price regions are connected. This suggests that in our model price index is more important than geographical constraints.



(a) Paris without price index

(b) Price index without inheritance

(c) Price index + inheritance

Figure 6: Applying the price model to Paris

3.4 The effect of having agents moving close to their family

Adding a preference for agents to move closer to their family produces small clusters of related individuals, but not the large, tight family blocs one might expect. Instead, family members settle near each other without becoming fully "sticky," reflecting real-life situations where relatives live within reach but not side by side. Because some family members start in different parts of the city, they form several small family groups scattered across the grid rather than a single unified cluster. This leads to weaker convergence and a more realistic, fragmented spatial pattern.

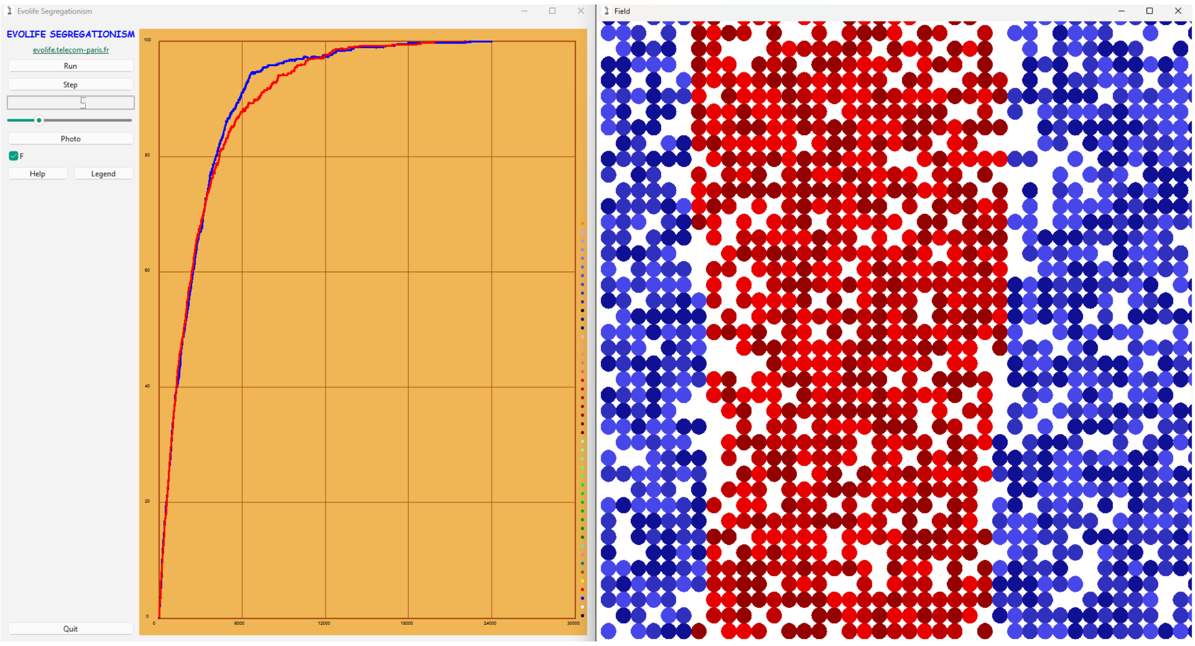


Figure 7: Simulation with agents trying to move closer to their family

4 Discussion

The results of the price index model largely align with our expectations. Agents prefer to live in a cheap house even if it is surrounded by different agents, which increases diversity in cheap areas. Just as in real life, there is a fierce competition for affordable housing areas, which leaves some poor agents stuck in expensive neighborhoods. Some possible improvements to our model would include a dynamic price index, based on demand in each area and a possibility of increasing their budget over time. These improvements should lead to a more uniform distribution of agents over the map.

Concerning the agents moving closer to their family, the results differ from the initial expectation of forming one large, cohesive family cluster. Instead, only small pockets of related agents emerge, highlighting that family proximity encourages local grouping without producing full "stickiness." As an improvement, we think that refining the movement rule or introducing multi-step relocation strategies could lead to more realistic long-range family aggregation or richer emergent structures and some new interesting results may emerge.

5 Bibliography

References

[1] Hatna, E.; Benenson, I. Geosimulation of Income-Based Urban Residential Patterns. In Advanced Geo-Simulation Models; J. Marceau, D., Benenson, I., Eds.; BENTHAM SCIENCE PUBLISHERS, 2011; pp 111–125. https://doi.org/10.2174/978160805222611101010111.

[2] Schelling, T. C. (1971). Dynamic models of segregation†. The Journal of Mathematical Sociology, 1(2), 143-186. https://doi.org/10.1080/0022250X.1971.9989794

[3] Mantzaris, A. V. (2020). Incorporating a monetary variable into the Schelling model addresses the issue of a decreasing entropy trace. Scientific Reports, 10(1), 17005. https://doi.org/10.1038/s41598-020-74125-6

November, 2025

ATHENS course: **MOB_0AT09_TP**

Emergence in Complex Systems

Micro-study

teaching.dessalles.fr/ECS

Name: Christel Al Hage, Hongxu Chen, Christelle Clervilsson, Alessa Mayer

Modeling Ant Foraging Behavior

Abstract

Ant colonies are remarkable examples of collective intelligence and natural problem-solving, especially the nomadic colonies of army ants (Eciton hamatum). Inspired by their behavior, we model a scenario using Evolife where ants must forage for food while overcoming an obstacle: in this case, a river or a barrier. Individual ants explore randomly but avoid previously visited areas marked with repulsive pheromones. When a food source is found, ants return to the colony after draining this source. The presence of the river introduces challenges that require cooperative strategies, including temporary bridge construction and adaptive path selection, highlighting how simple local rules can lead to complex collective solutions.

Problem

In the natural world several examples of collective intelligence and emergent behavior can be observed, where complex, functional structures arise from the local interactions of numerous simple agents. One of the most fascinating examples of this phenomenon is found within the nomadic colonies of army ants (Eciton hamatum). As described in recent studies, these ants employ a remarkable strategy known as self-assembly to optimize their foraging trails: they literally become the architecture of their environment by linking their own bodies together to form temporary, dynamic structures. [1]

Specifically, when a foraging column encounters a gap or obstacle, individual ants will use their bodies to construct living bridges across the void. These bridges serve to smooth the path and significantly increase the speed and efficiency of resource transport. This is an extreme example of collective problem-solving, wherein relatively simple, localized decisions by individual ants culminate in a massive, adaptive communal structure. [1]

Our study aims to model and simulate the emergence and dynamic regulation of this army ant living bridge phenomenon using the Evolife platform. By introducing a critical gap (representing a river or barrier) into the simulated foraging environment, we seek to replicate the collective decision-making process that governs bridge initiation, maintenance, and adaptation. The results of this simulation will allow us to explore the clear-cut relationship between individual interaction rules and collective structural efficiency.

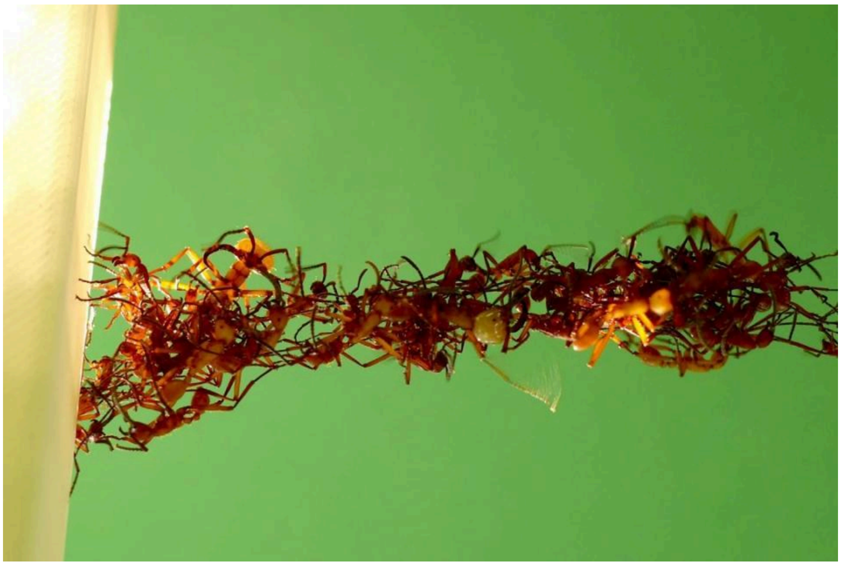


Fig. 1: Army ants using their bodies to building a “bridge”. *Source: https://magazine.gwu .edu/ants-dont-just-build- architecture-they-become -it*

Method

1. **Food source with barriers**

A first implementation, is a slight adaptation of the Ant Application. To simulate the emergent bridge-building behavior observed in army ants, the base Ants.py module was systematically modified to introduce barriers and a new collective aggregation logic. The ants might now encounter food sources behind a “barrier”. In this case, the ant that found the food source, will stay in place and needs to wait for other ants to arrive to build the bridge. These modifications ensure that the simulated gathering of stationary ants acts as an analog for the physical construction of the living bridge to bypass an obstacle and access a resource. The simulation entities were augmented by adding specific attributes: the FoodSource class now includes self.barrier = True, a flag signaling inaccessibility, and self.count = 0, a counter used to count the ants that arrived at the barrier and stayed. Correspondingly, the Ant class gained self.staying_fs = None, allowing an individual ant to maintain a necessary reference to the specific blocked food source where it is aggregated.

The environment visualization was also adapted within the Landscape.__init__ method. Food sources flagged with the barrier=True attribute are now displayed in blue in the observer record, to visualize that the food source is not accessible for the moment.

The most critical changes occurred within the ant's decision-making process, defined by the Ant.moves() method. The core change is the introduction of the new action state, elif self.Action == 'Stay'. This ‘Stay’ action is the simulation's direct representation of the bridge-building process; an ant entering this state becomes stationary, contributing its "body" to the growing structure that spans the modeled gap. An ant remains committed to this structure, holding its position, as long as the associated self.staying_fs contains food. This commitment ensures the bridge persists while the resource it provides access to is still available. Only once the food source is completely exhausted (FoodAmount <= 0) does the ant clear this food source and goes back to the general Move action, simulating the structural ants dispersing after their collective task is completed and the resource is empty.

The final element of the implementation governs the initiation of this bridge-building. Upon detecting a food source, the ant's logic checks the barrier status. If the barrier is present, the ant immediately transitions its action state to Stay and increments the source's interaction counter. This mechanism forces the individual to abandon immediate foraging and dedicate itself to the emerging structure. In turn, this facilitates access to the resource (either through the existing trigger where a certain number of ants clear the barrier), successfully translating a complex biological phenomenon into a minimal set of decentralized simulation rules.

Fig. 1 shows an instance of the implementation. Blue food sources are not yet accessible and ants that encounter them, will gather, until the right amount is met and the food source becomes yellow.

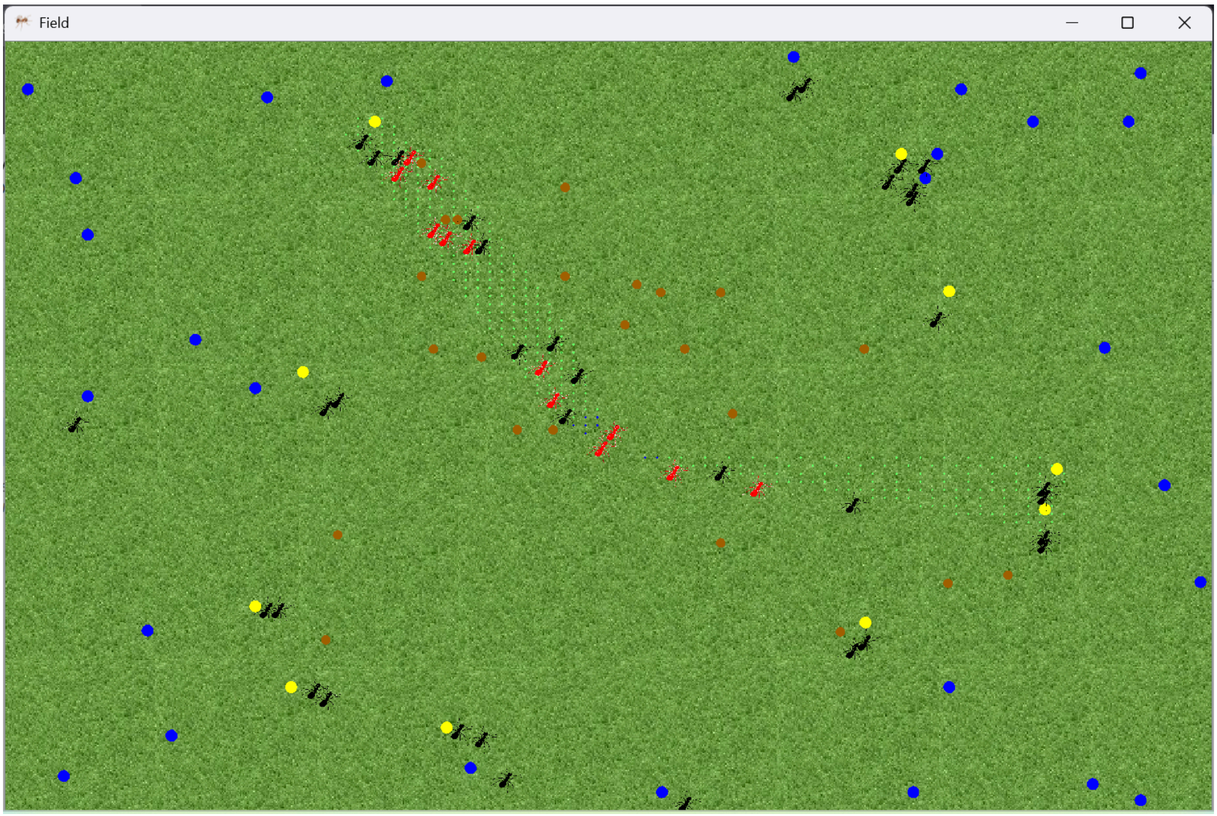


Fig. 2: Implementation of the food sources behind the barrier. Now, inaccessible food sources are present (blue).

2. Food sources behind a river

To implement this simulation, we had to consider multiple factors. First, we needed to structure the landscape to clearly separate the ground from the river. We divided the landscape into three parts: the upper part for the food sources (covering ⅖ of the landscape), the middle part for the river (⅕ of the landscape), and the lower part for the nest location (⅖ of the landscape).

The first challenge was the toric parameter, which allowed ants to exit one side of the frame and re-enter from the opposite side. This behavior undermined the simulation, so we removed it and instead assigned a random location to any ant that moved out of bounds. After resolving this issue, the main goal was to define the river, so the ants could stop as soon as they got to the river, so the bridge could be built. We modify the landscape class to create a river. The main idea is that the landscape is composed of landcell objects and each of them have a value called stay_count Each river cell stay_count is initialized to -1 and the other cells to 0. When the value of the stay_count is equal to zero, it means that this position is accessible.

Once the limitations were defined, the next step was to make ants stay next to the river until a sufficient number had accumulated to build the bridge. A key issue arose: if ants moved according to the original move function, there was no guarantee that enough ants would gather to form the bridge. We had to ensure that following ants would join those already waiting and that ants requesting assistance would not wait indefinitely.

To achieve this result, we modify the function "moves". When an ant's state was set to "stay," it would remain in place until its waiting time exceeded a limit, at which point it could move. If the ant was moving, it would check whether the next cell's stay_count was greater than 0 indicating that another ant was already waiting. If so, the ant would stay in place; otherwise, it would move to the next cell.

In Fig. 3 we can see the interface of the implementation. Food sources are behind the river and the ants need to cross the implemented river to access them.

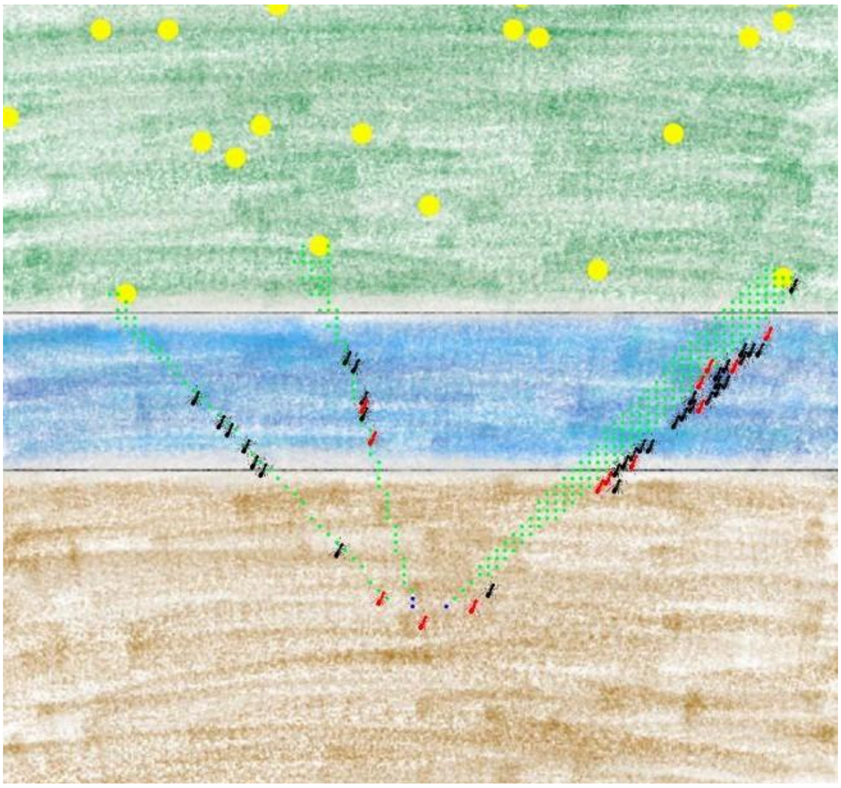


Fig. 3: Simulation, in which the ants build "bridges" to cross the river and reach the food sources.

Results

1. **Food source with barriers**

While the implementation allows to have any amount of food sources with or without a barrier initially, we used a scenario with only inaccessible food sources to compare how the number of ants used to remove the barrier changes the number of collected food. Initially, we started with the number 2 as a necessity and incrementally increased it until 5, then 10 and 20.

In Fig. 4 we compare the scenarios with the different number of ants needed. As expected, the most food is collected for only needing 2 ants. With increasing numbers, the food gathering gets less and less efficient. For high values, such as 10, the ants get stuck at some point and wait for other ants in vain. The whole mechanism is stopped and no ant moves anymore. For 20, this phenomenon arrives even earlier (after approximately 250 steps).

2. **Food sources behind a river**

Running the implementation, we can see that upon arriving at the river, the ants start forming bridges, allowing the rest of them to cross the obstacle. As expected the ants cannot gather food at the first few steps of the simulation, as they are occupied with overcoming the obstacle, but soon, as the first ants arrive at the food source, the amount of gathered food keeps increasing, similar to previous simulations. Our implementation moreover allows the ants to are efficient with the building of the bridges, since they tend to form groups to overcome the river. We can quickly notice ants forming bridges and next ants using these bridges to access the food. Just few of the ants do not follow these groups.

Discussion

Because of the limited time available for this mini-project, several perspectives could not be fully explored. In our first experiment, all ants were given the same requirement to cross the barrier, but a more realistic approach would involve food sources with different values. Depending on the richness of each food source, the number of ants required to cross the river and form a bridge could vary dynamically.

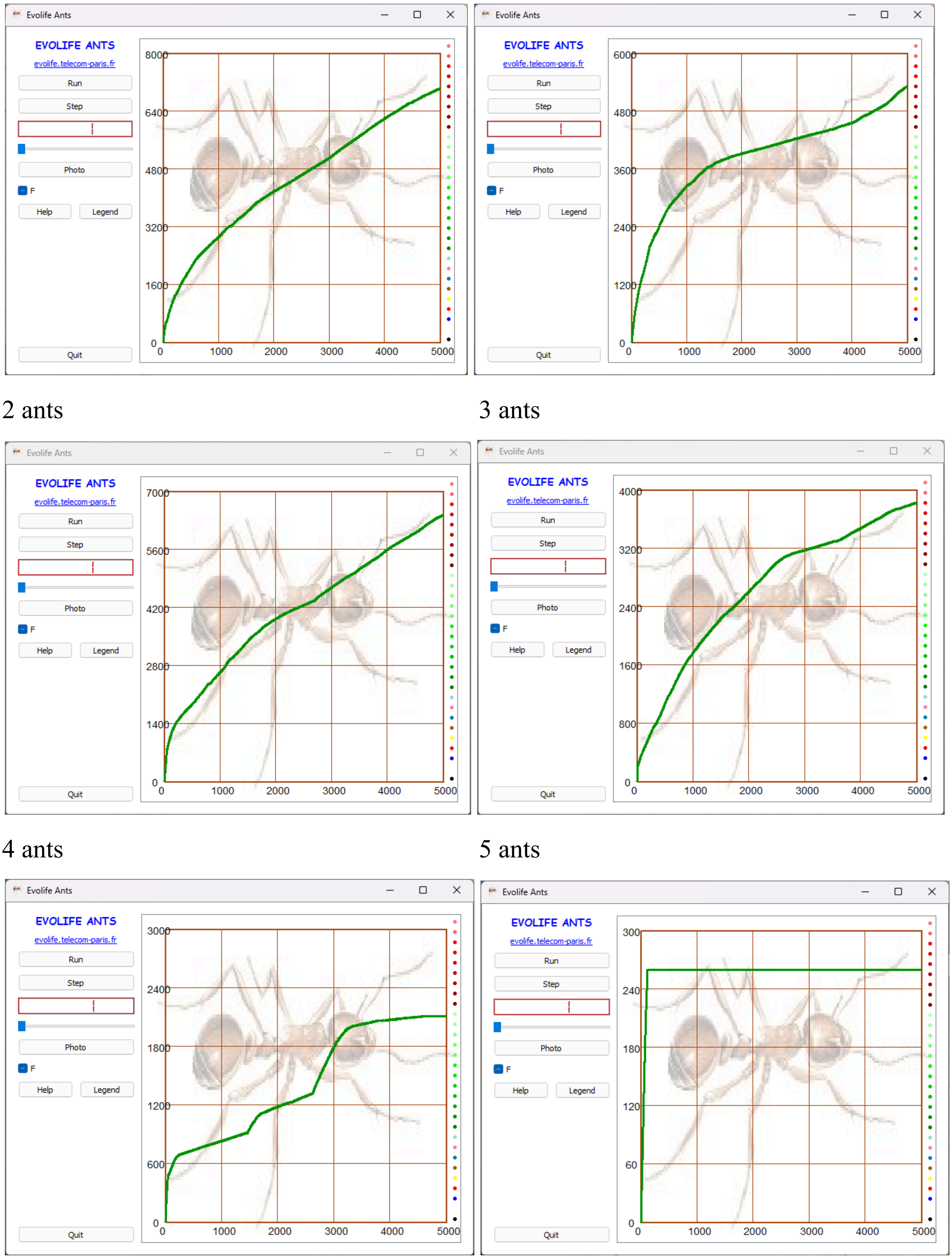
Another improvement would be to design more realistic bridges. In the current model, the bridge is represented as a simple line of ants placed one after another. Future work could incorporate more complex structures, such as layered or adaptive bridges that change shape depending on the width of the river.

A further extension would involve randomly changing food placement over time, instead of relying exclusively on fixed food locations defined at the start of the simulation. This would allow the colony to demonstrate more natural exploration behavior.

It would also be interesting to compare the colony's efficiency with and without obstacles, such as the river, in order to evaluate the impact of terrain complexity on foraging performance. The river itself could also be given a non-fixed or irregular position and shape to study how ants adapt to varying landscape configurations.

Testing the impact of parameters such as the sniffing distance or the minimum number of ants required to start crossing the bridge could also be interesting.

These additions would provide a deeper understanding of how simple local rules can produce adaptive and collective solutions in more complex and dynamic environments.



2 ants

3 ants

4 ants

5 ants

10 ants

20 ants

Fig. 4: Results from the barrier-based approach (first implementation) with different numbers of ants necessary for overcoming the barrier.

Bibliography

[1] Fields, Helen. "Ants That Don't Just Build Architecture, They Become It." GW Magazine, Spring 2016. https://magazine.gwu.edu/ants-dont-just-build-architecture-they-become-it.

YouTube links:

- https://www.youtube.com/watch?v=Wp3Gau-Aljs

November, 2025

ATHENS course : **MOB_0AT09_TP**

Emergence in
Complex Systems

Micro-study

teaching.dessalles.fr/ECS

Names: **Arthur Andrade, Diego Fleury, Guilherme Caporali, Lucas Martim**

Repeated Games in Small World Networks

Abstract

We study an Axelrod-style tournament spanning multiple two-strategy games (Prisoner's Dilemma, Mismatch, Stag Hunt, and Battle of the Sexes) played on networks with varying connectivity, from nearest-neighbor to small-world to fully connected graphs.

Problem

A central question in the study of emergence in complex systems is how *local strategic interactions* (especially when embedded in heterogeneous network topologies) give rise to *global behavioral patterns* that are nonlinear, path-dependent, and difficult to predict from the microscopic rules alone. While classical game-theoretic analysis typically assumes well-mixed populations, real-world systems exhibit *spatial constraints*, *community structure*, and *limited information*, all of which shape how cooperation, coordination, or antagonistic behavior emerge.

In this project, we address the problem of understanding how the *topology of interaction networks* influences collective outcomes in repeated two-strategy games, extending the classical Axelrod tournament beyond the Prisoner's Dilemma to include three qualitatively different games:

- **Mismatch** (zero-sum anti-coordination game),
- **Battle of the Sexes** (asymmetric-gain coordination game),
- **Stag Hunt** (assurance game with risk-sensitive equilibria).

Because each game encodes distinct incentive structures, it is not obvious how *connectivity*, *local majority influence*, and *strategy heterogeneity* interact to shape emergent macroscopic patterns. Limited connectivity can foster cluster formation, fragmentation, or local consensus, while high connectivity may suppress diversity or drive global synchronization. The problem, therefore, is to characterize how these structural and dynamical ingredients interact across a spectrum of graphs (from local nearest-neighbor lattices to small-world networks to fully connected populations).

Ultimately, we aim to understand and explain the collective behaviors that should emerge from each game under varying connectivities. In particular, we expect:

- **Mismatch (zero-sum anti-coordination):** Since no stable pure equilibrium exists, we do not expect persistent global cooperation or competition. Instead, we anticipate *locally oscillatory* or *cluster-biased* patterns, potentially driven by transient dominance of one action in local neighborhoods.
- **Battle of the Sexes (asymmetric coordination):** Fragmented clusters should favor *mixed* or *correlated* equilibria, where populations alternate between preferred actions. Network heterogeneity may sustain coexistence of different local conventions rather than enforcing a single global norm.
- **Stag Hunt (assurance game):** This case is the most structurally sensitive: risk-averse and risk-seeking strategies have asymmetric incentives, making their interactions difficult to analyze analytically. Nonetheless, we expect that, under higher connectivity or sufficiently stable clusters, *high-trust strategies* should dominate by maintaining superior long-run payoffs (although low-trust basins of attraction may persist in fragmented regions). This game is the closest one to the **Prisoner's Dilemma**, making it by far the most interesting.

By formulating the problem in this way, we seek to uncover how *emergent cooperation, coordination, or conflict* arise from the coupling between *game dynamics* and *network structure*, and how small changes in connectivity can induce large qualitative shifts in collective behavior.

Note on strategy mapping: In order to simplify the implementation, the original Prisoner's Dilemma actions (C, D) were naively mapped to the other games as follows: $(C, D) \mapsto (S, H)$ for Stag Hunt, $(C, D) \mapsto (A, B)$ for Battle of the Sexes, and $(C, D) \mapsto (H, T)$ for Mismatch. This mapping allowed the same codebase and strategy definitions to be reused without modification. Importantly, no adjustments were made to the strategies themselves, which were originally conceived for Axelrod-style Prisoner's Dilemma tournaments. Despite this, the simulations still reveal nontrivial and interesting dynamics, demonstrating that the behavior of these strategies in networked repeated games can be informative even when the original context is only approximately preserved.

Method

The core of the simulation relies on repeated local interactions among players embedded in a network. Each agent is implemented as a dedicated `Player` class that stores its strategic identity, action history, accumulated score, and neighborhood structure. At every iteration, a player observes only the most recent moves made by its neighbors and filters this information using the statistical mode, which represents the predominant local behavior. This aggregated signal is then passed to the player's strategy function, which returns the next action according to classical Axelrod-style update rules such as Tit-for-Tat, Grudger, Grofman, Joss, or purely random behavior. All payoffs are obtained by interfacing the chosen moves with a payoff matrix stored in the simulation configuration, and the entire interaction process is kept internally consistent through unique player identifiers that disambiguate the symmetry of the payoff tuples.

The network on which these strategic agents interact is generated through a configurable topology module that supports both Watts–Strogatz small-world graphs and Erdős–Rényi random graphs. In the small-world case, the initial lattice is constructed using $k = 6$ nearest neighbors for $n = 50$ total players, in accordance with the connectivity constraints highlighted by Watts and Strogatz in [3], who require $n \gg k \gg \ln(n) \gg 1$ to avoid disconnected regimes while retaining sparse structure. Several values of k and n were tested empirically, but this particular configuration achieved the best compromise between computational feasibility and adherence to the characteristic signatures of small-world networks, namely high clustering coefficients and short characteristic path lengths relative to the regular ring lattice baseline. The rewiring probability is treated as the experiment's main control parameter, sampled logarithmically to explore qualitatively distinct dynamical regimes.

Finally, all components are integrated into a repeated tournament loop that drives the evolution of the system. For each sampled rewiring probability, the simulation initializes the graph, assigns strategies to nodes, runs the prescribed number of interaction rounds, and records histories, scores, and degree information for all players. The final dataset is compiled per round into a structured format, capturing both the steady-state behavior of each agent and the global distribution of strategies. This modular pipeline ensures that changes in topology, strategy selection, or game specification propagate cleanly through the experimental setup, enabling systematic comparison across connectivity levels and game-theoretic environments.

Results

The final results are presented in the following graphs. For each game, we plotted (under a statistical sample size of 100 runs per sample) the ratio of one of the actions to the other (in alphabetical order) as a function of the connectivity, as well as the overall average score per play for each strategy. The game matrices are presented as well as a reminder:

$$\begin{array}{c|cc} & C & D \\ \hline C & (3,3) & (0,5) \\ D & (5,0) & (1,1) \end{array}$$

Figure 1: Payoff matrix for the Prisoner's Dilemma.

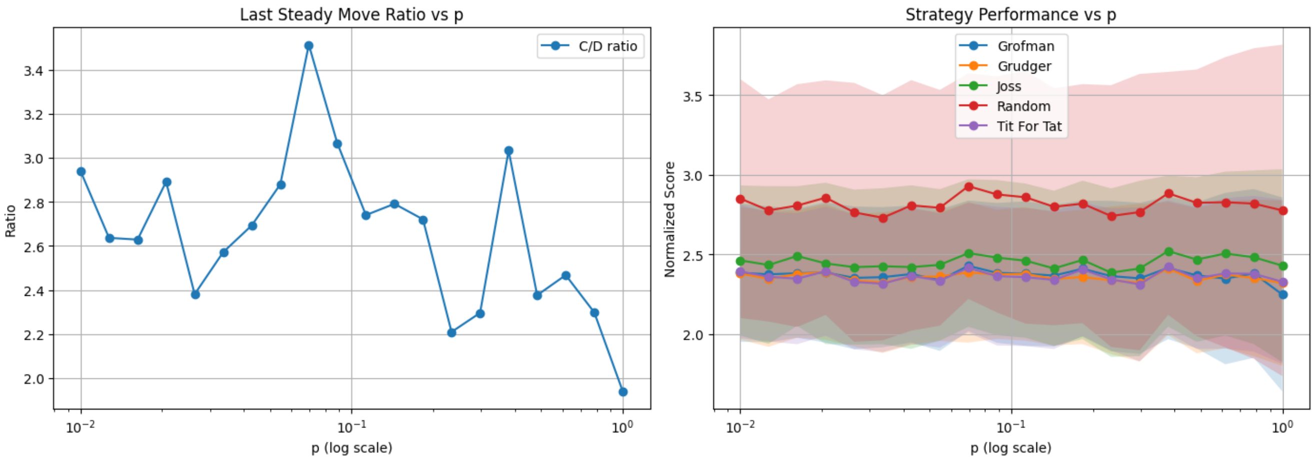


Figure 2: Prisoner's Dilemma graphs

$$\begin{array}{c|cc} & S & H \\ \hline S & (5,5) & (0,3) \\ H & (3,0) & (1,1) \end{array}$$

Figure 3: Payoff matrix for the Stag Hunt game.

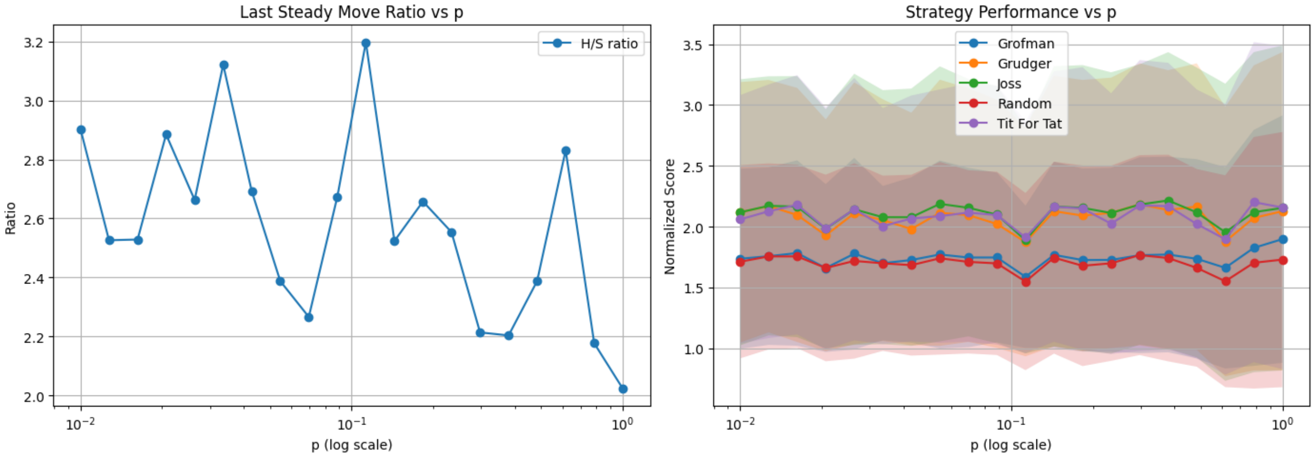


Figure 4: Stag Hunt graphs

$$\begin{array}{c|cc} & A & B \\ \hline A & (3,2) & (0,0) \\ B & (0,0) & (2,3) \end{array}$$

Figure 5: Payoff matrix for the Battle of the Sexes coordination game.

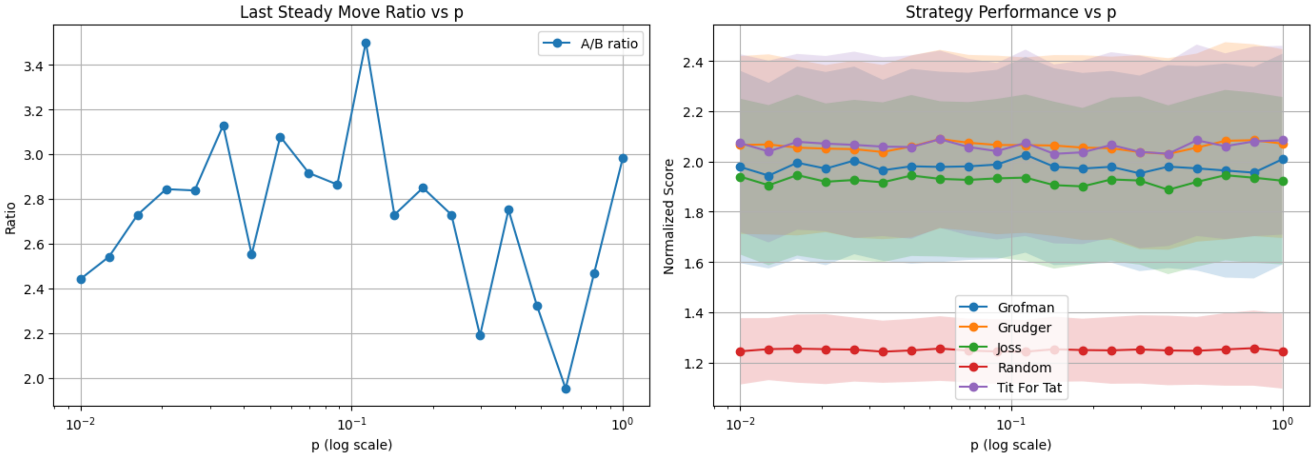


Figure 6: Battle of the Sexes graphs

$$\begin{array}{c|cc} & H & T \\ \hline H & (-1,1) & (1,-1) \\ T & (1,-1) & (-1,1) \end{array}$$

Figure 7: Payoff matrix for the Mismatch game.

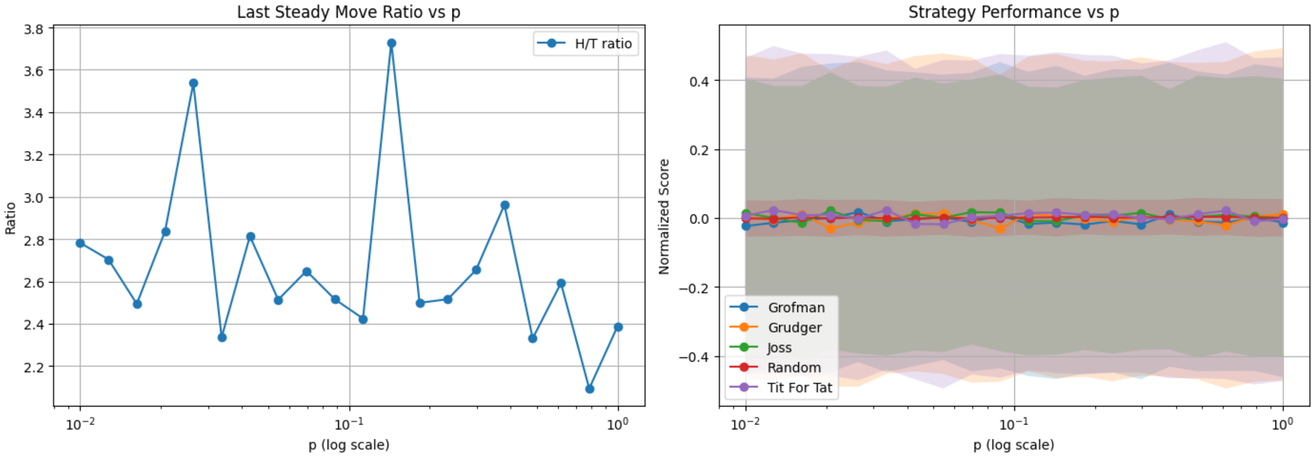


Figure 8: Mismatch graphs

Discussion

0.1 Prisoner‘s Dilemma

As we can see from figure 2, it seems that as the connectivity varies (as represented by the increase in the hyperparameter p for the probability of redirecting an edge) there seems to be an non-monotonic overall decrease in the amount of times that the system tended towards defection in the steady state. These findings suggest that in real world situations, where social circles tend to form small world networks [2], conditioning one's behavior on the local mode of others' actions may lead to collective outcomes that do not vary monotonically with network density. In such networks, the combination of highly clustered neighborhoods and occasional long-range social ties can amplify, dampen, or fragment local patterns of defection, depending on how information and behavioral imitation propagate.

As the overall amount of defectors in steady state seems to increase as the network becomes progressively more random, this would suggest that more chaotic structures tend to favor more defective behavior, as clusters become more fragmented and the average path length decreases, allowing defection to spread more rapidly across the graph. However, the overall proportion of cooperators still remains dominant (varying throughout the experiment from roughly 3.5 to 2 times the number of defectors), suggesting that it may still be the case that "nice" strategies can prevail in the long run, in line with the seminal findings of Axelrod's tournament [1].

Looking into the particular strategies performances', we can see that (surprisingly !) *tit-for-tat* didn't seem to prevail in this setting. This doesn't exactly contradict Axelrod's results [1] as the network topology and the type of strategies do influence a lot which strategies win, and *tit-for-tat* only won **on average**. It's interesting to see though that most of the more complicated strategies seem to be significantly behind the simpler 50-50 mixed strategy, with an average payoff of about 2.8.

If we were to assume that the overall effect of the other players' choices are a result of a Bernoulli distribution $\mathbb{B}(p_c)$ (that is, that one can assume that they cooperate with probability p_c and defect with probability $1 - p_c$) we can estimate its latent overall probability by this. The expected payoff for the mixed strategy player, given that it plays against such a mixed strategy, is given by :

$$\mathbb{E}[\mathbb{B}(p_c)] = \frac{1}{2}\mathbb{E}[\mathbb{B}(p_c)|C] + \frac{1}{2}\mathbb{E}[\mathbb{B}(p_c)|D] = \frac{1}{2}(3p_c + (4p_c + 1)) = \frac{7p_c + 1}{2} \tag{1}$$

If we set that its final payoff (experimentally) was given by about 2.8, we can therefore infer that $p_c \approx 0.66$. These finds are interesting, not just because they suggest an $\approx \frac{1}{3}$ probability of defection, but also because they disagree with the steady state ratios (which would suggest that $p_c/p_d \approx 3$ for highly regular networks, and just at the random end would tend towards 2). This suggests a significant shift in the dynamics from the departing state, which could explain why the more complex (rule-based) strategies tend to concentrate themselves at the lower end of the normalized score table (with around 2.4 as a payoff).

The fact that connectivity produced a noticeable shift in the population steady state while producing only minor changes in the relative performance of individual strategies suggests several (non-exclusive) interpretations (as well as a high **dynamicity**). First, it may indicate that the *ranking* of strategies is relatively robust to global rewiring: although the aggregate composition of C vs D changes with topology, the payoff differences between strategies are largely preserved because all strategies experience the same change in the background environment (i.e. the same shift in the effective p_c they face). Second, our specific decision rule (agents choosing according to the local *mode* of recent neighbour actions) can act as a strong homogenizing mechanism that reduces the value of sophisticated contingent rules which suggest that simple stochastic strategies (like the 50–50 mixed rule) may therefore achieve competitive average payoffs across topologies. Third, aggregate steady-state ratios (global p_c) can conceal spatially heterogeneous effects: it is possible that some strategies do much better in specific network niches (high-clustering cores, hubs, or periphery) while doing worse elsewhere, so that the population average hides complementary successes and failures.

0.2 Stag Hunt

For this game, we analyse the impact of adding another pure strategy Nash equilibrium, which coincides with the global optimal. This is done by reducing the magnitude of the equivalent to the "defective" action in this game (unilaterally deviating to hunting a hare when the other hunter decided to hunt the stag, and you've already agreed to hunt the stag) to the point that is no longer advantageous to do that. As we can see in figure 4 on the steady state graph, the dynamics are very similar to what was observed in the Prisoner's Dilemma, but inverted : the defective action becomes less prevalent in the population as connectivity increases, going from three times more likely initially to about twice at the end. This aligns with the coordination nature of Stag Hunt, where higher connectivity facilitates convergence to the globally optimal cooperative strategy. Overall, network structure again plays a key role in shaping the population dynamics and the success of different strategies.

We can also see, looking at the strategies' performances, that the dynamics more or less conformably shift : random becomes the least successful strategy (interestingly enough, extremely correlated in score with the Grofman strategy), with an average payoff of about 1.75, while most of the other strategies seem to be averaging about 2.1 as a payoff.

Again, modeling the sum of the other strategies' contributions as equivalent to a Bernoulli mixed strategy, we can estimate the latent probability of playing Stag, p_S and a 50-50 mixed strategy, the expected payoff is

$$\mathbb{E}[\mathbb{B}(p_s)] = \frac{1}{2}\mathbb{E}[\mathbb{B}(p_s)|S] + \frac{1}{2}\mathbb{E}[\mathbb{B}(p_s)|H] = \frac{1}{2}\left(5p_S + (2p_S + 1)\right) = \frac{7p_S + 1}{2}. \tag{2}$$

Setting the experimental payoff to about 2.1, we infer

$$2.1 = \frac{7p_S + 1}{2} \quad \Rightarrow \quad p_S = \frac{3.2}{7} \approx 0.46. \tag{3}$$

This suggests that approximately 46% of the time the non-random population chooses Stag on average.

In order to further inquire why Grofman scored closely to random, we need to first put into context what it does: in the Prisoner's Dilemma setting, Grofman is essentially a conditional cooperator with a small probability of forgiveness, playing C by default and only defecting occasionally in response to opponent defection. In the Stag Hunt setting, however, the strategy is inverted: $m_1 = H$ and $m_2 = S$, meaning it defaults to the "risk-averse" Hare action and only occasionally attempts to coordinate on Stag after an opponent played Stag previously.

This inversion has important consequences. While in the Prisoner's Dilemma Grofman could sustain cooperation by exploiting reciprocity and the forgiving mechanism, in the Stag Hunt it is biased toward the safer Hare action, which limits its ability to consistently reach the globally optimal Stag outcome. As a result, Grofman's behavior aligns closely with that of a random 50-50 strategy in terms of payoff, since the strategy rarely converges to the high-reward Stag coordination, yet still occasionally "forgives" and plays Stag with low probability.

The consequence of this risk-averse tendency is that Grofman rarely converges to the globally optimal Stag outcome, instead favoring the safer Hare action. While this limits its potential payoff, it also protects it from the low-reward outcomes associated with unilateral Stag deviations. In contrast, more risk-seeking strategies, such as Tit-for-Tat or Joss, are willing to attempt Stag coordination, which can yield higher payoffs in clusters where neighbors also attempt Stag (which seems to be the case, as 3/5 strategies do indeed favor stag over hare from the looks of the average payoffs), but at the cost of occasionally receiving low payoffs when coordination fails. Random, which is effectively agnostic to risk, ends up performing similarly to Grofman because the networked environment spreads Stag attempts thinly, preventing consistent high-payoff coordination for either. We can see that in the brute normalized scores, as the value of 2.1 is very far from the optimum 5 in a fully cooperated scenario.

0.3 Battle of the Sexes

This game is an interesting one. As we can see, coordination games are an interesting case study for the current set of strategies : Joss and Grofman are pretty much slightly modified versions of Tit-for-tat (the only difference being that Joss certainly chooses B if the opponent chooses A, while most likely choosing A - with probability 90% - if the opponent has chosen A, meanwhile Grofman certainly chooses A if the opponent has chosen A, while having a chance of 2/7 A if opponent has chosen A). These slight probabilities of not being in accordance to the opponent in certain cases is most likely to be the reason to explain why they present slightly lower average scores than tit-for-tat and Grudger.

The strategies tit-for-tat and Grudger have a common characteristic : they are both retaliatory. This means that they'll certainly continue to play B in the case that the mode of their opponents keeps playing B, being therefore more likely to keep the game in a predictable B state, letting the payoffs from coordination be more likely.

The random strategy, not surprisingly, scored the worst, being centered around a poor 1.25 average payoff (which contrasts very much with the ≈ 2.1 from tit-for-tat/Grudger and ≈ 1.9 from Joss/Grofman), yielding (by the previous assumptions of a mixed Bernoulli equilibrium as an approximation of the collective behavior of the other 4 strategies) we have :

$$\mathbb{E}[\mathbb{B}(p_A)] = \frac{1}{2}\mathbb{E}[\mathbb{B}(p_A)|A] + \frac{1}{2}\mathbb{E}[\mathbb{B}(p_A)|B] = \frac{1}{2}\left(3p_A + (2 \cdot (1 - p_A))\right) = \frac{1}{2}\left(2 + p_A\right) \quad (4)$$

As we have that, on average, this strategy yielded about 1.25 normalized average return, we have that $p_A \approx 2 \times 1.25 - 2 = 0.5$. Therefore, the empirical results show that the other 4 strategies have converged on a mixed 50-50 strategy as a whole. This, alongside the fact that the actual 50-50 strategy hasn't achieved such performance suggests that about half of the network has converged to (A,A) and the other half to (B,B) during the procedure.

The last steady state move ratio graph shows that, across different levels of connectivity, the ratio A/B changes a lot, but remain around approximately 2.7. This indicates that, although the network allows for fluctuations in local coordination, there is a persistent bias toward action A at the population level. In combination with the Bernoulli estimate, this suggests that while half of the network tends to coordinate on (A, A) and the other half on (B, B), action A enjoys a slight predominance, which could be attributed to the "niceness" of most strategies (almost never being the first to change to B).

0.4 Mismatch

For this game, we have that (as pretty much expected) there was no clear coordination/defection imbalance that influenced the outcome in any significant way. In this setting, most players default to playing "H", while unilateral changes to "T" can be beneficial to one party while being onerous to the other one. As one player tries to agree on the other's prediction in order to score while the other tries to disagree on the other's prediction, the initial configuration of the graph (alongside some small clustering differences influenced by the change in connectivity) pretty much dictates the final state.

As we can see from the last steady state move ratio, it seems that "H" has been favored overall (as expected by the "niceness" of the strategies), with about a 2.6 prevalence over "T". This, alongside the findings that the average normalized payoff for every strategy is centered around zero, suggests that the remaining defining element for the expected payoff for each player (whether or not it is player 1 or player 2, defined pretty much just by it's relative count of nodes "higher" than it versus "lower" than it, in the ring topology ordering) is relatively balanced in its nature, that is, the change in connectivity during the crafting of the small world network was as likely to reconnect it clockwise than it was to connect it counterclockwise. As discussed in the Prisoner's Dilemma subsection, the mode aggregator is also likely to play an important role as a strong homogenizing mechanism here.

Bibliography

References

[1] Robert Axelrod and William D. Hamilton. The evolution of cooperation. *Science*, 211(4489):1390–1396, 1981.

[2] Social networks tend to organize themselves in the form of small world networks, as popularized by Stanley Milgram‘s average path length observations.

[3] Duncan J Watts and Steven H Strogatz. Collective dynamics of ‘small-world’ networks. *Nature*, 393(6684):440–442, 1998.

November, 2025

ATHENS course : **MOB_0AT09_TP**

Emergence in Complex Systems

Micro-study

teaching.dessalles.fr/ECS

Ana Margarida Almeida, Vasil Georgiev, Josef Rabmer, Séthy Herlidou

Schelling Segregation with probabilistic multi-variable tolerance

November 2025

Abstract

Investigate how the diversity of a population changes in a probabilistic model of Schelling Segregation, where agents want to move if they are too different but also too similar to their neighbors. Additionally, we want to see what happens when agents move depending on some probabilistic distribution instead of always moving right away.

Context

The group follows the tracks of the Schelling segregation problem. Just a reminder :

- Hypothesis: Agents have only weak segregationist local behavior, in the following sense: each agent wants at most 50% neighbors that differ from her/him; otherwise agents are indifferent.
- Observation: people tend to form groups and neighbourhoods that are not diverse at all, meaning that they tend to stay only with the people who are similar to them.
- Conclusion: Small individual bias leads to large collective bias. In our example, it leads to a fractured and segregated society.

Problem

The group aims to enrich this model, mainly regarding two aspects :

- what does happen if the people start demanding a bit of diversity in their neighbourhood?
- the age of the people: as they get older, people tend to stay in the same neighbourhood they grew up in. We'll implement that and see what effect it has on society. Does it make it more or less segregated? Does something unexpected happen?

Method

The group will implement different ideas and test them one by one to estimate the effects it has on the simulation.

Although the model has stochastic movement decisions, the group wants to make the initial state deterministic to make sure that comparisons are relevant.

Results

Lower threshold

Definitions :

- Tolerance: Percentage of individuals that are different (= different / (same + different)) above which one decides to move.
- Lower threshold : Percentage of individuals that are different (= different / (same + different)) below which one decides to move.

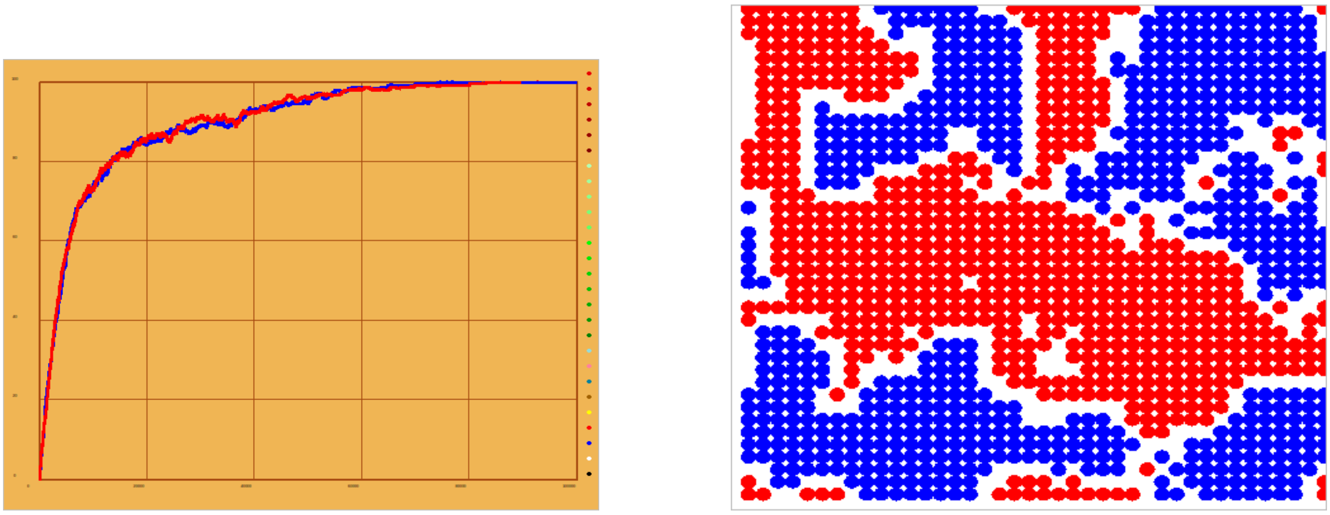


Figure 1: Tolerance 25, No Lower

People are unhappy and move if more than 25% of their neighbour are different than them.

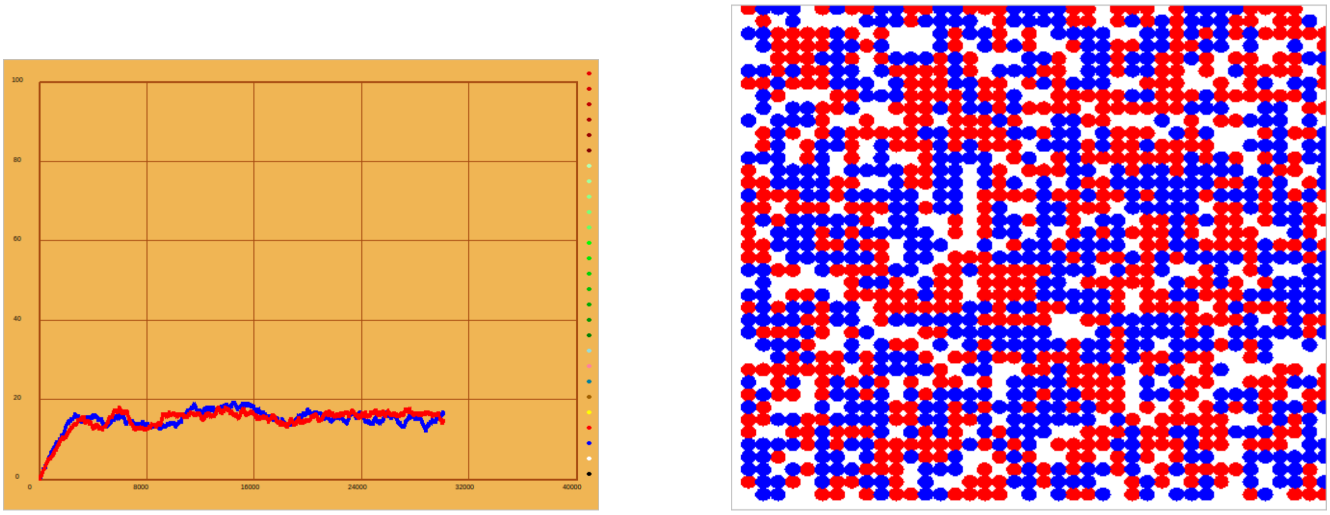


Figure 2: Tolerance 25, Lower 10

Observations and comments : we can see that folks are not satisfied at all with this configuration, whether they are blue or red. That makes sense : people move if less than 10% of the neighbours are different than them, or if more than 25% of their neighbour are different than them. They can never be satisfied with such a tight window.

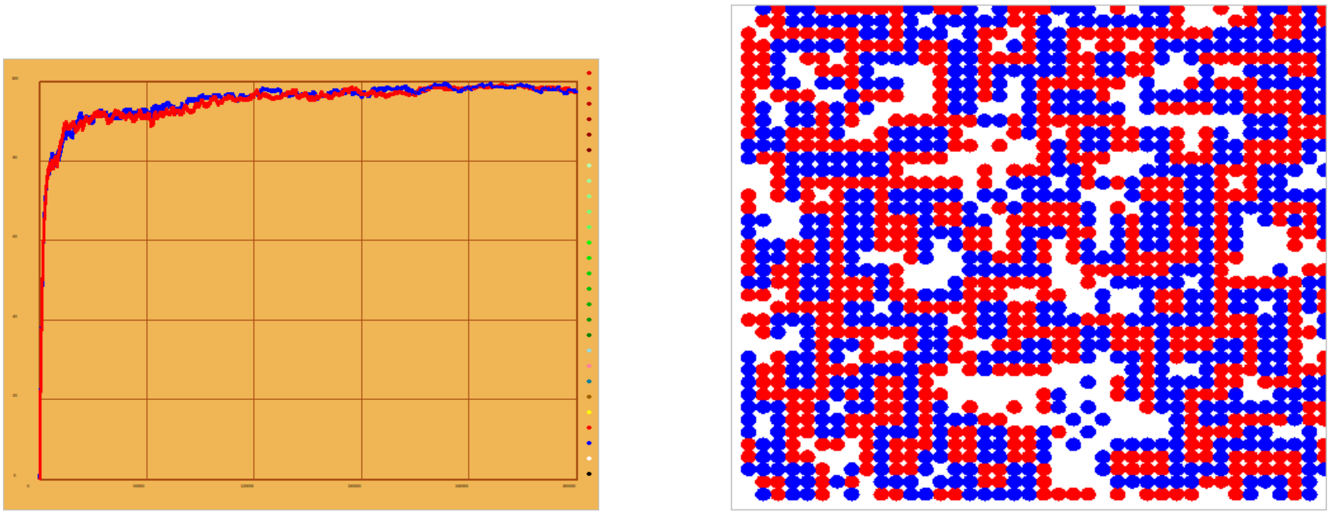


Figure 3: Tolerance 70, Lower 20

People move if less than 20% of the neighbours are different than them, or if more than 70% of their neighbours are different than them. Because the lower tolerance level is quite high, red cannot just form red neighbourhood, and the same goes for blue. Because there is still a threshold on the number of different people they accept, a much larger in between exists. That allows for very specific spacial patterns to appear. Red and blue neighbourhood are less crowded, but they tend to cross with neighbourhood of the opposite color.

Agents with Age

Abella et al. [1] introduce the concept of agent age to the basic Schelling model.

In this context, an agent's "age" is defined not by biological time, but by the time it has remained (satisfied) in its current location. The central idea is that the longer an agent stays in a satisfactory neighborhood, the more it develops an emotional or economic attachment to that place, simulating real-world factors like establishing links with schools, public goods, and the overall community.

This attachment translates into a mechanism where the probability of a satisfied agent moving out is inversely proportional to its age in that location. Essentially, the older the agent is in a satisfying spot, the more resistant it becomes to relocation, even if other factors might prompt a move. This adds a form of agent memory to the otherwise stateless agents and helps simulate real world phenomena more closely.

In our instance we model the agents willingness based on its age with an inverse relation:

$$P_{move}(j) = \frac{1}{\tau_j + 2}$$

where τ_j is the age of agent j. In our context the age is calculated using the simulation steps in Evolife.

Agents become less likely to move with age

In this instantiation an agent j checks if it is satisfied based on its neighbors like in the original model. If it is unsatisfied, it wants to move with a probability of $P_{move}(j)$.

This simulates the agent's desire to move. As time goes on and the agent stays at this location, its probability to move decreases. This is supposed to model a real person becoming familiar in an environment and its desire to move decreasing.

If it does not move, it is still unsatisfied and will try to move again the next time it gets selected again. Under the assumption that its satisfaction in the location does not change because of other agents moving.

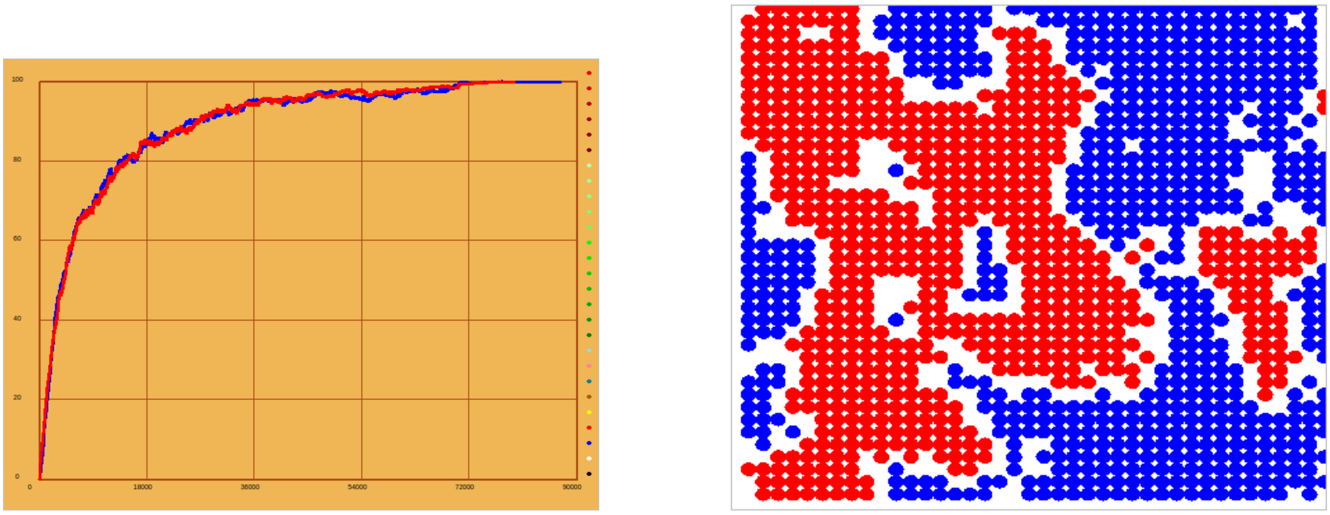


Figure 4: Tolerance 30, No Lower Tolerance, Without Age

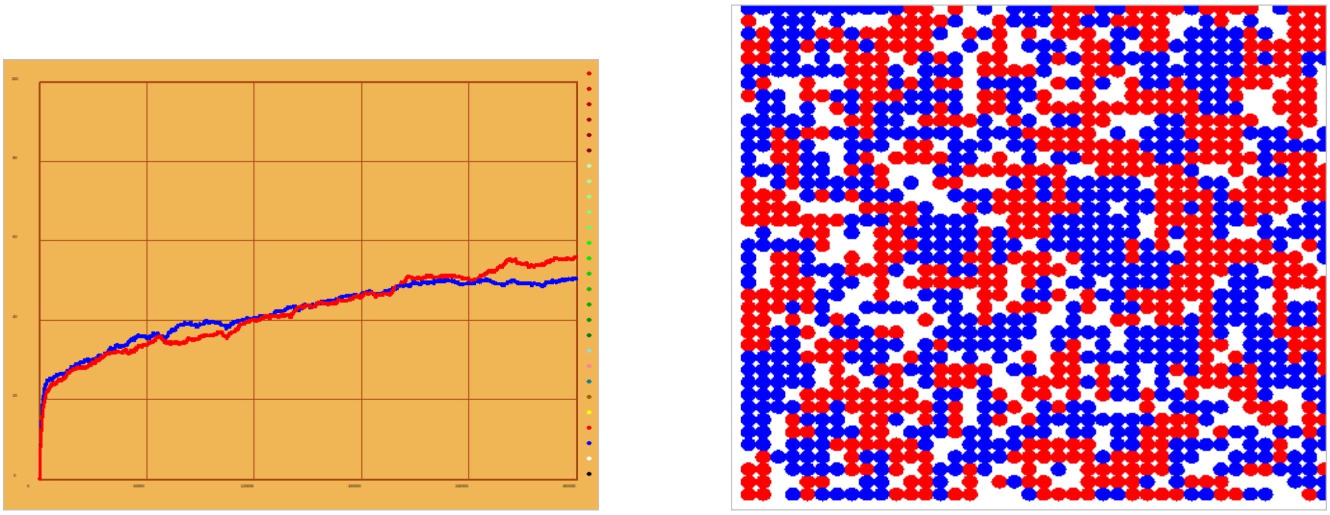


Figure 5: Tolerance 30, No Lower Tolerance, With Age

Agents get tolerant to their neighbors

This instantiation is conceptually similar to Agents become less likely to move with age, but here age influences satisfaction rather than movement probability. Instead of remaining unsatisfied but still not moving, the longer an agent stays in a location, the more tolerant it becomes toward its neighbors, independently of their color.

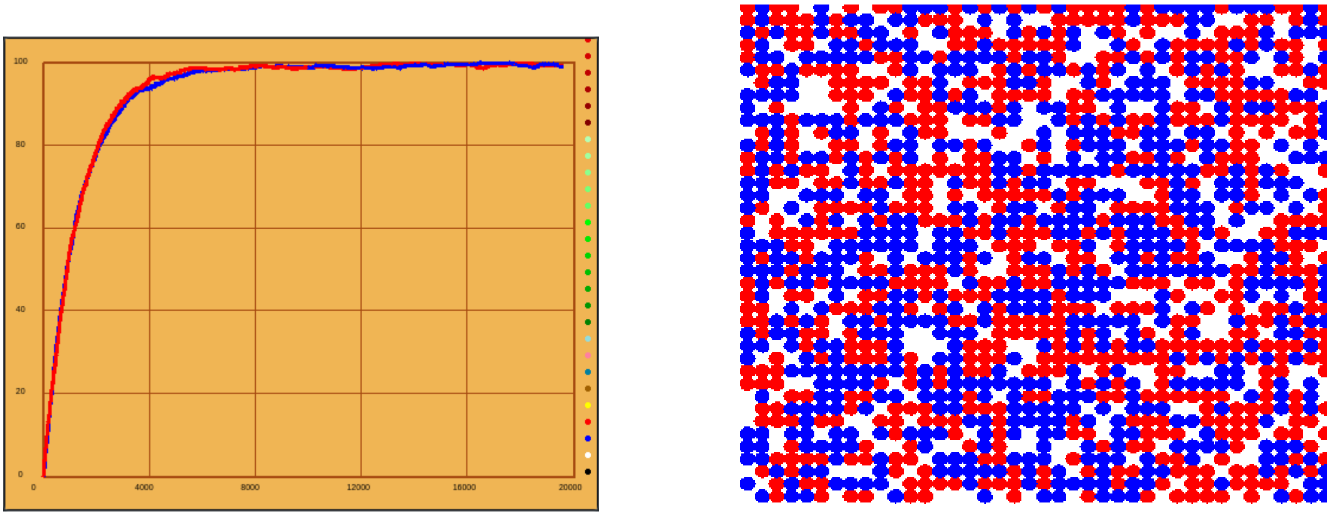


Figure 6: Tolerance 30, No Lower Tolerance, With age affecting tolerance toward neighbors

Agents Tolerance scales with age

In this simulation, an agent becomes more tolerant to the place they live the longer they live there. In the beginning, all agents start with the same base tolerance, but as time progresses and they do not move, their tolerance increases. However, if an agent decides to move, his tolerance is reset to the base level.

This model is supposed to simulate an agent's growing satisfaction with their neighborhood. As time progresses, agents become more accustomed to their neighborhood, so they need stronger reasons to move. As this model provides only 'racism' as a reason, the natural move would be to reduce that statistic.

This is different from the other simulations because the agents are actually satisfied in their neighborhoods. In the first model, the agents may never be satisfied, and they move based on random chance. In this model, movement declines naturally over time, not because the neighborhoods become perfectly balanced, but because the agents adapt psychologically to where they live. Even if their surroundings are not ideal, their rising tolerance makes them increasingly willing to stay, so segregation (if it forms) tends to stabilize due to habituation rather than persistent dissatisfaction. The other model appears to make them more tolerant, but that is actually on to their neighbors, so if someone new moves in, that would automatically reset them.

Neither model is more accurate or realistic, but they serve the purpose of illustrating different behaviors and interactions between age and segregation.

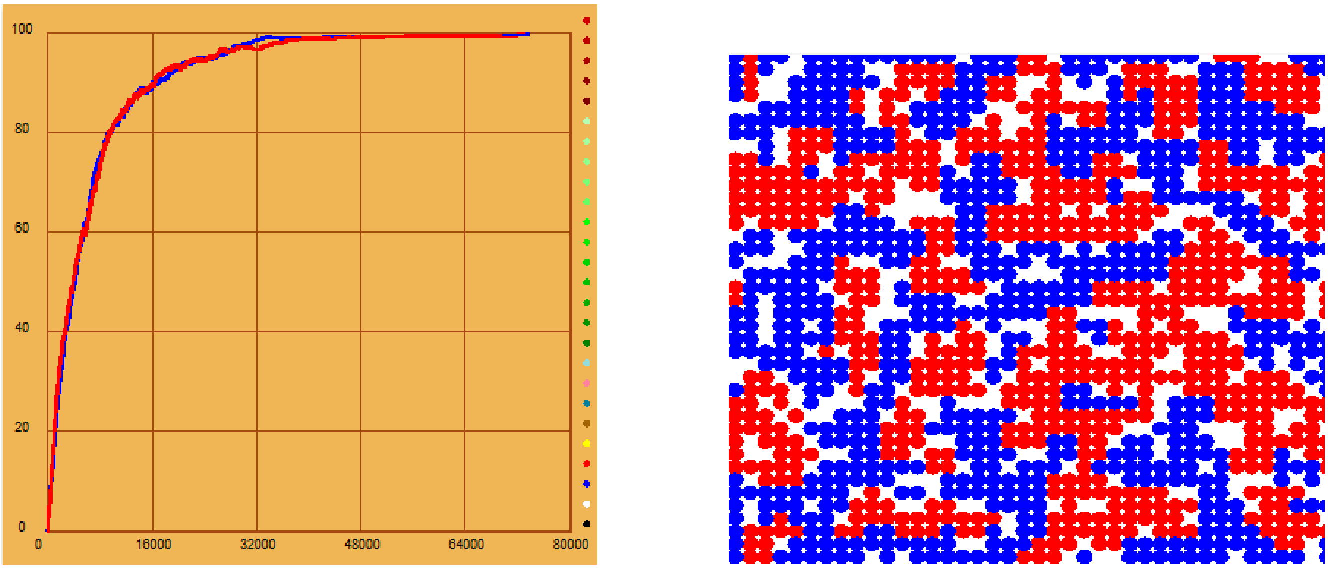


Figure 7: Tolerance 30, No Lower Tolerance, With Age increasing Tolerance

Discussion

Neither model is more accurate or realistic, but they serve the purpose of illustrating different behaviors and interactions between age and segregation.

Comparing these models is challenging because age influences each of them in different ways. Therefore, measuring only the final degree of segregation is insufficient to assess the dynamics of segregation or to understand the role that age plays in those outcomes.

Regarding the approach Agents become less likely to move with age, we initially assumed that while agents would move a lot at first, they would eventually settle into a stable outcome where most agents are satisfied. This turned out to not be the case. The simulation did slow down because agents became less and less likely to move but they did not reach a very high level of satisfaction and the simulation never actually terminated.

In contrast, in the Agents get tolerant to their neighbors approach, the outcome of the simulation shows a neighborhood with significantly less segregation and a higher overall satisfaction level.

References

[1] David Abella, Maxi San Miguel, and José J. Ramasco. “Aging effects in Schelling segregation model”. en. In: *Scientific Reports* 12.1 (Nov. 2022). Publisher: Nature Publishing Group, p. 19376. ISSN: 2045-2322. DOI: 10.1038/s41598-022-23224-7. URL: https://www.nature.com/articles/s41598-022-23224-7 (visited on 11/20/2025).

November, 2025

ATHENS course: **MOB_0AT09_TP**

Emergence in

Complex Systems

Micro-study

teaching.dessalles.fr/ECS

Name: **Sam Pegeot, Yangtao Fang, František Špaček, Jan Svoboda**

Social bubbles

Abstract

Segregation by niche, genre or type is not desired in many systems (instagram recommendations, spotify algorithm, movie recommendations, extremism on facebook).

This has been shown in the "bubbles" experiment. We want to research ways to prevent this extreme division while preserving some quality of "recommendation".

Problem

We want to research interactions between individuals, which produces clusters (genres, niches, recommendations), but does not lead to extremely dense clusters. We focus on influence between individuals and not the effect of recommended movies (green dots), see the Discussion/Other observations sections for explanations about this choice.

First part of the problem is deciding how to objectively compare methods, since visual differences between results are not reliable. We will propose an empirical metric, which balances two opposite effects. We want to encourage clustering, but penalize clusters, which are too tight.

Second part is establishing baselines for comparison. Note, that we do only change the parameter “neighborhood radius”, since other parameters change how individuals interact with new content (green dots) and we need to fix these parameters.

Lastly, we show several new methods of sampling neighbors with the resulting metric.

Method

Adaptation of Evolife:
We spent some time with Evolife, so we can display colours of different clusters, display different objective metrics and print the results to the console. All changes are in the file Bubbles.py

Proposed metric:

We decided to use K-means clustering (and its objective function) as a proxy to compactness of the clusters, which we want to improve. In each step, when we move some members of the population, we do K-means clustering starting from previous cluster centers (for stability). We take the WCSS criterion of the K-means clustering and use sqrt() function to counteract the second power in the formula for WCSS. We then normalize by the number of data.

As a proxy for how packed the data is (which we want to discourage), we calculate the average number of neighbours of each data point (8 directional).

As a final metric, which balances these 2 values (which have very similar ranges), we use a weighing factor of lambda. For our experiments, we selected lambda 1, to produce a simple sum. The objective is to be minimised. The problem is, that you need to decide beforehand, which weight you want to attribute to each effect.

$$\mathrm{Obj} = \frac{\sqrt{\mathrm{WCSS}}}{N} + \lambda \cdot \mathrm{PatchPenalty}$$

$$\mathrm{WCSS} = \sum_{k=1}^{K} \sum_{x_i \in C_k} \|x_i - \mu_k\|^2$$

$$\mathrm{PatchPenalty} = \frac{1}{N} \sum_{i=1}^{N} \mathrm{Neighbors}_8(i)$$

Image 1: Metric for forming loose clusters.

Sampling of neighbors

Default sampling:

After one individual moves due to a new trend being introduced (green dot), we select its direct neighbors which will be moved to. The default method of moving the individuals, which is implemented in evolife is probabilistic selection with uniform kernel. We select the size of the neighborhood, and the percentage of the neighbours to select. Each individual in the neighborhood has the same chance of being moved with the group.

Other sampling methods:

The sampling happens at the same phase as with the default sampling, after an individual moves, we select some of its neighbors and move them according to a sampling rule. The images are purely for illustrative purposes. Since the movement happens on a grid, the functions are actually discrete and not continuous.

Sampling case 1

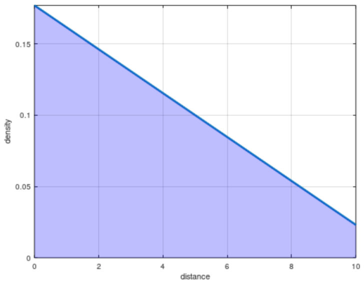


Image 2: Linear decreasing sampling density, based on distance from the individual

Sampling case 2

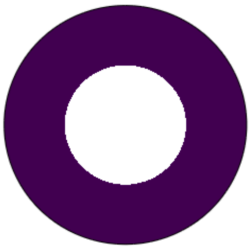


Image 3: In this case, the closest data is not moved, and others up to a certain distance may be moved with uniform probability.

Sampling case 3

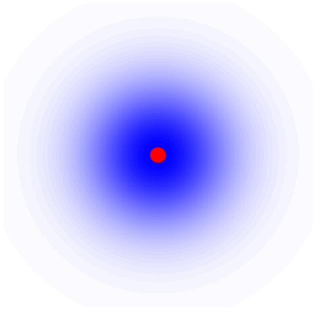


Image 4: Gaussian probability density used for sampling.

1. Linear decreasing sampling probability

This method is biased towards influencing the closer individual, while still giving a chance to move those who are further away. It scales based on the neighborhood and overall retains the same odds of some individual being influenced as the default method.

The idea is to keep some cohesion without instantly pulling whole neighborhoods, which reduces the tendency for groups to fuse too aggressively.

2. Donut sampling

In this case, we try an opposite approach, which means that we ignore the neighbors who are in close proximity to the individual and only influence those who are further away.

The idea behind this is that we want the groups to stay coherent, but want to discourage the individuals from forming a solid core. This means that we move the larger neighborhood but don’t influence those who are too close, which should prevent the formation of highly concentrated clusters.

3. Gaussian sampling

This method is similar to the linear sampling method, in the sense that we influence the close neighbors strongly while retaining some amount of influence over the broader neighborhood. The difference is that we use a normal distribution, which causes higher bias towards close neighbors.

Sampling is still limited to the neighborhood area, so we have to cut the tailing values, which means it's not a true normal distribution. The intention is to retain smaller compact groups but prevent huge solid neighborhoods.

4. **Global sampling**

For this method, we decided to completely ignore distance limitations and took samples from the entire population. The distribution is uniform, so it acts basically as the default method with an infinite neighborhood.

The idea was to test whether broad influence can prevent the formation of tight local clusters by constantly mixing population members.

Random noise

The issue we discovered with the sampling methods was that the movement was only initiated by movies. So when a cluster emerged, it stayed that way until some force acted on it. And this pulled the whole neighborhood together. So once an individual became part of a cluster, they were unlikely to leave. This could be somewhat mitigated by reducing the radius of influence, but this often left stranglers.

In contrast to the sampling methods, this approach doesn't work based on influence of other individuals. Instead, each individual has a small chance to move slightly, regardless of what's happening around them. That means that there's movement, even when no changes are being caused by movies and other individuals.

We set a 10% chance for a given individual to move during the simulation step, the direction and distance of the random movement were combined, we simply randomly choose one of the eight neighboring cells.

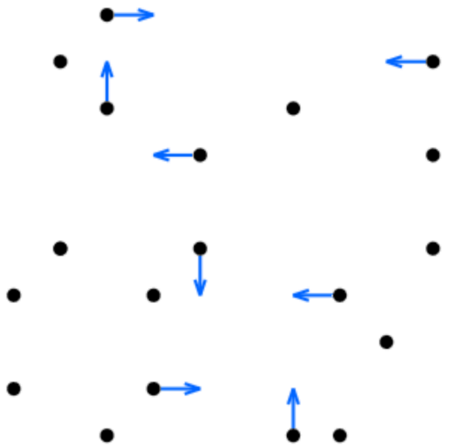


Image 5: Visualisation of the random noise. The individuals move in random directions with probability p.

Results

Exploration of different hyperparameter combinations:

Here we can see the achieved values of WCSS, patch penalty and the full metric for different combinations of parameters. This is to get a feel for how the parameter balance between influential neighbors and influential trends (movies etc. = green dots) changes the result of the simulation.

The experimental results are shown in Images 6, 7, 8, and 9. Each graph has one fixed Influence Radius (IR) and only Neighborhood radius (NR) changes between 1,5,10,25 and 50. Note that the X axis is on a logarithmic scale. These graphs are based on data, which can be found in the appendix in Table 3.

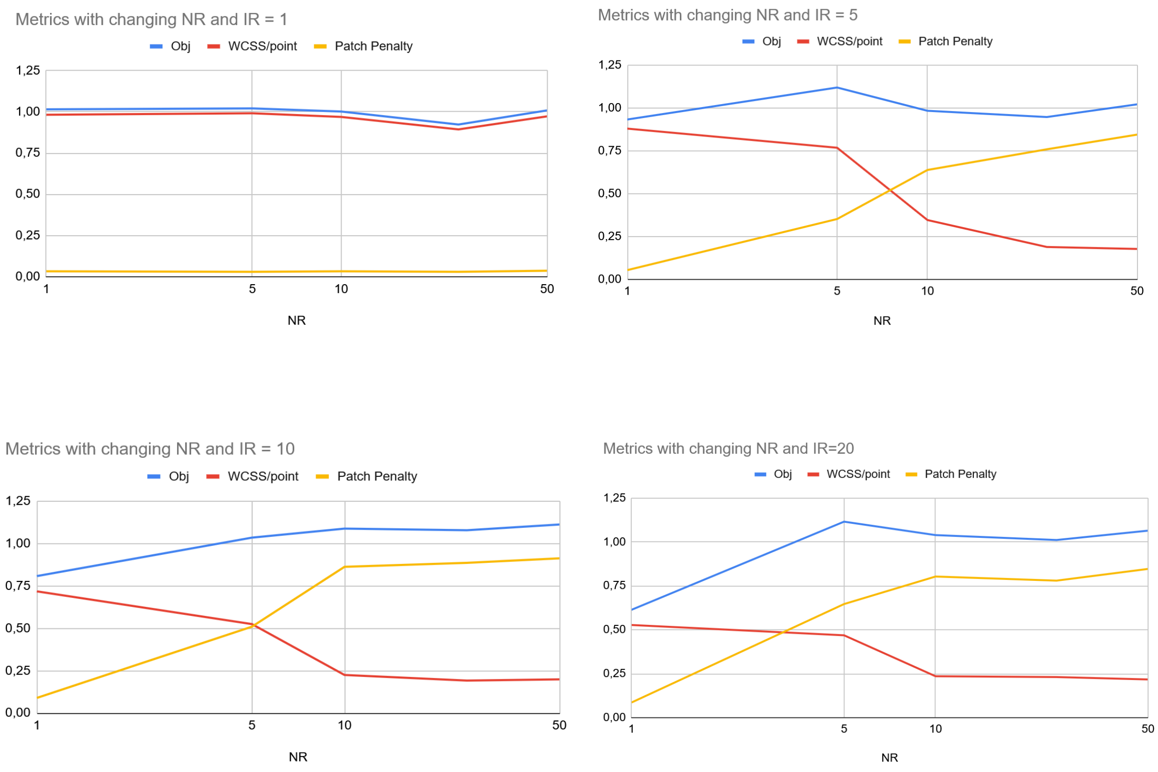


Image 6: Metrics for different NR with fixed IR (1,5,10,20)
Red = WCSS/N
Yellow = patch penalty
Blue = Metric

Baseline:

The core goal of these baselines is to demonstrate how the scope of social imitation (the radius) affects the two counteracting forces in your metric: compactness (WCSS/point) and over-packing (PatchPenalty), while keeping the external film influence fixed.

Table 1: Baseline performance with
InfluenceRadius:10 InfluenceRatio:50 and Influence:20

	NR	**Objective**	**WCSS/point**	**PatchPenalty**
B1: Dominant External Force	5	0.7640	0.5590	0.2050
B2: The Balanced Trap	10	0.8606	0.3256	0.5350
B3: Dominant Social Force	30	0.9910	0.1802	0.8108

Correction methods:

In the graph below, we show the objective value and each component in relation to different sampling techniques. Neighborhood ratio is fixed on 10 and all other parameters are the same as when calculating baselines. This graph is based on Table 2, which can be found in the appendix and also the table above.

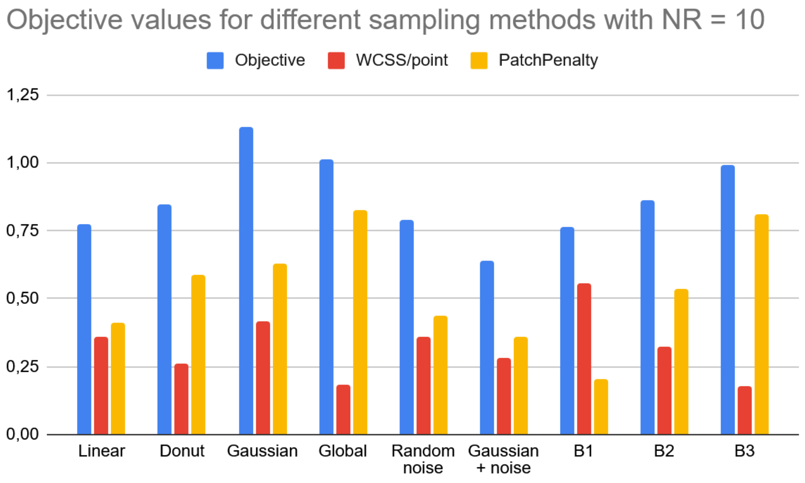


Image 7: the performance of each method of neighbor sampling and our baselines.

Discussion

Shortcoming of the metric

The metric was handcrafted for our scenario and does not have a root in theory. Please know that it is a proxy, and not a definitive measure to compare models. The need to set a fixed number of clusters and lambda is a detriment of the method, we needed to tune it by hand and might not be applicable in every situation. We did not consider multiobjective optimization.

While experimenting, we realised that to get the full picture, you need to see both of the terms and not just the final objective, so all three curves are visualised.

Result discussion:

Baseline:

None of these radius combinations achieves the optimal state (lowest metric value). We think that manipulation of the underlying parameters (like PopulationSize, LandSize, Influence) is essential for fine-tuning the isolation effect. Our project keeps a fixed influence of the external trends (green), so we can study how the interaction between neighbors changes the emerging phenomena.

The values of *NeighbourhoodRadius = 5, 10, 30* and related results shown in Table 1 (above) show that as the radius increases, WCSS/point decreases (clusters get tighter), but PatchPenalty increases dramatically (clusters get more packed and less isolated).

- **NeighbourhoodRadius=5 (B1):** The social influence is very limited. Only the immediate neighbors of the film's direct targets are influenced. This limits the size of the cohesive social unit, resulting in loose clusters (High WCSS/point) but few spatial connections between different groups (Low PatchPenalty). The effect of the film's attraction (InfluenceRadius=10) is relatively dominant in initiating movement.

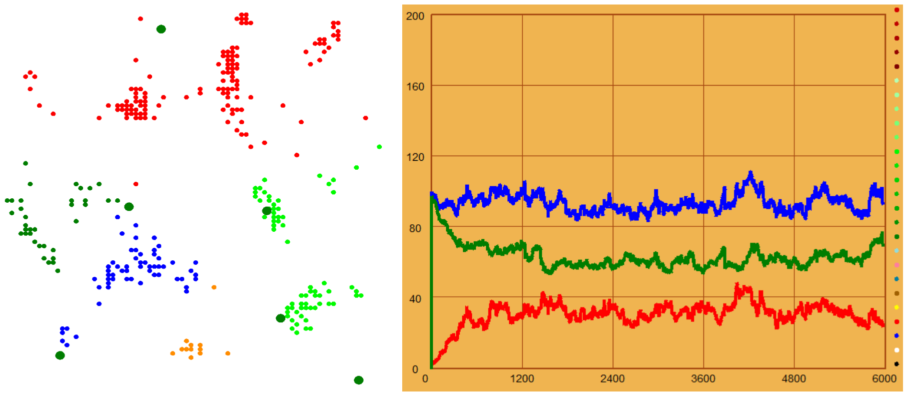


Image 8: Baseline 1 (NeighbourhoodRadius=5)

- **NeighbourhoodRadius=10 (B2):** The social influence extends further, allowing groups to establish stronger internal cohesion (lower WCSS/point). This moderate radius is an attempt to strike a balance: strong enough to overcome the internal spreading effect of the film, but not so large that it forces groups to aggressively fuse into giant patches.

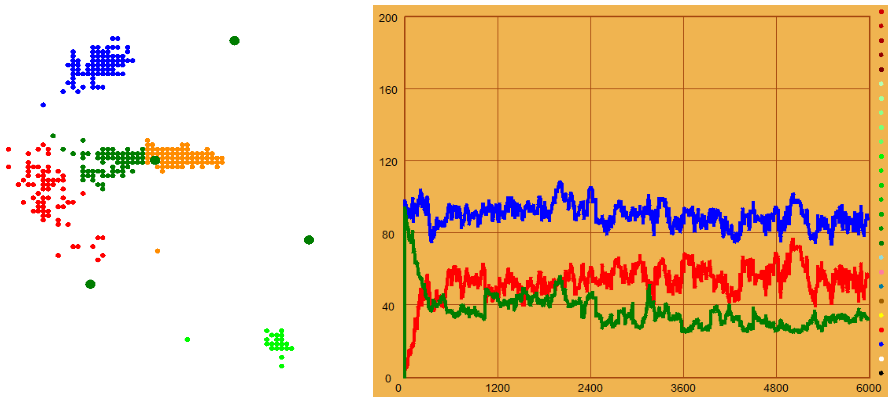


Image 9: Baseline 2 (NeighbourhoodRadius=10)

- **NeighbourhoodRadius=30 (B3):** The social influence is massive. A single individual moving toward a film can pull in a huge section of the population. This leads to rapid and excessive aggregation, meaning once groups form, they quickly merge spatially. This results in the lowest compactness score (WCSS/point) but the highest penalty for isolation failure (PatchPenalty=0.8108). The social force completely dominates the fixed film force.

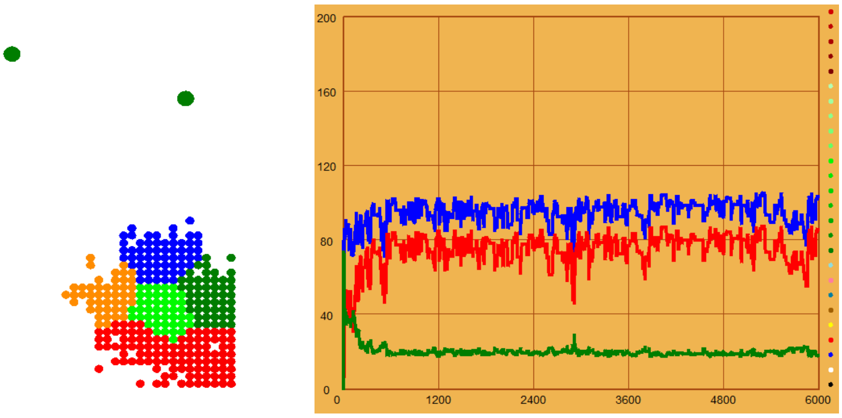


Image 10: Baseline 3 (NeighbourhoodRadius=30)

The default method (uniform sampling) successfully demonstrated the trade-off we intended to study: increasing social interaction scope (NeighbourhoodRadius) improves one objective (compactness) while worsening the other (dense clusters).

Our correction methods:

We refer to **Image 7** in the previous pages.

The metrics observed suggest that the combination of **random noise + gaussian sampling** (M1+M2) provides the best results. It minimises both the within cluster distances and the patching penalty. This method does not produce one giant group, which tends to happen with some of the other methods. You can see that we achieved better performance than the baselines.

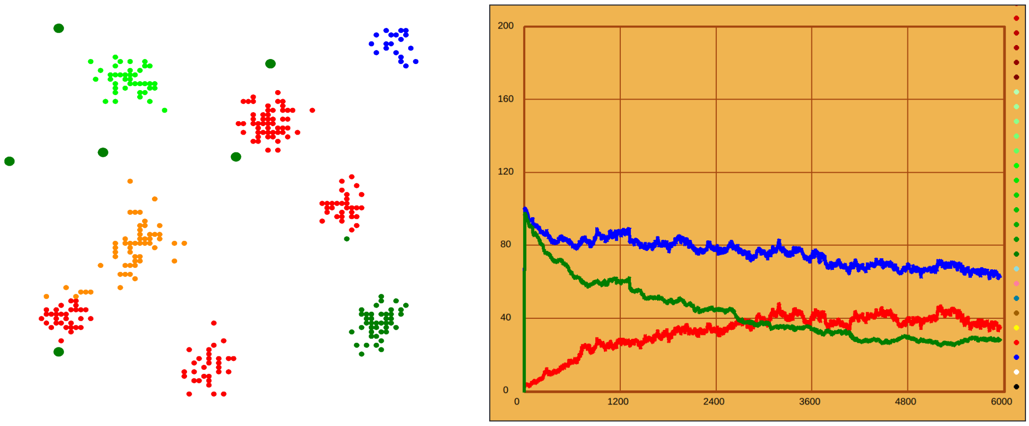


Image 11: Gaussian sampling + random noise

You can see on the chart that not only are the results better, the process to reach them is much smoother and the metrics don’t jump between different values nearly as much as with some of the other methods.

Linear sampling and **donut sampling** don't really improve the result, but are still better than gaussian sampling alone. The **gaussian sampling** creates wells which trap the data and they form very tight clusters. Worse result is with **global sampling**, which leads all of the data together into one ball. The performance of gaussian sampling is greatly improved with random noise, which leads to escaping local optima (wells) created.

Random noise linear and donut sampling are themselves comparable to our best baseline setting (NR=5) , and are an improvement over uniform sampling with the same NR (which is B2).

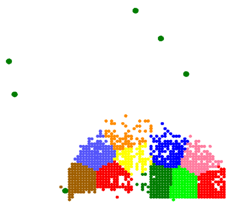
Note, that the results are not very robust, since it's the result of one run of the algorithm and different initialization can have different results. We combined the results with visual inspection of the resulting population.

Other observations:

Why do we choose to focus on the neighbourhood effect?

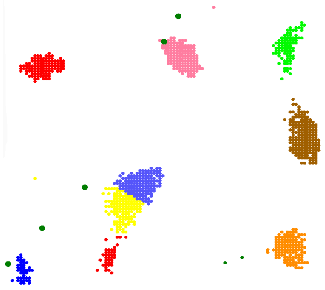
While exploring different sets of parameters, we observed that bubbles (or silos) formed due to a cohesion effect stronger than the attraction effect of the movies, which results in keeping individuals stuck together. The few examples below sum up the sensitivity study we performed and show what the long-term situation looks like, considering different radius values for the neighbor (N) and film (F) influence effect.

The combined effect of similar attractions (N:10/F:10)



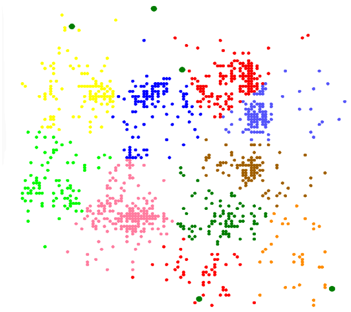
Clusters are initiated by films' attraction, then the interaction between neighbors keeps them next to each other. Finally, groups merge due to the influence of film at the frontier between groups.
Intermediate medium groups are never as well defined as in the right case because films' influence also causes some spreading effect.
Ultimately, individuals are always part of a unique community.

With a stronger neighborhood effect (N:20/F:3)



Neighbors form strong and isolated communities.
Individuals located between two groups can attract two groups together, although communities are little affected by movies not directly close to them.
Therefore, after the first merges instigated by individuals, communities do not come across each other and just get tighter and tighter.
Finally, weak films are not able to fold clusters and communities tend to line up.

With a stronger film influence (N:3/F:20)



Temporary clusters emerge around movies, and then are scattered by the next movie.

The social cohesion is too weak to keep group members together.

Further work, shortcomings of the experiment:

We did not average the results over multiple runs, so we don't have a solid idea about the variance of the results. For the metric comparison, we read the objective value at some number of iterations, which we decided beforehand. For a more robust measure, we should, for example, average metric at various thresholds.

Conclusion

We formalised the problem, proposed a metric to try to measure opposing effects and then tested the baselines as well as new methods with this metric. We focused on changing the neighborhood sampling, rather than the movie/ content recommendation effects.

We saw that the best new method was Gaussian sampling from the fixed size neighborhood with the introduction of random noise. It outperformed all of the baselines and other methods, decreasing both within cluster squared distances (WCSS) and the penalty for forming patches. The resulting structure is small clusters moving loosely together, but not densely packed.

The metrics proved useful and the average number of filled neighbors seems like a good proxy for dense clustering, since the clusters are very rotund. For full information about the population, one should consult both measures, objective function and the visuals.

Member contributions: Jan and Samuel were responsible for the main idea, comparison metrics and visualisations. Samuel studied extreme scenarios and the balance of influences between two major forces, which guided the experiment. Yangtao was in charge of exploring and explaining the experimental baselines. František conceptualised and implemented all of the different sampling methods, which were the final achievement of the project. Each member contributed to the slides, presentation and report with the part they worked on. Jan was responsible for the organisation of the group. The commitment of each member was quite balanced, and we achieved what we think is a good result in a limited time window.

Bibliography

No external sources, other than Evolife were used for the research and experimentation. Wolfram Mathematica was used for the illustrations of methods.

Appendix

In this table, we have the concrete values of comparison between different methods of neighbor sampling. All of these have fixes NR equals 10, so the size of the neighborhood is fixed.

Table 2: Performance with different sampling methods

	NR	**Objective**	**WCSS/point**	**PatchPenalty**
Linear	10	0.7750	0.3609	0.4142

Donut	10	0.8488	0.2613	0.5875
Gaussian	10	1.1327	0.4192	0.6314
Global	10	1.0108	0.1833	0.8275
Random noise	10	0.7915	0.3582	0.4375
Gaussian + noise	10	0.6423	0.2813	0.3608

Table 3: Exploring Antagonistic Hyperparameters

IR = InfluenceRadius
NR = NeighbourhoodRadius
	Objective	**sqrt(WCSS)/point**	**Patch Penalty**

IR:1 NR:1	1.0153	0.9828	0.0325
IR:1 NR:5	1.0209	0.9917	0.0292
IR:1 NR:15	1.002	0.9695	0.0325
IR:1 NR:25	0.9229	0.8937	0.0292
IR:5 NR:50	1.0089	0.9731	0.0358
IR:5 NR:1	0.9331	0.8798	0.0533
IR:5 NR:5	1.1198	0.7681	0.3517
IR:5 NR:15	0.9838	0.3455	0.6383
IR:5 NR:25	0.9473	0.1881	0.7592
IR:5 NR:50	1.0216	0.1766	0.8450
IR:10 NR:1	0.8085	0.7185	0.0900
IR:10 NR:5	1.0354	0.5254	0.5100
IR:10 NR:15	1.0886	0.2253	0.8633
IR:10 NR:25	1.079	0.1923	0.8867
IR:10 NR:50	1.1128	0.1995	0.9133
IR:20 NR:1	0.6138	0.5271	0.0867
IR:20 NR:5	1.1152	0.4685	0.6467
IR:20 NR:15	1.0387	0.2354	0.8033
IR:20 NR:25	1.0109	0.2309	0.7800
IR:20 NR:50	1.064	0.2173	0.8467

November, 2025

ATHENS course : **MOB_0AT09_TP**

Emergence in Complex Systems

Micro-study

teaching.dessalles.fr/ECS

Name: **Doğa Selin Damar, Emine Göksu Yıldız, Dennis Waniek**

Modeling a more complex ant behavior

Abstract

The goal of this micro-study is the enhancement of a simulated ant colony model originally designed for collective foraging behavior. The project introduced key extensions, including pheromone diffusion, nonlinear evaporation rates, and stochastic path attraction, renewable food source option to improve biological realism and model robustness in dynamic resource environments.

Problem

The existing ant colony simulation model relies on simplistic and deterministic mechanisms, which limit its biological realism. To influence the exploratory capability, path selection and interplay with the environment, further biologically plausible dynamics should be implemented.

Method

Pheromone evaporation

The 1st model enhancement refines the evaporation dynamic for pheromones. Although this mechanic is less relevant for real ants, it is essential for the use of artificial ants in solving various kinds of problems [1]. The current implementation is an evaporation with a constant rate.

However, pheromones can be assumed to be droplets of liquid evaporating over time, where smaller amounts decay significantly faster than larger amounts. This can be modeled using a power-law. As this simulation is time discrete the change in pheromone concentration from the time point t_1 to the time point t_2 is given as

$$\Delta p(t_2) = -kp(t_1)^{\alpha} \tag{1}$$

Using different factors for the positive- and negative-feedback pheromones different evaporation rates can be selected.

Local Pheromone Diffusion

We modified the ant foraging model by adding local pheromone diffusion directly inside `Ants.py`. When an ant deposits pheromone P_0 on its current cell, a fraction of this pheromone spreads to the surrounding 3×3 neighborhood. The amount received by each neighboring cell is computed as:

$$P_{i,j} = \frac{\alpha P_0}{d(i,j) + 1},$$

where α is the diffusion coefficient and $d(i,j)$ is the Chebyshev distance to the neighboring cell. The central cell ($d = 0$) receives the full pheromone amount. This modification increases the width of explored paths by reducing pheromone concentration gradients.

Periodic Food Regeneration

In the second scenario, food resources are fixed at specific coordinates, and each has an initial quantity $Q_i(0)$. Their quantity decreases when ants eat:

$$Q_i(t+1) = Q_i(t) - C_i(t),$$

Where $C_i(t)$ is the amount consumed at time t. When a food source is depleted ($Q_i = 0$), it remains empty until a global regeneration event occurs. Every 1000 simulation steps, all depleted food sources are restored to their original quantity:

$$t \bmod 1000 = 0 \Rightarrow Q_i(t) = Q_i(0).$$

This schedule produces visible jumps in food availability and leads to an overall increase in total food intake.

Probabilistic pheromone attraction

Following the existing logic, ants always choose the path with the highest score $s = p_+ - p_-$ computed by the difference of positive pheromone concentration p_+ minus negative pheromone concentration p_-. A more realistic approach would be a weighted probabilistic choice, as pheromone concentration can not be determined exactly by the ants.

To model this circumstance, firstly the score s is computed for every possible path choice. By adding a bias to the score s, it is shifted to positive values only in order to be used as weight for the path choice. Based on these weights one path is chosen randomly following a non-uniform distribution.

Results

Pheromone evaporation

Running the simulation with a constant evaporation rate leads to smaller paths existing for longer time. This in turn increases the chance for ants to follows these lesser frequented paths after a tapped food source has been exhausted. Figure 1 shows such a scenario, in which most of the ants move along higher frequented paths with a lesser-frequented path coexisting. After the depletion of the currently used food sources ants moved to the shown less frequented path.

With a power-law determining the pheromone decay, less frequented path exist for a much shorter time and therefore mostly vanish before being followed by a majority of ants. This leads to a focus on a few main paths, as Figure 2 shows, and an increased scattering of the ants as soon as the search for new food sources is necessary.

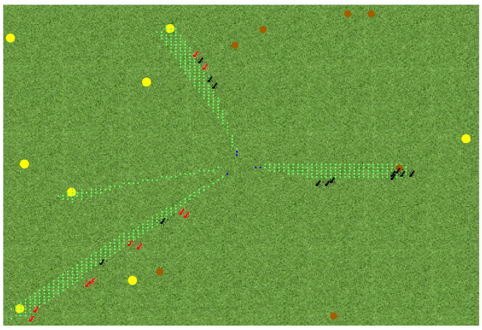


Figure 1: Simulation of ants with a constant pheromone evaporation rate.

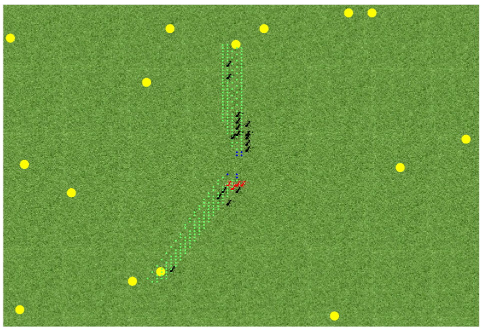


Figure 2: Simulation of ants with a pheromone evaporation rate following a power law.

Periodic Food Regeneration

In the second simulation, food sources were replenished every 1000 steps. Unlike random replenishment, only the depleted sources were restored. This led to observable jumps in the collected food during the replenishment steps, as ants quickly exploited the renewed sources that they formerly discovered near their nest. Figure 3 shows the food uptake by ants in the model without food renewal and Figure4 shows the food uptake with renewable food sources in every 1000 time steps. Overall, this scenario increased the total food intake compared to the first simulation, demonstrating that the timing and location of food renewal can significantly affect foraging efficiency.

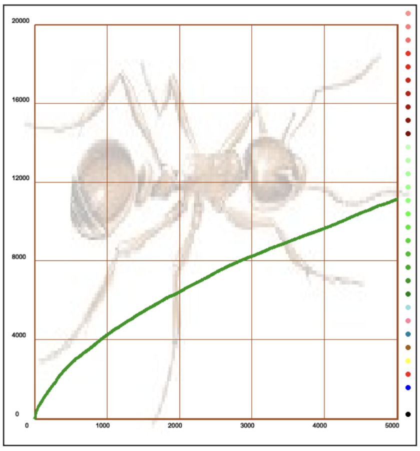


Figure 3: Graphic from the model with non-renewable food sources.

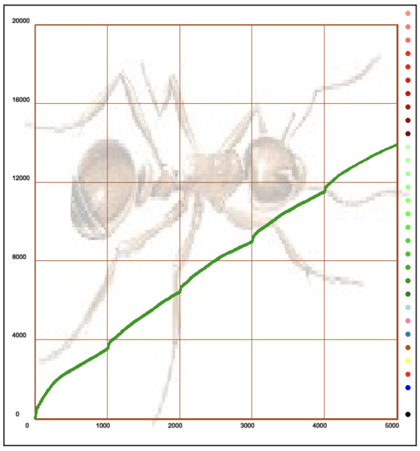


Figure 4: Graphic from the model with certain amount renewable food sources in every 1000 time steps .

Local Pheromone Diffusion

In this simulation, pheromones deposited by ants diffused to neighboring cells in a 2×2 square, proportionally to the distance from the ant. This modification allowed ants to explore a wider area around the food sources. As shown in Figure 5 (wide path scenario) and Figure 6 (narrow path scenario), the diffusion resulted in broader trails, but the overall food collected over 5000 steps was slightly lower compared to the standard simulation without diffusion. This indicates that while the ants explored more, their efficiency in returning to food sources decreased due to the spread of pheromone cues.

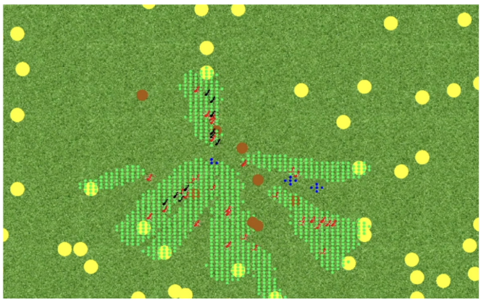


Figure 5: Ants form wider paths with the pheromone diffusion scenario.

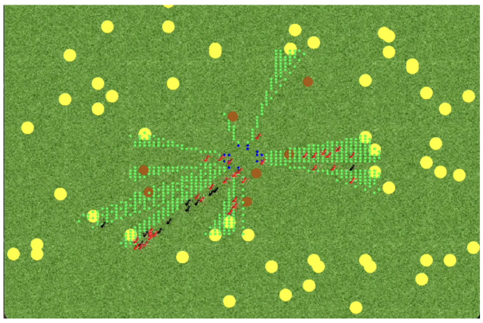


Figure 6: Ants form narrow paths in the original scenario.

Probabilistic pheromone attraction

With the deterministic path-selection-logic the modeled ants will always choose the path with the highest score s. Therefore less frequented paths become less likely as they will only emerge if an ant has no better path to follow. With a probabilistic path-selection highly frequented paths still have the highest chance to be followed by an ant, but less frequented paths are not completely ruled out as options. Therefore more of the less frequented paths, indicated by smaller pheromone trails, exist as shown in Figure 7 compared to Figure 8.

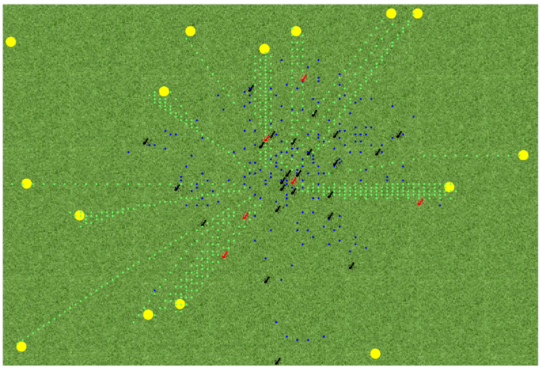


Figure 7: Simulation of ants with a probabilistic path-selection-logic.

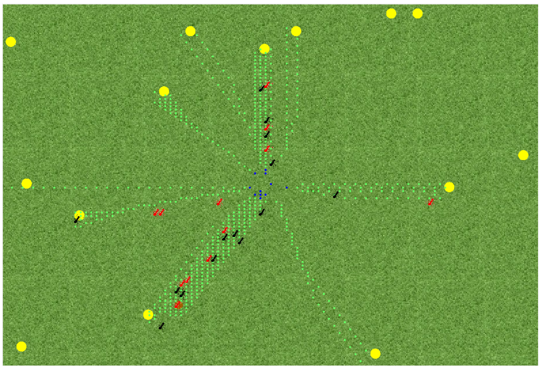


Figure 8: Simulation of ants with a deterministic path-selection-logic.

Discussion

Pheromone evaporation

A power law for pheromone evaporation therefore focuses the ant movement on fewer paths with higher frequentation and lets them tend to exploration only when already visited food sources have been depleted. In addition, when the evaporation rate set higher, the path discovery of ants almost seems like a pulsation in the model caused by disruption of the continuity of exploitation of the foods.

Pheromone Diffusion

Introducing pheromone diffusion to neighboring cells allowed ants to explore a wider area around the colony. This resulted in broader trails and more spatial coverage of the environment. However, the overall food intake decreased compared to the standard simulation over 5000 steps. This suggests that although diffusion facilitates exploration, it can dilute the pheromone signal, making it less precise for guiding ants directly to food sources. In other words, the trade-off between *exploration* and *exploitation* becomes apparent: while ants cover more area, their efficiency in exploiting known food sources is reduced.

Renewable Food Sources

Replenishing only depleted food sources every 1000 steps led to sharp increases in food collected during replenishment events. This scenario improved overall food intake compared to the diffusion-only simulation. The results indicate that targeted and periodic resource renewal can enhance foraging efficiency by ensuring that ants consistently encounter exploitable resources. It also shows that environmental dynamics, such as predictable replenishment, strongly affect collective behavior and the emergent patterns of resource exploitation.

Probabilistic pheromone attraction

with a probabilistic path-selection logic, once again exploration is emphasized as the chance for less frequented paths is nonzero in presence of highly frequented paths. Through the weighted choice of paths the logic still follows realistic ant principles of preferring paths with high (positive) pheromone concentration.

References

[1] Marco Dorigo, Mauro Birattari, and Thomas Stutzle. Ant colony optimization. *IEEE Computational Intelligence Magazine*, 1(4):28–39, 2006.

November, 2025

ATHENS course : **MOB_0AT09_TP**

Emergence in Complex Systems

Micro-study

teaching.dessalles.fr/ECS

Name: **Nikola Dobricic, Tia Manoukian, Margherita Necchi, Ada Yetis**

Decentralizing Navigation

Abstract

This project replaces the unrealistic hard-coded global coordinate tracking in ant foraging simulations with a decentralized olfactory mechanism. We do this by introducing a third pheromone called *Colony Pheromone*. We implement a diffusive pheromone gradient and demonstrate how precise homing behavior emerges solely from local agent-environment interactions, removing the need for internal cognitive maps or global positioning.

Problem

The global foraging simulations as currently defined hard-code the return journey home for the ants. That is, there is no actual mechanism implemented or explored for how ants find their way back to their nest after finding a food source. We want to explore a potential mechanism for such a return.

Background

Homing is defined the ability of an animal to return to its nest after traveling away from it. Foraging animals have been shown to display various different homing mechanisms as they are necessitated to wander away from their nests to find food sources. For ants, methods of homing are mainly olfactory.

Foragers establish pheromone trails on their way back to the nest after finding a food source to incentivize other ants from the nest to exploit the food source. These ants reinforce this pheromone trail on their return journeys, incentivizing more ants to exploit the food source. When the food source is exhausted, foragers deposit negative pheromones to signal to the ants to not take that pheromone trail. Although ants that follow a pheromone trail to a food source can follow the same pheromone trail back to the nest, the first ant that finds the food source does not have access to that trail. Instead, in some ant species, the foraging ant leaves behind a separate pheromone as it explores which it can follow back to its point of origin, the nest.

Furthermore, some ant colonies deposit a very specific blend of chemicals around their nests in a process called 'Home Range Marking'. This serves the purpose of both warning other ants and intruders that they are in the territory of that colony, which is generally defended, and aids colony members in finding their way back once they are in the vicinity of the colony.

Even if there are no colony specific pheromones in the area, ants may learn the specific olfactory profile of the area around their nest. They can then use various olfactory landmarks in the area to help them navigate.

Finally, some species of ants have been shown to rely on visual cues. These ants perform walks around the nest before becoming foragers. During these walks, they look in the direction of the nest and in different compass directions. Then, during foraging walks, they can match what they are seeing with their takeaways from these learning walks to align themselves towards the nest.

Although the most prevalent method seems to be pheromone trailing, it comes with steep costs. If the pheromone trail the ants are following is severely distracted, groups of ants have been shown to form an orderly circle and follow each other indefinitely. In these cases, they generally neither make their way back to the nest nor go back to foraging.

Method

We decided to replace the hardcoded vector for returning home with a *Colony Pheromone* gradient. To do so, we introduced a third type of pheromone called the *Colony Pheromone.* We wanted the pheromone to be secreted at a gradient so that the ants are secreting less pheromone the further they are from the nest. This was meant to ensure that the concentration of the *Colony Pheromone* would get higher as the ants approached the nest, thus ensuring that following the path of the increasing pheromone would lead the ants back home.

Specifically, ants were given a stock of *50,000* pheromone units while at the nest. This value decays by *10* each step an ant takes to create the aforementioned decaying gradient. After an ant finds food, it looks for neighbouring cells with a higher concentration of the colony pheromone. It continues to follow this increasing gradient until it reaches the nest. A percentage of the pheromone evaporates each time step to prevent overloading the environment with the pheromone and to maintain relative concentrations, which are necessary for the pheromone to be effective.

We also modified the simulation so all ants acted simultaneously instead of the old asynchronous structure as this made the simulation more consistent with the real-world and made analysing what was happening at each time step easier.

The following overview summarises the main functions involved and how the updated behaviour is distributed across classes.

- Ant behaviour pipeline

```
Ant.moves()
  If carrying food → returnHome()
  Else → exploration logic:
         Sniff()
         Random escape move
         Deposit Colony Pheromone
  Deposit Negative Pheromone
  Update taboo list (History)
  Switch to BackHome mode if food found
```

- Returning to the nest

```
returnHome()
  If current position = nest:
         Reset PPStock and CPStock
         Clear taboo list
  Determine next cell via SniffForHome()
  If no candidate or random escape trigger:
         Move randomly
  Deposit Positive Pheromone (food trail)
  Decrease PPStock proportionally to nest distance
  Update taboo list
```

```
SniffForHome()
  Retrieve neighbourhood cells
  Exclude current cell
  Exclude taboo list cells
  Select neighbour with highest Colony Pheromone
  Return None if no valid candidate exists
```

- Pheromone dynamics

```
evaporate()
  Exponential decay of NP
  Exponential decay of PP
  Exponential decay of CP
  If any pheromone < threshold:
         Snap to zero
  Clean cell if all pheromones = 0
```

```
cpheromone()
  Modify Colony Pheromone concentration
  Add cell to ActiveCells for evaporation
  Update rendering (pink/red intensity)
```

Results

Across all simulation runs we performed, the introduction of the diffusive *Colony Pheromone* produced a rich variety of emergent behaviors that depended strongly on the chosen parameter configurations.

A first set of results concerns the behaviour of the pheromone field itself. When the system relied on linear evaporation, *Colony Pheromone* levels near the nest grew indefinitely, as returning ants deposited pheromone faster than it could dissipate. This produced unstable concentration profiles that did not reflect biologically plausible conditions.

Switching to exponential evaporation stabilised the system: pheromone quantities decreased proportionally to their magnitude, preventing unbounded accumulation and creating a self-balancing dynamic. However, exponential decay also introduced long-lasting residual traces. Extremely small pheromone values never reached exactly zero and persisted as faint tails on the map, cluttering the environment and consuming unnecessary computational resources.

To address both issues simultaneously, we introduced a cutoff threshold below which pheromone values are snapped to zero. This preserved the stability of high-intensity regions while eliminating negligible residues, resulting in a clearer and more efficient pheromone field.

```
if self.cp() > 0:
    # Exponential decay:
    evap_rate = self.cp() * (Gbl['ColonyEvaporation'] / 100.0)
    self.cp(-evap_rate)
    if self.cp() < Threshold: self.cp(-self.cp())  # Snap to 0
```

The most significant behavioural results involve the emergence of circular motion patterns. In the initial stages of development, ants frequently became stuck in small local loops: oscillating between neighbouring high-pheromone cells or remaining confined inside minor local maxima. These micro-loops can be interpreted as early manifestations of the same mechanism that later gives rise to full circular milling.

To reduce these preliminary forms of oscillatory trapping, we introduced two mechanisms:

- a “Taboo List”, where ants remember their last six positions and avoid revisiting them;
- a small probability of random escape movement.

These modifications substantially reduced the formation of small oscillatory loops by interrupting self-reinforcing motion patterns before they could stabilise.

```
self.History = []
self.HistorySize = 6
...
if NewPos in self.History: continue
...
self.History.append(self.location)
if len(self.History) > self.HistorySize:
    self.History.pop(0)
```

Despite these improvements, in some parameter configurations the system still produced persistent circular formations: once a group of ants entered rotation, the pattern tended to self-sustain for the remainder of the simulation. Crucially, this behaviour is not an artifact of the model: circular milling, or “ant mills”, is a well-documented biological phenomenon, occurring when ants become trapped in self-reinforcing pheromone loops. Its appearance in our simulation therefore reflects a realistic emergent behaviour.

The biological interpretation of this phenomenon will be explored in detail in the following section.

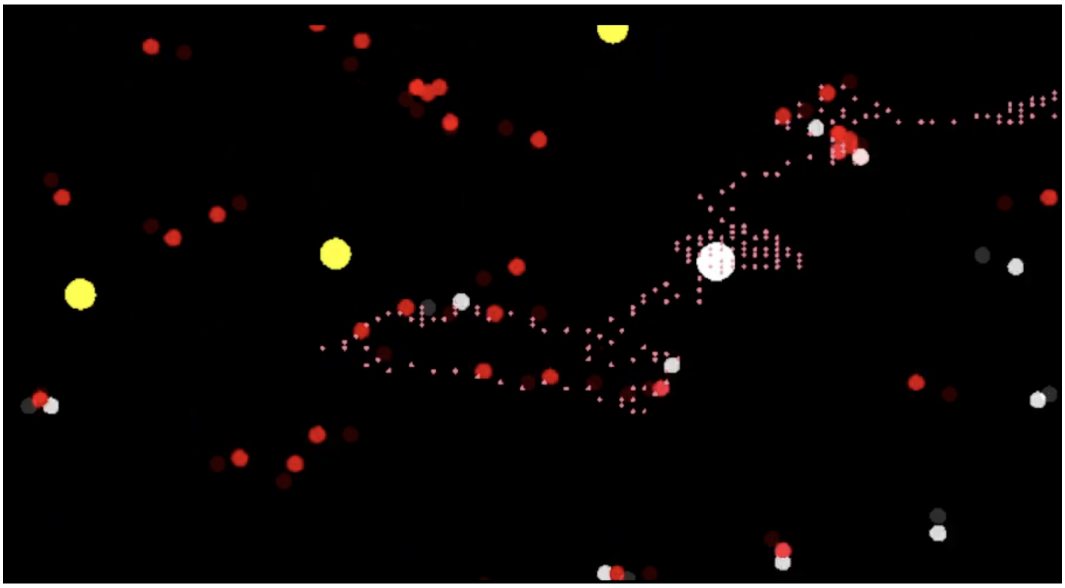


Figure 1: Circular milling behaviour observed in the simulation.

Rather than attempting to eliminate circular milling entirely, which would require imposing additional artificial constraints and would reduce biological plausibility, we focused on reducing its frequency. By tuning key hyperparameters, specifically the pheromone evaporation rate and the amount of *Colony Pheromone* deposited, we were able to make milling rare across the vast majority of runs. This reduction is sufficient to consider the system stable while still allowing it to reproduce characteristic emergent behaviours found in real ant societies.

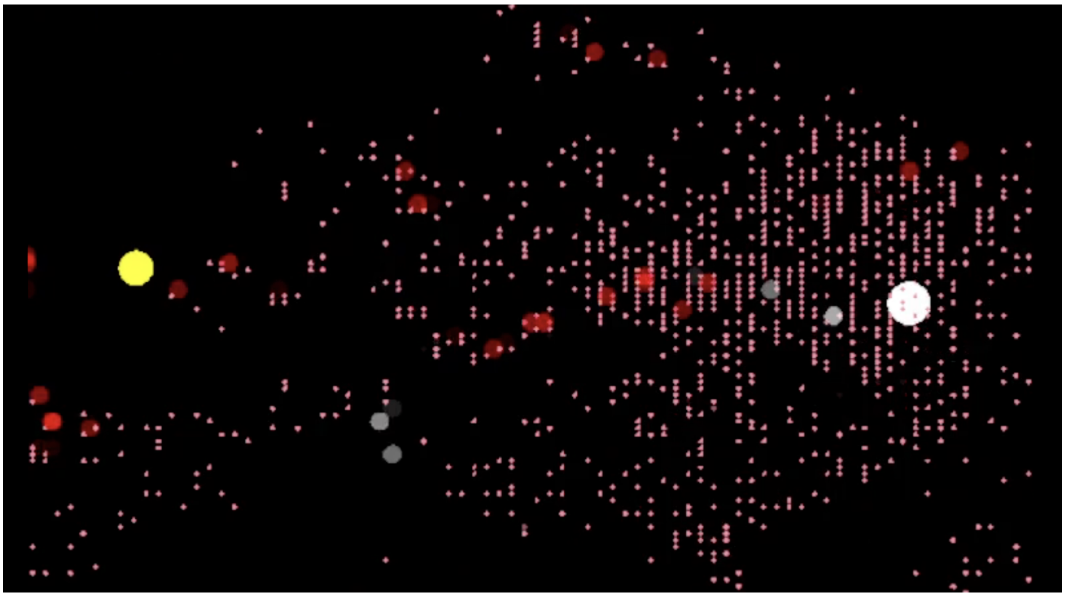


Figure 2: Example of a successful simulation without circular milling.

Discussion

The simulation results confirm that ants can find their nest without using a "hard-coded" GPS or coordinate subtraction. In the original model, ants used global knowledge—they "magically" knew exactly where the nest was (X, Y coordinates) relative to themselves. In our modified model, we replaced this with local knowledge. By simply following the trail intensity, the ants navigate successfully. The "pink fog" of *Colony Pheromone* essentially functions as an externalized cognitive map; by depositing pheromones during exploration, the colony constructs a physical memory of its history. This demonstrates that the "intelligence" of the system resides not in the individual agent but in the ants as a collective.

The "Ant Mill" Phenomenon: As noted in the results, early versions of our simulation failed because ants got stuck in infinite circles (circular mills) theoretically, this occurs because the local rule ("follow max pheromone") creates a positive feedback loop: the more a circular path is traversed, the stronger it becomes, trapping agents in a local maximum. This confirms that pure gradient ascent without error correction is inherently fragile in complex geometries.

Our Solution: Memory and Randomness To break these loops, we implemented two simple changes in the code, often referred to as adaptive stochasticity (or "useful randomness").

1. We gave each ant a short-term memory of its last 6 steps. If a move would take it back to a spot it just visited, it is forbidden. This prevents the ant from vibrating back and forth between two cells.
2. *The Random Noise Factor:* For larger circles, memory isn't enough. We added a 5 percent chance for the ant to ignore the pheromone entirely and move randomly. This "noise" is crucial because it breaks the symmetry of the circle, kicking the ant out of the loop so it can find the real path home.

-

Main Limitation: Because the pheromone evaporates or decays over time, there is a maximum distance from the nest where the signal works. If an ant goes beyond this "horizon," the trail home has already vanished by the time it turns around. Navigation is only possible within the radius where the pheromone signal exceeds the pheromone evaporation rate, limiting the effective foraging range compared to the infinite-range vector model. This suggests that in real decentralized systems, exploration must happen slowly and incrementally, expanding the territory only as fast as the pheromone network can support it.

Further Directions

Although functional, the paths created by the ants in our simulation were suboptimal as they were a direct result of the pathways they took during exploration. We know that in the physical world, ants trails are more efficient. A potential future direction for the work would be adding multiple other homing mechanisms to work in tandem with the trail pheromones, as there is no indication that ants only use one mechanism at a time and in fact some evidence that they do combine methods.

Another potential direction could be exploring having various ant colonies within the simulation environment. Since ant trails may intersect and interact with each other and this would introduce competition for food sources, an interesting direction would be exploring these interactions and how trails may evolve differently to accommodate the presence of other colonies.

Bibliography

Delsuc, F. (2003). Army ants trapped by their evolutionary history [Publisher: Public Library of Science]. *PLOS Biology*, *1*(2), e37. https://doi.org/10.1371/journal.pbio.0000037

Steck, K. (2012). Just follow your nose: Homing by olfactory cues in ants. *Current Opinion in Neurobiology*, *22*(2), 231–235. https://doi.org/10.1016/j.conb.2011.10.011

Li, L., Peng, H., Kurths, J., Yang, Y., & Schellnhuber, H. J. (2014). Chaos–order transition in foraging behavior of ants. *Proceedings of the National Academy of Sciences of the United States of America*, *111*(23), 8392–8397. https://doi.org/10.1073/pnas.1407083111

Zeil, J., Narendra, A., & Stürzl, W. (2014). Looking and homing: How displaced ants decide where to go [Publisher: Royal Society]. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *369*(1636), 20130034. https://doi.org/10.1098/rstb.2013.0034

Murray, T., Kócsi, Z., Dahmen, H., Narendra, A., Le Möel, F., Wystrach, A., & Zeil, J. (2020). The role of attractive and repellent scene memories in ant homing (myrmecia croslandi). *Journal of Experimental Biology*, *223*(3), jeb210021. https://doi.org/10.1242/jeb.210021

November, 2025

ATHENS course : **MOB_0AT09_TP**

Emergence in Complex Systems

Micro-study

teaching.dessalles.fr/ECS

Name: **Fares Boudelaa**

Emergent Phenomena in Conway's Game of Life

Abstract

We investigate emergent phenomena in Conway's Game of Life to determine if macroscopic outcomes depend on microscopic initial conditions. Results indicate that despite random starts, the population density deterministically converges to a stable equilibrium, mirroring biological survival under constraints.

Problem

Conway's Game of Life is made up of pretty simple rules: too many neighbors and the cell dies from overcrowding, too few and it starves, and with enough neighbors the cells live and reproduce. At a microscopic (cell) level there isn't much interesting observation to be made, yet at a macroscopic (interaction between cells) level the grid generally converges towards permanent or cyclical states and we observe the emergence of multiple phenomena such as recurring structures.

What is even more interesting is that the final density of the population seems to depend on the initial density of a random population. Does the macroscopic outcome (ρ_{final}) depend on the microscopic initial condition ($\rho_{initial}$)?

Method

We first implement the Game of Life using the Evolife framework. The Game of Life is a 2-dimensional cellular automaton with the rules B3/S23:

- **B3:** If a cell is dead and it has 3 neighbors, then it becomes alive.

- **S23:** If a cell is alive and it has 2 or 3 neighbors, then it keeps living.
- **Else:** The cell dies or stays dead.

We use `Landscape.py` from `Evolife.Graphics` to implement the simulation. While it would be much more efficient to implement the game on NumPy, use convolution matrices, and simply update the landscape where it needs to be updated, time constraints didn't allow for it.

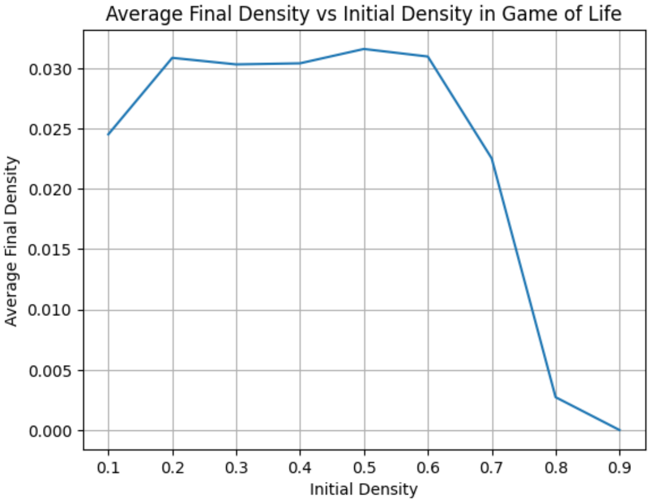


Figure 1: Visualization of the Game of Life grid using Evolife.

To test the given hypothesis the following parameters were used. A grid of 30 × 30 was used, initialized randomly with a given probability. The choice of using a probabilistic initialization was to ensure that the hypothesis was robust to noise. The simulation runs until convergence to either a constant or 2-cyclical state. There was a limit on the number of iterations set to 100,000 steps for time constraints and rare cases where the cycle was longer than 2 steps (such as a glider). Initial densities were between $0.1, 0.2, \dots, 0.9$ and for each, 600 simulations were run.

Results

The results are quite apparent. When considering the initial densities between 0.2 and 0.6, the final density is convergent towards 3%, just like hypothesized. Past 0.6, the final density starts plummeting, reaching an average of 0 when initialized with 0.9.

Furthermore, the number of iterations for convergent simulations in reference to the initial density follows the same curve as the latter. The average number of iterations for initial densities between 0.2 and 0.6 is around 400, and as we increase past this point the number of iterations drops.

Discussion

This emergent phenomenon is unexpected, since naturally one would assume that such a simple game with random initialization wouldn't demonstrate deterministic behaviors. This emergent phenomenon however, taken from a biological point of view is coherent; we recognize that the optimal densities for population survival under resource constraint stabilize.

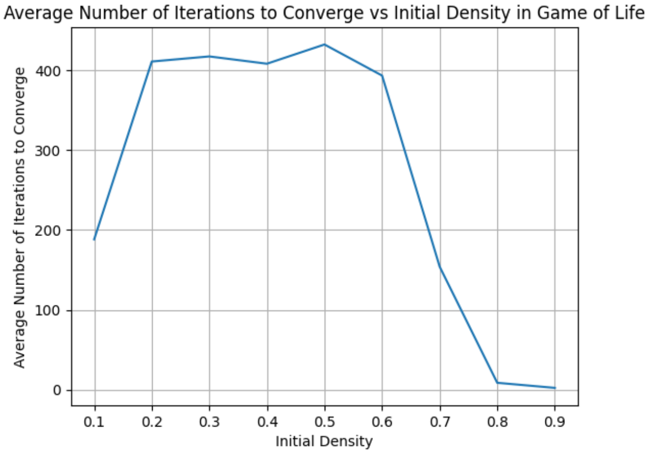


Figure 2: Analysis of final density and iterations relative to initial density.

For example, if we consider the cells as cities, a density that is too high creates competition for finite regional resources (water, electricity, trade routes), leading to a collapse of the weaker cities until a sustainable equilibrium is reached. Conversely, if cities are too sparse, they lack the necessary trade neighbors to sustain growth (reproduction), leading to isolation and eventual "starvation." The system naturally self-organizes to the maximum sustainable carrying capacity of the grid topology.

There are limitations however. Firstly, the method of stopping is very limited. There are only checks for 2-cycles and constant grids; perhaps checking when density becomes constant can be a better criterion, however a proof would be necessary. Secondly, the grid size of 30×30 is relatively small. In the study of complex systems, boundary conditions (edges) often distort the behavior of the system. Unless a toroidal (wrap-around) grid was used, the cells at the edges have fewer neighbors, artificially lowering the survival rate compared to an infinite plane. Finally, the definition of "convergence" misses complex oscillators or "spaceships" (like gliders) that move across the grid. These are strictly speaking repeating patterns, but because they shift position, a simple static check might misinterpret them as non-convergent chaos until the iteration limit is hit.