Learning in open adaptive networks

Learning in Open Adaptive Networks

Guoli Yang
LFCS, School of Informatics
University of Edinburgh

Vincent Danos
LFCS, School of Informatics
University of Edinburgh

September 13, 2016

Abstract

We propose a learn-and-adapt model for building efficient and resilient networks of cooperative agents. Agents are involved into three interconnected types of activities. Firstly, agents bid for handling random structured tasks. Secondly, agents *learn* the (exogenous) features of the random task source, by aggregating local information (such as success rates, average load, etc). And, thirdly, agents *adapt* the composition of their neighbourhoods following the (endogenous) targets set by their learning process. Neighbourhood readjustment proceeds by judicious rewiring steps which stay entirely local. Thus an agent continuously works, adjusts its neighbourhood, and based on his local metrics, learns how to inflect its own adaptation targets. Because of this tight coupling of all three activities, the network as a whole can reconfigure in a fully decentralized way to cope with changes in: the network composition (node failures, new incoming nodes, etc), and the parameters of the task source (changes in the size, structure, and frequency), while attaining robustly a near-optimal performance level (compared to the centralised solution).

Keywords: Adaptive networks; Distributed learning; Agents coordination

1 Introduction

A large body of research explores local adaptive dynamics on networks (**????**). Of particular interest are self-improving organisational models relying only on local adaptive dynamics **?**. Indeed, centralised solutions may incur high communication costs to ensure coherence, synchronization, dependability, and systemic resilience. Decentralised solutions, on the other hand, may attain comparatively good performance levels at a fraction of these costs (**????**).

Another, and perhaps less conspicuous advantage of local systems is that they are easier to evolve. In this paper, we propose a simple *learn-and-adapt* model designed to illustrate the benefits of this higher evolvability. In our model, agents continuously work to solve tasks fed to them by a random task source, while at the same time they adjust their neighbourhood, and learn how to inflect their own adaptation targets based on simple local metrics. Because of this tight coupling of all three activities, the network can reconfigure itself in a decentralized way and cope with unexpected changes in: the network composition (node failures, new incoming nodes, etc), and the parameters of the task source (changes in the size, structure, and frequency), while attaining robustly a near-optimal performance level compared to the centralised solution.

1.1 Blocking

Theoretical and experimental studies on task allocation abound in the field of multi-agent systems. In particular, agent-organised networks (AONs) (**?**) are organisations resulting from interacting agents, where team formation is essential to performance. In this context, a team consists of a number of related cooperative agents, equipped with skills and which work together toward a common complex task that none of the team members can complete independently. AON studies have established that simple local strategies to adjust structures dynamically may lead to remarkable and resilient performance with potential applications to: business alliances (**?**), dynamic supply chains (**?**), robotic teams (**?**), sensor networks (**?**), and distributed resource allocation (**??**).

Methods from operational research have been used to optimise task allocation in static networks (**?????**). Most strategies, however, experience intermittent blocking problems caused by insufficient or unreachable skills. This is especially so in networks with unfavourable structures. To see this, let us consider an example. Suppose that tasks T_1 and T_2 need to be completed by two teams (red and blue) with the required skills to do so. Fig. 1(a) shows a first type of blocking caused by insufficient skill supply. Both teams are in contention for skill S_2, but neither can successfully grab it. This is traditional blocking by contention for resources. Fig. 1(b) shows another, more subtle, form of 'spatial' blocking caused by skills being unreachable. That is to say, in this example all required skills are present, but some are separated from one team by the growth of the other one.

To work around both forms of blocking we will keep on the broad principles of AON to which we will add *division of labour*. Some agents will be (dynamically) distinguished as leaders. This slight modification will completely avoid contention for resources by accumulating resources around leaders. Spatial contentions will be avoided by organising the neighbourhoods of leaders into non-overlapping hubs.

As the task flow usually follows some distributions in open environments, and may change over time, a static network is manifestly disadvantageous to effective team formation. There are other reasons to favour designs that will be self-organising, such as resilience and the lack of a need to register new participants globally. Recently, various adaptation strategies to reshape connectivity and achieve better organisation performance were proposed (**????**). The effects of network structure on team formation was in particular investigated by Gaston *et al.* in Refs. (**??**), and many extensions (**???**) were provided to handle various issues regarding organisation performance and network adaptation.

In the following, changes of connectivity are based on learning processes which update the parameters driving the (stochastic) adaptation process. We start by describing the static aspects of the model in the next Section. Then we move on to the detailed description of the learn-and-adapt dynamics.

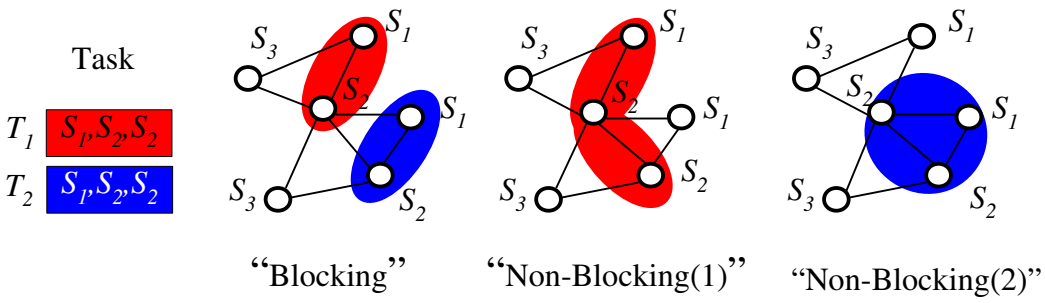
2 The static model

Our model of team formation and task allocation follows closely that of AON (**??**).

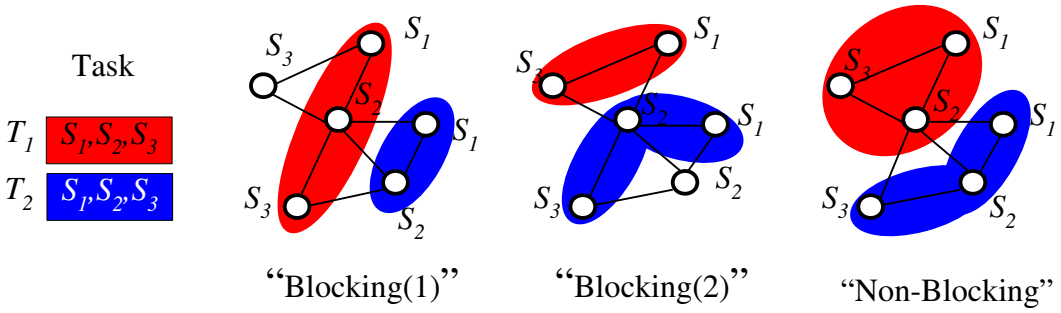
We consider a finite set of agents N, and a finite set of skills S, together with a *skill map* $s : N \to S$ which assigns a skill to each agent.

A *state* of the model is: 1) a set of undirected edges E on N, and 2) a *role* map $r : N \to \{0, 1\}$.

We say an agent x is a *leader* if $r(x) = 1$, a *follower* if $r(x) = 0$. We write: L (F) for the current set of leaders (followers); $\mathcal{N}(x)$, for the current set of neighbours of an agent x; and $d_1(x)$ for the



(a) Example 1: Blocking by insufficient skills. If both of the two incomplete teams: both the red team and the blue team are trying to complete their tasks, but neither can succeed because of insufficient skills. Would one give up, the other team could implement its task successfully.



(b) Example 2: Blocking by unreachable skills. Even if the available skills are enough to form two complete teams, it is still possible that some skills required by the blue (or red) team are blocked by the red (or blue) team, and then become unreachable because of unfavourable team structures.

Figure 1: Blockings and non-blockings in agents team formation.

number of neighbours of x which are leaders. That is to say $d_1(x) = |\mathcal{N}(x) \cap L| \leq |\mathcal{N}(x)|$.

We say that y is a follower of x, if y is a follower, x a leader, and y is in $\mathcal{N}(x)$

We say the state is *fully centralised* if there is just one leading agent connected to all other followers.

Later on, leaders will have access to additional information. Specifically, every x in L will have attached 3) a *load* $\lambda(x) \geq 0$, and 4) an S-vector *target* θ_x.

A *task* is simply represented as a multiset of skills (equivalently, a vector with integer coordinates in the free vector space generated by S). The idea is that for a team of agents to successfully complete a task during a round, the team needs to provide the skill set required by the task. The state of the system constrains the way in which teams can be assembled. Specifically, an agent x can only recruit another agent y if y is a follower of x.

Thus, the role function divides nodes between active recruiters or leaders (role 1), and passive

followers (role 0), and the edge structure limits the formation of teams within the immediate neighbourhood of a leader.

As we will see, both r and E can change over time: roles and connectivity adjust to demand. Even if one starts from a fully centralised state, the system will gradually adopt a more resilient, yet efficient, configuration. We will always assume that the initial graph structure N, E is connected and simple (at most one edge between two agents). Changes in connectivity will preserve this constraint.

2.1 Task allocation and completion

At the beginning of each round, γ new tasks are generated randomly (recall a task is a multiset of skills in S) and are dispatched evenly at random to leader nodes (randomized round robins are run until no task remains). Then, leaders recruit teams according to the constraints above. It is possible that many tasks are allocated to one leader. The process is illustrated in Fig. 2.

To avoid overloading the network, we will only consider situations where the average task size $|T|$ is such that $\gamma|T| < N$. This is a regime where, on average, the demand on skills placed by tasks in a given round is lower than the global work supply, i.e. the total number of nodes.

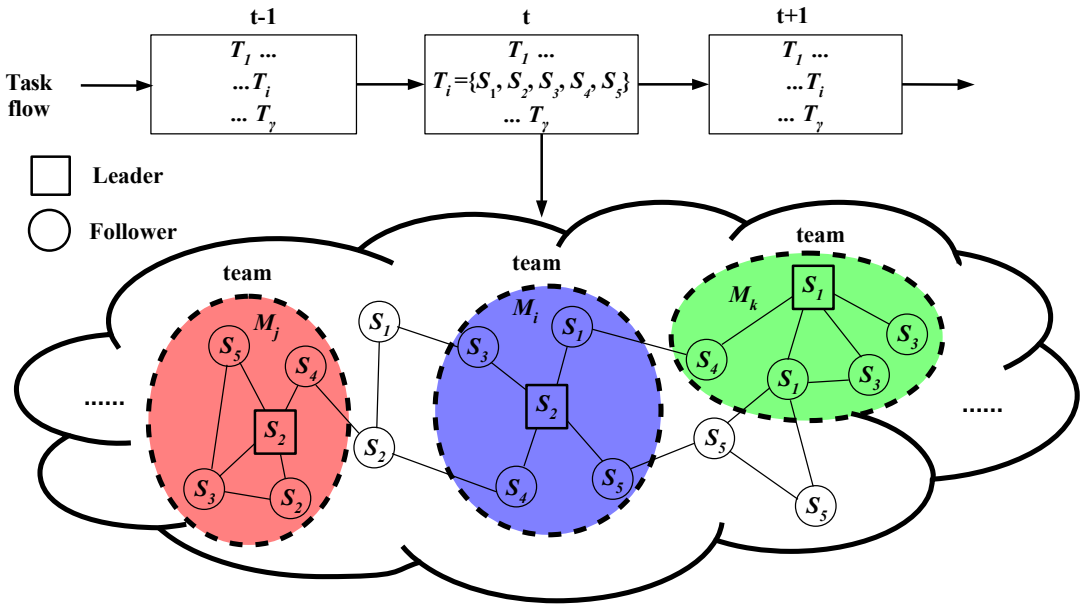


Figure 2: Team formation: each round γ new tasks are allocated evenly at random to leaders (square nodes). A task T allocated to a leader x will be completed if x recruits among her followers a team which provides the skills required by T.

2.2 Global performance metrics

We now introduce two global indicators of performance. These indicators are solely for the purpose of evaluating performance from outside the system. Agents will use other indicators to adapt the

r, E structure.

First, we define *task success rate*, i.e. the long-time average ratio between the number of completed tasks (CT) and the number of tasks offered (Fig. **??**):

$$TSR = \frac{1}{t} \lim_{t\to\infty} \sum_{u\leq t} \frac{CT(u)}{\gamma}$$

Second, we would like to define a metrics that captures the idea that the skill supply of a leader, i.e. the total number of skills available to her, should not be too large on average. More specifically, the skill pool of a leader should not exceed too much the actual demand placed on her. Indeed, if it does, the surplus work supply should be better allocated to other parts of the network.

This is what we call the *team waste rate*, i.e. the long-time average ratio between the surplus team size and the actual task size. To that effect, given r and E, we first define $M(x)$ as the size of the skill pool (the total number of skills) available to a leader x:

$$M(x) \quad = \quad 1 + \sum_{y\in\mathcal{N}(x)\cap F} 1/d_1(y)$$

Note that the formula above is always well-defined, by definition y has at least one neighbouring leader, namely x, so that $d_1(y) > 0$.

We can also remark that followers that belong to the neighbourhoods of multiple leaders are shared evenly thanks to the discount factor $1/d_1(y)$ (one can imagine that they work part-time). The reason we add 1 is because x itself provides a skill.

Set $M = \sum_{x\in L} M(x)$. It is easy to see that:

$$M \quad = \quad N - |\{y \in F \mid d_1(y) = 0\}|$$

Hence M counts the complement of the number of 'lonely' followers - ie followers who follow no leader. One can think of it, by analogy with 'free energy', as the 'free work force' - that which is actually recruitable in the current state of the system.

With this definition of skill supply in place, we can define our second metrics (also illustrated Fig. **??**):

$$TWR = \lim_{t\to\infty} \frac{1}{t} \sum_{u\leq t} \frac{M(u)}{\gamma|T|} - 1$$

Note that $\gamma|T|$ is the average skill demand, hence $M/\gamma|T|$ is the ratio of total skill supply to mean total skill demand. The further away this ratio is from 1, the more leaders are wasteful in skill resources. In this sense, TWR is the long term value of wastefulness. For a fully centralised organisation, $|L| = 1$, $M = N$, hence $TWR = N/\gamma|T| - 1$; again a measure of the excess of agents (skill supply) compared to the average skill demand.

We turn now to the core part of our model, namely the adaptive part of the dynamics.

3 Learn-and-adapt dynamics

To design an efficient adaptive strategy, we need first to specify what we want. Indeed, two problems should be addressed: (1) how many leaders (or teams) are required; (2) how do leaders adjust their teams.

Clearly, too many leaders lead to numerous small-size teams with an insufficient provision of skills, and, therefore, a low success rate. Too few leaders, on the other hand, will bring about large teams and a fragile network (Fig. **??**).[1]

In other words, the adaptive dynamics should drive towards states where: 1) (resilience) the typical skill pool of a leader is relatively small, i.e. $M(x) \ll N$, while 2) (performance) remaining large enough for task completion, i.e. $M(x) \geq \gamma|T|/|L|$.

Here we will realize the above constraints with the most decentralised state, where the task load $\gamma|T|/|L|$ on a leader is kept minimal. That is to say, the adaptive dynamics will drive towards states where: 1) $|L| = \gamma$ (meaning each leader receives on average one task per round), and 2) the skill pool of leaders is kept (somewhat) above their task load: $M(x) \geq |T|$.

To achieve this, our agents need to be equipped first with local metrics.

3.1 Learning one's role

For each leader x, we define a *load* $\lambda(x)$ which tracks the mean number of tasks taken by x:

$$\lambda(x) \leftarrow \alpha\lambda(x) + (1-\alpha)\gamma(x,u) \tag{1}$$

where $\gamma(x,u)$ is the number of tasks taken by x at time u, and α is a discount parameter. The weighted moving average smooths out accidental fluctuations and highlights the trends. A smaller α discounts the older observations faster, and a bigger one discounts them slowly. The load λ obtained through this simple update strategy is used to decide whether to expand or contract the population of leaders.

Say a leader is *overloaded* if $\lambda > 1$ (more than one task per team), and say it is *underloaded* if $\lambda < 1$ (less than one task per team). (Recall that we have chosen to target the most decentralised organisation.)

3.2 Adapting one's role

Let $0 < p_r < 1$ be a (small) real number. At the end of each round, overloaded leaders promote one of their followers (chosen uniformly at random) with probability $p_r \min(\lambda - 1, 1)$ (initialised with load zero). Likewise, underloaded leaders demote themselves with probability $p_r \min(1 - \lambda, 1)$ (Fig. **??**). Note that the flipping behaviour is entirely distributed and based only on local information. The idea is that at stationary state, the state will be such that $\lambda(x) \sim 1$ for all x, and the teams will receive one task each. The choice of α, p_r allows one to balance out fluctuations. Typically $\alpha = 0.01$, and $p_r = 1/N$ which guarantees that with high probability, there will be at most one change of role per round.

3.3 Learning one's friends

We now need to drive leading agents to arrange their set of followers so that it has the right skill composition. For each leader x, we define x's *composition target* as an S-indexed vector θ_x. We think of $\theta_x(s)$ as the number of team members with skill s which x is currently targeting. Recall that each round, x is allocated a list of tasks (Fig. 2) to handle. Let T be any of the tasks received by x

[1] As said in the introduction, large teams also come with higher costs of communication, cognition, management, etc. More reasons to keep away from them.

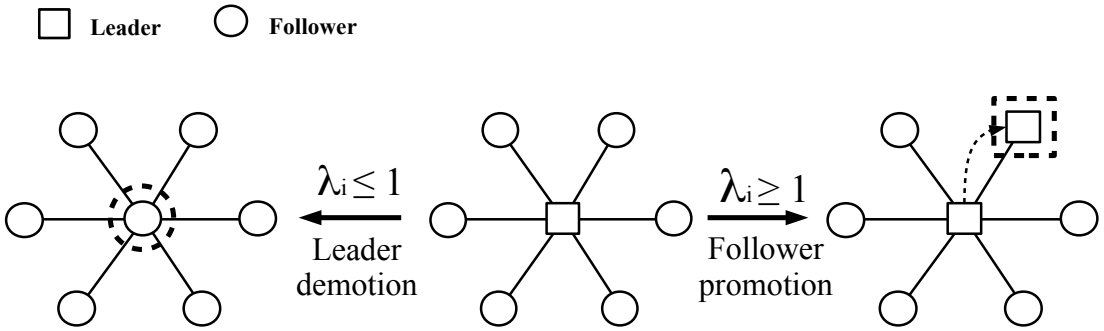


Figure 3: Changing roles: the evolution of roles is driven by the load estimate $\lambda(x)$: if $\lambda(x) \geq 1$, too many tasks are allocated to x's team, and more leaders are required; if $\lambda(x) \leq 1$, x's team is underloaded, and fewer leaders are required.

(supposing x receives at least one). The task T is also an S-indexed vector (with integer coordinates, aka multisets over S). Thus, one can define the current *deviation* at x as the difference:

$$\epsilon_x = T - \theta_x$$

The idea is to use this deviation to update x's target. Let η_-, η_+ small positive real numbers which will serve as our learning rates (analogous to α for role learning). For s in S, given T, the s-component of θ_x is updated as follows:

$$\theta_x(s) \leftarrow \theta_x(s) + \eta_-[\epsilon_x(s)]_- + \eta_+[\epsilon_x(s)]_+ \tag{2}$$

where we have used the following notation: $[x]_+ = \max(0, x)$, and $[x]_- = \min(0, x)$.

In words: when $\theta_x(s)$ is *above* $T(s)$ (equivalently when $\epsilon_x(s) \leq 0$), then x decreases its target by $\eta_-\epsilon_x(s)$. Conversely, when $\theta_x(s)$ is *below* $T(s)$ (equivalently when $\epsilon_x(s) \geq 0$), then x increases its target by $\eta_+\epsilon_x(s)$.

It is important to note that there is an assymetry in the problem at hand. When T's demand is below x's supply, T is completed, and the worst that can happen is that x's team is oversized. On the other hand, when T's demand exceeds x's supply, then T cannot be 'somewhat' completed, it is a complete failure. Thus a parameterisation where $\eta_- \leq \eta_+$ (meaning θ_x gets kicked harder when below target, than when above target) seems reasonable. Indeed, it will implement a risk-averse strategy (**???**). The bigger the ratio η_-/η_+, the larger the surplus of skill supply. The dependence of the performance of the system on this choice is studied in the next Section under simple assumptions.

3.4 Adapting one's friends

For connectivity adaptation, we opt for *triangle rewiring* (aka edge rotation) where an agent x disconnects from one neighbour y to connect to a neighbour z of y which is not already a neighbour of x (**?**). We call such a triple x, y, z a *redex*. Note that if x, y, z is a redex in graph G, so is z, x, y in the transformed graph G'. The rotation is fully reversible.

This class of transformations has three interesting properties: *(i)* it preserves the number of edges, *(ii)* it preserves connectedness, and *(iii)* it is ergodic in the sense that any two connected graphs on the same set of vertices can be related by a series of such rewirings.

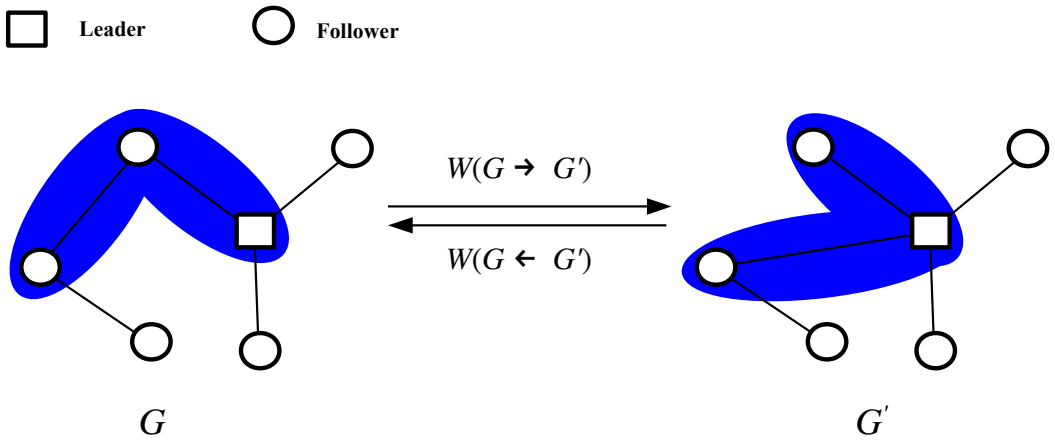


Figure 4: Changing friends: agents can change friends by triangle rewiring; the probability of a rewiring is determined by the incurred change of energy (see next Subsection).

Let $0 < p_E < 1$ be a (small) real number, and let $\mathcal{E}$ be an *energy* function from states to the real numbers. At the end of each round, for each leader x, with probability p_E, a candidate redex x, y, z is chosen uniformly at random (if any), and the corresponding rotation is performed with probability $\min(1, \exp(-\beta\Delta\mathcal{E}))$ with $\beta > 0$ an "inverse temperature" parameter. The energy difference $\Delta\mathcal{E}$ is the difference of the energy after and before rewiring the redex.

This probabilistic rewiring is an example of Metropolis algorithm (**?**) on graphs. Such indirect definitions of rewiring dynamics were investigated recently for generic graph-rewriting (**?**). Of course, in our application, everything hinges on the choice of a good energy function. This is what we need to define now.

3.5 Rewiring energy

Let ϕ_x be the S-vector describing the current *skill supply* available to x; that is to say, for s in S, $\phi_x(s)$ is the number of x followers with skill s. Its is important not to confuse ϕ_x, x's *actual* skill supply, with θ_x, x's *desired* skill supply. One can think of $|\phi_x - \theta_x|$ as the *tension* perceived by agent x. Accordingly, we will engineer an energy landscape which favours edge rotations that most reduce this difference.

For x a leader agent, we set its local energy to be:

$$\mathcal{E}_x = \sum_{s \in S} \exp(\theta_x(s) - \phi_x(s)) \tag{3}$$

Then the global energy is just $\mathcal{E} = \sum_{x \in L} \mathcal{E}_x$.

A point worth noting is that the energy local to a leader is computable with *only local information*: θ_x is directly part of the agent state, and ϕ_x can be obtained from its immediate neighbourhood. As a consequence, the energy (and thus probability) of a rotation step can be computed entirely based on local information. This justifies our claim of having defined a distributed learn-and-adapt process.

Using this non-linear form of energy, will lead by energy minimisation to balancing the distribution of skills within the network. The network energy will decrease more when the required skills are attached to the more "needy" agents. In addition, this definition of energy also makes it possible for skills from "rich" agents to be transferred to "needy" ones. By generic Metropolis properties, and assuming that roles do not change any more, and neither do the target vectors θ_x (ie the agents stop learning), we know that the rewiring process will converge to a probabilistic steady state where $p(G)$ is proportional to $\exp(-\beta\mathcal{E}(G))$ for any G reachable from the initial state via rotations as defined in Fig. **??**.

3.6 Algorithm

We can now summarise our full distributed algorithm. We start from a connected random network with one arbitrary leader agent, and iterate *ad libitum* the following round. The 'role change' and 'edge rotation' probabilities are defined above and depend respectively on learned parameters λ_x and θ_x.

Algorithm 1 learn-and-adapt

Require: initial state r, E, λ, θ
$t \leftarrow 0$
while $t < t_{\max}$ **do**
  $t \leftarrow t + 1$
  distribute tasks to leaders evenly uniformly
  for x in L **do**
    update λ_x and θ_x
    with probability $p_r(\lambda_x)$ try role change
  end for
  for x in L **do**
    pick uniformly a redex x, y, z (if any)
    with probability $p_E(\theta_x)$ try edge rotation
  end for
end while

4 Calculations

In this section, we analyse the effects of our learn-and-adapt framework on the network structure and the global organisation performance by some mathematical calculations.

We write k for the number of skill types: $k = |S|$.

To make our problem amenable to (approximate) hand calculations, we suppose: 1) that the tasks generated all have the same size n, and 2) that they are distributed according to a multinomial distribution with parameters p_s for s in S (with n trials over k categories).

In other words, each round, tasks are obtained by drawing (with replacement) k times from an urn where skill s is represented in proportion to p_s.

When n is large enough, by the central limit theorem, the number of occurrences of skill s in a task follows to a very good approximation a distribution of the form:

$$T(s) \quad \sim \quad np_s + \sqrt{n}\,\mathcal{N}(0, \sigma_s^2)$$

where we have written $\sigma_s^2 = p_s(1-p_s)$ for the variance of $T(s)$ and $\mathcal{N}(0, \sigma^2)$ for the centred normal distribution with standard deviation σ.

For s in S, x in L, recall that we have defined the s-component of the deviation as the random variable $\epsilon_s = T(s) - \theta(s)$ (we no longer write the more correct ϵ_x, and θ_x to keep notations lighter). Hence, at stationary state:

$$\epsilon_s \quad \sim \quad \sqrt{n}\,\mathcal{N}(0, \sigma_s^2) + np_s - \theta^\star(s) \quad \sim \quad \mathcal{N}(np_s - \theta^\star(s), n\sigma_s^2)$$

where $\theta^\star(s)$ is the stationary value of x's target $\theta(s)$, and the approximate probability density function (pdf) can be computed as:

$$p_{\epsilon_s} \quad = \quad \frac{1}{\sqrt{2\pi n\sigma_s^2}} e^{-\frac{(\epsilon_s - np_s + \theta^\star(s))^2}{2n\sigma_s^2}}$$

Stationarity for $\theta^\star(s)$ means that the average kick-up and kick-down cancel out:

$$\eta_+ \int_0^{+\infty} \epsilon_s p_{\epsilon_s}\, d\epsilon_s \quad = \quad \eta_- \int_{-\infty}^{0} -\epsilon_s p_{\epsilon_s}\, d\epsilon_s$$

Using integration by parts, we can rewrite this stationarity constraint as:

$$\frac{np_s - \theta^\star(s)}{-\sigma_s\sqrt{n}} \left[\eta_+ - (\eta_+ - \eta_-)\Phi(\frac{np_s - \theta^\star(s)}{-\sigma_s\sqrt{n}})\right] = (\eta_+ - \eta_-)\frac{1}{\sqrt{2\pi}} e^{-\frac{(np_s - \theta^\star(s))^2}{2n\sigma_s^2}}$$

where $\Phi(x)$ is the *cumulative distribution function* of the normal distribution. And with the change of variable below to rescale $\theta^\star(s)$, the stationarity constraint becomes:

$$\begin{aligned} a &= \frac{\theta^\star(s) - np_s}{\sigma_s\sqrt{n}} \\ a\big[\frac{\eta_+}{\eta_+ - \eta_-} - \Phi(a)\big] &= \frac{1}{\sqrt{2\pi}} e^{-\frac{a^2}{2}} \end{aligned}$$

It is easy to see that there is a unique a which verifies this equation. We will call $a^\star$ this unique solution. Thanks to our rescaling, the solution does not depend on the initial parameters n, p_s, σ_s. In fact, $a^\star$ only depends on the 'kick ratio' η_-/η_+. Fig. **??** plots $a^\star$ for various values of the ratio in the interval $0 \leq \eta_-/\eta_+ \leq 1$.

One can also prove that $a > 0$ if and only if $\eta_+ > \eta_- > 0$. This has a direct intuitive interpretation. It amounts to saying that the steady state value of $\theta(s)$ will be above the average number of occurrences of s, in a task, namely np_s, if and only if x kicks harder up than down.

Returning to the definition of a:

$$\frac{\theta^\star(s)}{n} = p_s + a^\star \frac{\sigma_s}{\sqrt{n}} \tag{4}$$

Hence one sees that $\frac{\theta^\star(s)}{n}$ decreases and converges monotonically to p_s as 1) $a^\star$ decreases, 2) n increases, or 3) the skill variance decreases.

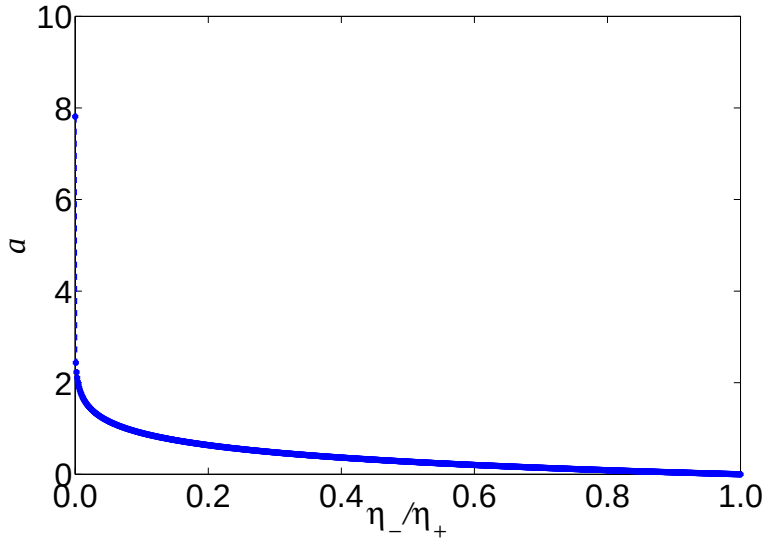


Figure 5: The solution $a^\star$ decreases as η_-/η_+ goes from 0 to 1.

4.1 The uniform case

If, in addition, we suppose $p_s = 1/k$, that is to say all skills are equally represented in the multinomial, we can aggregate all skills, and define $\theta^\star = \sum_{s\in S} \theta^\star(s)$. Then, Eq. **??** gives:

$$\frac{\theta^\star}{n} \quad \sim \quad 1 + a^\star \frac{\sum_s \sigma_s}{\sqrt{n}} \quad = \quad 1 + a^\star \frac{\sqrt{k-1}}{\sqrt{n}}$$

Suppose further that the team skill supply of a leader, $\phi_x(s)$, matches x's target θ_x (through network adaptation) at stationary state. That is to say, suppose $\phi_x^\star(s) \sim \theta_x^\star(s)$. Again we can aggregate skills and define $\phi_x^\star = \sum_s \phi_x^\star(s)$. Assuming there are exactly γ leaders, we can approximate the team waste rate as:

$$TWR \quad = \quad \sum_{x\in L} \frac{\phi_x^\star}{\gamma n} \sim \frac{\phi^\star}{n} - 1 \sim \frac{\theta^\star}{n} - 1$$

which gives (by plugging the equation right before):

$$TWR \sim \frac{a^\star \sqrt{k-1}}{\sqrt{n}} \tag{5}$$

We will see in the next Section that these approximations are quite accurate.

4.2 Larger skill set decreases the task completion

Given a task T, for a team to complete it, one must have $\phi^\star(s) \geq T(s)$. Hence, still assuming $\phi_x^\star \sim \theta_x^\star$ at steady state, one needs for each s in S:

$$np_s + a^\star \sigma_s \sqrt{n} \geq np_s + \sqrt{n}\,\mathcal{N}(0, \sigma_s^2)$$

The probability of this event of excess supply happening is by definition of Φ:

$$p(\phi_x^\star(s) \geq T(s)) = \Phi_{0,\sigma_s^2}(a^\star \sigma_s)$$

Hence the favourable event where the task can be completed can be approximated by:

$$\prod_{s \in S} p(\phi_x^\star(s) \geq T(s)) = \prod_{i=1}^{k} \Phi_{0,\sigma_s^2}(a^\star \sigma_s)$$

When skills are picked uniformly, this leads to an approximation of the task success rate:

$$\begin{aligned} TSR &\simeq \Phi_{0,\sigma^2}(a^\star \sigma)^k \\ &\simeq \left(\frac{1}{2} + \frac{1}{2}\mathrm{erf}(\frac{a^\star}{\sqrt{2}})\right)^k \end{aligned} \tag{6}$$

where $\mathrm{erf}(x)$ is the so-called error function.

It is interesting to observe that, for a given $a^\star$, the probability for task completion decreases exponentially with k, the size of skill set, but does not depend on the task size n. Skill diversity may damage the performance, but skill quantity does not.

5 Numerical Simulations

This section presents a series of numerical simulations to verify the above calculations. Several important observables are captured during the process of network evolution and adaptation, such as leader's internal targets θ, skill supply ϕ, and overall performance using TSR and TWR. Simulations use a random connected network with $N = 500$ agents and an average degree $\langle d \rangle = 4$ (ultimately, by ergodicity of edge rotation, only the initial number of nodes and edges matter). At each step, γ tasks are introduced into the network. Role and edge changes both use the same probability $p_r = p_E = 0.002$. Such low values reflect the fact that the adaptation module should have a *slow time scale* relative to the learning one. The kick-up learning rate is $\eta_+ = 0.01$ and the kick-down rate η_- ranges in $\{0.001, 0.002, \ldots, 0.009, 0.01\}$. The whole simulation lasts 10^5 steps and follows Algorithm 1 (empirically the simulation reaches steady state an order of magnitude faster - see for instance Figs. **??** and **??** for the time scale of adaptation of team size).

5.1 Baseline

In this part, we have the size of the task $n = |T| = 100$ and the number of tasks introduced at each step is $\gamma = 4$, so that $\gamma |T| < N$. The size of skill set is $k = 2$ and each skill is selected with same probability $p_i = 0.5$. We run the distributed learn-and-adapt framework described above and the ratio of learning rates is $\eta_-/\eta_+ = 0.1$. At stationary state, we have the snapshot of network in Figure **??** (left panel), where 4 evident hubs are obtained indicating the leaders. The red and blue nodes indicate the two different skills, which are uniformly assigned to the agents at random.

The numerical and analytical values for $\theta^\star/n$ under varying η_-/η_+ are presented in Fig. **??** (right panel). As one can see, the simulation results match the analytical solutions of the preceding Section closely.

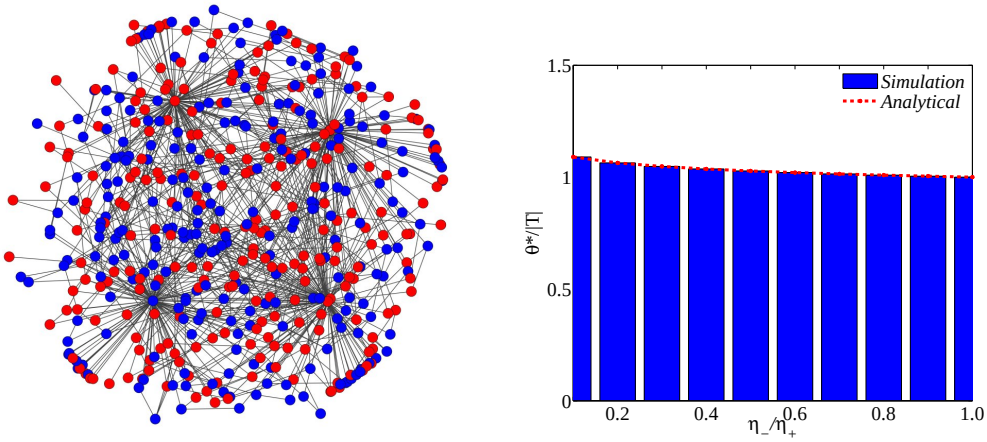


Figure 6: For a kick ratio $\eta_-/\eta_+ = 0.1$, a snapshot of the connnectivity structure is shown. We have set $N = 500$ and $\langle d \rangle = 4$. As η_-/η_+ increases, the numerical and analytical values for $\theta^\star/n$ at stationary state are presented in the right panel. The other parameter simulations are $\gamma = 4$, $n = |T| = 100$, and $k = 2$ (skills are indicated in red and blue colours). The self-organized convergence to four leaders reflect the design of the learning procedure.

5.2 The evolution of teams

With $n = |T| = 100$, $\gamma = 4$ and $k = 2$, Fig. **??** displays the evolution of the average team size when the ratio η_-/η_+ varies. Ideally, the size of a team should be about equal to the size of a task n at stationary state. Considering the uncertainty in task demand, however, some surplus skill suppliers are necessary. In particular, we find an evident gap between team supply and task demand for values of η_-/η_+ that are too low.

Fig. **??** displays the dynamics of the task success rate. Clearly, when the ratio η_-/η_+ is smaller, team supply is bigger, and task success rate is higher; on the contrary, when the ratio η_-/η_+ is bigger, team size is close to task size gradually and task success rate drops very low. There is a trade-off between wastefulness (TWR) and success in completion (TSR), where higher completion performance usually comes along with bigger resource wastefulness. Note that our system can reach stationary state in a quite quick manner.

5.3 Increasing the number of edges

As said, too many edges might lead to higher communication cost when forming teams but this is not visible in our model. What one can clearly see however is that too few edges fragment the states. The task success rate (TSR) changes with the increase of the initial (and conserved) number of edges (Fig. **??**). Simulations are run on an Erdos-Renyi random network, with $N = 500$, $|E| = \binom{N}{2}p$ for various values of p the probability to connect any two nodes. The abrupt transition in performance arises at $|E| \approx N$ and corresponds to the transition in connectivity in the underlying (initial) random graph (**?**).

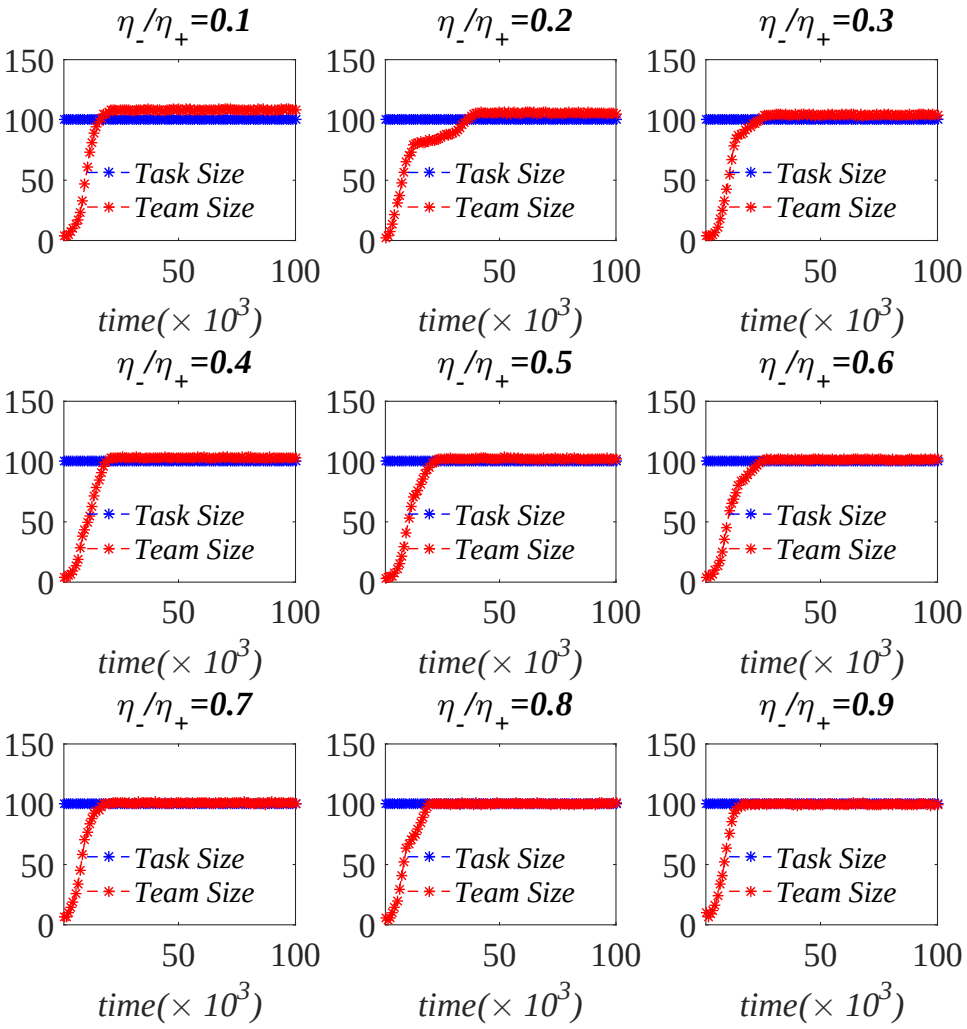


Figure 7: The changes of average team size depend on η_-/η_+. As η_-/η_+ increases, typical team size becomes smaller as leader targets θ shrink. Accordingly the difference between the two curves diminishes as η_-/η_+ gets closer to 1.

5.4 Node failures

In the centralised case, there is only one leader followed by everyone. This unique leader must solve all the tasks by providing the required skills. This strategy is optimal in terms of TSR, but it deteriorates in the presence of node failures.

Starting from an initial network, which is connected and random with $N = 500$ and $\langle d \rangle = 4$,

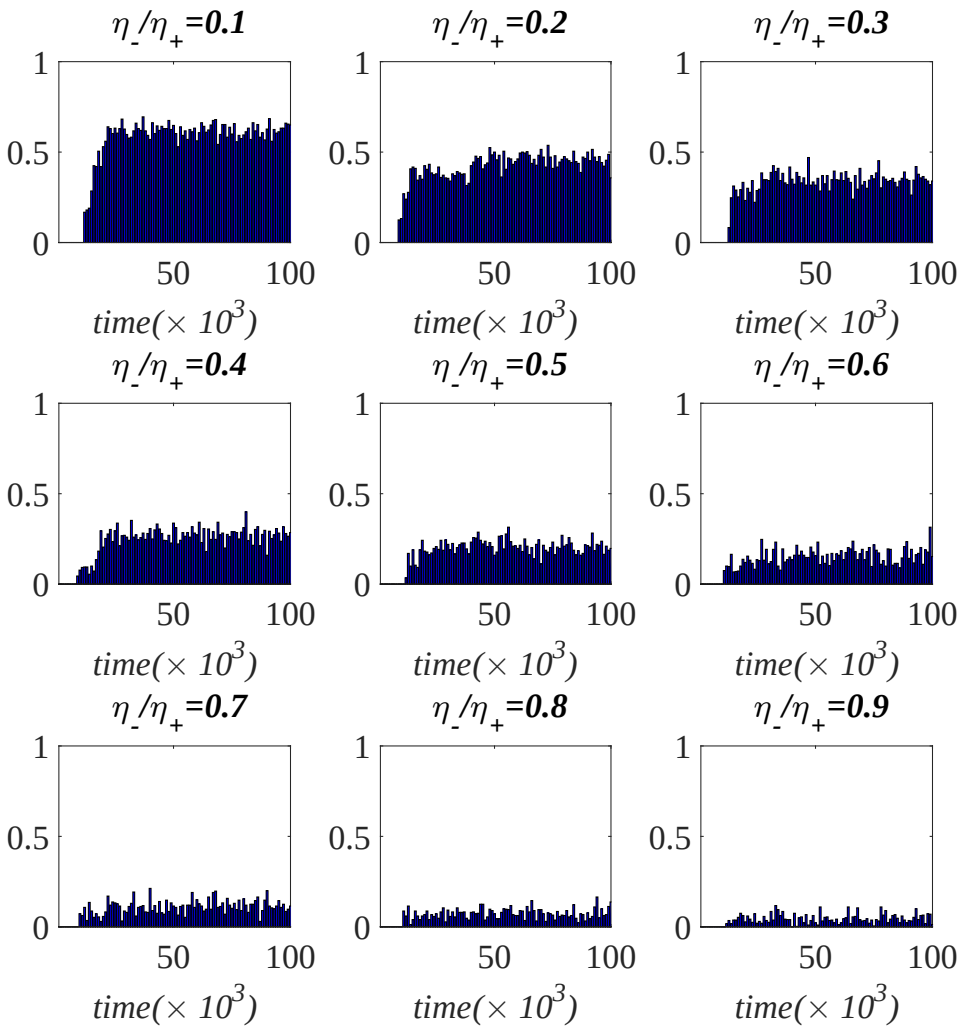


Figure 8: Task success rate (TSR) changes with the evolution of network for different η_-/η_+. A higher task completion is obtained when η_-/η_+ is smaller, where team supply exceeds typical task demand. With the increase of η_-/η_+, the TSR drops sharply.

we implement the centralised and decentralised leader-follower model respectively. In this situation, $\gamma = 4$ tasks are introduced at each round, each of which has size $n = |T| = 100$. We set node failures to occur with a low frequency (here, a small probability $p_f = 0.0001$). A node is selected to fail with a probability proportional to its degree. As shown Fig. **??**, and perhaps unsurprisingly, the centralised network is very fragile because of its single point of failure. Its performance collapses

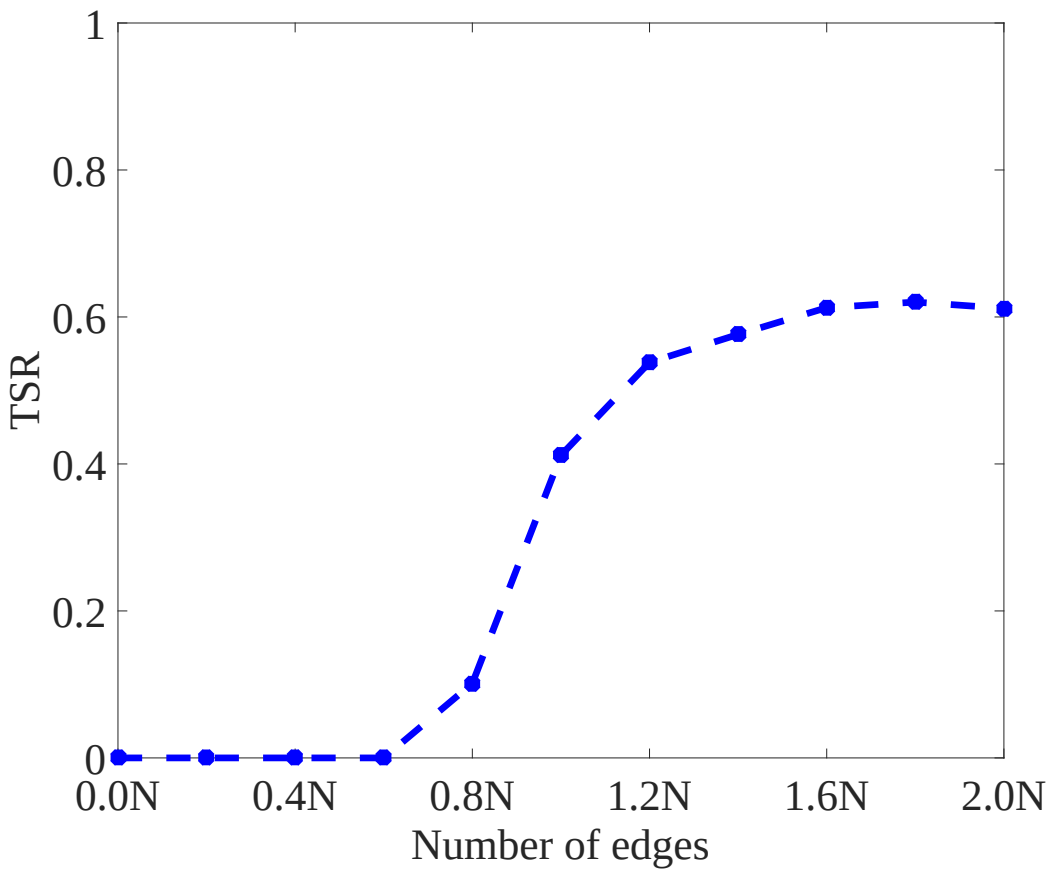


Figure 9: For kick ratio $\eta_-/\eta_+ = 0.1$, we track success rate (TSR) as the number of edges increases. In the simulations, $\gamma = 4$ tasks are created at each step, and the size of each task is $n = |T| = 100$. The skill set has size $k = 2$, and skills are allocated to agents uniformly.

abruptly when the hub is turned in to a follower. However, in the case of a decentralised mechanism, the overall performance is more resilient. Note that there are more visible drops in the decentralised case (10 on average for a total of 10^5 steps). This reflects the fact that in the centralised case, half of the failures concern followers and are imperceptible. Centralised drops are more rare, but much more detrimental to the overall performance.

5.5 The influence of task source

In this part, we consider the above case as a baseline and investigate the impact of the key parameters: k the size of the skill set S, the task size n, and the kick ratio η_-/η_+.

In Fig. **??**, we plot the team waste rate (TWR) and task success rate (TSR) for various cases. The baseline case is in the upper row ($\gamma = 4$, $n = |T| = 100$ and $k = 2$); the case of a larger skill set is in the middle row ($\gamma = 4$, $n = |T| = 100$ and $k = 4$); and, the case of increasing task sizes is

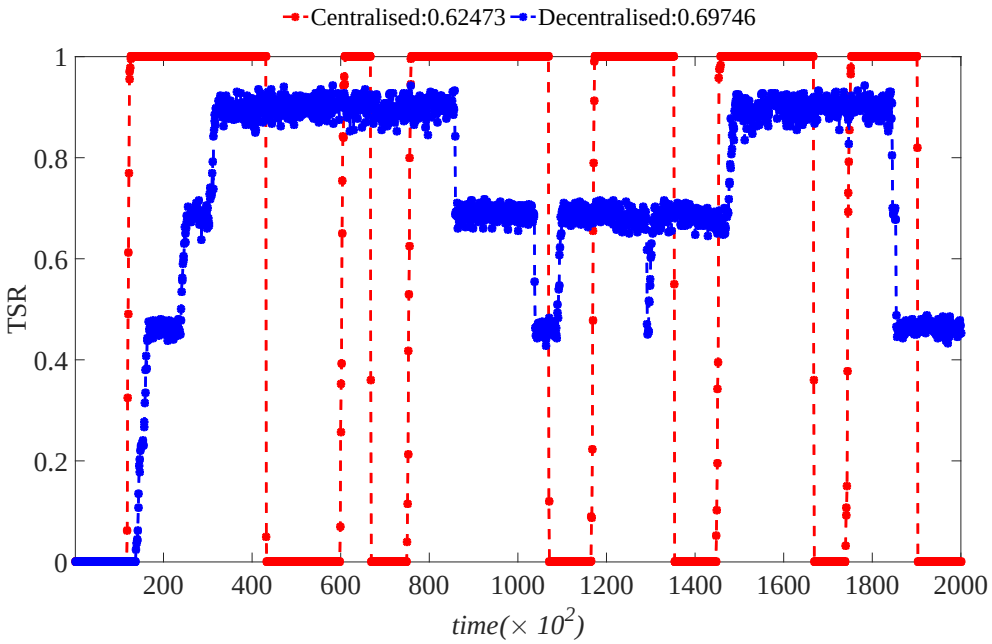


Figure 10: The evolution of task success rate (TSR) under centralised and decentralised strategies. Degree-proportional node failures occur in the network with a small probability each round, leading to the fall of organisation performance, especially in the case of centralised mechanism. The simulation parameters are: $\gamma = 4$, $n = |T| = 100$, $\eta_- = 0.0001$, $\eta_+ = 0.01$ and $k = 2$.

in the lower row ($\gamma = 1$, $n = |T| = 400$ and $k = 2$).

At stationary state, the resource wastefulness (TWR) is higher when the skill set increases, especially so for small η_-/η_+. This phenomenon can be explained by Eq. **??**, where team waste rate is associated with both task size (inversely proportional to $\sqrt{|T|}$) and skill set size (proportional to $\sqrt{k-1}$). The probability for task completion (TSR) is sharply reduced as the size of skill set (namely k) increases, which is also well explained by Eq. **??**. As for the case of increasing the task size, TWR decreases, but the TSR remains the same as the baseline's, whence one sees that task completion indeed is independent of task size.

5.6 Dynamical environments

Considering the dynamics of task requirements in real world, an intelligent and robust distributed algorithm is able to detect those changes and react to them quickly. The adaptive network mechanism explored in this paper enables leader agents to aggregate historical information and adjust the internal targets continuously to keep up with the task requirements. To see this in action, we set up a simulation where we start from the baseline settings with $k = 2$, $|T| = 100$ and $\gamma = 4$. We then change task requirements *during* the course of the evolution of the system, by stepping up and down $|T|$. Fig. **??** shows the result of this extended adaptation experiment.

5.7 Selfish agents

When the agents in the system are selfish, some agents in a team are *cooperators* and some are *cheaters*. When doing the task in a team, each member has a probability p_c to be a cheater. A cooperator provides the equipped skill to do the task with a cost of c, and the cheater is a free-rider and pays nothing. When the task is successfully accomplished, every agent in the corresponding team can gain a benefit of b. In short, the *payoff* of an agent can be computed through:

$$\Pi = \begin{array}{c} \\ \textit{cooperator} \\ \textit{cheater} \end{array} \begin{array}{c} \begin{array}{cc} \textit{success} & \textit{unsuccess} \end{array} \\ \begin{pmatrix} b-c & -c \\ b & 0 \end{pmatrix} \end{array}$$

With a given $b = 10$, Fig. **??** displays the task success rate (TSR) and average agent payoff at stationary state.

6 Conclusion

This paper incorporates state evolution, structure adaptation and distributed learning into adaptive networks, for high-performance and resilient agent coordination. Through continuous learning, a series of local adjustments reshape the roles of agents and their connectivity.

Key to the overall performance, is a dynamic division of labour between leaders and followers. This local form of order avoid locks caused by insufficient or unreachable resources. To reach an efficient configuration, a risk-averse learning mechanism is strapped on the leaders so that they continuously refresh the picture of the local structure they are targetting, and seek to attain by rewiring their neighbourhoods. For radical changes in patterns of demand, leaders also track their loads and are able to demote themselves or promote neighbours as the case may occur. The ratio of target learning rates η_-/η_+, the probabilities of adaptation, together with the task size, and the number of different skill types, have an influence on both team wastefulness (over-accumulation of followers) and task completion rates.

Increasing the number of skill types decreases the organisation performance. A diverse demand in skills seem to require larger assemblies to be dealt with well. A situation familiar to people living in small groups. In contrast, increasing the typical size of a task will lead to more efficient teams with smaller resource waste rate. Our self-organizing decentralised method, where no one exerts global control, does not know a priori about the type of skill demand it is going to face. As a result, it works well in a dynamical environment while incurring a relatively low loss of performance compared to a centralised solution.

In the real world, more and more systems are open to new incoming agents and built by largely interdependent components, who are supposed to manage and adapt their relationships with others intelligently based on local and historical information. Elaborations of our simple distributed learn-and-adapt model might be useful in these more complicated settings. One limitation, is that our agents are cooperative. There are many situations in organisations where this is not the case. A line of research we feel would be worth pursuing, would be to take self-interested agents (**?**) into consideration, and integrate to the design of the learn-and-adapt model elements of game theory and incentives to handle these more realistic situations.

7 Acknowledgments

GY gratefully acknowledges financial support from the University of Edinburgh and the China Scholarship Council; GY and VD were partly funded by the ERC project RULE 320823.

References

Sherief Abdallah and Victor Lesser. Multiagent reinforcement learning and self-organization in a network of agents. In *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, page 39. ACM, 2007.

Julie A Adams et al. Coalition formation for task allocation: theory and algorithms. *Autonomous Agents and Multi-Agent Systems*, 22(2):225–248, 2011.

Estefanía Argente, Holger Billhardt, Carlos E Cuesta, Sergio Esparcia, Jana Görmer, Ramón Hermoso, Kristi Kirikal, Marin Lujak, José-Santiago Pérez-Sotelo, and Kuldar Taveter. Adaptive agent organisations. In *Agreement Technologies*, pages 321–353. Springer, 2013.

L. Barton and V.H. Allan. Methods for coalition formation in adaptation-based social networks. *Cooperative Information Agents XI*, pages 285–297, 2007.

M. Bowling and M. Veloso. Multiagent learning using a variable learning rate. *Artificial Intelligence*, 136(2):215–250, 2002.

Yuriy Brun, Giovanna Di Marzo Serugendo, Cristina Gacek, Holger Giese, Holger Kienle, Marin Litoiu, Hausi Müller, Mauro Pezzè, and Mary Shaw. Engineering self-adaptive systems through feedback loops. In *Software engineering for self-adaptive systems*, pages 48–70. Springer, 2009.

B. Bulka, M. Gaston, and M. Desjardins. Local strategy learning in networked multi-agent team formation. *Autonomous Agents and Multi-Agent Systems*, 15(1):29–45, 2007.

Vincent Danos, Russ Harmer, and Ricardo Honorato-Zimmer. Thermodynamic graph-rewriting. *Lecture Notes in Computer Science*, 11(2):380–394, 2013.

Mathijs M de Weerdt, Yingqian Zhang, and Tomas Klos. Multiagent task allocation in social networks. *Autonomous Agents and Multi-Agent Systems*, 25(1):46–86, 2012.

Daniela Scherer Dos Santos and Ana LC Bazzan. Distributed clustering for group formation and task allocation in multiagent systems: a swarm intelligence approach. *Applied Soft Computing*, 2012.

Richard Durrett, James P Gleeson, Alun L Lloyd, Peter J Mucha, Feng Shi, David Sivakoff, Joshua ES Socolar, and Chris Varghese. Graph fission in an evolving voter model. *Proceedings of the National Academy of Sciences*, 109(10):3682–3687, 2012.

Paul Erdős and A Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hungar. Acad. Sci*, 5:17–61, 1960.

Matthew E Gaston and Marie Desjardins. *Organizational learning and network adaptation in multi-agent systems*. University of Maryland at Baltimore County, 2005.

M.E. Gaston and M. DesJardins. Agent-organized networks for dynamic team formation. In *Proceedings of the Fourth International Conference on Autonomous Agents and Multiagent Systems*, pages 230–237. ACM, 2005.

R. Glinton, K. Sycara, and P. Scerri. Agent organized networks redux. *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence*, pages 83–88, 2008.

Robin Glinton, Paul Scerri, and Katia Sycara. Agent-based sensor coalition formation. In *Information Fusion, 2008 11th International Conference on*, pages 1–7. IEEE, 2008.

Thilo Gross and Bernd Blasius. Adaptive coevolutionary networks: a review. *Journal of the Royal Society Interface*, 5(20):259–271, 2008.

Petter Holme and Mark EJ Newman. Nonequilibrium phase transition in the coevolution of networks and opinions. *Physical Review E*, 74(5):056108, 2006.

Charles A Holt and Susan K Laury. Risk aversion and incentive effects. *American economic review*, 92(5):1644–1655, 2002.

Teuvo Kohonen. Self-organization and associative memory. *Self-Organization and Associative Memory, 100 figs. XV, 312 pages.. Springer-Verlag Berlin Heidelberg New York. Also Springer Series in Information Sciences, volume 8*, 1, 1988.

Oliver Kosak, Gerrit Anders, Florian Siefert, and Wolfgang Reif. An approach to robust resource allocation in large-scale systems of systems. In *Self-Adaptive and Self-Organizing Systems (SASO), 2015 IEEE 9th International Conference on*, pages 1–10. IEEE, 2015.

Ramachandra Kota, Nicholas Gibbins, and Nicholas R Jennings. Decentralized approaches for self-adaptation in agent organizations. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 7(1):1, 2012.

Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

Yuki Miyashita, Masashi Hayano, and Toshiharu Sugawara. Self-organizational reciprocal agents for conflict avoidance in allocation problems. In *Self-Adaptive and Self-Organizing Systems (SASO), 2015 IEEE 9th International Conference on*, pages 150–155. IEEE, 2015.

Matjaž Perc and Attila Szolnoki. Coevolutionary gamesa mini review. *BioSystems*, 99(2):109–125, 2010.

Christian Prehofer and Christian Bettstetter. Self-organization in communication networks: principles and design paradigms. *IEEE Communications Magazine*, 43(7):78–85, 2005.

Onn Shehory and Sarit Kraus. Methods for task allocation via agent coalition formation. *Artificial Intelligence*, 101(1):165–200, 1998.

Attila Szolnoki and Matjaž Perc. Emergence of multilevel selection in the prisoner's dilemma game on coevolving random networks. *New Journal of Physics*, 11(9):093033, 2009.

HP Thadakamaila, Usha Nandini Raghavan, Soundar Kumara, and Réka Albert. Survivability of multiagent-based supply networks: a topological perspect. *Intelligent Systems, IEEE*, 19(5):24–31, 2004.

Lovekesh Vig and Julie A Adams. Coalition formation: From software agents to robots. *Journal of Intelligent and Robotic Systems*, 50(1):85–118, 2007.

D. Ye, M. Zhang, and D. Sutanto. Self-organization in an agent network: A mechanism and a potential application. *Decision Support Systems*, 53:406–417, 2012.

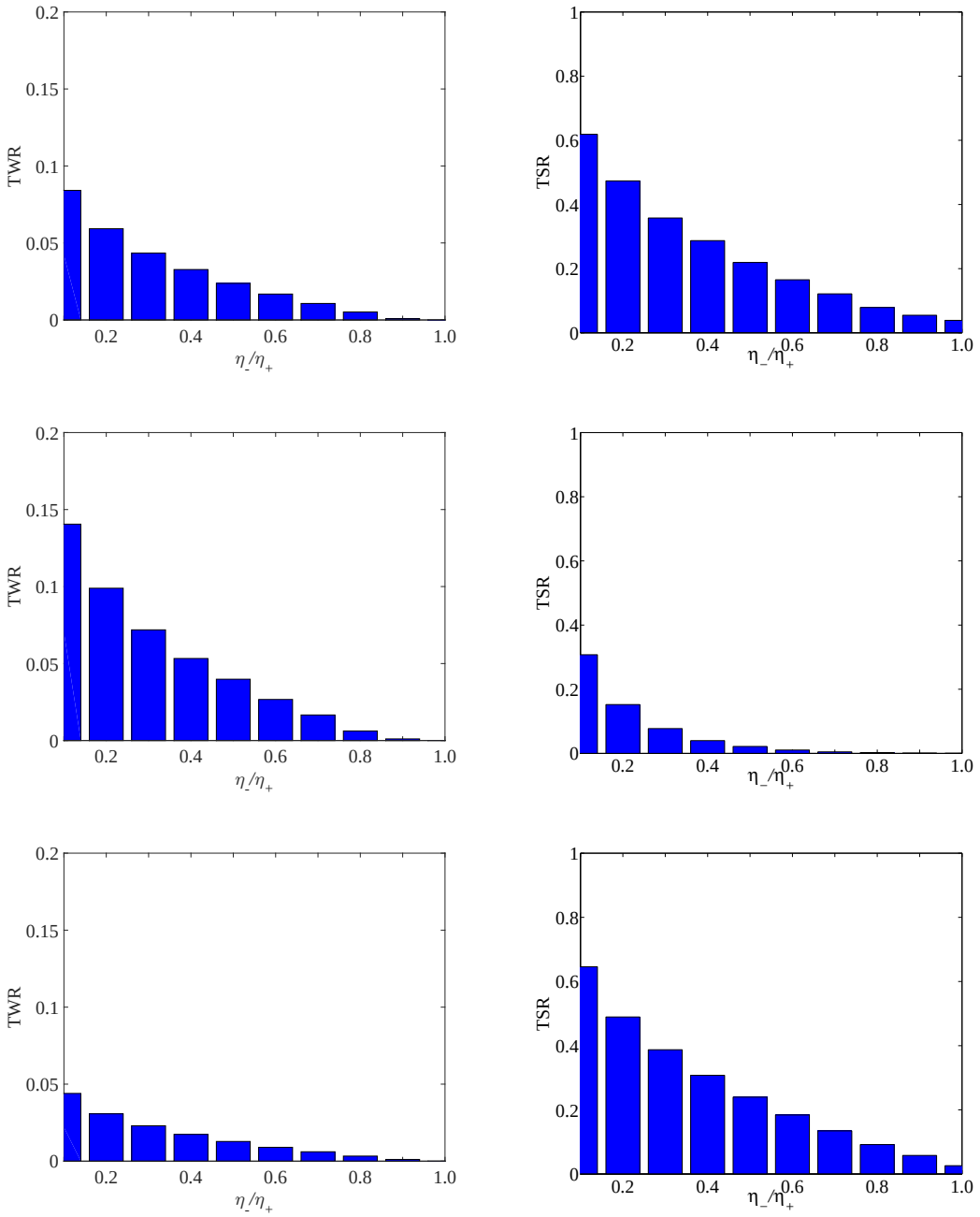


Figure 11: Organisation performance (at steady state) changes as the learning ratio η_-/η_+ increases. Baseline case (upper), increasing skill set (middle) and increasing task size (lower).

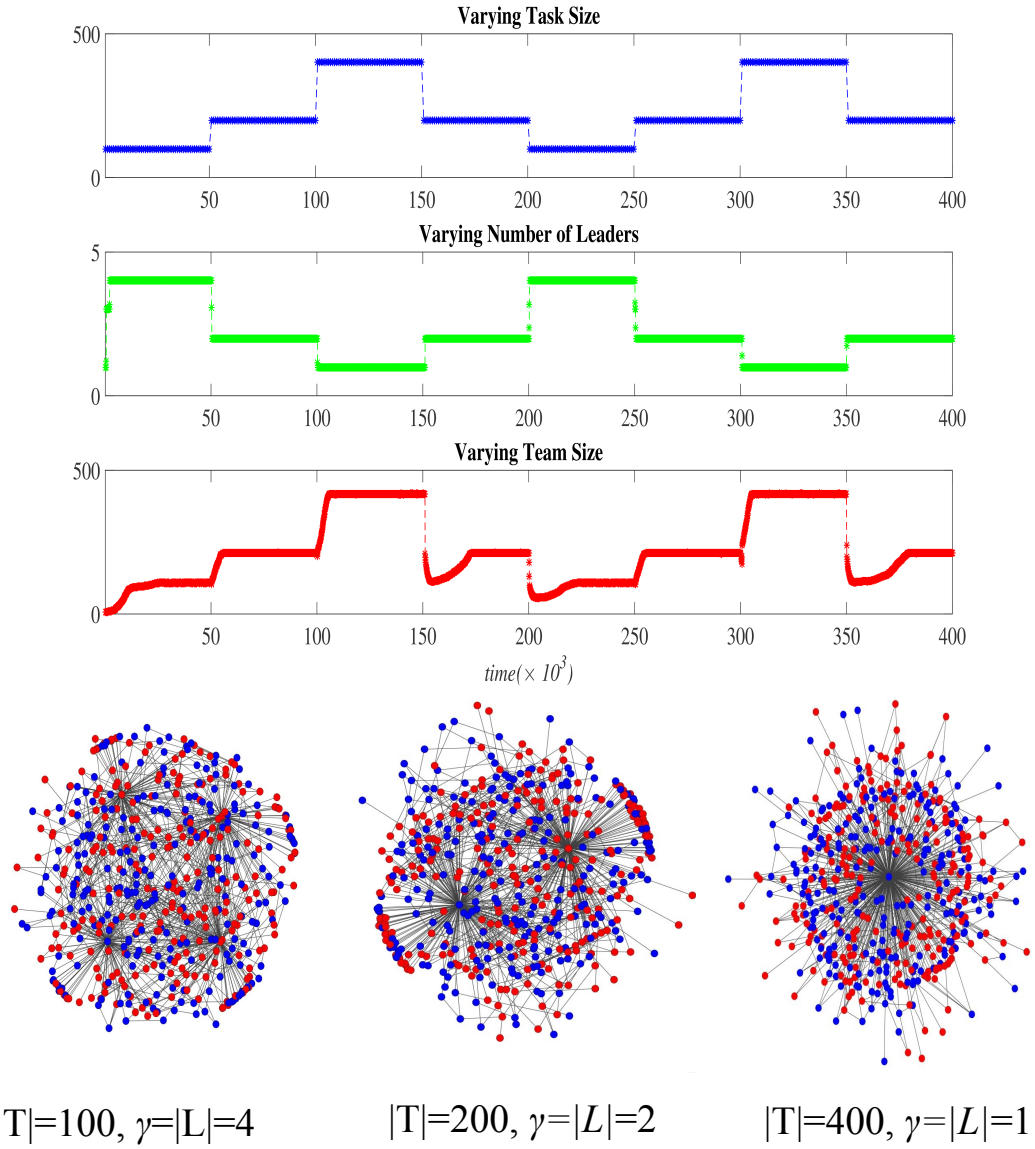


Figure 12: Team size follows dynamic changes in skill demand. The upper row shows changes in task size along the simulation. The middle rows shows the number of leaders. The lower row shows the corresponding changes in actual team size. We use a ratio of learning rates $\eta_-/\eta_+ = 0.1$ and a size of skill set $k = 2$. Three snapshots show the radical reorganisation of the system under changing demand.

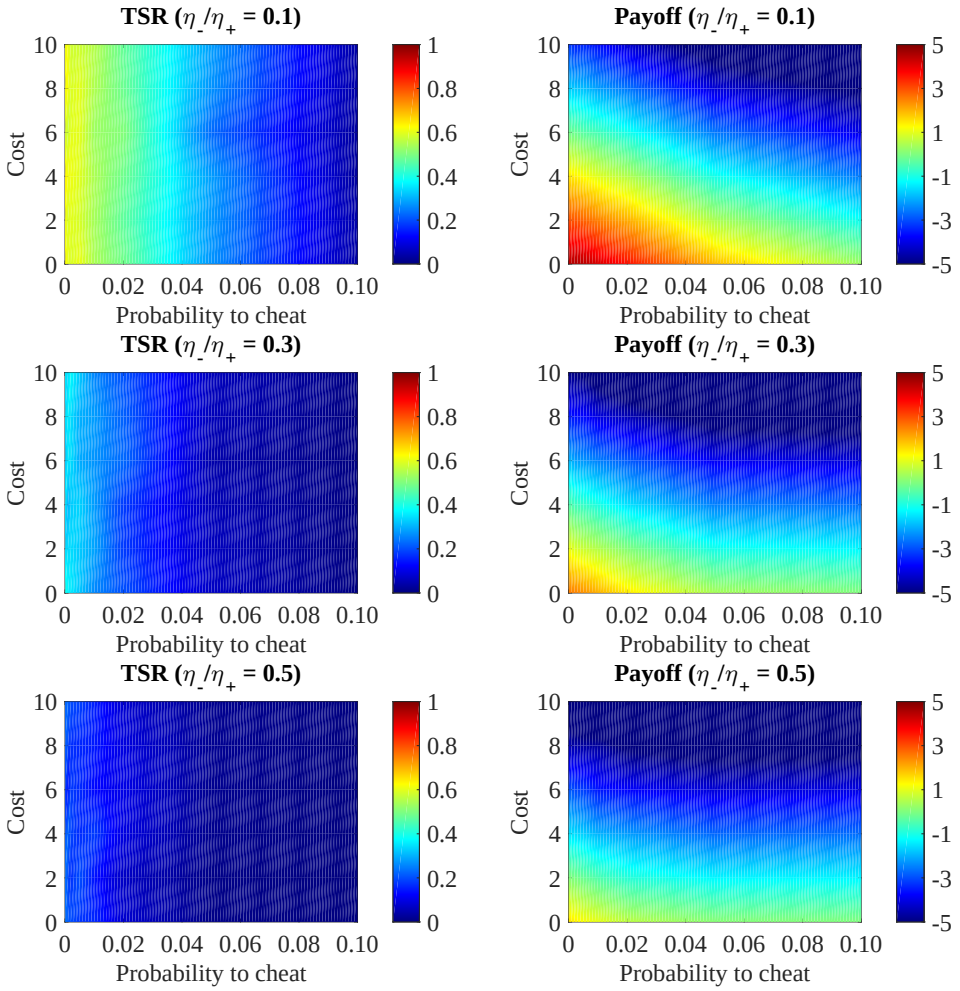


Figure 13: Task success rate and average agent payoff at stationary state with the change of cost c and probability to cheat p_c. A higher payoff is obtained when c and p_c are smaller. As for the TSR, it is dependent on the probability to cheat and the ratio of learning rates η_-/η_+.