The Impact of Multi-scale Control Topology on Asset Distribution in Dynamic Environments

Payam Zahadat
Computer Science Department
IT University of Copenhagen
Copenhagen, Denmark
paza@itu.dk

Ada Diaconescu
Telecom Paris, LTCI
Institute Polytechnique de Paris
Paris, France
ada.diaconescu@telecom-paris.fr

***Abstract*—In many self-organising systems the ability to extract necessary resources from the external environment is essential for growth and survival. E.g., extracting sunlight and nutrients in organic plants, monetary income in business organisations and mobile robots in intelligent swarms. When operating within competitive, changing environments, such systems must distribute their assets wisely, to improve and adapt their ability to extract available resources. As the system size increases, the asset-distribution process often gets organised around a multi-scale control topology. This topology may be static (fixed) or dynamic (enabling growth and structural adaptation) depending on the system's constraints and adaptive mechanisms. In this paper we expand on a plant-inspired asset-distribution model and study the impact that the topology of the multi-scale control process has upon the system's ability to self-adapt asset distribution when resource availability changes within the environment. Results show how different topological characteristics and different competition levels between system branches impact overall system profitability, adaptation delays and disturbances when environmental changes occur. These findings provide a basis for system designers to select the most suitable topology and configuration for their particular application and execution environment.**

***Index Terms*—self-adaptive asset distribution, multi-scale control topology, dynamic environment**

I. Introduction

Autonomic and self-organising systems must manage available resources from their environment and invest them efficiently into internal assets, to ensure their growth, competitiveness and survival [2]. Examples range from natural systems (e.g. trees) through social systems (e.g. business organisations) to cyber-physical systems (CPS) (e.g. robot swarms). To improve their resource intake within changing environments, such systems must often self-adapt their structures and their *asset distribution* within those structures. To ensure viability as the amount of managed assets increases, the system's self-adaptive control often takes the form of a multi-scale topology.

Multi-scale structures (e.g. [6], [7]) include multiple abstraction levels, where each level increases the granularity of observation of the level below – i.e. information about the lower scale (micro) is lost in the abstraction to the higher scale (macro). This allows multi-scale systems to increase their scopes, or operation domains, while limiting the amount of resources needed to handle information at each scale. Hence, multi-scale schemes can achieve system-wide coordination among many self-adaptive processes in large systems [7].

While many self-adaptive algorithms exist for context-aware asset distribution across application domains (sec. II), much less is known about the impacts that the *topology* of the multi-scale control system has on the self-adaptation process. This paper aims to provide the basis for such analysis by identifying key topological features and linking them to generic characteristics of the self-adaptation process (e.g. reactivity and costs).

We illustrate three examples (Fig.1) of systems featuring self-adaptive asset distribution to help select the initial topologies to analyse. First, we consider a natural system example [22]: trees absorb sunlight, water and mineral resources and transform them into organic matter, in turn forming internal structures that are essential to growth and survival. Hence, depending on resource availability in the environment, trees self-adapt their growth process to prioritise development towards resource-rich areas (e.g. leafy branches growing towards sunny patches and roots towards moist, mineral-rich soils). Similarly, within the socio-economic realm, business organisations must employ their assets (e.g. workers) to fulfill service requests from their market environment, so as to grow and survive within a competitive context [20]. They must self-adapt to unexpected fluctuations in market demands by reallocating assets to the most popular service sectors. Finally, we consider a cyber-physical system (CPS) example: a robot swarm must coordinate their actions across several rooms to achieve a shared task (e.g. cleaning). Robot distribution across the rooms should self-adapt to the level of cleaning services required within each room, which may dynamically change [19], [23].

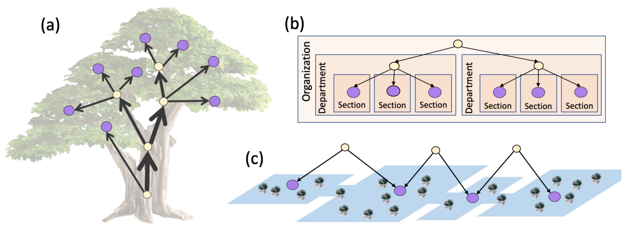

Fig. 1: Examples of systems that feature an adaptive distribution of internal assets. a) adaptive nutrient flows within a tree; b) adaptive budget investments within a business organisation; c) adaptive distribution of robots across various regions.

In previous works (e.g. in [18], [20] and generalised in [21]), we proposed decentralised asset distribution algorithms for systems with multi-scale control structures. The system's internal assets were employed to maintain and to grow system branches. Feedback from the branches indicated their efficacy in acquiring external resources. This feedback was used to skew asset distribution towards more successful branches.

Based on [21], here we aim to study the impact of system topology on the context-aware self-adaptation of asset distribution. We consider a control system's topological variation along three main dimensions:

- *Single or multiple roots*: resulting in single-rooted tree graphs or general directed acyclic graphs, respectively;
- *Static or growing topology*: in static cases, assets are relocated within a pre-existing system structure; in growing cases, assets also impact structural self-adaptation.
- *Single or multiple control scales*: resulting in single or multiple decision levels, respectively.

From the combinatorial space of these dimensions, we select several concrete topologies that we consider representative for natural and artificial systems (Fig. 3). On the one hand, we analyse several tree topologies for plants and business organisations – i.e. single-root with growing multi-scales, with static single-scale, or with static multi-scales. On the other hand, we consider multi-root topologies with a single control scale, as relevant to robot swarms carrying-out collective tasks – i.e. 'linear', 'circular' or 'complete' topologies. Here, control coordination between roots only occurs indirectly via observable impacts on the shared entities they control.

We study the self-adaptability of these topologies to variations in resource availability within the environment. We focus on environment changes that require system-wide coordination between self-adaptation processes (rather than being addressable via local self-adaptation). This choice allows studying the impact of the entire system control topology (rather than just local control sub-trees). In all cases, we introduce self-adaptation delays within controlled entities to emphasise the role of the control topology in the self-adaptation behaviour.

Results show the impact of system topology on key self-adaptation features, notably including: the *extent* of self-adaptability to context changes; the *duration* of self-adaptation; and the *disturbance in efficiency* during self-adaptation. Some of the key findings include (Cf. sec. VI):

1) Growing topologies self-adapt better to new opportunities (via asset re-specialisation) and provide more profits than static ones. This comes at the price of longer adaptation delays and profit disturbances during adaptation.
2) Multi-root topologies achieve global control coordination slower than tree topologies. This incurs longer adaptation delays and more profit disturbance. Similar effects would be observed in multi-scale topologies when including inter-scale communication delays (e.g. studied in [12]).
3) While some level of performance-oriented competition is necessary to drive self-adaptation, too much competition actually hampers self-adaptation by creating too much inertia and preventing the detection of new opportunities.

These findings provide a basis for system designers to reflect upon and select the most suitable topology for their particular application and execution environment. They also provide opportunities for further studies on larger and more complex topologies as in real-world systems.

II. Related Works

Asset distribution is a broad research topic covering numerous application domains (e.g. from dynamic VM allocation in cloud systems [8] and process scheduling in mixed-criticality real-time systems [11]; through adaptive data-mediation [4] and power networks [1]; and all the way to group organisation in social insects [10] or swarm robotics [23]). We can only mention a few of these approaches here, emphasising some of their key characteristics relevant to our study. Still, to the best of our knowledge, only limited studies assess the importance of the control topology in asset distribution approaches that are self-adaptive and multi-scale. e.g. [13] studies the impact of topology on decentralised data-collection in complex networks, yet these are non-adaptive and single-scale.

Concerning self-adaptive resource allocation in static structures, [5] proposes decentralised resource allocations in grid networks based on a spatial algorithm optimisation approach. [3] targets power grids to optimally distribute produced electricity to users, minimising their costs while covering their demands. [8] proposes self-adaptive resource-allocation in virtual environments, to deal with dynamically deployed services and their workload fluctuations by adding/removing application servers to clusters and virtual CPU cores to virtual machines. [15] also deals with resource allocation in clustered servers, by self-adapting the number of replicated databases in a clustered J2EE application when the load varies.

Resource distribution solutions with self-adaptive structures (e.g. growing), may concern, for instance, the long-term extension and restructuring of energy system infrastructures, by investing into new power plants to match shifting demands [1]. In data-mediation systems, [4] investigate self-growing and self-adapting structures as means to address unexpected changes in data sources/consumers, workloads and servers.

Another category of resource allocation solutions concerns self-adaptive group formation from members of social organisations that aim to achieve a collective task. This involves the self-adaptive distribution of members into different task groups in, e.g. social insects (honeybees [14]), wasps [16]and ants [10]). Similarly, in technical systems inspired by social insects, swarm robots self-coordinate to dynamically allocate themselves to various regions depending on the dynamic demands imposed internally by the swarm or externally by the environment [9], [17], [23].

In all relevant cases, distributed assets are conserved (energy, money, physical resources), whereas control information may not be (because of various multi-scale abstractions).

III. Background and Previous Work

A. Multi-scale control systems

Multi-scale structures (e.g. [6], [7]) include multiple abstraction levels, where each level increases the granularity of observation of the level below (i.e. coarse graining), allowing to scale-up with the size of the observation domain. This principle also applies to multi-scale *control* systems, which are characterised by two main information flows [7]: i) *observation abstraction flow* (bottom-up), where an entity at a higher-scale (macro) collects and abstracts information about several entities at the lower scale (micro); and, ii) *control flow* (top-down) – where a higher entity (macro) provides self-adaptation directives to lower entities (micro). Directives rely on decisions taken based on abstracted information from their micro entities (bottom-up flow) and adaptation directives from their macro entities (top-down control flow). Hence, the two flows form multiple control feedback-loops between subsequent system scales (e.g. Fig.2). Such multi-scale control schemes enable system-wide coordination among self-adaptive processes while limiting the amount of resources required at each decision node (e.g. [7]). This explains their wide-spread occurrence in complex self-adaptive systems, including the large-scale asset distribution systems studied here.

B. Distribution of internal assets

A system contains various internal assets. The manner in which these assets are distributed within the system is essential to its development, adaptability and survival in the face of changing conditions. The effective investment of available assets depends on the system's internal structure (e.g. topology) and external environment (e.g. available resource distribution). To illustrate these, we consider three examples, from the natural, business, and artificial world (Fig. 1), and use an analogy between them to highlight the essential roles and behaviours of internal asset distribution.

The natural example is a plant, where water and minerals provided at the root represent internal resources, or assets. These are distributed throughout the plant and 'invested' in various branches to produce sugars (for energy provision) and hence to enable growth. The way in which available assets are allocated among various branches depends on the local state and context of each branch – e.g. local sunlight availability, meaning that the more access to sunlight a branch has the better it grows. A similar phenomenon can be observed within a business organisation. Here, internal assets consist of workers and the budget available to hire them. The budget is allocated to hire and remunerate workers for each service that the organisation offers. It is distributed among these services depending on the demand and profitability of each service. The final example is a collective of robots that are assets for taking care of given tasks in different regions of an arena. The distribution of the robots among these different regions depends on the amount of work needed in each region.

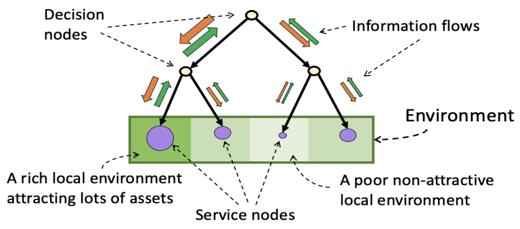
Decision nodes
Information flows
Environment
A rich local environment attracting lots of assets
Service nodes
A poor non-attractive local environment

Fig. 2: An example multi-scale system distributing internal assets in a way that is well-adapted to its environment.

C. Various types of system structures

Many complex systems, such as plants and large-scale business organisations, feature some form of tree-like structure (Fig. 1a and b). On the other hand, some systems do not have a tree-like structure. An example is a group of robots (Fig. 1c) distributed across a number of rooms to perform some collective tasks, e.g. keeping the rooms clean. Once there is a lack of tasks in a room, the robots may start moving to the neighboring rooms where there are more tasks to perform. The decision for robot redistribution can be taken by an exogenous agent located at the gates between neighboring rooms, or it can happen via direct interactions between robots that exchange information through the gates. In either case, the decision-making process can be represented as a decision node. In the example in Fig. 1c, decision nodes are the roots of the graph. The system has multiple roots, meaning that there is no higher level of coordination amongst those decision nodes.

When systems need to adapt to internal and external changes – e.g. internal growth, external resource fluctuations – their structure may change accordingly, e.g. similarly to the shaping of a plant crown. For example, an organisation may subdivide one of its sections into subsections if the number of workers in that section becomes too large to administer by a single manager; or in case further specialisation of experts is needed.

D. Multi-Scale Asset Distribution model

In a previous work [21] we have proposed a Multi-Scale control system for Asset Distribution (MSAD), loosely inspired by processes in plant growth. Here the system structure is represented by a directed acyclic graph (see Fig. 2), that can change (grow or shrink) under certain conditions. The leaf nodes, called *service nodes*, are the main points of interaction between the system and the environment, i.e. micro entities. The information flows are initiated at these nodes as a function of the assets interacting with the local environment. The non-leaf nodes, called *decision nodes*, are entities of higher scales providing self-adaptation directives for asset reallocation.

The structure can only grow at the leaf nodes by branching and if the assets locating there cross a given threshold. Likewise, the structure can shrink by removing all the leaves of a node, if their total assets is below a given threshold.

The asset distribution process relies on the competition between sibling nodes, tunable by a competition factor β. By

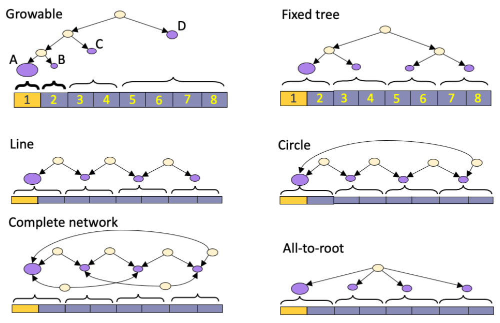

Fig. 3: Illustration of topologies used in the experiments.

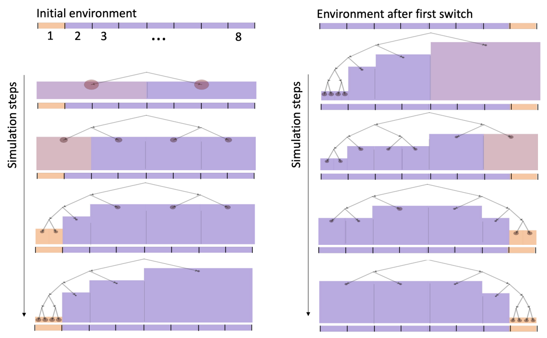

Fig. 4: A series of selected stages of growth.

increasing β to values larger than 1, the competition between sibling nodes is intensified. Additionally, the model employs a series of costs and delays, e.g. gradual release of assets for reallocation (in contrast to immediate release), described by a factor α. The MSAD model is described in details in [21].

IV. Experimental Setup

A. Environment

An environment is implemented as a sequence of regions. In Fig. 3, the bars below each graph represent a sample environment with 8 regions, numbered $1..8$. Every region has a profitability value that indicates how much profit can be produced per simulation step if one unit of asset is fully invested there. They are color-coded in Fig. 3, with yellower colors indicating higher profitability for the region.

A service node can support services demanded in a single or multiple regions. In Fig. 3, the regions that are supported by each service node are indicated using curly brackets. In the example of the 'fixed tree' topology, every service node supports two regions. On the other hand, in the example of 'growable' topology, the various service nodes support different numbers of regions. The profitability of a service node is computed as the product of its assets and the average profitability of the regions it supports. The averaging is equivalent to a worker equally dividing their time between different services.

In the examples of Fig. 3, the 'growable' topology benefits from the further branching into service nodes A and B that specialise to correspondingly support regions 1 and 2. This branching allows larger amounts of assets to be invested specifically in the more profitable region 1.

The bars on top of Fig. 4 show the environments used in all the experiments here. Initially, the leftmost region has a high profitability, 0.3, and all the other ones have a lower profitability, 0.1 (Fig. 4 left). After a given period of T simulation steps, the environment is reversed (Fig. 4 right). In all the experiments $T = 400$. The first change of the environment occurs at step $t = T$ and then it switches back to the initial state at $t = 2T$. The experiments end at $t = 3T$.

B. Topologies

As illustrated in Fig. 3, the experiments are conducted for the following topologies: a) growable tree: a binary tree that initially starts with a root and two leaves and can adapt its shape by further growing or trimming branches; b) fixed tree: a static balanced binary tree with 4 leaves, where every leaf supports 2 regions of the environment; c) line: 4 leaves where the neighboring leaves share a root; d) circle: similar to a line except that the first and the last leaves share a root as well; e) complete network: 4 leaves where every two leaves share a root; f) all-to-root: 4 leaves all sharing a single root.

C. Model parameters

All the experiments are initiated with 100 units of assets uniformly distributed among the service nodes. The growable topology is initiated with one root and two leaves. To grow new branches on a leaf node, the amount of assets in the node must exceed a threshold of 25 units. Likewise, to trim a branch, a threshold of 20 must be crossed. The value of the competition factor β is explicitly stated for each experiment.

V. Experimental Results

In the first set of experiments, a competition factor $\beta = 0.7$ is used. This particular value of β is selected based on preliminary experiments (not shown here) to represent a case where the behavioural dynamics are not drastically different for the various topologies. The experiments start with the region 1 of the environment being the most profitable region. At simulation step 400, the environment changes and region 8 becomes the most profitable region. At simulation step 800, the environment switches back to the initial setup. Fig.5 shows a) the asset investment per each region of the environment, b) the total profit produced within the system, and c) the percentage of relocated assets over time, for various topologies. The results indicate a relatively slow response of line topology to the changes in the environment. The system with growable topology produces highest profit by effectively investing most of the assets into the best region. Note that unlike growable topology that can grow deeper to specialise in service node regions, in all other topologies (static topologies) each service node is extended in a pair of neighboring regions – indicated in the figure by different shades of the same color.

Fig. 4 shows a series of snapshots from various stages of structural adaptation of growable topology. The system starts with 2 leaves, each supporting 4 regions of the environment. The structure grows toward region 1 with the highest profitability while retracting from all the other regions. After the first switch, the structure starts retracting from the previously best region and grows toward the new best region 8.

The second set of experiments investigate the effects of various values for the competition factors β on the dynamics of asset investment and profit production in systems with various topologies. Here, the release factor of assets (α) is set to 0.2 delaying the asset relocation process. Fig. 6 shows the asset investment dynamics. Table. I shows the average profit productions over the first 800 simulation steps of the same experiments. The highest and the lowest profit production levels (indicated in bold) belong to the growable topology at $\beta = 0.8$ and the line topology at $\beta = 1.1$ respectively.

The first row in Fig. 6 shows the results for $\beta = 0$, leading to no competition between sibling branches irrespective of their profitability. That results in a uniform asset distribution in all the regions of the environment. With larger β, the neighboring branches start to compete with each other based on their profitability. This leads to the attraction of assets to more profitable regions. As shown in the figure, when β gets too large, the system loses its flexibility and adaptability to the environmental change. For different topologies, the effect kicks in at different β values – e.g. growable topology fails to adapt at $\beta = 0.9$, while the fixed tree fails at $\beta = 1.1$. For all values of β, the Line topology shows slower response to changes and a relatively lower profit production.

Another interesting observation is in the fixed tree at $\beta = 0.9$ (and with less clarity at $\beta = 0.8$, also in growable topology at $\beta = 0.8$ and 0.9). In this case, after the environment switches, the asset relocation to the newly profitable regions is almost invisible until a certain period. Meanwhile, assets move from the nodes in the previously profitable regions to their siblings (e.g. from blue regions to the pink ones), which is not beneficial in terms of profit production. After this first period and as soon as enough assets are relocated to the newly profitable regions, the relocation process gets a high speed until most of the assets are in the newly profitable regions.

TABLE I: Mean profit over 800 simulation steps

β	Growable	Fixed	Line	Circle	Complete	All-to-root
0.0	12.5	12.5	12.5	12.5	12.5	12.5
0.6	20.2	15.0	14.5	14.8	14.8	14.8
0.7	24.5	16.5	15.3	16.0	16.1	16.0
0.8	**26.2**	18.3	15.9	17.9	18.1	18.0
0.9	19.6	17.6	14.9	19.0	19.4	19.4
1.0	19.6	17.6	14.9	17.5	14.9	14.9
1.1	19.6	14.9	**12.3**	13.4	14.9	14.9

VI. Discussion

The most important insights we draw from the obtained results include the following.

Growable versus static topologies: topologies that self-adapt via growth ('Growable') have higher potential to produce profits under changing conditions compared to static topologies. This is because growth allows to reallocate and re-specialise more assets in areas that become profitable. This comes at the cost of longer adaptation periods and lesser profits during

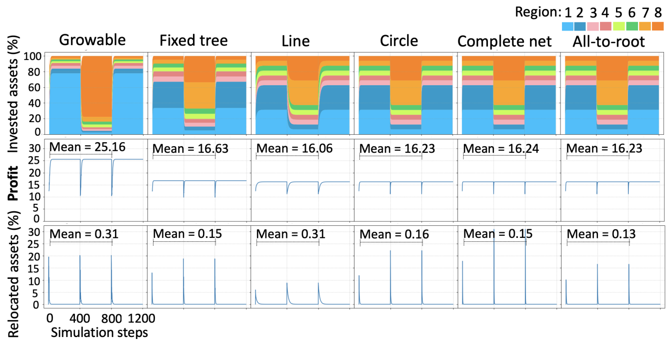

Fig. 5: competition factor = 0.7.

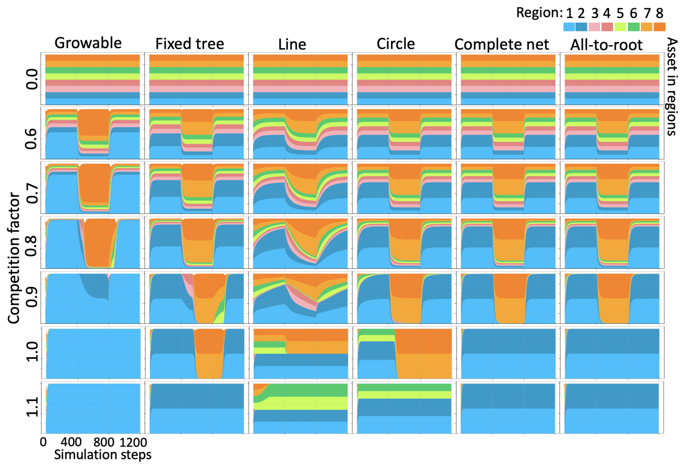

Fig. 6: Asset investment in various environmental regions for various competition factors and topologies, with $\alpha = 0.2$.

these periods. In business organisations, reallocation and re-specialisation practices may also negatively impact worker experience. In comparison, static topologies can reallocate assets but cannot re-specialise them. This limits their ability to adapt to (and profit from) new opportunities that were not predicted in their fixed structures. At the same time, they incur fewer asset re-allocations, because of lower generated profits; and shorter self-adaptation profit disturbance.

Tree versus multi-root topologies: in multi-root topologies (e.g. 'line'), global control coordination occurs via indirect propagation of control information across roots (i.e. indirect horizontal coordination). This causes less effective asset re-allocation, inducing longer self-adaptation periods and impact on profits. This is similar to the impacts of communication delays in multi-scale structures (e.g. [7]), except propagating horizontally rather than vertically.

As these delay and perturbation effects are directly related to global control coordination, they tend to disappear when self-adaptation only requires local coordination. In the presented experiments, this was the case in the 'circle' and 'complete' topologies. Here, the system parts concerned by the environmental changes were under the control of a single root, hence removing the need for coordination among several roots. For the same reasons, we note that multi-root topologies with complete meshing amongst roots behave similarly to single-scale tree topologies because all environment changes concern system parts that are observed directly by one root, so no root coordination is needed. Yet, the former topology ('complete') induces more management and communication costs.

Competition level: Some level of profit-oriented competition amongst system branches is necessary to drive self-adaptation. However, 'extreme' competition hampers self-adaptation – the actual degree depending on the competition degree and topological features. The main reason is that high competition favours high adaptation for short-term profits (e.g. over-fitting). Thus, when profitability changes, there aren't enough assets to detect the new profit opportunities, as most assets have been allocated to the previously profitable areas. This incurs increasingly long adaptation delays, with inefficient intermediary adaptation states. e.g. in Fig. 6 this can be observed for the 'fixed tree' and 'line' topologies, with competition factor $= 0.9$, via large pink areas, which indicate significant asset allocations to non-profitable areas (i.e. located in the middle of the environment bar, whereas the profitable areas are at the extremes). Extreme competition ultimately leads to the inability to adapt, as the amount of assets allocated to the previously profitable area (that has become less profitable) is large enough to generate higher profits than the fewer assets allocated to the newly profitable area. This causes high inertia and blocks reactivity.

Single versus multi-scale of control: Importantly, we only considered communication delays of one step across multi-scale topologies (just as for single-scale topologies). In most real systems this will not be the case as more scales will incur heavier control delays and potentially significant impacts on the self-adaptation outcomes (as we showed in previous work [12]). Similarly, considering management costs at the decision nodes will weight heavier on multi-scale structures with more decision nodes (not shown in the results).

VII. Conclusions

Within the asset-distribution domain, this paper studied the relation between multi-scale systems topology and the ensuing self-adaptive behaviour. We used a novel asset-distribution algorithm and studied topologies with single or multiple roots; with growing or static capacities; and with multiple or single control scales. We also considered internal competition as an important concern across all topologies. Results led to the following key insights: 1) topologies that allow asset re-specialisation (e.g. by 'growing') are more adaptable to unforeseen opportunities and can provide higher profits than static topologies; 2) multi-root topologies incur higher delays than trees, due to extra inter-root communication delays (i.e. horizontal, same-scale communication) – this is similar to delays in multi-scale topologies (i.e. vertical inter-scale delays); 3) while some level of competition between system branches is needed for self-adaptation, too much competition actually hampers adaptation, by causing over-fitting and too much inertia. In all cases, more adaptation is achieved via more asset relocation and/or re-specialisation and hence incurs higher delays and disturbance. These insights provide a basis for further studies, aiming to offer reusable guidance for selecting suitable system topologies in various dynamic environments.

References

[1] Catherine S.E. Bale, Liz Varga, and Timothy J. Foxon. Energy and complexity: New ways forward. *Applied Energy*, 138:150–159, 2015.

[2] Adrian Bejan and J. Peder Zane. *Design in Nature: How the Constructal Law Governs Evolution in Biology, Physics, Technology, and Social Organizations*. Doubleday, 2012.

[3] Raffaele Carli and Mariagrazia Dotoli. A decentralized resource allocation approach for sharing renewable energy among interconnected smart homes. In *IEEE Conf. Dec. and Control (CDC)*, pages 5903–5908, 2015.

[4] Bassem Debbabi, Ada Diaconescu, and Philippe Lalanda. Controlling self-organising software applications with archetypes. In *IEEE Int.Cnf. SASO, 2012*, pages 69–78.

[5] Andrea Di Stefano and Corrado Santoro. A decentralized strategy for resource allocation. volume 27, pages 295– 300, 07 2005.

[6] Ada Diaconescu, Louisa Jane Di Felice, and Patricia Mellodge. Multi-scale feedbacks for large-scale coordination in self-systems. In *IEEE Intl.Cnf. on Self-Adaptive and Self-Organizing Systems, SASO, Umea, Sweden, June 16-20, 2019*, pages 137–142, 2019.

[7] Ada Diaconescu, Louisa Jane Di Felice, and Patricia Mellodge. Exogenous coordination in multi-scale systems: How information flows and timing affect system properties. *FGCS*, 114:403–426, 2021.

[8] Nikolaus Huber, Fabian Brosig, and Samuel Kounev. Model-based self-adaptive resource allocation in virtualized environments. In *Intl. Symp. on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, page 90–99, 2011.

[9] Chris Jones and Maja J. Mataric. Adaptive division of labor in large-scale minimalist multi-robot systems. In *IEEE/RSJ Intl.Cnf. on Intelligent Robots and Systems (IROS)*, pages 1969–1974, LA, 2003.

[10] G. E. Julian and Sara Cahan. Undertaking specialization in the desert leaf-cutter ant acromyrmex versicolor. *Anim Behav*, 58(2), 1999.

[11] Roberto Medina, Etienne Borde, and Laurent Pautet. Scheduling multi-periodic mixed-criticality dags on multi-core architectures. In *2018 IEEE Real-Time Systems Symposium (RTSS)*, pages 254–264, 2018.

[12] Patricia Mellodge, Ada Diaconescu, and Louisa Jane Di Felice. Timing configurations affect the macro-properties of multi-scale feedback systems. In *IEEE Intl. Cnf. ACSOS*, 2021.

[13] Arles Rodriguez, Jonatan Gomez, and Ada Diaconescu. Self-healing networks via self-organising mobile agents. *Journal of Autonomous Agents and Multi-agent Systems (JAAMAS)*, 01 2021.

[14] Thomas D. Seeley. Adaptive significance of the age polyethism schedule in honeybee colonies. *Behavioral Ecology and Sociobiology*, 11:287–293, 1982.

[15] Christophe Taton and et al. Self-sizing of clustered databases. In *Intl. Symp. WoWMoM, Buffalo, New York, USA*, pages 506–512. IEEE, 2006.

[16] VO. Torres, TS. Montagna, J. Raizer, and WF. Antonialli-Junior. Division of labor in colonies of the eusocial wasp, mischocyttarus consimilis. *Journal of insect science*, 12:21, 2012.

[17] Yongming Yang, Changjiu Zhou, and Yantao Tian. Swarm robots task allocation based on response threshold model. In *ICARA*, pages 171–176, Wellington, 2009. IEEE.

[18] Payam Zahadat. Self-adaptation and self-healing behaviors via a dynamic distribution process. In *2019 IEEE 13th International Conference on Self-Adaptive and Self-Organizing Systems (SASO)*. IEEE, 2019.

[19] Payam Zahadat, Karl Crailsheim, and Thomas Schmickl. Social inhibition manages division of labour in artificial swarm systems. In *EU Cnf. on Artificial Life (ECAL)*, pages 609–616. MIT Press, 2013.

[20] Payam Zahadat and Ada Diaconescu. Reactive or stable: A plant-inspired approach for organisation morphogenesis. In *The 2020 Conference on Artificial Life*, pages 614–622, 07 2020.

[21] Payam Zahadat and Ada Diaconescu. Multi-scale asset distribution model for dynamic environments. *arXiv preprint*, 2022.

[22] Payam Zahadat, Daniel Nicolas Hofstadler, and Thomas Schmickl. Vascular morphogenesis controller: A generative model for developing morphology of artificial structures. In *Genetic and Evolutionary Computation Conference*, GECCO, pages 163–170. ACM, 2017.

[23] Payam Zahadat and Thomas Schmickl. Division of labor in a swarm of autonomous underwater robots by improved partitioning social inhibition. *Adaptive Behavior*, 24(2):87–101, 2016.