Empowering Self-Driving Networks

Patrick Kalmbach[1] Johannes Zerwas[1] Peter Babarczi[1] Andreas Blenk[1]
Wolfgang Kellerer[1] Stefan Schmid[2]

[1] Technical University of Munich, Germany [2]University of Vienna, Austria

ABSTRACT

As emerging network technologies and softwareization render networks more and more flexible, the question arises of how to exploit these flexibilities for optimization. Given the complexity of the involved network protocols as well as the context in which the network is operating in, such optimizations are increasingly difficult to perform. A particularly interesting vision in this regard are "self-driving" networks: networks which measure, analyze and control themselves in an automated manner, reacting to changes in the environment (e.g., demand), while exploiting existing flexibilities to adjust and optimize themselves as needed.

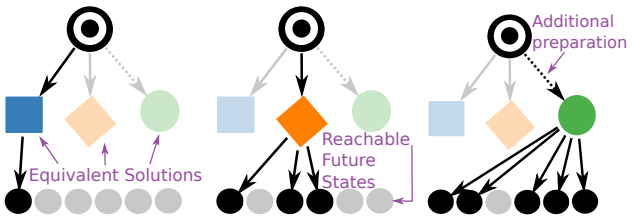
A fundamental challenge faced by any (self-)optimizing network concerns the limited knowledge about future changes in the demand and environment in which the network is operating. Indeed, given that reconfigurations entail resource costs and may take time, an "optimal" network configuration for the current demand and environment may not necessarily be optimal also in the near future. Thus, it is desirable that (self-)optimizations also *prepare* the network for possibly unexpected events in the near future.

This paper makes the case for *empowering* self-driving networks: empowerment is an information-centric measure which allows to account for how "prepared" a network is and how much flexibility is preserved over time. While empowerment has been successfully employed in other domains such as robotics, we are not aware of any applications in networking. As a case study for the use of empowerment in networks, we consider self-driving networks offering topological flexibilities, i.e., reconfigurable edges.

KEYWORDS

Self-driving Networks, Optimization, Network Intelligence

SelfDN' 18, August 2018, Budapest, Hungary



(a) Traditional. **(b) Empowerment-Guided.** **(c) Empowerment-Optimized.**

Figure 1: Approach comparison.

1 INTRODUCTION

The increasing complexity of communication networks, their continuously changing requirements (e.g., in terms of demands and workloads), and their complex objective functions, render their management *in real time* almost impossible for human operators with today's tools. Indeed, there is an increasing consensus that network operations should be supported by data-driven, machine-learning-based models revolving around more high-level goals and a holistic view of the underlying network [4]. An increased automation bears the potential to not only simplify network operations but also enable more fine-grained optimizations, fully leveraging the available network data rather than relying on predefined models. A vision emerges of fully self-driving networks which measure, analyze and control themselves continuously.

Another major motivation for self-driving networks comes from the increasing flexibilities offered in modern communication networks. Indeed, over the last years, networks have become more and more software-defined and reconfigurable. However, exploiting such flexibilities is non-trivial. Even with state-of-the-art machine learning algorithms and exploiting all the available information about the current network state and demands, any algorithm optimizing the network is faced with the challenge that there is an inherent uncertainty regarding the future: there may be an unexpected spike in the demand, an edge failure or any other unexpected behavior of a component internal or external to the network. Moreover, given that even in the most flexible and highly reconfigurable self-driving network, configuration changes

typically come at a cost, e.g., in terms of resource consumption or time (and hence service disruption). Accordingly, in addition to adjusting optimally to the *current* situation, a self-driving network should also be *prepared* for the requirements that may come up in the near future. That is, networks should be optimized *robustly*, accounting both for the present and possible future demands.

This paper initiates the discussion of how to rigorously optimize (self-driving) communication networks while keeping them flexible in the near future. Indeed, while the need for this additional "preparedness" may seem intuitive and known on an anecdotal level, little is known on how to actually prepare a self-driving network and enhance it with the intelligence it will need to perform well also in the future. Worse, today, we even lack good definitions and models to capture such properties.

In order to fill this gap, we in this paper establish a connection to an intriguing notion of *preparedness*, so far only used and successfully applied in other domains, such as robotics. In particular, we make the case for applying the framework of *empowerment* [11] to communication networks. Empowerment is an information-theoretic measure to quantify the influence of an "agent" (or actor) on its environment.

Our Contributions. In this position paper, we make the case for introducing concepts of empowerment in future self-driving networks: networks for which specifying complex objective functions is undesirable and cumbersome and which should not only optimize for the current demand but also be prepared for upcoming changes. We describe the challenges involved in enhancing communication networks with empowerment and present a concrete proposal based on a basic case study revolving around emerging reconfigurable network designs. Our preliminary experiments demonstrate that using empowerment as a driver for action selection, high reconfiguration costs and potentially harmful situations may be avoided.

Novelty and Related Work. Our approach radically differs from classical approaches which use utility functions to guide optimizations and suffer from the drawback of having to design and tweak the functions on a case by case basis. The concept of empowerment itself was introduced in [11], and studied by many authors subsequently, e.g., [1, 14, 16]. empowerment can be used as a task-independent, intrinsic motivation to restructure the environment: in our case *the network*. In general, the need and benefits of being "flexible" [9] or "prepared" is an ever-green topic in networking research, and e.g., studied intensively in the context of oblivious routing [2] (where routes need to be defined without full knowledge of the traffic matrix) or resilient routing [17]. However, to the best of our knowledge, we are the first to consider empowerment in the context of communication networks, motivated by the advent of self-driving networks.

2 EMPOWERMENT IN NETWORKS

In this section, we introduce the concept of *empowerment*, discuss its application in communication networks, and initiate the discussion of a case study of a reconfigurable network which needs to serve routing requests.

2.1 General Concepts

empowerment is a concept from agent (game) theory. It is motivated by the observation that living organisms strive for states that give them maximum control or impact over their environment: Everything else being equal, states are preferable which (1) keep as many options as possible open, or (2) whose actions have the greatest influence on the direct environment [11, 15]. The concept of empowerment is an attempt to formalize and quantify the influence an agent has on its environment. This is in stark contrast to current approaches in communication systems. Fig. 1 illustrates this difference abstractly as a state transition diagram. The middle layer in each subfigure corresponds to a system state associated with one objective value. The bottom layer corresponds to future system states. Fig. 1a shows a traditional optimization. Traditional systems usually only consider the imminent objective value. But multiple solutions with the same objective value can exist, which one a traditional system chooses is undefined. All solutions are equally good. Thus, the traditional approach might select a solution that moves the system into states with few future options. Fig. 1b shows a possible state transition that is guided by empowerment. A system that additionally considers empowerment for selecting a solution transitions into states, from which more future states can be reached. The system has thus more options and may be in a better shape to react. Fig. 1c shows a system that not only considers empowerment for the selection of equivalent solutions, but additionally maximizes empowerment. That is, arranges free resources, *or restructure the environment*, in such a way, that it is *prepared* for the future, further maximizing the number of reachable states.

A key aspect of empowerment is the agent's *embodiment*: The sensor and motor capacities of an agent in an environment [14]. The interplay of the agent with the environment can be represented as a perception-action loop, where the agent influences the environment through its actuators, and receives a perception of the resulting environment state through its sensor [14].

An agent, situated in a time-discrete model, chooses an action for the next time step based on the sensory information of the current time step. The action influences the state of the environment, which in turn influences the sensor input of the agent in the next time step. The cycle then repeats itself. This perception-action loop can also be modeled formally:

- the sensor S taking values $s \in \mathcal{S}$,

- the actuator A taking values $a \in \mathcal{A}$,
- the rest of the environment R taking values $r \in \mathcal{R}$.

Note that both the sensor variables can be random (e.g., due to measurements) as well as the actuators (enabling randomized agents). The relationship between the random variables can be expressed as a time-unrolled *Causal Bayesian Network (CBN)* . The perception-action loop can be understood as a probabilistic channel, and empowerment is defined as the channel capacity between the agent's own actuator A and sensor S at a later time step:

$$\mathrm{E} := C(A_t \rightarrow S_{t+1}) = \max_{p(a_t)} I(S_{t+1}, A_t), \tag{1}$$

where $I(S_{t+1}, A_t)$ is the mutual information between two random variables [15]:

$$I(S_{t+1}, A) := \sum_{s \in \mathcal{S}} p(s) \sum_{a \in \mathcal{A}} p(a \mid s) \log p(a \mid s) - \sum_{s \in \mathcal{S}} p(s) \log p(s). \tag{2}$$

Eq. (2) is called mutual information, the first term corresponds to the conditional entropy, and the second term to the standard Shannon entropy. The Shannon entropy measures the uncertainty of a random variable. The conditional entropy measures the uncertainty in S_{t+1} once A is known. Mutual information then measures the average information one can gain about S_{t+1} by observing A [15].

For a general setting with noise present in the channel, e.g., random node and link failures, empowerment is calculated using the Blahut-Arimoto Algorithm, which is, however, computationally expensive [15]. More efficient techniques exist, but are not straight forward to analyze [8].

It is important to note that empowerment represents only the *potential* information flow. The agent calculates how it *could* affect the environment, and does not materialize its potential [16]. The use of channel capacity provides a number of desirable properties [14]:

- *It is* **agent centric:** Only information accessible to the agent is used, i.e., (samples) from the perception action loop (sensi-motoric data).
- *It features* **locality:** No global knowledge of the world is necessary.
- *It is* **well-defined and computable:** Due to the channel formulation, standard information-theoretic quantities and established methods can be used for its calculation
- *It is* **semantically unbiased:** No external value system is introduced.

Especially the last point sets empowerment apart from usual reinforcement learning approaches such as [12, 13]. In reinforcement learning, the designer has to define a specific reward signal (e.g., related to Quality of Service or Quality of Experience parameters). Defining a specific objective function is, however, often non-trivial, as it is multi-dimensional and depends on potentially many aspects (e.g., on routing latency, resilience, etc.). In contrast, empowerment depends only on the agent embodiment and the environment. This does not only radically simplify the design of self-driving networks but also renders it more flexible (e.g., the same agent can be used in different contexts) and "prepared".

2.2 Application: Reconfigurable Networks

We are concerned with the question of how to leverage the ideas of empowerment in the context of (self-driving) communication networks. To this end, we specify the different components, that is, the environment, actions and sensors, for a specific application in networking: *routing in reconfigurable networks*. We choose this case study because it is rather general (allowing not only to select and adapt routing requests but also edges) and challenging; hence, it shows the potential and limitations of empowerment. Moreover, reconfigurable networks are an emerging and not well-understood paradigm [3, 5, 6].

The Context. We model the network as a capacitated graph $G = (\mathcal{N}, \mathcal{E})$ and are given a set of routing requests (to be served on the graph). The function $b : \mathcal{E} \rightarrow \mathbb{N}$ gives the capacity of an edge and the function $c : \mathcal{E} \rightarrow \mathbb{R}$ the cost of using an edge. We model a request as a source-destination pair given by the triple (s, t, d), where $s \in \mathcal{N}$ is the source, $t \in \mathcal{N}$ the destination and $d \in \mathrm{N}$ the demand (in routing units). A set of requests (or demand) is denoted by $\mathcal{D}$. We consider reconfigurable networks whose edges can be adjusted, e.g., leveraging emerging technologies in datacenters [5, 6] or in Wide-Area Networks [3]. Reconfigurable topologies are helpful in the face of changing traffic patterns, since they allow the network to adapt to a new pattern, and thus mitigate e.g., congestion.

The Problem. The problem is to design a topology that can serve as many requests with as few reconfigurations as possible. To achieve this, the topology needs a measure of *preparedness* with respect to the traffic in the near future. In particular, we argue that a self-driving network should be capable of using available reconfigurable edges in such a way that the perceived average latency of flows is minimized, or the number of successfully routed flows is maximized, given a limited number of reconfigurations.

The Solution. Empowerment presents itself naturally as a principled solution to this problem, where an agent restructures the network, consequently actions correspond to the configuration of edges, and the sensor relates to routed flows or packets. Intuitively, an agent has high empowerment if the edges it can reconfigure result in, say, many routed flows.

The Challenge. The actuators and sensors for the agent need to be formulated with care. If, for example, the changes

made in the environment are not perceivable with the sensor, then the agent has empowerment 0 (by Eq. (2)) and the approach is bound to fail. If, on the other hand, the actuators and sensors are too complex, then calculation of empowerment can be computationally expensive since the perception-action loop needs to be sampled many times, which can be costly itself. The calculation of empowerment must also be fast to allow the agent to react quickly.

The Environment includes the graph G as well as the routed requests. Thus, for a given demand $\mathcal{D}$, the environment is defined as $\mathcal{R} := \{(\mathcal{D}_r, \mathcal{E}') \mid \mathcal{D}_r \in 2^{\mathcal{D}}, \mathcal{E}' \in 2^{\mathcal{N}\times\mathcal{N}}\}$, where D_r is the set of routed requests and $\mathcal{E}'$ the set of realized edges.

The Agent is equipped with a set of actuators and one sensor, which together form its embodiment.

The Actuators: To achieve a large empowerment, the agent's actuators must be able to actually have an impact of the environment. We define multiple ones:

- EdgePlacer (EP) establishes an arbitrary edge in the network, taking values $\mathcal{A}_{EP} := \mathcal{N} \times \mathcal{N}$. If an existing edge is chosen, the placement of the edge corresponds to an increase of the capacity on that edge.
- EdgeRemover (ER) removes an arbitrary existing edge. It takes values $\mathcal{A}_{ER} := \{e \mid e \in \mathcal{E}^t\}$.
- RequestPlacer (RP) chooses an arbitrary request from $\mathcal{D}$ that is not yet routed at time t, $\mathcal{A}_{RP} := \mathcal{D}_r^t / \left(\mathcal{D}_r^t \cap \mathcal{D}\right)$ and tries to find a shortest path in the graph. On success, the demanded resources are allocated.
- RequestRemover (RR) chooses an arbitrary request routed, and removes it from the graph, $\mathcal{A}_{RR} := \mathcal{D}_r^t$.
- Idler correspond to "*do nothing*"; this actuator does not change the environment.

The Sensors: Similar to the actuators, if empowerment is to be meaningful, the sensors must relate to the environment. We define the following two sensors as functions of an environment state r and a positive number $l \in \mathbb{R}^+$:

- ExactRequests (ER): $\mathrm{ER}(r, l) := \mathcal{D}_r$ `if` $\mid \mathcal{D}_r \mid > l$ `else` $\emptyset$ returns the set of currently realized requests, if the number of realized requests is larger than a specific value. Else, the empty set is returned.
- NumRequests (NR): $\mathrm{NR}(r, l) :=\mid \mathcal{D}_r \mid$ `if` $\mid \mathcal{D}_r \mid > l$ `else` 0, returns the number of currently realized requests, if that number is larger than l. Else zero is returned.

In addition, we vary the sensor threshold l. In one set of experiments we keep $l = 0$ constant, in another set of experiments the agent dynamically adapts the threshold based on the perceived routed requests. Threshold adaptation is abbreviated with a F. Agents can achieve high empowerment by accepting at each sequence only a small number of requests that vary between sequences. Given two states r_1, r_2 that result in the same number of different sensor readings, an agent has no incentive to choose r_1 over the r_2, even if the number of accepted requests is higher in r_1. By adapting the threshold the agent himself discovers without external influence states with high empowerment in which different large request sets are accepted. It is important to note that we did not introduce any external value or signal.

We examine four embodiments in our evaluation: `ExactBuilder (EB)`, `SimpleBuilder (SB)`, `ExactController (EC)`, `SimpleController (SC)` with the following actions/sensor:

- EB := ({EP, ER, Idler}, ER)
- SB := ({EP, ER, Idler}, NR)
- EC := ({EP, ER, Idler, RP, RR}, ER)
- SC := ({EP, ER, Idler, RP, RR}, NR)

Exact indicates that the embodiment uses the ER sensor and Simple the NR one. Builder indicates that the agent has only control over the graph of the network, i.e., can place and remove edges. In contrast, the Controller embodiment has additionally control over the routed requests.

As a consequence, the builders can influence their corresponding sensor only indirectly by placing or removing edges and thus enabling or disabling requests. A sensor reading is obtained by sequentially performing shortest path routing for all requests or a subset of requests of demand $\mathcal{D}$. To mitigate the effect of request ordering, we take the sensor reading corresponding to the largest amount of routed requests out of ten random permutations or subsets of $\mathcal{D}$. The controllers, in contrast, have direct influence on their sensor due to their ability of adding/removing requests.

3 PRELIMINARY EXPERIMENTS

In order to shed some first light on the potential benefits but also limitations of using empowerment in networks, we conducted a case study of a self-driving network which adjusts to the routing requests it has to serve. We consider a simple discrete system where an agent chooses, at each time step, the action maximizing the empowerment.

3.1 Algorithms

The main objective considered in literature is to maximize the number of accepted and routed requests in the network [5, 7]. Thus, for comparison, we introduce an Integer-Linear-Program (ILP) that serves as an optimal baseline. Since ILPs are computationally expensive to compute, we introduce a simulated annealing heuristic. Both, the ILP and the heuristic are prototypical: they optimize for point estimates of demand, and do not actively prepare for future demand. In particular, they do not take uncertainty in the request distribution into account, or prepare the network with

respect to possible future changes in demand. Due to space we keep the algorithm descriptions short, but will release our source code to aid reproducibility.

Exact Baseline (ILP). A canonical goal is to compute a network which maximizes the number of routed requests, while minimizing the bandwidth cost as a secondary objective:

$$\min \sum_{d\in\mathcal{D}} \sum_{e\in\mathcal{E}} c(e)\cdot x_d(e) - \gamma \sum_{d\in\mathcal{D}} r(d), \tag{3}$$

where $r(d)$ and $x_d(e)$ are binary variables representing if demand d is routed and its flow value along edge e, respectively; while γ is a large positive constant, e.g., $\gamma > \sum c(e)$.

For the sake of modelling, we extend G with unit cost and zero capacity ($b(e) = 0$) edges to a full-mesh graph. In this graph every edge is replaced by two anti-parallel arcs with identical cost and capacity values with its corresponding (undirected) edge. The following constraints are required:

$$\forall d \in \mathcal{D}, \forall i \in \mathcal{N}:$$

$$\sum_{(i,j)\in\mathcal{E}} x_d(i,j) - \sum_{(j,i)\in\mathcal{E}} x_d(j,i) = \begin{cases} r(d) & \text{, if } i = s \\ -r(d) & \text{, if } i = t \\ 0 & \text{, otherwise} \end{cases}, \tag{4}$$

$$\forall (i,j) \in \mathcal{E} : \sum_{d\in\mathcal{D}} x_d(i,j) + x_d(j,i) \le p(i,j), \tag{5}$$

$$\forall (i,j) \in \mathcal{E} : p(i,j) = p(j,i), \tag{6}$$

$$\forall i \in \mathcal{N} : \sum_{j\in V:(i,j)\in\mathcal{E}} [p(i,j) - b(i,j)] \le 3, \tag{7}$$

$$\sum_{e\in\mathcal{E}} [p(e) - b(e)] \le 2I, \tag{8}$$

$$\sum_{e\in\mathcal{E}} |p(e) - b(e)| \le 2B. \tag{9}$$

Eq. (4) formulates the flow conservation for every routed demand (i.e., $r(d) = 1$). The empowered capacity $p(e)$ is set in Eq. (5)-(6) for every undirected edge by summing up the flow values on its corresponding directed arcs. The total number of reconfigurable edges per node is bounded in Eq. (7). In Eq. (8)-(9) the capacity increase (I) and reconfigurations (B) are bounded (multiplied by two because we have to perform them on both bi-directional arcs), respectively.

Heuristic Baseline (HEU). Simulated Annealing is a heuristic approximation method motivated by the Metropolis-Hastings algorithm [10]. We use it to place and move edges in an initial topology in such a way that the number of routed flows is maximized.

The algorithm works as follows: Starting from an initial solution, it randomly decides to place or remove an edge. The probability of the two events depends on the current inventory. If the inventory is empty an edge is taken with high probability and vice verca. In case of an edge placement the algorithm filters out all those nodes in the network that already have a maximum degree. The starting node of the edge is sampled with probability proportional to the number of request sources and destinations located at each node. Only nodes with a degree smaller than the limit are considered. A target node is then drawn uniformly at random from all requests that have the source node of the edge as source. In case of an edge removal any existing edge is removed uniformly at random.

Empowerment Algorithm. We will focus on a deterministic setting and algorithm in our case study: for every action the agent performs in a given state, always only one sensor reading is perceived. In this case Eq. (2) reduces to the logarithm of all perceivable sensor readings given an initial state [16]. This greatly simplifies computation and analysis.

We also note that the characteristics of the agent-environment interaction, that is, the effect of placing or removing an edge, might become distinguishable only after several steps. Accordingly, we consider *n-step* empowerment: We consider not a single action A_t but a sequence of variables for the n next time-steps, $(A_t, \ldots, A_{t+n})$, and consider only the sensor reading S_{t+n+1} [15].

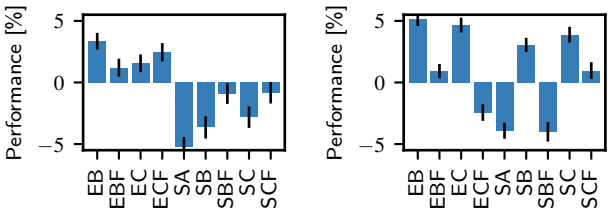
With increasing sequence length calculation, computing exact empowerment values becomes quickly intractable, since the evaluation of $|\ \mathcal{A}\ |^n$ action sequences would be required. Therefore we use *sparse sampling* [16] to approximate n-step empowerment.

3.2 Evaluation

In our scenario, we consider an undirected graph with 30 nodes. Each node can have at most three reconfigurable edges, resulting in a maximum number of 45 edges. Edges have unit capacity in this scenario, and we consider all simple s-t pairs with unit demand as requests, i.e., $\mathcal{D} := \{(s,t,1) \mid (s,t) \in \mathcal{N}\times\mathcal{N} \wedge s \neq t\}$. Starting from an empty network, we let agents maximize their empowerment for at least 500 time steps, or if no new best empowerment value was observed for 100 time steps. The agent chooses that action resulting in the state with highest empowerment, and breaks ties by randomly sampling an action. ILP and SA maximize the number of accepted requests out of $\mathcal{D}$. We give the ILP an infinite action budget for this purpose.

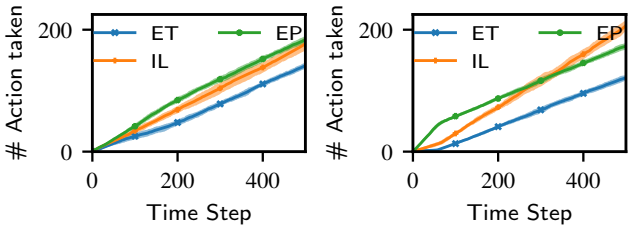
Comparing Preparedness. To evaluate preparedness, we draw 100 demands $\mathcal{D}'_1, \ldots, \mathcal{D}'_{100}$ having 45 requests without replacement from $\mathcal{D}_s$. We then use the ILP with action budgets of $B \in \{5, 15\}$ to get the maximum number of routable requests for each $\mathcal{D}'_i$ for the topologies previously generated by ILP, SA and empowerment. The action budget B and sequence length n coincide, e.g., for $B = 5$ we consider for the agents topologies created with $n = 5$.

Fig. 2 shows the difference in the accepted requests in percent relative to the ILP as bar plot with standard error. Greedy maximization of empowerment can indeed result



(a) $n = 5$ (b) $n = 15$

Figure 2: Accepted requests in percent relative to the ILP for sequence length $n = 5$ and $n = 15$. A Positive value indicates an improvement, a negative value a degradation compared to the ILP.



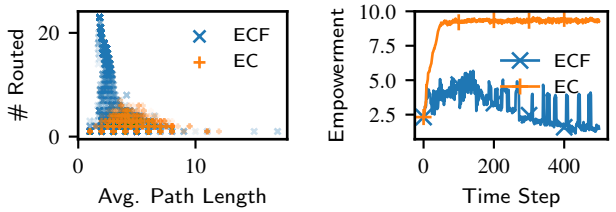
(a) EB, $l = 0$ (b) EB, l adaptive

Figure 3: Chosen actions and empowerment for the EB with sequence length $n = 10$.

in topologies, which are flexible with respect to varying demands. Fig.2a shows that for $B = 5$ the topology designed by the EB is able to accept 2.5 % more requests on average than the ILP, and for $n = 15$ even 5 % more. Embodiments with the ER sensor always outperform the ILP. In contrast, Fig. 2a shows that embodiments with the NR sensor always perform worse than the ILP for $n = 5$, while for $n = 15$ (Fig. 2b) SB and SC outperform the ILP. All embodiments always perform at least as good as SA.

This analysis shows that depending on the embodiment empowerment-driven agents are indeed able to structure a network towards the specific purpose of accepting varying future demands.

Chosen Actions. Now one may wonder whether the agent simply stumbled over a good state, or whether it chose its actions with purpose? Fig. 3 shows how often the EB with- and without sensor adaptation chose an action as cumulative sum over time, averaged across ten executions. The shaded area corresponds to one standard deviation. The results are exemplary and similar for the respective actuators across all embodiments. The agent chose actions with purpose. Fig. 3a, and Fig. 3b show that the corresponding agent placed edges during the beginning of each run. This behavior is more pronounced for the EB with sensor adaptation. After about 100 time steps, the curve for the Idler actuator increases, indicating that the EB actively chooses to do nothing. The



(a) Routing Quality (b) Empowerment

Figure 4: Number of routed requests and average path length for EC with and without filtering, with $n = 10$.

curves for EP and ER become parallel, indicating that they are sampled in equal measure due to random tie breaks.

Thus, the agent displays purpose behind the choice of actions. In the beginning placing edges increases the empowerment, since this allows routing of requests for the first time. The EB with sensor adaptation has a stronger incentive to place edges, since a network with more edges can host more requests. After a certain time empowerment cannot be improved any further by placing edges, and the agents stay where they are or take actions at random due to tie breaks.

Sensor Adaptation and Request Selection. To understand how threshold adaptation impacts selection of routed requests and topology design, we investigate the average path length and number of paths in the graph for the EC and ECF with sequence length $n = 10$. Fig. 4a plots the average path length against the number of routed requests, and shows that threshold adaptation leads to an increase in routed requests, and a decrease in the path length. Thus the ECF successfully learned to place edges and choose requests such that requests are routed with low resource footprint.

The increased acceptance ratio comes at a cost. Fig. 4b shows the average empowerment across ten runs for the EC and ECF. The empowerment for the ECF is significantly less than for the EC, and decreases as the filtering threshold increases. Comparing the maximum values relativizes this gap. The EC achieves Empowerment of 9.88, and the ECF of 7.54. Thus, the ECF is able to find states with relative high empowerment, allowing the routing of different large request sets.

4 CONCLUSION

We understand our work as a first step and believe that network empowerment opens many interesting directions for future research. So far, we have focused on a single and simple case study only, and it would also be interesting to consider to study the use of empowerment to deal with faulty networks where, e.g., links may fail.

REFERENCES

[1] Tom Anthony, Daniel Polani, and Chrystopher L. Nehaniv. 2011. Impoverished Empowerment: 'Meaningful' Action Sequence Generation

through Bandwidth Limitation. In *Advances in Artificial Life. Darwin Meets von Neumann*, George Kampis, István Karsai, and Eörs Szathmáry (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 294–301.

[2] Yossi Azar, Edith Cohen, Amos Fiat, Haim Kaplan, and Harald Räcke. 2004. Optimal oblivious routing in polynomial time. *J. Comput. System Sci.* 69, 3 (2004), 383–394.

[3] S. Das, G. Parulkar, and N. McKeown. 2013. Rethinking IP core networks. *IEEE/OSA Journal of Optical Communications and Networking* 5, 12 (Dec 2013), 1431–1442.

[4] Nick Feamster and Jennifer Rexford. 2017. Why (and How) Networks Should Run Themselves. *arXiv preprint arXiv:1710.11583* (2017).

[5] Monia Ghobadi, Ratul Mahajan, Amar Phanishayee, Nikhil Devanur, Janardhan Kulkarni, Gireeja Ranade, Pierre-Alexandre Blanche, Houman Rastegarfar, Madeleine Glick, and Daniel Kilper. 2016. ProjecToR: Agile Reconfigurable Data Center Interconnect. In *Proc. ACM SIGCOMM*. ACM, New York, NY, USA, 216–229. https://doi.org/10.1145/2934872.2934911

[6] Navid Hamedazimi, Zafar Qazi, Himanshu Gupta, Vyas Sekar, Samir R Das, Jon P Longtin, Himanshu Shah, and Ashish Tanwer. 2014. Firefly: A reconfigurable wireless data center fabric using free-space optics. In *Proc. ACM SIGCOMM Computer Communication Review (CCR)*, Vol. 44. 319–330.

[7] Xin Jin, Yiran Li, Da Wei, Siming Li, Jie Gao, Lei Xu, Guangzhi Li, Wei Xu, and Jennifer Rexford. 2016. Optimizing bulk transfers with software-defined optical WAN. In *Proc. ACM SIGCOMM*. 87–100.

[8] M. Karl, J. Bayer, and P. van der Smagt. 2015. Efficient Empowerment. *ArXiv e-prints* (Sept. 2015).

[9] Wolfgang Kellerer, Arsany Basta, and Andreas Blenk. 2015. *Flexibility of Networks: a new measure for network design space analysis?* Technical Report. Lehrstuhl fÃijr Kommunikationsnetze.

[10] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. 1983. Optimization by Simulated Annealing. *Science* 220, 4598 (1983), 671–680. arXiv:http://science.sciencemag.org/content/220/4598/671.full.pdf http://science.sciencemag.org/content/220/4598/671

[11] A. S. Klyubin, D. Polani, and C. L. Nehaniv. 2005. Empowerment: a universal agent-centric measure of control. In *2005 IEEE Congress on Evolutionary Computation*, Vol. 1. 128–135 Vol.1.

[12] Hongzi Mao, Mohammad Alizadeh, Ishai Menache, and Srikanth Kandula. 2016. Resource Management with Deep Reinforcement Learning. In *Proceedings of the 15th ACM Workshop on Hot Topics in Networks (HotNets '16)*. ACM, New York, NY, USA, 50–56.

[13] Hongzi Mao, Ravi Netravali, and Mohammad Alizadeh. 2017. Neural Adaptive Video Streaming with Pensieve. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication (SIGCOMM '17)*. ACM, New York, NY, USA, 197–210.

[14] Phillipe Capdepuy. 2010. *Informational Principles of Perception-Action Loops and Collective Behaviours*. Phd Dissertation. University of Hertfordshire.

[15] Christoph Salge, Cornelius Glackin, and Daniel Polani. 2013. Empowerment - an Introduction. *CoRR* abs/1310.1863 (2013).

[16] Christoph Salge, Cornelius Glackin, and Daniel Polani. 2014. Changing the Environment Based on Empowerment as Intrinsic Motivation. *CoRR* abs/1406.1767 (2014).

[17] Martin Suchara, Dahai Xu, Robert Doverspike, David Johnson, and Jennifer Rexford. 2011. Network architecture for joint failure recovery and traffic engineering. In *Proceedings of the ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems*. ACM, 97–108.