UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

JEFERSON JOSÉ BAQUETA

A TASK DELEGATION MODEL IN MULTI-AGENT SYSTEMS: MANAGING SUB-DELEGATIONS AND DELEGATION CHAINS UNDER AGENT BEHAVIORAL CHANGES

CURITIBA

2025

versão 11.0 (abr. 25)

JEFERSON JOSÉ BAQUETA

A TASK DELEGATION MODEL IN MULTI-AGENT SYSTEMS: MANAGING SUB-DELEGATIONS AND DELEGATION CHAINS UNDER AGENT BEHAVIORAL CHANGES

Um Modelo de Delegação de Tarefas em Sistemas Multiagente: Gerenciamento de Subdelegações e Cadeias de Delegação diante de Mudanças Comportamentais dos Agentes

Trabalho de pesquisa de doutorado apresentado como requisito para obtenção do título de Doutor em Ciências do Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial da Universidade Tecnológica Federal do Paraná.
Orientador(a): Prof. Dr. Cesar Augusto Tacla

Coorientador(a): Profª. Drª. Miriam Mariela Mercedes Morveli Espinoza

CURITIBA

2025

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Campus Curitiba

JEFERSON JOSE BAQUETA

A TASK DELEGATION MODEL IN MULTI-AGENT SYSTEMS: MANAGING SUB-DELEGATIONS AND DELEGATION CHAINS UNDER AGENT BEHAVIORAL CHANGES.

Trabalho de pesquisa de doutorado apresentado como requisito para obtenção do título de Doutor Em Ciências da Universidade Tecnológica Federal do Paraná (UTFPR). Área de concentração: Engenharia De Computação.

Data de aprovação: 04 de Julho de 2025

Dr. Ayslan Trevizan Possebom, Doutorado - Instituto Federal de Educação, Ciência e Tecnologia do Paraná (Ifpr)

Dr. Fabricio Enembreck, Doutorado - Pontifícia Universidade Católica do Paraná (Pucpr)

Dr. Gustavo Alberto Gimenez Lugo, Doutorado - Universidade Tecnológica Federal do Paraná

Dr. Marco Aurelio Wehrmeister, Doutorado - Universidade Tecnológica Federal do Paraná

Dra. Miriam Mariela Mercedes Morveli Espinoza, Doutorado - Umeå University (Umu)

Documento gerado pelo Sistema Acadêmico da UTFPR a partir dos dados da Ata de Defesa em 31/07/2025.

Dedico este trabalho a minha família e aos
meus amigos, pelos momentos de ausência.

AGRADECIMENTOS

Gostaria de expressar meus sinceros agradecimentos a todos que estiveram ao meu lado e contribuíram para esta jornada de crescimento e conquistas. Gostaria de agradecer à minha mãe, Roseni Ferreira Baqueta, e ao meu pai, Anito Baqueta, pelo apoio que recebi durante os anos de doutorado.

Gostaria de agradecer a minha querida esposa, Aline Silva de Lima, por estar ao meu lado durante toda essa jornada. Seu apoio constante, companheirismo e compreensão foram essenciais nos momentos difíceis. Agradeço por sempre estar disposta a me ouvir nos momentos de desabafos e por me incentivar a seguir em frente, mesmo diante das adversidades.

Ao meu estimado orientador, Professor Cesar Augusto Tacla, gostaria de expressar minha mais profunda gratidão. Sua paciência, orientação e comprometimento foram fundamentais para o meu crescimento acadêmico e profissional. Agradeço pelo tempo dedicado para nossas reuniões e pelos conselhos que recebi ao longo deste percurso.

Também gostaria de agradecer à pesquisadora Miriam Mariela Mercedes Morveli Espinoza. Sua ajuda e suas valiosas sugestões desempenharam um papel significativo na elaboração deste trabalho e em outros projetos que desenvolvi ao longo do período do doutorado.

Por fim, não posso deixar de agradecer a todos aqueles que, direta ou indiretamente, contribuíram para esta conquista. Suas contribuições foram verdadeiramente valorizadas e me fortaleceram ao longo do caminho.

O presente trabalho foi realizado com apoio do Conselho Nacional de Desenvolvimento Científico e Tecnológico - Brasil (CNPq) - Projeto nº 409523/2021-6.

“A verdadeira delegação não é apenas
distribuir tarefas, mas transferir confiança.”
(Stephen Covey)

RESUMO

Neste trabalho, propomos um modelo de delegação de tarefas capaz de suportar subdelegações por meio de decomposição de tarefas e delegação recursiva. Nosso modelo considera componentes sociais e cognitivos derivados das teorias de confiança e reputação (*e.g.*, taxa de sucesso, preferências dos agentes, imagem social, reputação e know-how), além da forma como os agentes se conectam e estabelecem relacionamentos sociais, com o objetivo de refinar o processo de seleção de parceiros. O modelo proposto foi projetado para lidar com cenários de delegação de tarefas em que os agentes podem alcançar seus objetivos por meio de ações de delegação ou execução. Especificamente, o modelo de delegação é empregado no processo de seleção de parceiros, onde um agente delegador deve decidir qual parceiro selecionar como delegado. Nesse processo, além das avaliações pessoais e compartilhadas sobre um parceiro, o delegador pode extrair informações diretamente das cadeias de delegação. Isso inclui dependências diretas e transitivas com base nas conexões dos agentes, a probabilidade de sucesso acumulada à medida que as tarefas são subdelegadas e os graus de penalização atribuídos aos parceiros em casos de propagação de falhas. Em particular, uma cadeia de delegação é uma estrutura comum em aplicações onde os agentes trabalham como uma equipe e dependem uns dos outros para alcançar seus objetivos. No entanto, poucos modelos de delegação na literatura consideram as subdelegações como um processo abrangente e integrado. A maioria das abordagens existentes aborda a delegação de tarefas a partir de uma perspectiva monoepisódica, desconsiderando a formação e o impacto das cadeias de delegação e tratando as subdelegações como meras repetições de instâncias isoladas de delegação. Em nossos experimentos, avaliamos a eficácia do modelo de delegação proposto em um ambiente dinâmico, onde os agentes podem mudar seu comportamento social ao longo do tempo. Nesse contexto, comportamento social refere-se à capacidade de um agente de cumprir com precisão suas estimativas de desempenho ao atuar como parceiro, bem como à probabilidade de falha associada ao agente durante a execução da tarefa. Essas características podem ser aprendidas pelos delegadores à medida que interagem com os demais agentes do sistema. Como principal contribuição deste trabalho, demonstramos que os agentes se beneficiam de um modelo de delegação que lida explicitamente com subdelegações e cadeias de delegação. Nossa abordagem leva a melhorias significativas no cumprimento das tarefas quando comparada a abordagens monoepisódicas. Especificamente, observamos aumentos nas taxas de sucesso e níveis mais altos de satisfação entre os agentes em relação às suas tarefas, além de uma redução no tempo necessário para que os delegadores aprendam os novos comportamentos sociais adotados por seus parceiros em um ambiente dinâmico. Argumentamos que, ao gerenciarem explicitamente as cadeias de delegação, os agentes obtêm melhor desempenho e se adaptam mais rapidamente aos comportamentos dos demais. Análises estatísticas confirmaram que o modelo proposto superou significativamente as abordagens de referência em taxa de sucesso, satisfação e eficiência de aprendizagem, em diferentes topologias e estratégias de delegação.

Palavras-chave: delegação de tarefas; modelos de delegação; modelos de confiança e reputação; cadeia de dependência; relações de dependência social.

ABSTRACT

In this work, we propose a task delegation model capable of supporting sub-delegations through task decomposition and recursive delegation. Our model considers social and cognitive components derived from trust and reputation theories (*e.g.*, success rate, agents' preferences, social image, reputation, and know-how), as well as how agents connect and establish social relationships, with the aim of refining the partner selection process. The proposed model was designed to deal with task delegation scenarios in which agents can achieve their goals through delegation or execution actions. Specifically, the delegation model is employed in the partner selection process, where a delegator agent must decide which partner to select as the delegatee. In this process, in addition to personal and shared evaluations of a partner, the delegator can extract information directly from delegation chains. This includes direct and transitive dependencies based on agents' connections, the accumulated success probability as tasks are sub-delegated, and the penalty degrees assigned to parnters in cases of failure propagation. In particular, a delegation chain is a common structure in applications where agents work as a team and depend on one another to achieve their goals. However, few delegation models in the literature consider sub-delegations as a comprehensive and integrated process. Most existing approaches address task delegation from a mono-episodic perspective, disregarding the formation and impact of delegation chains, and treating sub-delegations as mere repetitions of isolated delegation instances. In our experiments, we evaluate the effectiveness of the proposed delegation model in a dynamic environment where agents can change their social behavior over time. In this context, social behavior refers to an agent's capability to accurately fulfill its performance estimations when acting as a partner, as well as the likelihood of failure associated with the agent during task execution. These characteristics can be learned by the delegators as they interact with other agents in the system. As the main contribution of this work, we demonstrate that agents benefit from a delegation model that explicitly handles sub-delegations and delegation chains. Our approach leads to significant improvements in task accomplishment when compared to mono-episodic approaches. Specifically, we observe increased success rates and higher satisfaction levels among agents regarding their tasks, as well as a reduction in the time required for delegators to learn the new social behaviors adopted by their partners in a dynamic environment. Therefore, we argue that when agents can explicitly manage delegation chains, they achieve better performance and adapt more quickly to the evolving behaviors of other agents over time. Statistical analyses confirmed that the proposed model significantly outperformed baseline approaches in terms of success rate, satisfaction, and learning efficiency under different network topologies and delegation strategies.

Keywords: task delegation; delegation models; trust and reputation models; dependence chain; social dependence relations.

LIST OF ALGORITHMS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

MAS	Multi-Agent System
MAB	Multi-Armed Bandit
DL	Delegation Likelihood
DG	Dependence Graph
XML	Extensible Markup Language
API	Application Programming Interface
ANOVA	Analysis of Variance

Acronyms

DS-graph	Delegation Scenario Graph
DS-net	Delegation Scenario Network
QuAD-V	Quantitative Argumentation Debate with Votes
CNP	Contract-Net Protocol

LIST OF SYMBOLS

ι	It represents an impression stored in an agent's belief base.
σ	It is a default startup value used to define the initial success rate for a partner.
Λ	It represents a delegation chain capable of connecting roots to delegatees.
ρ	It controls the slope of the squashing function used to aggregate an agent's impressions.
ξ	It controls the relevance of an impression generated at time t, relative to the current time.
τ	A task to be delegated.
α	Delegator agent.
β	Partner or potential delegatee agent.
g	Goal pursued by an agent.
$d(\tau)$	Delegation action for task τ.
$e(\tau)$	Execution action for task τ.
$\prec$	Dependence relation between two agents in a task.
V	Vector of performance estimations for task criteria.
c_i, v_i	Task criterion and the estimated value provided by an agent.
$SR(\tau)$	Success Rate of a task τ based on direct dependencies.
$ASR(\Lambda)$	Accumulated Success Rate of a delegation chain.
UT	Utility of an offer.
TM	Trust Measure.
CM	Competence Measure.
SL	Success Likelihood.
SI	Social Image of an agent.
RP	Reputation of an agent.
KH	Know-how of an agent.
SAT	Satisfaction level of a delegator.
$MSAT$	Maximum satisfaction level reached by a delegator.
REG	Regret experienced by a delegator.
$NREG$	Normalized regret relative to maximum satisfaction.
itr	Iteration of the simulation.
W	Weight assigned to an evaluation dimension.

SUMMARY

1 INTRODUCTION

In a Multi-agent System (MAS), task delegation is a fundamental mechanism adopted by agents to solve problems that involve teamwork. Such a mechanism contributes to agents accomplishing private or collective objectives beyond their capabilities. A critical stage of task delegation is partner selection, where the agents must choose partners capable of performing the tasks delegated to them. Generally, this process is oriented by the trust degree associated with each partner. In particular, the trust establishment relies on observations about the partners' social behavior and the environmental conditions.

The simplest type of delegation is called *mono-episodic* (Castelfranchi; Falcone, 2010). In this type of delegation, an agent, referred to as a *delegator*, must delegate a task τ to another agent, referred to as *delegatee*, because it does not have the conditions to complete τ alone. The mono-episodic approach assumes that a delegatee can execute τ directly. However, in the context of sub-delegations, if a delegatee is not able to complete a task τ by itself, it may delegate τ onward until another agent performs it (*recursive delegation*) (Afanador; Baptista; Oren, 2019) or decompose τ into sub-tasks (τ_1, τ_2, ..., τ_n) and then delegate them (*task decomposition*) (Karimadini; Lin, 2010). It is important to remark that sub-delegations imply the formation of *delegation chains* as the agents sub-delegate tasks to each other.

Delegation chains admit the representation of complex social structures that express the dependence relations among the agents, such as direct, transitive, AND and OR social dependencies (Costa; Dimuro, 2006). These structures define the connections among the agents and are essential for task delegation scenarios where the agents must work together to accomplish goals they could not achieve alone (Dong; Ota; Dong, 2021) (Wang *et al.*, 2022) (Pettitt; Elliott; Swiecicki, 2017) (Yliniemi; Agogino; Tumer, 2014) (Barker; Whitcomb, 2016). A delegation chain can be seen as a sequence of delegations performed in a given order by a set of agents that aim to achieve their goals (An; Miao; Cheng, 2005) (An *et al.*, 2007). Along a delegation chain, an agent can act as a delegator by assigning tasks to others or as a delegatee by receiving a task from others. When a sub-delegation occurs, the delegatee becomes the delegator of a new delegation instance, and a new agent is selected as the delegatee. Besides, the agent who makes the first delegation request is called the chain's *root* (Afanador; Baptista; Oren, 2019). Distinct delegation chains can be integrated into a single network structure since an agent can make part of different delegation chains at the same time. In such a network structure, each chain represents a sequence of delegations that might lead a root to accomplish its goal according to an accumulated success rate (ASR). This measure indicates the probability of all agents along a delegation chain successfully completing their respective tasks.

In this work, we propose a task delegation model that explicitly supports sub-delegations and the formation of delegation chains through recursive delegation and task decomposition. We assume a task can be decomposed using conjunction (AND) or disjunction (OR) operations. A conjunction operation is applied when task decomposition requires the simultaneous selection of multiple partners (delegatees). Each partner's joint success implies the decomposed task's

success in this case. On the other hand, a disjunction operation is applied when a task can be broken down into alternative forms involving different partners. In this way, the success of any given partner is enough to ensure the success of the decomposed task. Additionally, in our delegation model, the partner selection process is modeled using a reinforcement learning approach known as Multi-armed Bandits (MAB) (Turgay; Oner; Tekin, 2018). In such an approach, the partners are selected based on a delegation likelihood. This likelihood is calculated through a multi-goal analysis that takes as input several social and cognitive components, which are better discussed later in this document, particularly in chapters 2 and 3. In a general way, these components involve agents' direct and transitive social dependencies, preferences, historical information about agents' successes likelihood, behavioral impressions, social image, reputation, and know-how (Castelfranchi; Miceli; Cesta, 1992) (Castelfranchi; Falcone, 2010) (Conte; Paolucci, 2002) (Buccafurri *et al.*, 2015).

To delimit the scope of this work, we assume a simplified and controlled multi-agent environment focused on the task delegation process through delegation chains. Tasks are announced in parallel and do not carry priority levels or differentiated rewards. Agents are capable of executing multiple tasks concurrently, and we assume that resources are unlimited and non-consumable. All agents possess the same set of skills, have unlimited availability, and adopt neutral behavior (*i.e.*, they are neither benevolent nor selfish) and do not act maliciously. Additionally, there are no strategic attacks on the delegation process, as agents are assumed to be honest. These assumptions allow us to isolate and analyze the impact of delegation chains and network topology on partner selection and overall system efficiency, without interference from external strategic factors.

1.1 Motivation

As pointed out by (Burnett; Oren, 2012) and (Afanador; Baptista; Oren, 2019), few delegation models in the literature consider sub-delegations as an independent process. In general, the delegation problem is addressed from a mono-episodic point of view, ignoring the formation of delegation chains or considering sub-delegations as a mere repetition of mono-episodic instances (Sabater; Sierra, 2001) (Huynh; Jennings; Shadbolt, 2004) (Griffiths, 2005) (Sabater; Paolucci; Conte, 2006) (Castelfranchi; Falcone, 2010) (Cho; Chan; Adali, 2015) (Castelfranchi; Falcone, 2020).

Consequently, key aspects regarding sub-delegations tend to be overlooked, such as:

- Task transitivity in scenarios where sub-delegations are required (*i.e.*, the effects that sub-tasks and recursive delegations, generated as ramifications of an initial delegation action d_1, have on the outcome obtained after executing the task associated with d_1 (Afanador; Baptista; Oren, 2019; Afanador; Oren; Baptista, 2019);

- Penalization strategies for propagating an agent's failure along a delegation chain (Burnett; Oren, 2012);

- Dependence relations and their effect on the establishment and maintenance of trust (Costa; Dimuro, 2006; Sichman; Conte, 2002; Kox *et al.*, 2021).

Even in works that consider sub-delegations as an independent process (Burnett; Oren, 2012) (Afanador; Baptista; Oren, 2019) (Afanador; Oren; Baptista, 2019), the sub-delegations are addressed only from the point of view of recursive delegations. In this kind of approach, agents are not able to decompose their tasks based on their dependence relations. On the other hand, in some approaches, like (Burnett; Oren, 2012), (Afanador; Baptista; Oren, 2019), and (Afanador; Oren; Baptista, 2019), the agents decide which partner to choose, taking into account just a single evaluation dimension. For example, in (Burnett; Oren, 2012), partner selection is performed through a reputational mechanism, in which the agents' reputation is estimated based on their interactions with others. Conversely, in (Afanador; Baptista; Oren, 2019) and (Afanador; Oren; Baptista, 2019), partners are chosen based on their success history, considering the tasks they have executed in the past.

As pointed out by (Castelfranchi; Falcone, 2010) and (Castelfranchi; Falcone, 2020), the trust needed by an agent to decide to delegate a task involves several other social and cognitive components besides reputational measures or success rate. These aspects can be divided into internal and external factors. In this work, we focus on internal factors, which are directly included in our delegation model. For example, direct experiences are represented through the agent's social image (Sabater; Paolucci; Conte, 2006), and indirect experiences are related to reputation and know-how (Conte; Paolucci, 2002), (Buccafurri *et al.*, 2015). External factors, like unexpected events or changes in the environment, are not directly included in the model. Instead, we represent them indirectly in our experiments by changing the behavior of the agents, which affects how trust is updated and how delegation decisions are made.

In addition, when a delegator selects a partner to delegate a task, they also create expectations about how that task will be performed (Castelfranchi; Falcone, 2020). These expectations usually include agreements established between the delegator and the chosen partner regarding the time, cost, or quality of task execution. If the partner fails to meet what was promised, for example, by delivering the task late or at a higher cost, the delegator's expectations are not fully met, even if the task is completed. In delegation chains, this problem can be even more complex, as each agent may sub-delegate its tasks, relying on others to fulfill their performance estimations (promises). Therefore, delegation models should not only focus on the probability of task success but also consider the partner's ability to meet the delegator's expectations regarding task execution.

In order to highlight the contribution of the delegation model proposed in this work, Table 1 provides a comparative overview of related delegation models based on the type and nature of their evaluation mechanisms and whether they support sub-delegation structures through delegation chains. This classification considers whether the model adopts a unidimensional or multidimensional evaluation strategy, the nature of the dimensions considered (social, cognitive, or performance-based), the specific evaluation criteria used (*e.g.*, success rate, repu-

tation, cost, and quality), and whether the model supports mono-episodic delegation or more complex structures involving sub-delegations and delegation chains.

As shown in Table 1, models such as those proposed by (Sabater; Sierra, 2001), (Huynh; Jennings; Shadbolt, 2004), and (Pinyol *et al.*, 2012) employ multidimensional evaluations grounded in social constructs like social image and reputation but treat delegation from a mono-episodic perspective. In contrast, (Griffiths, 2005) extend evaluation criteria to task-specific dimensions such as cost, timeliness, and quality, focusing on the performance characteristics of agents during task execution. On the other hand, (Castelfranchi; Falcone, 2010), incorporate cognitive elements such as dependencies, risk, and willingness into trust evaluation, yet also retain a mono-episodic perspective. More recent approaches, such as those by (Burnett; Oren, 2012) and (Afanador, 2019), support recursive delegation via sub-delegations and delegation chains. However, their evaluation models remain unidimensional, relying solely on reputation or agents' success histories.

1.2 Problem Definition

Considering sub-delegations in partner selection requires refining how trust is computed, relying solely on a partner's individual performance (*e.g.*, success or failure rate) may be inadequate, as delegation chains provide additional insights into its behavior and performance. For instance, when delegation chains are considered during partner selection, a partner's performance estimations do not need to be based solely on its individual capabilities. Instead, these estimations may take into account the performance capabilities of other agents at lower levels of the delegation chain. Thus, the promises made by a partner regarding task execution can be built considering the performance estimations of the agents it relies on to execute sub-tasks. This feature enables the partner to evaluate the entire chain's impact on its performance estimation regarding task outcomes, resulting in more accurate overall estimations, as it reflects the influence of all agents involved in fulfilling the initial promise. In our approach, the performance estimations made by a partner consider the capabilities of agents at lower levels in the delegation chain. However, the accuracy in fulfilling these estimations depends on the partner's accuracy level (*i.e.*, agents with higher accuracy levels tend to better meet the expectations of their delegators). In our experiments, we discuss this process in detail as part of the characterization of the agents.

Furthermore, the explicit representation of a delegation chain allows different degrees of penalization to be applied to agents along a chain based on their involvement in a failure event. In particular, an agent can be directly or indirectly responsible for a failure. A direct failure happens when the agent fails to perform a task due to individual reasons, like a lack of a resource or competence. In contrast, an indirect failure is caused by the propagation of someone's failure, which affects the agent due to its social dependence relations. For instance, if an agent in the middle of a delegation chain fails to complete a task τ, this failure must be propagated to the other agents at the higher levels of the chain (*i.e.*, agents that depend directly or indirectly on τ).

Table 1 – Comparison of delegation models based on evaluation dimensions and delegation type

Work	Evaluation Type	Dimension	Evaluation Criteria	Delegation Structure
(Sabater; Sierra, 2001)	Multi-dimensional	Social	Social image and reputation	Mono-episodic
(Huynh; Jennings; Shadbolt, 2004)	Multi-dimensional	Social	Reputation and know-how	Mono-episodic
(Griffiths, 2005)	Multi-dimensional	Performance-based	Success, cost, timeliness, quality	Mono-episodic
(Sabater; Paolucci; Conte, 2006)	Multi-dimensional	Social	Success, cost, timeliness, quality	Mono-episodic
(Castelfranchi; Falcone, 2010)	Multi-dimensional	Social + Cognitive	Willingness, social image, reputation, know-how, dependencies, risks	Mono-episodic
(Burnett; Oren, 2012)	Unidimensional	Social	Reputation	Sub-delegation and delegation chains
(Pinyol *et al.*, 2012)	Multi-dimensional	Social	Social image and reputation	Mono-episodic
(Afanador; Baptista; Oren, 2019)	Unidimensional	Performance-based	Success	Sub-delegation and delegation chains
Proposed Model (this work)	Multi-dimensional	Social + Performance-based + Cognitive	Success, social image, reputation, know-how, dependencies	Sub-delegation and delegation chains

Therefore, the degree of penalization for each agent may differ according to the transitive effects of the failure.

It is important to remark that the topology of the agent network plays a crucial role in the formation of delegation chains and how dependence relations are established. Different network structures (*e.g.*, hierarchical trees or probabilistic random graphs) impose distinct constraints on agent connectivity, affecting the agents' performance and how failures propagate. For instance, in a tree topology, the connections between agents are built on rigid parent-child relationships, which simplify the way agents define their dependencies, but at the same time limit the number of tasks that a partner can receive from distinct agents, considering that tasks are delegated from a parent to a child. In contrast, a random topology enables greater interconnectivity between network levels, allowing agents to establish multiple dependencies and communication paths, but this topology tends to increase the number of possible delegation chains that can be formed when an agent decides to delegate a task. These structural characteristics directly impact how delegation chains operate, how failures affect agents at higher levels of a failure chain, and how agents evaluate the reliability of their sub-delegations.

Another advantage of considering the delegation chain in the partner selection process is the possibility of delegating tasks to groups of partners through task decomposition. For example, in a group formed through a dependence relation of AND-type, the failure of a single partner results in the failure of the entire group. Thus, completing the task delegated to the group of agents requires all members to complete their sub-tasks. Nevertheless, a delegator can assess the performance of each group member separately, differentiating the individual performance of each delegatee from the overall group performance. This feature allows a delegator to have different views about its partners.

Therefore, in a delegation chain, the trust that a delegator has in a partner can not be computed merely by counting the number of successes or failures of the partner since the agents may fail due to the actions of others (*i.e.*, failures can be caused due to transitive social dependence relations or by other members of a group). Taking this type of refinement into account allows the delegators to be more sensitive to the delegatees participation in group fails during their deliberative process. Such a feature is crucial for optimizing agents' performance in situations where a partner's success depends not only on its individual capabilities but also relies on the behavior of other agents. This is particularly pertinent in dynamic scenarios where agents can change their behavior over time, such as those defined as follows:

- *Traffic Management*: Vehicles can change their behavior over time based on real-time traffic conditions, such as congestion levels, accidents, or road closures. Besides, vehicles may adjust their routes, speed, or driving behavior to optimize the traffic flow or minimize travel time (Nellore; Hancke, 2016) (Mushtaq *et al.*, 2023) (Serugunda *et al.*, 2021).

- *Financial Markets*: Aiming to maximize profits or minimize losses, agents in financial markets, like traders or trading algorithms, can adapt their strategies and behaviors

over time in response to market dynamics, news events, or the joining and leaving of investors investor (Shavandi; Khedmati, 2022) (Tao *et al.*, 2021) (Cohen, 2022).

- *Smart Grids*: Energy-producing and consuming agents (e.g., power plants, renewable energy sources, households) can dynamically adjust their energy production or consumption patterns based on factors like electricity prices, demand forecasts, or availability of renewable energy. Such an adaptive behavior helps balance supply and demand in the system, optimizing energy usage and improving grid reliability (Hossain *et al.*, 2016) (Camacho *et al.*, 2011) (Faheem *et al.*, 2018) (Dileep, 2020).

- *Robotics and Autonomous Systems*: Agents can learn and adapt their behaviors over time through learning mechanisms, refining their navigation strategies, object manipulation techniques, or task execution procedures based on feedback from sensors, past experiences, or environmental interactions (Yliniemi; Agogino; Tumer, 2014) (Barker; Whitcomb, 2016) (Wong *et al.*, 2017) (Dong; Ota; Dong, 2021) (Wang *et al.*, 2022).

- *Supply Chain Management*: Agents involved in supply chain management, such as manufacturers, distributors, and vendors, can adjust their production, stocking, or distribution strategies over time in response to changing market demand, supply chain disruptions, or variations in production costs (Manavalan; Jayakrishna, 2019) (Abdel-basset; Manogaran; Mohamed, 2018) (Margolis *et al.*, 2018) (Cui; Idota; Ota, 2019).

It is important to remark that task delegation is a central theme in multi-agent systems and intersects with broader research areas such as cooperative game theory (Branzei; Dimitrov; Tijs, 2008) and coalition function games (Aumann; Dreze, 1974), which explore how agents can collaborate, form coalitions, and share payoffs or responsibilities based on individual contributions and capabilities. While these approaches often focus on formal models for coalition formation and utility distribution, this work adopts similar collaborative principles to model how agents select partners and form delegation chains in dynamic environments. In contrast to traditional cooperative games, where coalitions are typically formed with global knowledge or static assumptions, our model addresses local, dynamic, and decentralized decision-making, where agents must evaluate others based on their social behavior, which may change over time.

In particular, in this work, social behavior is defined as an agent's ability to fulfill its performance estimation and its capability to complete tasks, which depend on the agent's accuracy level and the likelihood of failure associated with task execution, respectively. These characteristics may vary over time, as discussed later. Such changes affect how a delegator perceives its partners and may force it to relearn which partners are trustworthy as behavioral patterns evolve.

In this context, the research question adopted as a guide for this work is:

> *How do sub-delegations affect the task delegation process, and how can delegation models based on delegation chains improve partner selection and overall system efficiency in dynamic scenarios where agents can change their social behavior over time?*

To answer this question, we designed a delegation model that explicitly takes into account sub-delegations and the formation of delegation chains during the partner selection stage under two different types of network topologies (*i.e.*, tree and random networks). Specifically, our model aims to identify the most suitable partners for a given task by evaluating each agent's capabilities and competencies, along with the impact of social dependence relations and the partner's ability to meet the delegator's expectations regarding task execution.

1.3 Motivating Example

The dependence network shown in Figure 1 demonstrates the importance of considering the transitive dependence relationships during partner selection. Such a network represents a complex delegation chain, which is compounded of three distinct sub-chains. In turn, each sub-chain represents a sequence of delegations that can enable the root to complete the task τ_1. In this case, the delegation of a task τ is represented as **d**(τ), whereas the execution of a task τ is defined as **e**(τ). Moreover, in this example, for simplicity, we omit the social and cognitive elements mentioned previously and assume that a delegator's decision about which partner to select is based only on the success rate (SR) and accumulated success rate (ASR). The success rate is associated with the direct social dependencies established between a delegator and its partners, representing the probability of a partner completing a task based on its success history. On the other hand, the accumulated success rate expresses the success probability of a sub-chain as a whole, considering the transitive social dependencies. Such a measure is calculated as the product of the individual success rates of each agent along a sub-chain. For instance, ag_1 records that the $SR(\tau_1)$ assigned to ag_3 is 0.6, and the ASR for the chain from τ_1, followed by τ_3, is 0.54, which is calculated as the product of $SR(\tau_1)$ and $SR(\tau_3)$.

Note that, in a mono-episodic partner selection approach, the partners would be selected based on their success rates since this measure is obtained from the direct social dependencies between each delegator and its partners. In this case, the partners chosen as delegates would be those with the highest success rate. In contrast, when the transitive dependencies are considered, the success rate of all agents in a chain can be taken into account during the partner selection. In this case, the partners would be selected based on the path with the highest probability of success rather than solely on the success rate of a single partner. Despite the simplicity of this example, it is possible to observe that the local choice by the partner with the highest success rate (ag_4 with $SR(\tau_1)$ = 0.8) does not guarantee the best arrangement of partners (*i.e.*, the path with the highest success probability - $ASR(\tau_1, \tau_3)$ for ag_3 and ag_6), which can only be

achieved when the transitive social dependencies are considered during the partner selection process.

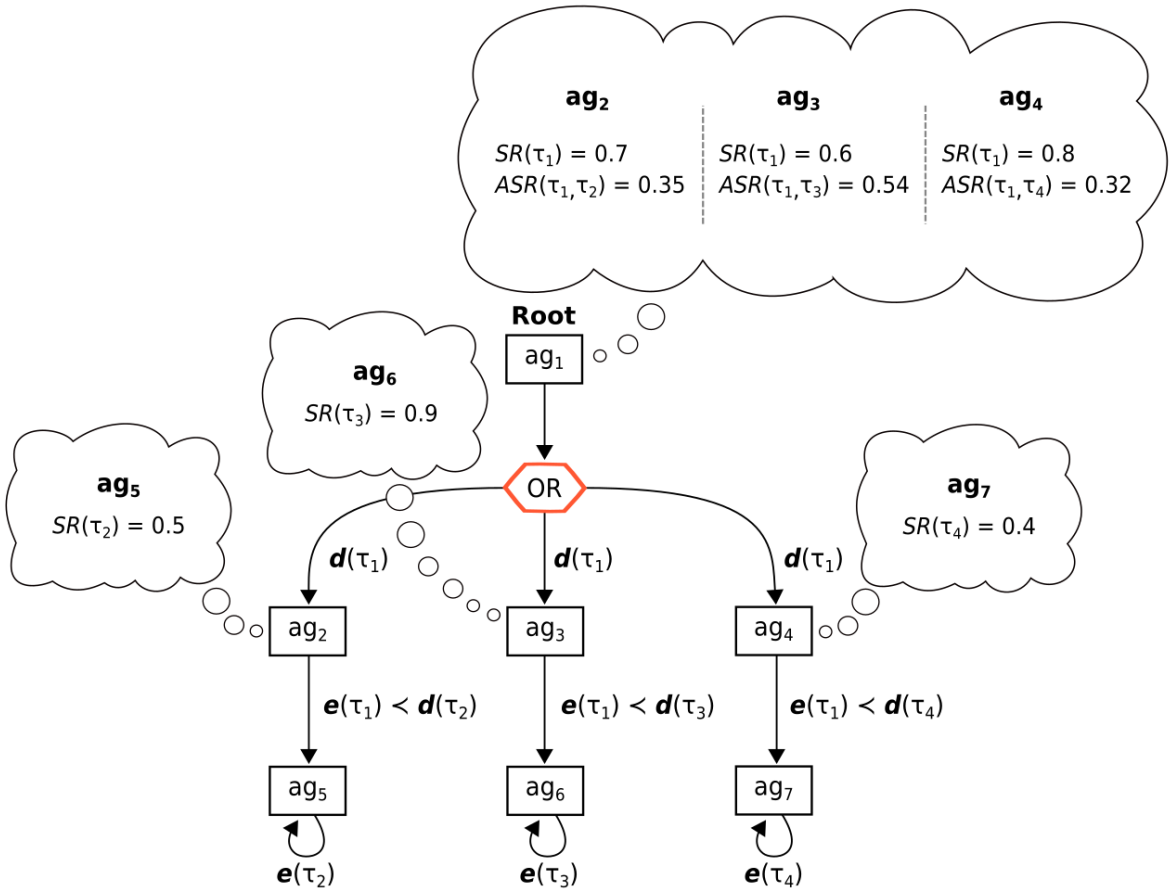


Figure 1 – Delegators' views on their partners, considering the individual success rates (*SR*) and the accumulated success rate for distinct delegation chains (*ASR*). The symbol $\prec$ denotes a dependence relation established between two agents concerning the execution of a task. Thus, the notation $\mathbf{e}(\tau_1) \prec \mathbf{d}(\tau_2)$ means that the execution of task τ_1 depends on the delegation of task τ_2, which, in turn, must be executed by ag_5. Source: Own authorship (2025).

1.4 Goals

General goal: Build a task delegation model with support for sub-delegations and the formation of delegation chains. In this model, partners are selected through a multi-goal analysis, which considers as dimensions social and cognitive components, like the agents' direct and transitive social dependencies, historical success data, behavioral impressions, social image, reputation, and know-how. Additionally, in this thesis, we achieve the following sub-goals:

- **Sub-goal 1**: Proposing a network structure capable of representing complex delegation chains. This network must express all dependence relations in the delegation chains formed from the sub-delegations performed among the agents. The networks produced are used to evaluate the performance of the proposed delegation model.

- **Sub-goal 2**: Evaluate the accuracy of the proposed delegation model concerning the partner selection considering dynamic scenarios, where the agents' behavior can change over time. These behavior changes affect the agents' performance concerning the execution of their tasks and, hence, the evaluations produced by their delegators (*e.g.*, social image, reputation, know-how, and success rate).
- **Sub-goal 3**: Evaluate the complexity and efficiency of the proposed delegation model compared to a mono-episodic approach.

1.5 Research Hypotheses

The hypotheses examined in this thesis originate from the following claims:

- **Hypothesis 1: A delegation model designed to cope with delegation chains can achieve better results than mono-episodic models.** Extracting additional information from a delegation chain gives a delegator a more comprehensive view of its partners' behavior, increasing its chances of selecting suitable partners. Moreover, since a delegation chain is constructed on the basis of transitive dependencies, the success rate of each agent along a chain can be aggregated into a single measure (*i.e.*, accumulated success rate (ASR)), which defines the probability of all agents in the chain successfully completing their respective tasks. Consequently, the accumulated success rate propagation along the chain prevents a delegator from selecting their partners using just local evaluations obtained through its direct dependence relations. This level of refinement can only be achieved when taking transitive dependencies into account, allowing for more informed decision-making in dynamic environments.
- **Hypothesis 2: The network's topology, created based on dependence chains, influences how often success rates are updated and impressions are shared by delegators, improving the effectiveness of social evaluation mechanisms and partner selection strategies based on delegation chains.** Delegation chains can be integrated into a complex network as dependence relations are established. These relations determine how agents connect with each other and with their neighborhoods, shaping the network's topology. This topology, in turn, influences how often success rates are updated and how many impressions are shared among delegators. As a result, it tends to improve the effectiveness of social evaluation mechanisms used to estimate agents' competences and the performance of partner selection strategies that rely on delegation chains to propagate agents' success rates.
- **Hypothesis 3: The use of transitive dependencies as a mechanism for propagating information can speed up the identification of good partners.** Considering the structure of delegation chains, formed from transitive dependencies, to propagate accumulated success rates allows agents (delegators) to choose their partners based on

the chain's success probability as a whole rather than solely on the individual success probability of each partner. Such a feature accelerates the perception of which partners are the most trustworthy, even in an environment where the agents' behavior changes over time.

- **Hypothesis 4: A multi-goal delegation model, based on distinct social and cognitive dimensions, enables delegators to select partners capable of ensuring a high likelihood of success while meeting their expectations regarding task execution.** Considering distinct evaluation dimensions, such as social dependencies, preferences, historical success rates, reputation, and know-how in a delegation model allows delegators to choose partners based not only on the likelihood of task completion but also on their ability to meet the delegator's expectations regarding task execution (*e.g.*, considering whether the task was performed within the time and cost estimated by the partner).

1.6 Document Outline

The remainder of the thesis is divided into six chapters. Chapter 2 reviews essential concepts employed in our proposed delegation model, such as situational dependence graphs, multi-armed bandits, and trust and reputation theory. Chapter 3 covers the modeling details, presenting the internal components of our delegation model, which are derived from trust and reputation theory. Chapter 4 outlines the methodology used to validate the hypotheses of this thesis, including evaluation metrics and experimental constraints. Chapter 5 presents the experiments designed to test the proposed hypotheses and discusses the results obtained. Finally, Chapter 6 concludes the thesis by summarizing the contributions of this work and proposing future research directions. An appendix is presented at the end of this document, which formalizes a file format to represent networks used as input for our experiments.

1.7 Publications

The following titles were published as direct products of this research:

- Baqueta, J. J. and Tacla, C. A. (2025). A Task Delegation Model: An Approach Focused on Sub-delegations and Delegation Chain Formation. Journal of Autonomous Agents and Multi-Agent Systems (JAAMAS) (*under review*).

- Baqueta, J. J., Morveli-Espinoza, M. and Tacla, C. A. (2025). Transparent Task Delegation in Multi-Agent Systems Using the QuAD-V Framework. Appl. Sci. 2025, 15, 4357.

- Baqueta, J. J. and Tacla, C. A. (2024). Explaining Task Delegation Through Argumentation Debates with Votes. Lecture Notes in Computer Science. 1ed.: Springer Nature Switzerland, 2025, v. , p. 372-383.

- Baqueta, J. J. and Tacla, C. A. (2023). A Task Delegation Model for Delegation Chains. C-MAS: Citizen-Centric Multiagent Systems, Londres. Proceedings of C-MAS. v.1. p.1 - 7

- Baqueta, J. J., Morveli-Espinoza, M., Gimenez-Lugo, G. A., and Tacla, C. A. (2022). An Adaptive Trust Model Based on Fuzzy Logic. Revista de Informática Teórica e Aplicada, 29(1), 54-67.

- Baqueta, J. J., Morveli-Espinoza, M., and Tacla, C. A. (2021). Trust Computing Based on Argumentation Debates with Votes for Detecting Lying Agents. In XVIII Encontro Nacional de Inteligência Artificial e Computacional (pp. 410-421). SBC.

- Baqueta, J. J. (2020). A Computational Trust Model for Task Delegation. In 1st Doctoral Consortium at the European Conference on Artificial Intelligence (DC-ECAI 2020) (Vol. 29).

- Baqueta, J. J., Morveli-Espinoza, M., Gimenez-Lugo, G. A., and Tacla, C. A. (2020). The Role of Social Image as a Behavior Filter: A Case Study Based on Cognitive Agents. In 14th Workshop-School on Agents, Environments, and Applications (WESAAC).

2 BACKGROUND

This chapter provides an overview of the fundamental definitions and concepts that form the foundation of this work, essential for comprehending our delegation model. Herein, we discuss the various types of social dependence relations that can be established among agents as they delegate tasks to to one another, present the social evaluation approaches integrated into our delegation model to facilitate the trust computing process, introduce a data structure known as Dependence Situation Network (DS-net), which can represent delegation chains, and describe the multi-armed bandit problem, discussing how it can be used as a partner selection mechanism.

2.1 Task Delegation

A task delegation scenario involves an agent (*delegator*) who aims to achieve a goal g but cannot perform the task τ necessary to accomplish g by itself. Thus, to achieve this goal, the delegator needs to delegate the task τ to a partner (*delegatee*) capable of completing τ from the delegator's perspective (Castelfranchi; Falcone, 2010; Cantucci; Falcone; Castelfranchi, 2019). In this scenario, the delegator will achieve its goal g only if the delegatee successfully completes τ (Castelfranchi; Falcone, 2020).

In particular, the delegation process described in this work is inspired by the Contract-Net Protocol (CNP) (Smith, 1980), which defines a negotiation-based mechanism for dynamic task allocation in multi-agent systems. Similar to CNP, our approach involves three main stages: task announcement (offer request), proposal evaluation (partner selection), and task execution. Although we adopt this structure, our model extends it by incorporating social evaluation mechanisms and support for delegation chains. These phases are described in detail as follows:

Offer request: a delegator notifies its partners of the intention to delegate a task τ. After receiving the notification, partners send their offers to the delegator, providing their performance estimations for the task criteria (*e.g.*, the time required to perform τ).

Definition 1. *An offer is a 4-tuple $\langle \alpha, \beta, \tau, V \rangle$, such as:*

- *α is the agent who requests the offer (delegator);*
- *β is the agent who produced the offer (partner);*
- *τ is the task that α intends to delegate to β if β is selected as its delegatee;*
- *$V = [(c_1, v_1), ..., (c_k, v_k)]$ is a vector that represents the performance estimations made by β, where each value $v_i \in [0, +\infty]$ is a β's estimation for a τ's criterion c_i.*

Partner selection: after receiving the offers from its partners, the delegator must select its delegatees. The choice of a delegatee involves evaluating the partners' offers for a task τ, their success history in executing τ (success likelihood), and their capability to execute τ according to their performance estimations (competencies).

Execution: the delegatees execute their tasks. Subsequently, the delegator evaluates its delegatees based on their performance. An evaluation performed by a delegator α concerning the performance of its delegatee β takes into account the outcome produced by β after executing the task τ. An outcome is a tuple similar to an offer. However, the values assigned to task criteria represent the delegatee's actual performance. In this sense, when β is capable of producing an outcome for τ, it indicates that τ was successfully completed. Otherwise, β failed to execute τ, preventing α from achieving its goal. We consider that a failure can be caused by two distinct situations:

- **Execution failure:** occurs when β is prevented from executing τ, resulting in the failure to produce the task outcome (*e.g.*, β cannot complete τ due to a lack of some skill or resource);
- **Delivery failure:** occurs when β successfully completes τ but is unable to transfer the results to α (*e.g.*, some environmental conditions, such as bad weather or communication problems, prevent β from delivering the outcomes produced from the execution of τ to α).

To illustrate the task delegation process phases, consider a project manager (delegator) who needs to organize a marketing campaign (task). Unable to do it alone, he decomposes the task into more specialized tasks and notifies potential team members (partners) about the campaign tasks that need delegation (*e.g.*, social media management, content creation, and content marketing). Interested team members respond with offers, indicating the hours needed to complete each task (*offer request phase*). The manager then evaluates the offers to select team members (delegatees). This evaluation considers the time constraints for project completion, the historical success of potential team members in completing their tasks, and their capability of completing their tasks on time (*partner selection phase*). Once selected, tasks are delegated, and the manager assesses each member's performance regarding the produced outcomes (*execution phase*). These evaluations can be used by the project manager in future similar projects for selecting team members.

2.2 Multi-Armed Bandits

Multi-armed bandit (MAB) is a machine learning paradigm that can be seen as the single-state reinforcement learning approach (Drugan; Nowe, 2013). In the MAB problem, a single agent repeatedly selects one among a finite set of actions aiming to maximize its obtained reward. An action is termed *arm*, while the act of choosing an arm is termed *arm-pull*

(Auer; Cesa-bianchi; Fischer, 2002). For each arm-pull, the agent must decide whether to choose a known arm from those already tried (exploitation) or select something new, expecting to get better rewards (exploration). Pulling an arm i n-times will yield the rewards $\mu_{i,1}, \mu_{i,2}, \ldots \mu_{i,n}$, which are independent of each other and associated with an unknown probability distribution.

In particular, task delegation can be seen as an exploitation/exploration problem and, thus, can be modeled as a MAB instance, as depicted in Figure 2. In such a scenario, a delegator intends to get a chair but cannot achieve it alone since it does not have the tools and materials to build a chair by itself. Consequently, the delegator must delegate the task of building a chair (τ) to a partner capable of performing it. However, as many partners are available, the delegator must decide which of them will be selected as its delegatee.

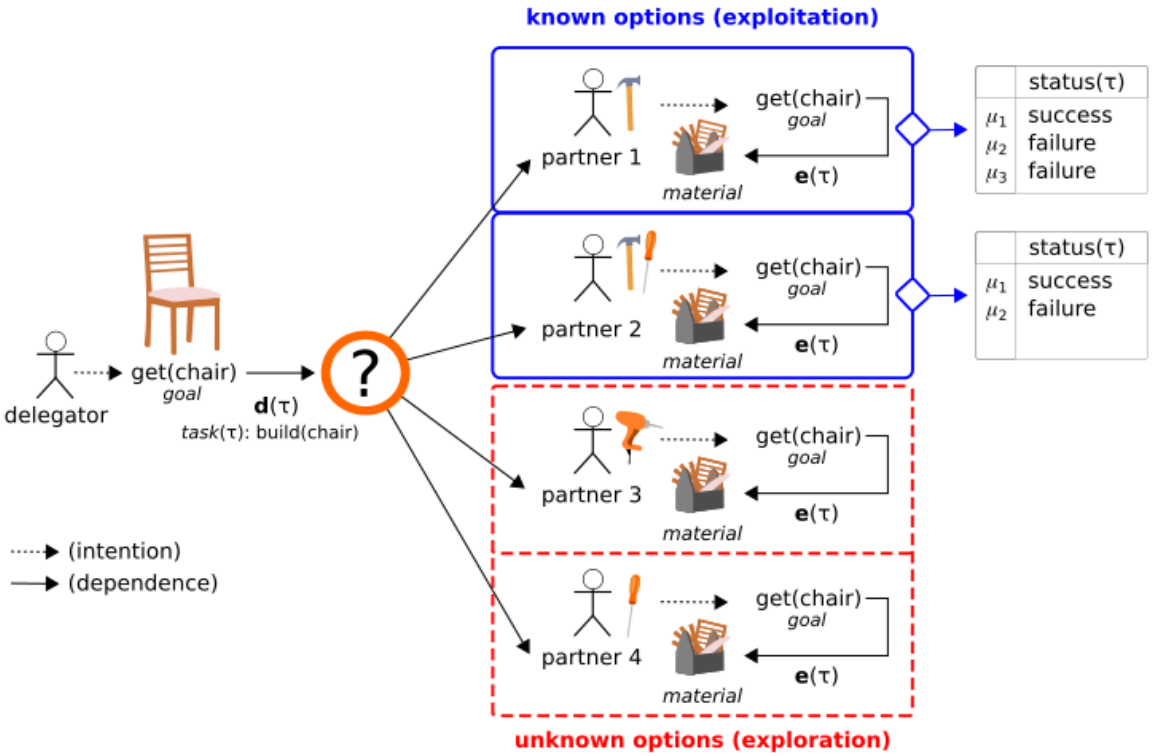


Figure 2 – Multi-armed bandit problem as a partner selection mechanism.
Source: Own authorship (2025).

In situation presented in Figure 2, the delegator must decide whether to delegate a task to a known partner, with witch it has already interacted, expecting a likely good outcome (exploitation), or selecting a new partner as its delegatee, hoping to get better results (exploration). This decision can be made through the success rate (SR) of each partner, taking into account the rewards obtained over time. The success rate can be computed based on the binary rewards (success/failure) obtained each time a partner executes the task τ. A simple way to calculate it is to divide the number of successes obtained by a partner by the number of times τ was performed by it (*e.g.*, $SR(partner1) = \frac{1}{3}$ and $SR(partner2) = \frac{1}{2}$). As discussed in the next chapter, besides the number of successes, other dimensions can be considered to assess a partner's

performance concerning the execution of a task, like the estimated time the partner takes to build a chair, which can vary based on the tools used by it to execute τ.

In order to explore the arms that have not been chosen yet, a MAB policy must be employed. Such a policy aims to determine a sequence of arm-pulls that maximizes the accumulated rewards over time, balancing the trade-off between exploitation and exploration (Hayes *et al.*, 2022) (Chapelle; Li, 2011) (Auer; Cesa-bianchi; Fischer, 2002). One of the most popular MAB policies is the ϵ-*greedy* (Sutton; Barto, 2018). This policy relies on a random variable ν, which represents the probability of selecting a random arm, and a threshold value defined as ϵ. When $\nu < \epsilon$, the arm that yields the highest expected reward so far is selected, whereas when $\nu \geq \epsilon$, a random arm is selected (Afanador; Baptista; Oren, 2019). In this case of task delegation, the value assigned to ϵ defines how much time each delegator will spend exploring new partners.

2.3 Social Control Mechanisms

Social control mechanisms offer agents means to penalize undesirable social behaviors by themselves (Pinyol; Sabater-mir, 2013). Trust and reputation models are considered good solutions concerning social control (Griffiths, 2005) (Castelfranchi; Falcone, 2010) (Huynh; Jennings; Shadbolt, 2004) (Sabater; Paolucci; Conte, 2006) (Cho; Chan; Adali, 2015). In computational studies about trust, the decision to trust in someone (partner) is a complex action that can be decomposed into internal and external components (Castelfranchi; Falcone, 2020). Internal components include the partner's attributes, such as its competencies and historical behavior. External components refer to an agent's beliefs about its partners, which might change due to dynamic environmental conditions (*e.g.*, obstacles, adversities, interference) (Castelfranchi; Falcone, 2010).

In the violin concert scenario discussed by (Castelfranchi; Falcone, 2010), even trusting a violinist for playing very well (*i.e.*, his abilities to play the violin are extraordinary, making him very competent to do great concerts), suppose he has to do a concert in an open environment with particularly bad weather conditions, and it is very cold (adversities). Such conditions may affect the specific hand abilities of the violinist and, hence, his performance (obstacles). Moreover, his performance could also be affected by a particularly distracted, inattentive, or noisy audience (interference). In such conditions, the violinist's performance could cause some frustration for his audience regarding their expectations for the concert, resulting in negative evaluations of the violinist (*e.g.*, negative assessments of the violinist's competencies).

2.3.1 Impressions

The delegation model proposed in this work does not directly address external factors. As discussed hereafter, we represent such elements in our experiments indirectly through changes in the behavior of the agents. These changes can alter the trust a delegator places in its partner,

depending on the new behavior assumed by the partner. In particular, we adopt the definition of trust suggested by (Castelfranchi; Guerini, 2007), where trust is defined by five components, which are represented by a 5-tuple $TRUST\langle\alpha, \beta, \gamma, \tau, g\rangle$. This tuple can be read as α trusts β based on the context γ to perform the task τ and achieve the goal g. In our case, the context γ for a delegator α is defined based on historical information about β's success in performing τ over time and evaluative beliefs (impressions) about β's competencies as a delegatee for τ. Such impressions are collected through direct interactions between α and β, during which τ was delegated to β (Conte; Paolucci, 2003).

Definition 2. *An impression is a 5-tuple $\langle\alpha, \beta, \tau, t, S\rangle$, such as:*

- *α is the delegator who formed the impression;*
- *β is the assessed delegatee;*
- *τ is the task performed by β.*
- *t is the time in which the impression was created by α.*
- *$S = [(c_1, s_1), ..., (c_k, s_k)]$ is a vector of ratings, where each pair (c_i, s_i) represents a score $s_i \in [0,1]$ given by α to β based on a criterion c_i for task τ (e.g., scores for completing τ within the agreed time and budget, considering time and cost criteria).*

2.3.2 Types of Impressions

Impressions can be grouped to build more comprehensive social measures of a partner's competencies, including its social image, reputation, and know-how (Conte; Paolucci, 2002) (Sabater; Paolucci; Conte, 2006):

Social image is a social measure calculated by the aggregation of the set of impressions produced by a delegator α regarding the competencies of a partner β as a delegatee of a task τ. The social image can be seen as α's personal opinion about β's competencies, taking into account a task τ (Pinyol; Sabater-mir, 2013).

Reputation is another social measure that attests to the competencies of a partner β concerning the execution of a task τ. However, unlike the social image, a delegator α calculates the β's reputation regarding τ by aggregating a set of impressions obtained from third parties (other delegators). Thus, reputation constitutes a collective evaluation circulating among a group where most members agree, without necessarily verifying the truth or sources of the impressions (Conte; Paolucci, 2002) (Pinyol; Sabater-mir, 2013). In this work, a group consists of delegators who assess a common delegatee concerning the execution of a task. Therefore, β's reputation is built as delegators share their impressions of β's competence in executing τ.

Know-how can be seen as a specialized form of reputation where impressions produced by delegators are shared with their delegatees (Huynh; Jennings; Shadbolt, 2004). As a

delegatee β interacts with various delegators, executing a task τ, β accumulates impressions regarding its competencies as delegatee of τ. These impressions can be send to a delegator α during the partner selection to attest the β's competencies, similar to job references (Botelho *et al.*, 2018) (Buccafurri *et al.*, 2015). By aggregating the impressions sent by β, α is able to compute a social measure referring to β's know-how.

From now on, we refer to an impression contributing to the social image of a partner β as a *personal impression*, to β's reputation as a *shared impression*, and to β's know-how as a *reference*. Moreover, aiming to distinguish the different types of impressions, consider a commercial scenario where an agent α (delegator/buyer) wishes to buy books and magazines from an agent β (delegatee/seller). If α has interacted with β before, it could use its previous personal impressions to form distinct social images of β. For instance, α may associate β with the social image of a good bookseller, considering the excellent quality and prices of the books sold by β in their past interactions. In this case, the competencies of β are estimated considering the task of selling books, taking into account two criteria: the quality and prices of the books. At the same time, α could assign the social image of a bad magazine seller to β due to the limited diversity of magazines for sale (*i.e.*, another task, for which β's competencies are estimated based on the available magazines). On the other hand, if α and β have never interacted with each other, α could estimate β's reputation through the shared impressions about β formed by other buyers, to decide whether to interact with β or not. Additionally, α could require references from β as a book or magazine seller before purchasing products from it.

2.3.3 Impression Filter

A delegator α can select specific types of impressions from its belief base about a partner β through an impression filter f_β^α. For instance, a filter can be configured to return only the α's personal impressions regarding an agent β who acts as a doctor ordering a treatment (task):

$$f_\beta^\alpha := (\alpha = delegator \wedge \beta = delegatee \wedge \tau = order\ a\ treatment) \quad (1)$$

Another possibility is to filter all shared impressions in the α's belief base about β (*i.e.*, impressions shared with α by other delegators of β):

$$f_\beta^\alpha := (\alpha \neq delegator \wedge \beta = delegatee \wedge \tau = order\ a\ treatment) \quad (2)$$

Finally, the references sent from β to α can be filtered by specifying the source of the impression (*i.e.*, the agent that sent the impression to α). Personal impressions do not have a source, as they are produced by the delegator itself. In contrast, the source of a shared impression is the delegator who produced and shared it. For references, the source is the delegatee being assessed in the impression.

2.4 Social Dependence Relations

A social dependence relationship occurs when a delegator relies on another agent (partner) to achieve a goal. This type of relationship creates a *direct dependence* from the delegator to its partner (Castelfranchi; Miceli; Cesta, 1992). Agents' social dependence relationships can be represented through a dependence situation, which describes the dependence relationship held between the agents based on the goals that they wish to accomplish (Sichman *et al.*, 1998). For a task delegation scenario where sub-delegations are allowed, a social dependence relation can be defined as follows (Costa; Dimuro, 2006):

Definition 3. *A delegator α that aims to achieve a goal g, through a task τ, socially depends on a partner β, denoted $(\alpha \prec \beta.\tau \mid g)$, if and only if:*

1. *g is a goal of α;*
2. *α cannot perform τ by itself;*
3. *β can perform τ by itself or through sub-delegations;*
4. *τ being completed by β implies g being achieved.*

As introduced by (Costa; Dimuro, 2006), *unilateral* dependence represents the simplest form of social dependence relation, where a delegator α relies on a partner β to achieve a specific goal g through the execution of a task τ. Formally, a unilateral dependence relation is expressed as:

$$\exists \tau, g.(\alpha \prec \beta.\tau \mid g) \wedge \forall \tau', g'.\neg(\beta \prec \alpha.\tau' \mid g') \tag{3}$$

where, α depends on β concerning some task τ and some goal g, but there is no task and no goal for which β depends on α.

More complex forms of social dependence relations can be built based on unilateral relations, considering the number of agents involved and how their tasks are combined to achieve a goal, as seen in *AND-dependence* and *OR-dependence* relationships. In an AND-dependence relation (*multiparty*), a delegator α requires multiple partners β_i to successfully complete their respective tasks τ_i for achieving its g:

$$(\alpha \prec \beta_1.\tau_1 \mid g) \wedge (\alpha \prec \beta_2.\tau_2 \mid g) \wedge ... \wedge (\alpha \prec \beta_n.\tau_n \mid g) \tag{4}$$

Conversely, an OR-dependence relation can be expressed through single or multiple-task variants. In a *single-task* OR-dependence, a delegator α needs at least one of its partners β_i to complete a common task τ for accomplish its goal g:

$$(\alpha \prec \beta_1.\tau \mid g) \vee (\alpha \prec \beta_2.\tau \mid g) \vee ... \vee (\alpha \prec \beta_n.\tau \mid g) \tag{5}$$

On the other hand, in an OR-dependence of the type *multiple-task*, the delegator α is able to achieve its goal g if any partner β_i completes its task τ_i:

$$(\alpha \prec \beta_1.\tau_1 \mid g) \vee (\alpha \prec \beta_2.\tau_2 \mid g) \vee ... \vee (\alpha \prec \beta_n.\tau_n \mid g) \tag{6}$$

2.5 Dependence Situation Graphs

Social Dependence relations can be represented by Dependence graphs (DG) (Costa; Dimuro, 2006). A $DG = (Ag, G, Pl, Ac, Ar, \Psi)$ is a structure where agents Ag, goals G, plans Pl, and actions Ac are nodes connected by arcs Ar, as defined by Ψ. This shows how agents rely on each other to achieve goals through plans requiring actions from others. As pointed out by (Costa; Dimuro, 2006), DGs can also be simplified by omitting plans and using goals as arc labels, such as presented in Figure 3.

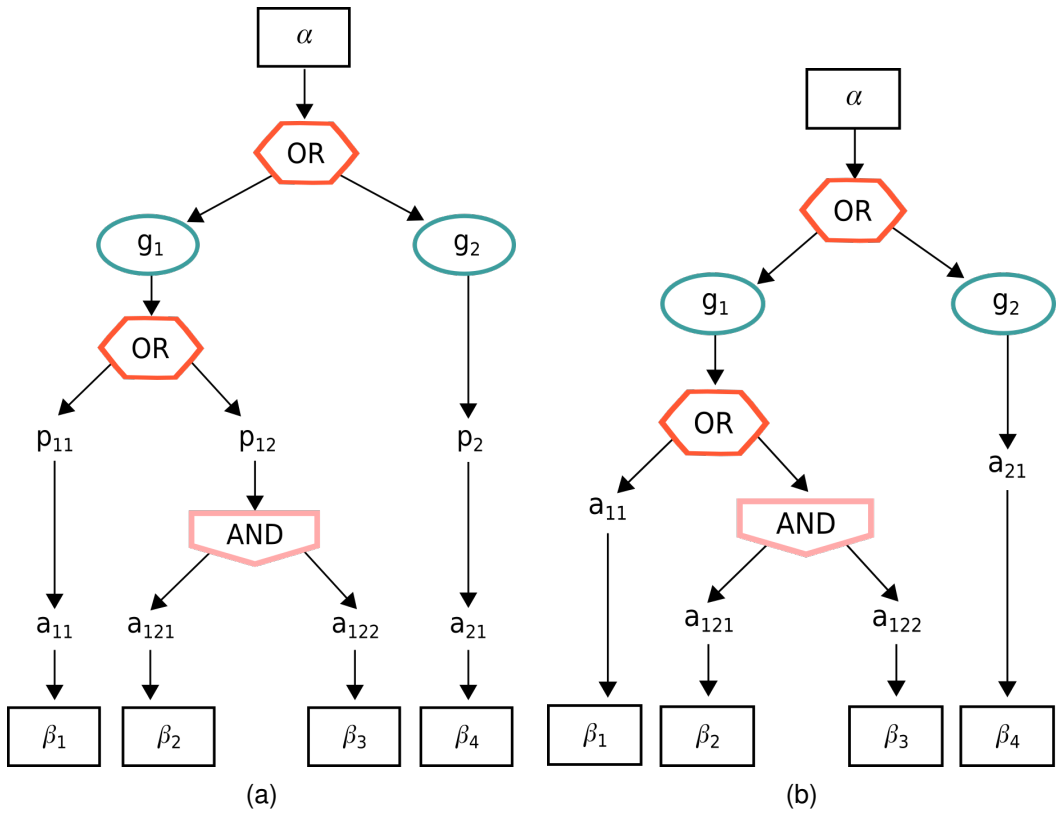


Figure 3 – *Dependence graphs* (DG): (a) dependence relations between agent α and agents β_1, β_2, β_3, β_4, taking into account the goals g_1, g_2, plans p_{11}, p_{12}, p_2, and actions a_{11}, a_{121}, a_{122}, a_{21}, and (b) the same DG represented in its reduced form.
Source: Costa e Dimuro (2006).

On the other hand, when dependence relations represented in a DG are defined based on the agents' goals, a more compact structure can be adopted, where plans and actions are abstracted away. Such a structure is known as the dependence situation graph (DS-graph) (Costa; Dimuro, 2006). A DS-graph is a particular case of DG that allows an agent α to calculate

its dependence situation[1] with respect to another agent β for a given goal (Sichman *et al.*, 1998). A DS-graph is defined as (Costa; Dimuro, 2006):

Definition 4. *Assuming Ag as the set of agents and G as the set of goals that those agents may have. A DS-graph over Ag and G is a structure DS = (Ag, G, Ar, Lk, Ψ, Δ) such that:*

- Ar *is a set of arcs, connecting either an agent to a goal or a goal to an agent;*
- Lk *is a set of links, connecting subsets of arcs;*
- $\Psi : Ar \rightarrow (AgXG) \cup (GXAg)$ *is a function assigning either an agent and a goal or a goal and an agent to each arc, so that if $\Psi(ar) = (ag, g)$ then arc ar indicates that agent ag has the goal g, and if $\Psi(ar) = (g, ag)$ then arc ar indicates that goal g requires ag to perform something action in order to be achieved;*
- $\Delta : Lk \rightarrow \wp(Ar)$ *is a function that links arcs together representing an AND-dependence, so that $\Delta(l) = ar_1, \ldots, ar_n$ iff either:*
  - *there are an agent ag and n goals $g_1, \ldots, g_n$ such that $\Psi(ar_1) = (ag, g_1), \ldots, \Psi(ar_n) = (ag, g_n)$ indicating that ag aims the achievement of all the goals $g_1, \ldots, g_n$; or,*
  - *there are a goal g and n agents $ag_1, \ldots, ag_n$ such that $\Psi(ar_1) = (g, ag_1), \ldots, \Psi(ar_n) = (g, ag_n)$ indicating that g requires the involvement of all the agents in the set $\{ag_1, \ldots, ag_n\}$ in order to be achieved.*

An example of DS-graph is presented in Figure 4, which expresses the dependence relations established among the agents α, β_1, β_2, β_3, and β_4.

2.6 Delegation Chains

A delegation chain can be seen as a sequence of delegations performed in a given order by a set of agents that aim to achieve their goals (An; Miao; Cheng, 2005) (An *et al.*, 2007). Along a delegation chain, the agents can act as delegators or as delegatees, and the agent who makes the first delegation request is called the chain's root (Afanador; Baptista; Oren, 2019). In this work, we adopt the Dependence Situation Network (DS-net) to represent potential delegation chains. A DS-net extends a DS-graph. Specifically, we extend the DS-graph by explicitly representing the instantiations of delegation chains in a DS-net, which are created based on the actions available to an agent concerning a task τ (*i.e.*, the delegation **d**(τ) and execution **e**(τ) actions) and the order in which agents must perform their tasks.

[1] Dependence situation refers to the relations of dependencies held among the agents based on the goals that they wish to accomplish.

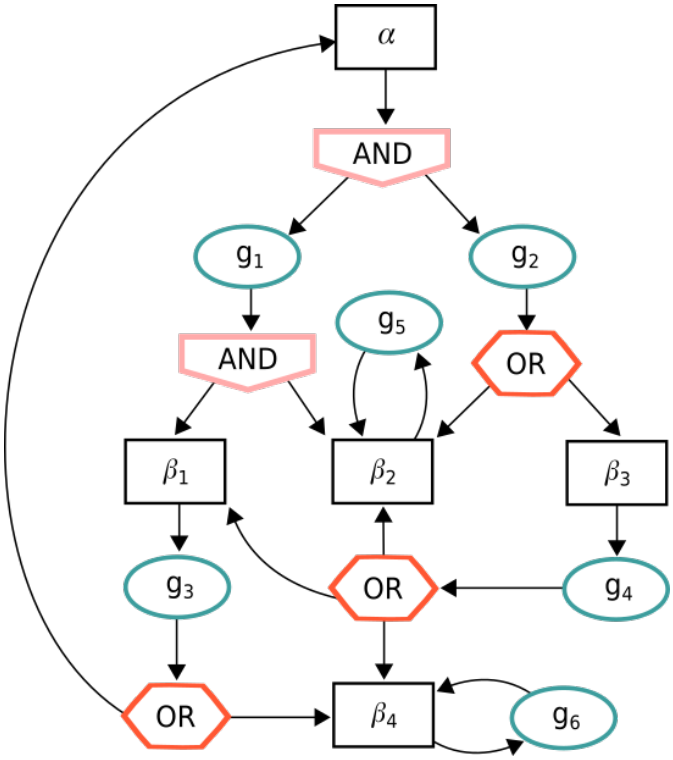


Figure 4 – DS-graph: dependence situation involving agents α, β_1, β_2, β_3, and β_4. Source: Costa e Dimuro (2006).

2.6.1 Dependence Situation Network

Given the following sets:

- Ag: set of agents; ag represents an element of the set, and there is precisely one root agent that initiates the delegation process, as it has the general goal to be pursued.
- G: set of goals that the agents can pursue; g represents an element of the set.
- T: set of tasks to accomplish the goals; τ represents an element of the set.

The DS-net is formalized as a directed acyclic bipartite graph with two types of nodes (N) (*i.e.*, agents and goals):

$$N = Ag \cup G \textit{ with } Ag \cap G = \emptyset \tag{7}$$

We distinguish the relations (the edges) between agents and goals based on their source and destination nodes:

- $HasGoal \subseteq Ag \times G$ (from agents to goals): each pair $(ag, g) \in HasGoal$ indicates that agent ag has goal g. Each goal is associated with a single agent, and every agent has at least one goal.
- $Act \subseteq G \times Ag$ (from goals to agents): each pair $(g, ag) \in Act$ indicates that the accomplishment of g requires either a delegation action, where a task τ is delegated to

ag (**d**(τ)), or an execution action, in which ag executes τ (**e**(τ)). Each act is associated with one action type (ActType), and one task (ActTask):

$$\text{ActType} : \text{Act} \mapsto \{\mathbf{d}, \mathbf{e}\} \tag{8}$$

$$\text{ActTask} : \text{Act} \mapsto T \tag{9}$$

The delegator of an act $Act(\mathbf{g}, ag)$ is the agent ag_d that $HasGoal(ag_d, \mathbf{g})$. We abuse when using the word delegator because ag_d can delegate a task τ or execute τ:

$$\text{ActDelegator} : \text{Act} \mapsto Ag \tag{10}$$

A path of the graph always starts with the root agent and finishes with a terminator agent (*i.e.*, an agent that has a goal and executes by itself a task to achieve that goal). We use the $\exists!$ as the unique existential quantification:

- There is only one root agent:

$$\exists! ag, g(HasGoal(ag, g_1) \wedge \forall g_2(g_1 \neq g_2 \rightarrow \neg Act(g_2, ag))) \tag{11}$$

- The terminator agents must execute a task:

$$\forall ag, g(HasGoal(ag, g) \wedge Act(g, ag) \rightarrow ActType(g, ag) = \mathbf{e}) \tag{12}$$

Acts can be grouped to represent OR-dependencies and AND-dependencies among a delegator and its partners. The delegator of a set of acts can be obtained by the function *ActDelegator* applied to any element of the set. Thus, an OR-dependence is composed of one or more acts, meaning that accomplishing any of these acts is sufficient for the delegator to achieve the goal. $\mathcal{P}$ denotes the potency set:

$$\text{OR-dependencies} \subseteq \mathcal{P}(\text{Act}) \tag{13}$$

An AND-dependence is composed of one or more acts, indicating that all of these acts must be accomplished for the delegator agent to achieve the goal.

$$\text{AND-dependencies} \subseteq \mathcal{P}(\text{Act}) \tag{14}$$

Some constraints apply to these dependencies. First, the dependencies sets are not empty, and the OR-dependencies and AND-dependencies are mutually exclusive:

$$\text{Dep} = \text{AND-dependencies} \cup \text{OR-dependencies}, \text{with Dep} \neq \emptyset \tag{15}$$

$$\text{OR-dependencies} \cap \text{AND-dependencies} = \emptyset \tag{16}$$

An act belongs to only one dependence group:

$$\forall X, Y \in \text{Dep},\ X \neq Y \Rightarrow X \cap Y = \emptyset \tag{17}$$

Additionally, we formalize a grammar that allows the representation a DS-net through a dependence expression, as defined below:

$$\begin{aligned}
&\langle DSN \rangle ::= (ag \prec \langle dep \rangle \mid g) \\
&\langle dep \rangle ::= ag.\tau \\
&\quad || \quad (ag.\tau \prec \langle dep \rangle \mid g) \\
&\quad || \quad \langle dep \rangle \vee \langle dep \rangle && \text{(OR-dependence)} \\
&\quad || \quad \langle dep \rangle \wedge \langle dep \rangle && \text{(AND-dependence)}
\end{aligned}$$

where ag is an agent, g is a goal, τ is a task that may be performed through a delegation or execution action, $\prec$ indicates a dependence between a delegator and its partners, and $\mid$ associates a delegator with a goal. In the grammar specification language $\langle\rangle$ represents a non-terminal element of the language, $::=$ is the definition symbol, and $||$ denotes alternative forms of sentence composition.

2.6.2 Delegation Instances

The dependence relationships described in a DS-net represent all possible relationships that can be established between agents. Specifically, these relationships express the dependencies that a delegator has concerning its partners. In this state, referred to as the *potential view*, a DS-net can be used during the offer request and partner selection phases, allowing message exchanges between delegators and partners. However, during the execution phase of the task delegation process, the state of the DS-net is referred to as the *instantiated view*. In this state, the DS-net is used to represent the instantiation of delegation chains, as tasks are delegated among agents during an interaction. An interaction (*itr*) comprises all delegation instances, starting with the root agent delegation until it reaches the terminator agents. In particular, a delegation instance is formed when a delegator selects its delegatees. Instances of OR and AND dependencies are called *simple* and *composed* delegation instances, respectively. These

instances satisfy the following constraints:

$$\forall D_{OR} \in \text{OR-dependencies}, \exists!(g, ag) \in D_{OR} \text{ such that } (\text{g}, \text{ag}) \in \text{Act}_{itr} \tag{18}$$

where, for each OR-dependence, the delegator selects only one act, which is considered successfully terminated if the associated task τ successfully ends.

$$\forall D_{\text{AND}} \in \text{AND-dependence}, \forall(\text{g}, \text{ag}) \in D_{\text{AND}}, (\text{g}, \text{ag}) \in \text{Act}_{itr} \tag{19}$$

where, for each AND-dependence, the delegator selects all acts, which are considered successfully terminated if all tasks associated with each act successfully end.

Some examples of delegation instances are shown in the DS-nets presented in Figure 5. We assume the self-delegation and delegation patterns are OR-dependencies. In particular, in Figure 5(a), a self-execution pattern is depicted, wherein agent ag_1 can achieve its goal g_1 by executing the task τ_1 by itself (represented by **e**(τ_1)). A simple delegation instance, which is equivalent to the mono-episodic task delegation pattern, is presented in Figure 5(b).

This situation represents a classic task delegation scenario, where ag_1 cannot perform τ_1 alone. Thus, to accomplish its goal g_1, ag_1 must delegate τ_1 (represented by **d**(τ_1)) to another agent (ag_2). When ag_2 receives τ_1, it pursues its own goal (g_2), associated with its goal to complete τ_1. In this simple delegation pattern, ag_2 can self-assess its performance after executing τ_1. However, even if ag_2 successfully completes τ_1, it might fail to deliver the task outcome to ag_1. As a result, while ag_2 achieves g_2 and positively self-evaluates its performance as the executor of τ_1, ag_1 cannot achieve g_1 due to ag_2's failure in delivering τ_1's outcome. From the point of view of ag_1, ag_2 failed to complete τ_1, resulting in a negative evaluation of ag_2's competencies as its delegatee for τ_1. In both cases, the potential view coincides with the instantiated view.

Another single delegation scenario appears in Figure 5(c), in which an 0R-dependence (single-task) defines the dependence relations among the agents. Here, ag_1 must choose whether to perform τ_1 itself or delegate it to one of its partners (ag_2, ag_3, or ag_4). Once a partner is selected, a simple delegation instance is formed.

Figure 5(d) shows a complex delegation chain formed through sub-delegations. Note that ag_1 can achieve g_1 through two alternative sub-chains connected by OR-dependence. In the left sub-chain, ag_2 decomposes τ_1 into τ_3, τ_4, and τ_5, relying on ag_4, ag_5, and ag_6 through AND-dependence. In the right sub-chain, ag_3 recursively delegates τ_2 to ag_7. Different task delegation patterns can be formed by combining AND and OR dependencies, enabling the creation of complex delegation chains.

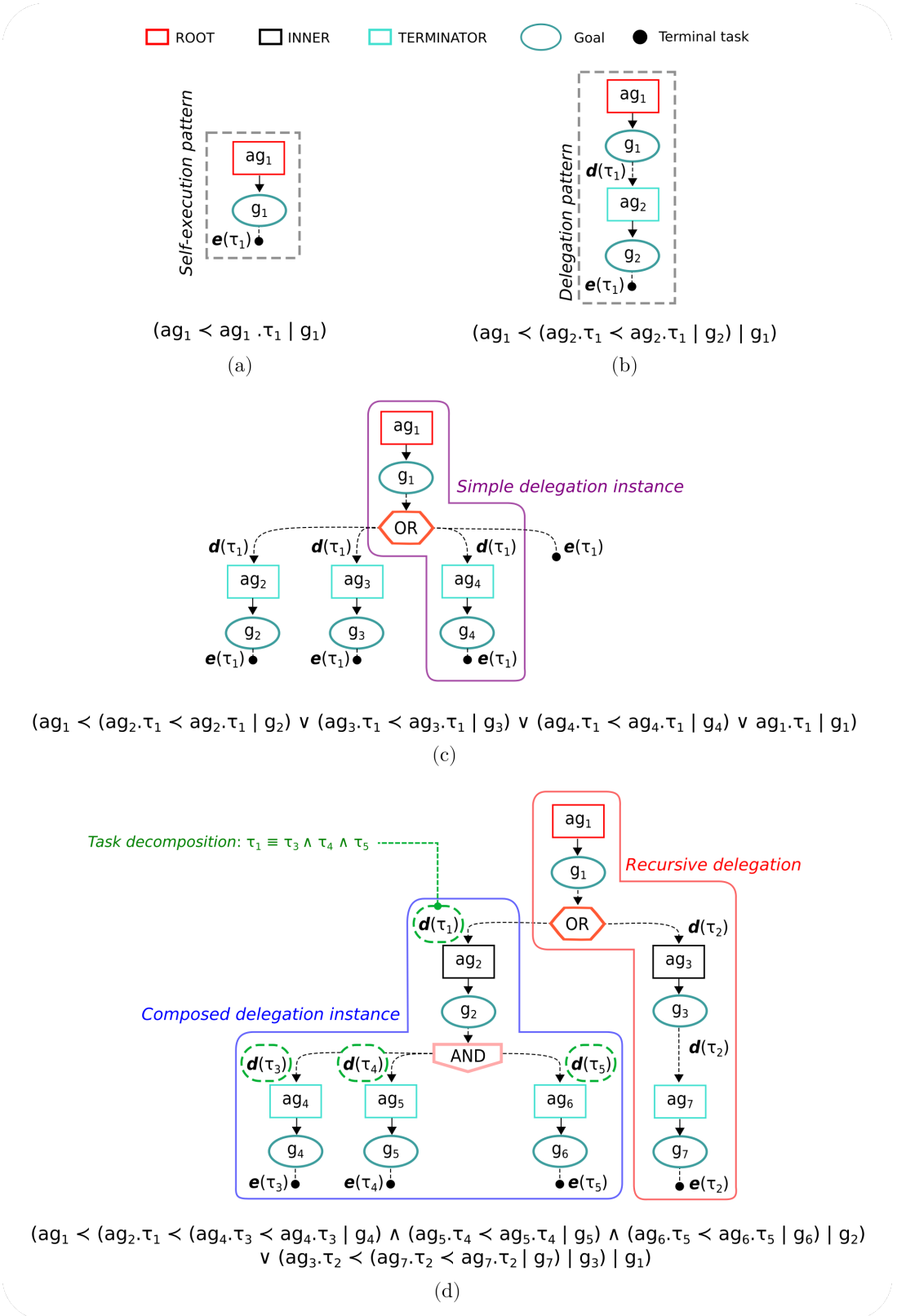


Figure 5 – DS-nets and their respective dependence expressions: (a) self-execution pattern, (b) simple delegation equivalent to the mono-episodic delegation, (c) simple delegation formed from an OR-dependence (single-task), and (d) delegation chains formed through sub-delegations.

Source: Own authorship (2025).

Table 2 – List of potential acts shown in Figure 5(d), taking into account the type and task associated with each act

Act	ActType	ActTask
(g1, ag2)	d	τ_1
(g2, ag4)	d	τ_3
(g4, ag4)	e	τ_3
(g2, ag5)	d	τ_4
(g5, ag5)	e	τ_4
(g2, ag6)	d	τ_5
(g6, ag6)	e	τ_5
(g1, ag3)	d	τ_2
(g3, ag7)	d	τ_2
(g7, ag7)	e	τ_2

To exemplify the concepts introduced in our DS-net structure, considering the DS-net presented in Figure 5(d), we assume that $Ag = \{ag_1, ag_2, ag_3, \ldots, ag_7\}$, $G = \{g_1, \ldots, g_7\}$, and $T = \{\tau_1, \ldots, \tau_5\}$. The goals pursued by the agents, AND-dependencies, and OR-dependencies are represented by the following sets:

$$\text{HasGoal} = \{(\text{ag}_1, g_1), (\text{ag}_2, g_2), (\text{ag}_3, g_3), (\text{ag}_4, g_4), (\text{ag}_5, g_5), (\text{ag}_6, g_6), (\text{ag}_7, g_7)\}$$

$$Act = \{(g_1, \text{ag}_2), (g_2, \text{ag}_4), (g_4, \text{ag}_4), (g_2, \text{ag}_5), (g_5, \text{ag}_5), (g_2, \text{ag}_6), (g_6, \text{ag}_6), (g_1, \text{ag}_3), (g_3, \text{ag}_7), (g_7, \text{ag}_7)\}$$

$$\text{OR-Dependencies} = \{\{(g_1, \text{ag}_2), (g_1, \text{ag}_3)\}, \{(g_4, ag_4)\}, \{(g_5, ag_5)\}, \{(g_6, ag_6)\}, \{(g_7, ag_7)\}\}$$

$$\text{AND-Dependencies} = \{\{(g_2, \text{ag}_4), (g_2, \text{ag}_5), (g_2, \text{ag}_6)\}\}$$

Additionally, Table 2 lists the actions needed to achieve the agents' goals, including the type of action and the associated task. Next, with the aid of our grammar, we present two instantiated views representing different concrete options, which includes only the selected partners:

- Option 1: $(ag_1 \prec (ag_2.\tau_1 \prec (ag_4.\tau_3 \prec ag_4.\tau_3 \mid g_4) \wedge (ag_5.\tau_4 \prec ag_5.\tau_4 \mid g_5) \wedge (ag_6.\tau_5 \prec ag_6.\tau_5 \mid g_6) \mid g_2) \mid g_1)$
- Option 2: $(ag_1 \prec (ag_3.\tau_2 \prec (ag_7.\tau_2 \prec ag_7.\tau_2 \mid g_7) \mid g_3) \mid g_1)$

2.7 Conclusion

To conclude this chapter, we presented some concepts from the literature that influenced the design of our proposal. Based on the reputation theory introduced in (Conte; Paolucci, 2002), we formalized the structure of impressions, which agents use to make social evaluations of their partners, as well as impression filters, which are adopted to extract specific types of impressions from an agent's belief base. Furthermore, we extended the Dependence Situation Graphs

(DG) (Costa; Dimuro, 2006) into a Dependence Situation Network (DS-net). This new structure not only represents agent dependencies (potential view) but also captures delegation instances formed along a delegation chain (instantiated view). Finally, we proposed a formal grammar for representing dependence expressions, enabling a DS-net to be described through a simple expression. These contributions provide the conceptual and formal basis for the delegation model developed in the following chapters.

3 PROPOSED DELEGATION MODEL

We model task delegation as an exploitation/exploration problem, considering sub-delegations and the formation of delegation chains. In this approach, a delegator must decide whether to delegate a task to a known agent (exploitation), expecting a likely good outcome, or to select an unknown partner (delegatee), expecting for better results (exploration). In this sense, we model the partner selection phase of the task delegation process using a MAB approach (Turgay; Oner; Tekin, 2018), where a delegator interacts with its partners (arms that can be pulled) to identify those with the highest likelihood of maximizing the expected reward (*i.e.*, the probability of a partner successfully completing a task while respecting the performance estimations of its offer).

The structure of our delegation model is presented in Figure 6. This model represents the reasoning flow adopted by a delegator to choose a delegatee during the partner selection phase. Through this reasoning, a delegator is able to calculate the delegation likelihood associated with a partner concerning a given task (dotted line). In particular, the structure of the model is divided into three layers (*i.e.*, competence measure (CM), trust measure (TM), and delegation likelihood (DL)). These layers are connected by scalarization functions[1] that combine the social and cognitive components of each layer into a single value. The scalarization process depends on an external parametrization regarding the weights assigned to the model's components. In general, the weight assignment is performed by taking contextual information into account. For instance, the relevance given to the different types of impressions (*i.e.*, personal impressions, shared impressions, and references) can be managed in the first layer of our delegation model by adjusting the weights assigned to the social image (W_{SI}), reputation (W_{RP}), and know-how (W_{KH}) of a partner.

As presented in Figure 6, the first layer is combined with the second to jointly represent the delegator's *decision to trust* (blue line). This decision is made based on the trust measure, $TM(\alpha, \beta, \tau) \in [0, 1]$, which represents the trust that a delegator α places in a partner β, taking into account β's competencies (CM) concerning the execution of a task τ and the likelihood of β successfully completing τ (SL) based on its success history as a delegatee of τ. In this case, when the trust measure assigned to β is 0, it means that α sees β as an untrustworthy partner with a high likelihood of failure in executing τ or causing some frustration due to β lack of competencies in making accurate estimations about its performance as a delegatee of τ (*e.g.*, taking longer than estimated to perform τ or executing τ at a higher cost than initially estimated). When the trust measure assigned to β is 1, it implies that α considers β a trustworthy agent with a significant probability of successfully completing τ and meeting all expectations concerning its performance as a delegatee of τ.

[1] Scalarization functions are widely used to convert a multi-objective optimization problem into a single-objective one. For instance, the Weighted Sum Scalarization function (f_{ws}) takes as input a vector of weights $W = (w_1, ..., w_n)$ and a vector of values $V = (v_1, ..., v_n)$, and returns a scalar value that represents the weighted sum of the input values, where $f_{ws} = \sum_{i=1}^{n} w_i * v_i \mid \sum_{i=1}^{n} w_i = 1$ (Chugh, 2020) (Agarwal; Aggarwal; Lan, 2022).

The third layer of the model represents the delegator's *decision to delegate* (red line). This decision is made by considering the delegation likelihood, $DL(\alpha, \beta, \tau) \in [0, 1]$, where a delegator α infers the probability of delegating a task τ to a partner β based on its trust measure (TM) in β and the utility of β's offer (UT), taking into account its preferences for the criteria of τ. The higher the delegation likelihood assigned to a partner, the more likely it will be chosen as a delegatee.

3.1 Competence Measure

The competence measure ($CM(\beta, \tau) \in [0,1]$) concerns the capability of a partner β in performing a task τ according with its performance estimations. This measure is calculated through a scalarization function (Figure 6), where values for β's social image, reputation, and know-how are aggregated into a single value through a weighted mean. In particular, these dimensions are computed through the aggregation of impressions produced by delegators regarding the performance of their delegatees.

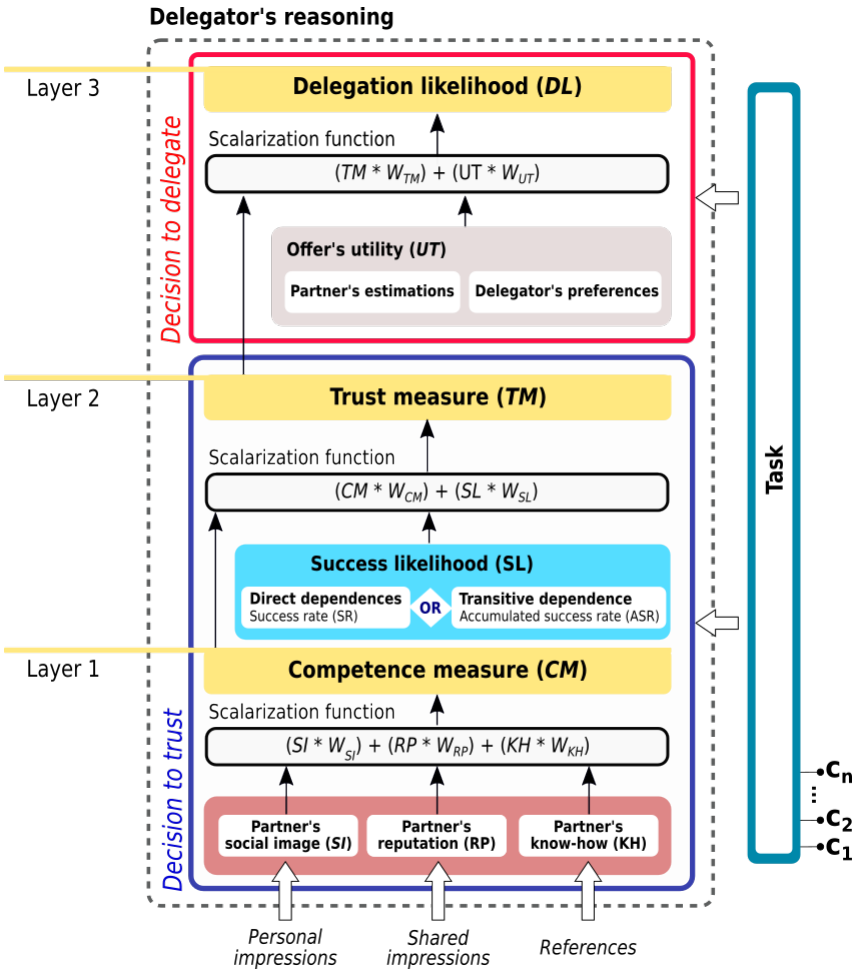


Figure 6 – Proposed delegation model and its components.
Source: Own authorship (2025).

3.1.1 Evaluating Delegatee's Performance

During the execution phase of the task delegation process, a delegator α can evaluate the performance of a delegatee β by comparing β's performance estimations for a task τ with the outcome obtained after executing τ. Such a comparison is made for each criterion associated with τ. The difference between the actual performance and the estimation for a task criterion c determines β's estimation error. If this difference is 0, β was accurate in its performance estimation for c. In contrast, if the estimation error for c is greater than 0, it means β did not execute τ according to its estimation for that criterion. In this case, in the impression produced by α, β will penalized regarding the criterion c based on its estimation error. Therefore, assuming an offer $O = \langle \alpha, \beta, \tau, V \rangle$ and an outcome $O' = \langle \alpha, \beta, \tau, V' \rangle$, the score assigned by α to β for a criterion c of an impression $Imp = \langle \alpha, \beta, \tau, t, S \rangle$ ($Imp(S[c]) \in [0,1]$), considering the execution of τ, is defined as follows:

$$Imp(S[c]) = \begin{cases} \frac{O(V[c])}{O'(V'[c])} & \textit{if } isMin(c) \wedge O'(V'[c]) \geq O(V[c]) \\ \frac{O'(V'[c])}{O(V[c])} & \textit{if } isMax(c) \wedge O'(V'[c]) \leq O(V[c]) \\ 1 & \textit{otherwise} \end{cases} \tag{20}$$

where $isMin(c)$ is a function that indicates if c is a minimization criterion (*i.e.*, a criterion whose value should be as low as possible, such as the time or cost of a task), while the function $isMax(c)$ indicates if c is a maximization criterion (*e.g.*, the quality of a task). Note that when β is able to accurately estimate your performance concerning the criterion c (*i.e.*, fulfilling or exceeding the α's expectation regarding the β's performance (Cantucci; Falcone; Castelfranchi, 2019; Cantucci; Falcone; Castelfranchi, 2021)), the score assigned to c is 1.

3.1.2 Impression Aggregation Process

The impression aggregation process takes into account a filter f^{α}_{β}, which selects the set of impressions from the belief base of a delegator α that will be aggregated, considering the delegation of a task τ to a partner β (Sabater; Sierra, 2001). Specifically, β's social image ($SI \in [0,1]$) is estimated by aggregating α's personal impressions about β. β's reputation ($RP \in [0,1]$) is estimated by aggregating the shared impressions received by α from other delegators concerning β. Finally, β's know-how ($KH \in [0,1]$) is estimated by aggregating its references sent to α.

A set of impressions is aggregated into a summary value through Equation 21. This process is based on the impression aggregation approach proposed by (Sabater; Sierra, 2001). While the original model defines a structure for impressions derived from social evaluations, in our case, the impression format has been adapted to represent task-specific criteria. Nevertheless, the aggregation mechanism itself, including the use of a squashing function to normalize

and time-weigh impressions, follows the same fundamental principle. In particular, this squashing function considers that impressions produced more recently in a given set are more relevant than older impressions. This consideration allows for the possibility that an agent judged as not very competent to perform a task in the past could be selected as a delegatee in more recent interactions with its delegator, depending on its current behavior (Botelho *et al.*, 2018). Moreover, the squashing function ensures that the resulting value is in the interval $[0,1]$. The impression aggregation process is performed as follows:

$$Agg^t(BB^\alpha) = \begin{cases} \delta & \textit{if } |BB^\alpha(f^\alpha_\beta)| = 0 \\ 1 - \exp\left(-\rho * \sum_{\iota_i \in BB^\alpha(f^\alpha_\beta)} d^t_{\iota_i.t} * M(\iota_i.S)\right) & \textit{otherwise} \end{cases} \tag{21}$$

$$M(S : [(c_1, s_1), ..., (c_k, s_k)]) = \frac{\sum_{i=1}^{k} s_i}{|S|} \tag{22}$$

where δ is the default startup value returned when there are no impressions about a given partner (*e.g.*, assigning 1 to δ implies that, initially, an unknown partner β will be considered very competent in the execution of a task τ, while assigning 0 means expecting that β will not be capable of fulfilling its estimation about τ). $Agg^t(BB^\alpha)$ is the aggregated value for a set of impressions obtained at time t from α's belief base (BB^α) by applying the impression filter f^α_β, $|BB^\alpha(f^\alpha_\beta)|$ is the size of the set of impressions obtained from α's belief base, ρ is the factor that controls the curve slope of the squashing function, $d^t_{\iota_i.t}$ [2] a time-dependent function that decreases the relevance of an impression ι, produced at time $\iota.t$, concerning the current time t, and $|S|$ is the number of ratings for the task's criteria associated with an impression ι.

3.2 Success Likelihood

The success likelihood (*SL*$(\alpha, \beta, \tau) \in [0,1]$) can be directly computed by either analyzing the direct dependence between α and β (*i.e.*, β's success rate (SR)) or the transitive dependencies that form the delegation chains rooted in β (*i.e.*, β's accumulated success rate (ASR)). The choice of which approach to adopt in determining the success likelihood associated with a partner is modeled as a parameter of the delegation model.

[2] $d^t_{\iota_i.t} = \exp -\frac{t-\iota.t}{\xi}$ is an example of a time-dependent function in which the factor ξ can be configured to determine the speed of the decrement of the function (Botelho *et al.*, 2018).

3.2.1 Success Rate

The success rate (*SR*$(\alpha, \beta, \tau) \in [0,1]$) expresses how successful an agent β has been in performing a task τ over time from the perspective of a delegator α. Such a measure is estimated based on the direct dependence between α and β, and hence, β's success means accomplishing the goal g associated with the task τ delegated by α. Thus, when β's success rate is 0, it implies β has never accomplished g while performing τ, whereas a success rate of 1 means β has succeeded in achieving g every time it performed τ. The success rate is computed as follows:

$$SR(\alpha,\beta,\tau) = \begin{cases} \sigma & \textit{if } |(\beta,\tau)| = 0 \quad \textit{(Case 1)} \\ 1 - I_U & \textit{if } |(\beta,\tau)| \geq I_n \wedge |(\beta,\tau)^+| = 0 \quad \textit{(Case 2)} \\ 1 - \Delta_I(\beta,\tau) & \textit{if } |(\beta,\tau)| < I_n \wedge |(\beta,\tau)^+| = 0 \quad \textit{(Case 3)} \\ \frac{|(\beta,\tau)^+|}{|(\beta,\tau)|} * \Delta_I(\beta,\tau) & \textit{if } |(\beta,\tau)| < I_n \wedge |(\beta,\tau)^+| > 0 \quad \textit{(Case 4)} \\ \frac{|(\beta,\tau)^+|}{|(\beta,\tau)|} & \textit{otherwise} \quad \textit{(Case 5)} \end{cases} \tag{23}$$

$$\Delta_I(\beta,\tau) = I_U + (I_L - I_U) * \left(\frac{I_n - |(\beta,\tau)|}{I_n}\right)^2 \tag{24}$$

where $|(\beta,\tau)|$ denotes the number of times β performed τ, and $|(\beta,\tau)^+|$ represents the number of times β successfully performed τ. Given the unknown success probability distribution of a partner, a reliability factor ($\Delta_r \in [I_L, I_U] \mid I_L \leq I_U$) is introduced, representing α's uncertainty about β's behavior, which decreases as they interact with each other (Ashtiani; Azgomi, 2014). I_L is the reliability lower bound, I_U is the reliability upper bound, and I_n[3] is the number of interactions needed for α to accurately infer β's success rate. Possible cases for β's success rate are:

- *Case 1*: if β has never performed τ, its success rate is assumed to take a default startup value defined as σ (*e.g.*, assigning 1 to σ results in an optimistic initialization because unknown partners are initially considered trustworthy);
- *Case 2*: if α has already interacted with β sufficiently to evaluate its success rate accurately ($|(\beta,\tau)| \geq I_n$) and β was unable to complete τ at least once during its interactions with α ($|(\beta,\tau)^+| = 0$), then α can infer that β has a low probability of success in completing τ in future interactions. In this case, α assigns the lowest possible value

[3] As pointed out by (Sabater; Sierra, 2001), the value of the I_n parameter is application-dependent. It depends on both the frequency of interactions among agents and the number of evaluations they produce. In the case of the success rate, these evaluations refer to the number of times a task is performed.

to β's success rate, defined as the complement of the reliability upper bound ($1 - I_U$), which represents the worst-case scenario for a success rate assessment;

- *Case 3*: if α is unable to accurately infer β's success rate ($|(\beta, \tau)| < I_n$) and β has never successfully completed τ over its interactions with α ($|(\beta, \tau)^+| = 0$), then β's success rate tends to approach the worst-case scenario as it fails to complete τ ($1 - \Delta_I(\beta, \tau)$);
- *Case 4*: if α is unable to accurately infer β's success rate ($|(\beta, \tau)| < I_n$) and β succeeded in completing τ in at least one of its interactions with α ($|(\beta, \tau)^+| > 0$). Thus, in this case the success rate is defined as $\frac{|(\beta,\tau)^+|}{|(\beta,\tau)|}$, adjusted by the reliability factor.
- *Case 5*: if the number of interactions between α and β is sufficient for α to have a precise understanding of β's success rate and β succeeded in completing τ in at least one of its interactions with α, then the success rate is $\frac{|(\beta,\tau)^+|}{|(\beta,\tau)|}$.

3.2.2 Accumulated Success Rate

The accumulated success rate (*ASR*$(\Lambda = [\alpha.\tau_0, \beta_1.\tau_1, ..., \beta_{n-1}.\tau_{n-1}, \beta_n.\tau_n]) \in [0,1]$) is calculated based on a delegation chain Λ of size *n*, where α is the root of the delegation chain, β_1 is α's delegatee and the delegator of the second delegation instance, while β_{n-1} and β_n are, respectively, the delegator and a delegatee of the last delegation instance in the chain. In particular, the accumulated success rate represents the likelihood of success along a delegation chain, considering the individual success rate of all intermediary agents between the chain's root and a terminator delegatee (*i.e.*, the last delegatee of the chain). As the accumulated success rate associated with a delegation chain approaches 1, there is a higher probability that all agents along this chain will complete their tasks successfully, allowing the root to achieve its goal. In contrast, when the accumulated success rate approaches 0, there is a higher probability of an agent in the chain failing, preventing its predecessors from completing their tasks.

Unlike the success rate, which considers only the direct dependence between a delegator and its partners, the accumulated success rate also takes into account the transitive dependencies established along a delegation chain.

Definition 5. *A transitive social dependence happens when an agent α depends on β to achieve some goal g, and β depends on γ to accomplish another goal g', where g' is instrumental to g. In this case, α indirectly depends on γ to achieve its goal (Costa; Dimuro, 2006).*

The accumulated success rate for a delegation chain Λ is calculated through a recursive process, where the individual success rates of the agents along the chain are integrated into a

single accumulated value, as follows:

$$\textit{ASR}(\Lambda) = \begin{cases} \textit{SR}(\alpha_0,\alpha_0,\tau_1) & \textit{if } |\Lambda| = 1 \\ \textit{SR}(\alpha_0,\beta_1,\tau_1) & \textit{if } |\Lambda| = 2 \\ \textit{SR}(\beta_{n-1},\beta_n,\tau_n) * \textit{ASR}(\alpha_0.\tau_0,\beta_1.\tau_1,...,\beta_{n-1}.\tau_{n-1}) & \textit{if } |\Lambda| > 2 \end{cases} \quad (25)$$

where $|\Lambda|$ indicates the size of the delegation chain Λ, defined as number of Acts that composed the chain. When the size of a delegation chain Λ is 1, it represents a self-execution pattern, where the delegator α_0 is able to execute its task τ on its own. In this case, the success rate of α_0 as the delegatee of τ is estimated based on its performance as the executor of τ. The second case indicates a situation where a simple delegation instance is formed. Thus, α_0 delegates τ to β_1 who acts as delegatee, executing τ. The third case expresses how the accumulated success rate is computed, considering a complex delegation chain formed through several delegation instances.

The success rate and the accumulated success rate are performance metrics used to assess a partner concerning its successes in executing a task over time. Both measures can be directly applied in dependence relationships of type OR (single-task or multiple-task), where multiple partners are available for selection, but only one will be chosen as the delegatee. Nevertheless, for a multiparty relationship defined through an AND-dependence, several partners are selected simultaneously. Even though the outcome for a delegator's task is conditioned on the success of all delegatees, each one is individually assessed based on its performance after executing its sub-task. This feature does not affect how the success rate is computed for each delegatee, since the delegator is able to directly assess a delegatee. However, for the accumulated success rate, AND-dependencies require special attention. Each delegation chain rooted in a delegatee has its own accumulated success rate, which accumulates from the end of the chain back to the delegatee. Therefore, the accumulated success rate that a delegator with several delegatees propagates upward (*i.e.*, to the delegator of its delegation instance) must be calculated by taking into account the accumulated success rate of each delegation chain rooted in each of its delegatees. The accumulated success rate upward propagated by a given delegator regarding their delegatees is calculated as follows:

$$ASR([\alpha.\tau_0, \beta_1.\tau_1, ASR(\Lambda_1), \ldots, ASR(\Lambda_n)]) = SR(\alpha,\beta,\tau_0) * ASR(\Lambda_1) * ... * ASR(\Lambda_n) \quad (26)$$

where α is the delegator of a simple delegation instance in which β is its delegatee. β is the delegator of a composed delegation instance formed by n delegatees, to whom τ_1 was decomposed into sub-tasks. $ASR(\Lambda_1)$ is the accumulated success rate for the delegation chain rooted by the first delegatee of β, and $ASR(\Lambda_n)$ is the accumulated success rate for the delegation chain rooted by the last delegatee of β.

3.3 Offer's Utility

The Offer's utility (*UT*$(\alpha, \beta, \tau) \in [0,1]$) is a measure that determines the interest of a delegator α in selecting a partner β as the delegatee for a task τ based on β's offer. When the utility of β's offer is 0, it indicates that α has no interest in β's offer. Conversely, a value of 1 indicates that β's offer is highly relevant to α because it contains the best performance estimations for τ among all offers received from α's partners, considering α's preferences.

Definition 6. *The preferences of a delegator are represented as a vector* $P = [(c_1, p_1), ..., (c_n, p_n)] \mid \sum_{i=1}^{n} p_i = 1$*, where* $p_i \in [0,1]$ *indicates the relevance of* c_i *for a delegator.*

In general, upon receiving all offers for τ from its partners, α analyzes them to identify those that best match its preferences. In this analysis, α evaluates each offer based on the task's criteria, aiming to identify the partner with the best performance estimations according to its preferences for the task criteria. For example, for a task evaluated through two criteria, cost and time, and considering that for α the criterion time is more relevant than the criterion cost (*e.g.*, $P = [(cost, 0.1), (time, 0.9)]$), α will give preference to partners who sent offers where the performance estimation for the criterion time is the lowest possible, assuming time as a minimization criterion, and largely disregarding the estimation for criterion cost.

To determine the most suitable partner for task delegation based only on received offers, each partner's offer is evaluated across multiple criteria such as time, cost, and quality. Values assigned to each criterion are normalized relative to the best-observed values. For criteria where lower values are preferred (*e.g.*, time, cost), each value is normalized relative to the smallest observed value. Conversely, for criteria where higher values are desirable (*e.g.*, quality), each value is normalized relative to the highest observed value. After normalization, a score is calculated for each offer using a weighted average of the normalized values, where the delegator's preferences are employed as the weights. Therefore, the offer with the highest score is selected as the best because it most closely aligns with the delegator's preferences.

3.4 Failure Propagation

Due to transitive dependencies, an agent's failure can impact several other agents along a delegation chain (Lau; Singh; Tan, 2015). Generally, when an agent at a lower level of a delegation chain fails, this failure needs to be propagated to the higher levels of the chain. Failure propagation may prevent other agents from completing their tasks, resulting in a chain of failures. Different penalization strategies may be adopted to penalize the agents in case of failure (Burnett; Oren, 2012). For example, in a full penalization strategy, all agents along the chain are fully penalized as the failure propagates. On the other hand, partial penalization strategies consider the agents' positions in the chain to determine the degree of penalization for each agent.

In our model, agents penalize the failure of their delegatees through impressions. A *full penalization* applied to an impression means assigning a score of 0 to each task's criterion. In contrast, partial penalization depends on the agents' positions in the failure chain, denoted as the failure position ($F_{pos} \in [1, +\infty]$), where 1 is the failure position of the agent that caused the failure. Specifically, as the failure propagates upwards towards the chain root, the failure position of an agent is computed by increasing the failure position of its immediate delegatee by one unit. Thus, considering an impression $Imp = \langle \alpha, \beta, \tau, t, S \rangle$, the score assigned by α to β, in the event of β's failure, for a criterion c ($Imp(S[c])$) is defined as follows:

$$Imp(S[c]) = (1 - (\frac{1}{F_{pos}(\beta)})) * pd \tag{27}$$

where $pd \in [0,1]$ is the penalization discount factor, which defines the maximum score an agent can receive in case of failure (*i.e.*, the severity of the penalization). Note that 0 implies maximum penalization, while 1 means no penalization. In particular, we adopt the β's success rate as penalization discount factor, which makes the degree of penalization vary according to β's history of success (*i.e.*, the severity of penalization for a failure depends on the number of past failures committed by β while executing τ). By adopting the success rate as a penalization discount factor, we assume that agents who fail frequently will suffer more severe penalties. On the other hand, an occasional failure will not have a significant effect on the agent's penalization, as this type of failure has a substantial impact on the success rate of an agent.

In a delegation chain composed only of simple delegation instances, a failure propagates upward from one delegation instance to another. However, for a composed delegation, where failures may occur in different chains rooted in distinct delegatees of a delegator α, α is penalized based on its distance from the farthest delegatee who failed, as follows:

$$F_{pos}(\alpha) = argmax(F_{pos}(\beta_1), F_{pos}(\beta_2), ..., F_{pos}(\beta_n)) + 1 \tag{28}$$

where the $argmax$ returns the delegatee's position with the highest index in a failure chain (*i.e.*, the delegatee's index with the greatest difference between its position and the agent's position that produced the failure, propagated along its respective chain). The argmax function was adopted to minimize the penalization degree for agents closer to the root, as the number of chains branching from these agents tends to be greater than for those farther from the root. This approach maximizes the exploration potential for agents with a high branching degree, from which several delegation chains are rooted. In this way, even if a failure occurs in one of these chains, other chains may have a lower chance of failure.

3.5 Conclusion

In summary, this chapter introduced our delegation model, which differs from other approaches in the literature by incorporating both social and cognitive mechanisms into the partner

selection process. These include the agents' direct and transitive dependencies, their success history, and shared evaluations such as personal impressions, shared impressions, and references. Furthermore, it is important to highlight, as a central contribution of this chapter, the explicit support for sub-delegations through the modeling of delegation chains. These elements enable a more flexible and adaptive decision-making process in dynamic environments, as explored in the following chapters.

4 METHODOLOGY

The purpose of this chapter is to describe how our experiments were conducted. Specifically, this chapter is organized as follows:

- Section 4.1 discusses the configurations and constraints we adopted concerning the agents and the environment employed in our experiments;
- Section 4.2 defines the evaluation metrics used to assess the results produced by the agents as they delegate and execute tasks;
- Section 4.3 describes the DS-networks that model the agents' neighborhood (*i.e.*, their dependence relations), discussing the topology of each network and its effect on the performance of our delegation model;
- Section 4.4 describes how the individual parameters of our delegation model were adjusted for each evaluation scenario in the experiments.

4.1 Simulation Details

To assess our delegation model, we have designed a simulator capable of handling task delegation scenarios in which delegation chains are explicitly represented[1]. The agents and their dependence relations are represented through a DS-net, which is the input for the simulator[2]. It means that the agents, goals, tasks, sub-tasks and the dependence relations are not dynamically generated during a simulation. Instead, they are predefined and specified in the input files. Therefore, in the simulation, each delegator knows their goals and the different ways to achieve them, as described by their dependence relationships (*i.e.*, whether through self-execution of a task, direct task delegation to a partner from an OR-dependence relation, or task decomposition based on an AND-dependence relation). Nevertheless, they need to find the best combination of partners based on these relationships (*i.e.*, the partners that allow them to achieve their goals according to their expectations).

Representing the delegation chains through DS-nets allows us to reduce the complexity of our simulation process while extracting essential information directly from a DS-net. The DS-nets used in our experiments were manually designed to assess our delegation model's sensitivity to some parameters, such as network topology and agent connectivity, which will be discussed later in this section.

For a simulation run, agents delegate and execute tasks over a maximum number of iterations (MAX_{itr}) based on an input DS-net. In each iteration itr, as described in Algorithm 1,

[1] The simulator employed in this work was implemented in Jason (Bordini; Hübner, 2005), an interpreter for an extended version of AgentSpeak (Rao, 1996). The simulator can be accessed through a GitHub repository available in: https://github.com/jjbaqueta/scdModel/tree/main

[2] The files that describe the DS-nets used as input for our experiments can be accessed in this repository: https://github.com/jjbaqueta/scdModel/tree/main/DSNetExamples

a root agent initiates the offer request phase in a cascading manner (line 5). In this phase, delegators request offers from their partners in each delegation instance. These requests propagate through the network until they reach terminator agents, who respond directly since they don't rely on other agents. Offers then move upward through the network, allowing inner-level agents to create their own offers based on a composition process.

The composition process is discussed hereafter, but in general, it involves formulating an offer based on the performance estimations propagated with the offers sent by agents from lower levels (*e.g.*, in the case of a task that involves cost, propagating the cost of a task upwards from a delegatee to a delegator or propagating the mean cost of the tasks delegated to a set of delegatees in the case of task decomposition).

Algorithm 1 – Task delegation simulation

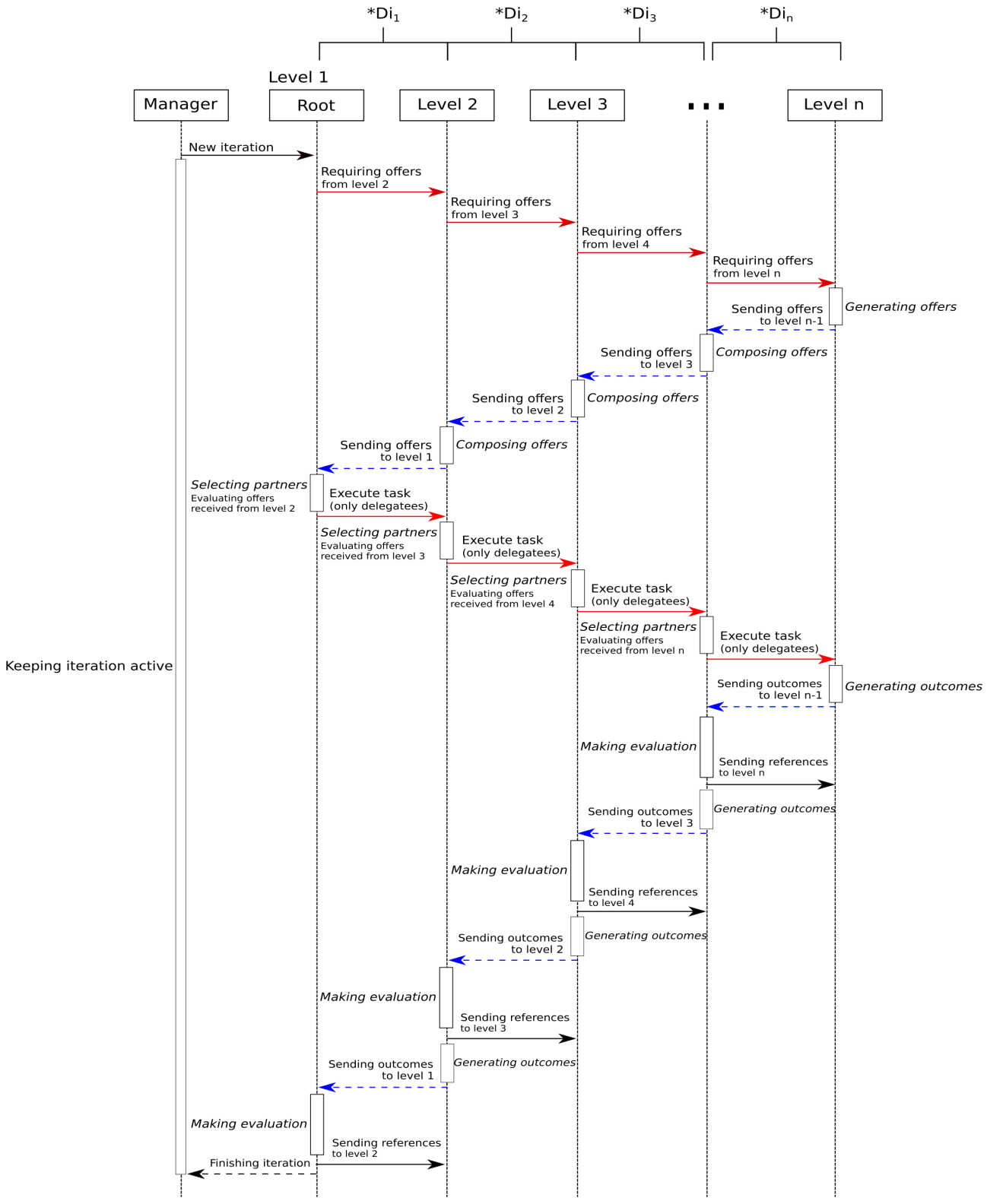
1: **Input:** DS-net structure, MAX_{itr}
2: Initialize root agent and other agents
3: $itr \leftarrow 1$
4: **while** $itr \leq MAX_{itr}$ **do**
5: **Offer request phase**
6: Root agent initiates offer requests
7: Propagate offer requests through delegation chains
8: **for** each terminator agent **do**
9: Answer the offer request with an offer
10: **end for**
11: **for** each agent in the inner levels of the network **do**
12: Compose offers based on received offers from lower levels
13: Propagate composed offers upwards
14: **end for**
15: **when** offers reach the root
16: **Partner selection phase**
17: Root agent selects delegatees
18: Propagate selection process downwards
19: **for** each delegator **do**
20: Select delegatees using the delegation model
21: Creation of simple and composed delegation instances
22: **end for**
23: **Execution phase**
24: Trigger task execution cascade
25: **for** each delegatee from terminator to root **do**
26: Execute tasks
27: Evaluate delegatees based on their offers and task outcomes
28: Update success rates (SR/ASR) and generate impressions
29: Share impressions with other delegators and the delegatee
30: **end for**
31: **end when**
32: $itr \leftarrow itr + 1$
33: **end while**

When offers reach the root, the partner selection phase begins (line 16), in which the root and subsequent delegators use our delegation model to choose their delegates, triggering a cascade of task executions (line 23). This chain of execution connects the root to termina-

tor agents, defining the task sequence for the root to achieve its goal. As tasks are completed, delegators evaluate their delegatees by comparing their performance estimations with their outcomes and updating the delegatees' success rates (SR/ASR) based on their success or failure in completing their tasks. An impression resulting from a delegator's evaluation is stored as part of its social image of the delegatee, shared with other delegators to contribute to the formation of the delegatee's reputation, and shared with the delegatee, serving as a reference and becoming part of the delegatee's know-how.

Besides the Algorithm 1, the task delegation stages are also illustrated in Figure 7 and Figure 8. Specifically, in Figure 7, the delegation chains created as the agents delegate task to each other are expressed through the delegation levels (Di_{i+1}). Note that each iteration starts with the manager, an agent that controls when an iteration starts and finishes. A new iteration starts when the manager signals the network's root to initiate the task delegation process. Moreover, a new iteration is initialized only if the iteration count is within the set limit (MAX_{itr}). In contrast, an iteration ends when the root evaluates its delegatees and signals the manager to conclude the current iteration.

On the other hand, Figure 8 illustrates the phases of an iteration in a three-level DS-net, showing each phase of task delegation while considering the timing (t_i) and sequence of message exchanges among agents. In particular, Figure 8 depicts the offer request phase from time t_0 to t_3, partner selection from t_4 to t_6, task execution from t_7 to t_9, and evaluation from t_{10} to t_{12}.



*Di: *Delegation instance*

Figure 7 – The flow of a simulation iteration represented through a sequence diagram. Source: Own authorship (2025).

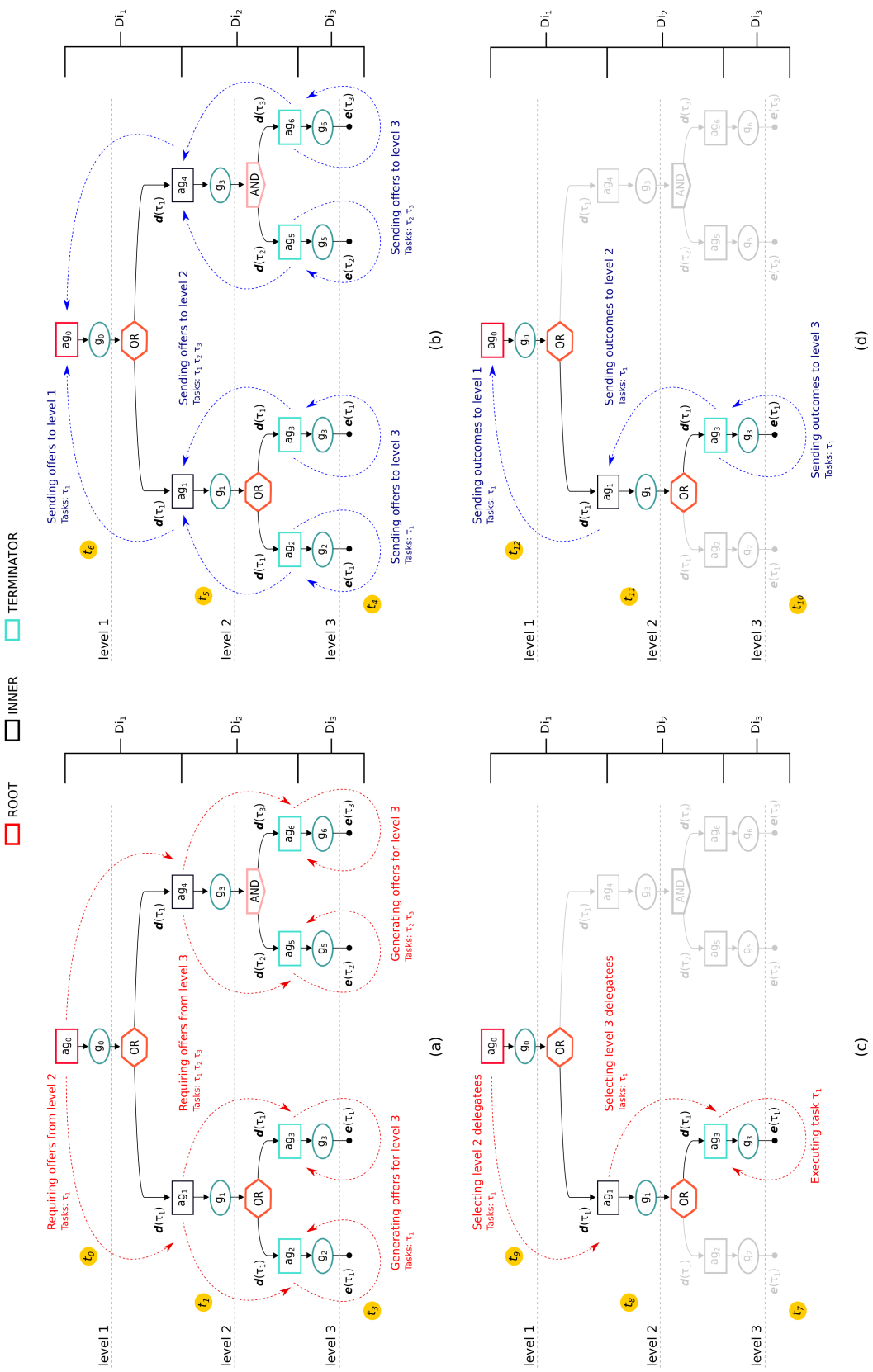


Figure 8 – The task delegation flow applied on a three-level DS-net: a) requesting offers, b) sending offers, c) partner selection, and d) task execution.

Source: Own authorship (2025).

4.2 Evaluation Metrics

We evaluate our delegation model based on the delegators' performance, as they are responsible for initiating the delegation process and relying on other agents to accomplish their goals. Note that a delegation chain may include several delegators, one for each delegation instance that composes the chain. Thus, considering an iteration (itr), we define the following:

- $Dl = \{\alpha_1, ..., \alpha_n\}$ is the set of all delegators of an iteration itr obtained from the Act_{itr}, while $|Dl|$ denotes the size of the set of delegators.
- Given a delegator α_i, its delegatees for a task τ_i are defined by $Delegatees(\alpha_i, \tau_i)$[3].

The following evaluation metrics are considered during our experiments to assess the performance of our delegation model:

Delegators' success rate average ($SR(Dl)_{itr} \in [0,1]$): this measure is calculated by averaging the success rates of all delegators in the set Dl, representing the number of tasks delegated by the delegators that were successfully completed up to iteration itr:

$$SR(Dl)_{itr} = \frac{\sum_{i=1}^{|Dl|} (SR(\alpha_i, \tau_i))}{|Dl|} \tag{29}$$

$$SR(\alpha, \tau) = \begin{cases} 0 & \textit{if } |(\alpha, \tau)| = 0 \\ \frac{|(\alpha,\tau)^+|}{|(\alpha,\tau)|} & \textit{otherwise} \end{cases} \tag{30}$$

where $SR(\alpha, \tau)$ represents the success rate of a delegator α concerning the task τ (*i.e.*, the number of times the τ was completed successfully ($|(\alpha, \tau)^+|$) concerning the number of times α delegated τ ($|(\alpha, \tau)|$)). In particular, the success of α regarding the execution of τ relies on the success of its delegatees, as follows:

$$f_{succ}(\alpha, \tau) = \begin{cases} true & \textit{if } \forall \beta_j \in Delegatees(\alpha, \tau), f_{succ}(\beta_j, \tau_j) = true \\ false & \textit{otherwise} \end{cases} \tag{31}$$

where, The function $f_{succ}(ag, \tau)$ is a function that returns *true* if an agent ag succeeds in completing its task τ, producing an outcome for τ.

Delegators' satisfaction average ($SAT(Dl)_{itr} \in [0,1]$): this measure is calculated by averaging the satisfaction of all delegators in the set Dl concerning their choice of delegatees

[3] We assume that a delegator α must have at least one delegatee, as each delegation instance requires both a delegator and at least one delegatee.

up to iteration itr. For a delegator α, satisfaction is a performance measure that indicates α's degree of satisfaction with the performance of its delegatees, considering both their performance estimations and the outcomes obtained after the execution of their tasks:

$$SAT(Dl)_{itr} = \frac{\sum_{i=1}^{|Dl|}(SAT(\alpha_i, \tau_i))}{|Dl|} \tag{32}$$

$$SAT(\alpha, \tau) = \begin{cases} 0 & \textit{if } |(\alpha, \tau)| = 0 \\ \frac{SAT_{acc}(\alpha,\tau)}{|(\alpha,\tau)|} & \textit{otherwise} \end{cases} \tag{33}$$

where $SAT_{acc}(\alpha, \tau)$ represents the accumulated satisfaction that α obtained by delegating τ to other agents over time (*i.e.*, $|(\alpha, \tau)|$ times). Additionally, the satisfaction of α regarding a task τ executed by a delegatee β, denoted by $SAT(\alpha, \beta, \tau)$, is computed similarly to the score computation shown in Equation 20. However, in this case, the individual scores estimated for τ's criteria (C) are combined into a single measure of satisfaction through a simple mean, such as follows:

$$SAT(\alpha, \beta, \tau) = \frac{\sum_{i=1}^{|C|} S[c_i]}{|C|} \tag{34}$$

where each criterion c_i belongs to the set $C = \{c_1, \ldots, c_n\}$, which is associated with τ, and $|C|$ denotes the size of the set C. In the case of task decomposition, α's satisfaction concerning a task τ is computed as the average of the satisfactions obtained from the execution of each sub-task τ_j performed by a delegatee $\beta_j \in Delegatees(\alpha, \tau)$.

Delegators' regret average ($REG(Dl)_{itr} \in [0,1]$): this measure represents the average regret of all delegators in the set Dl regarding their choice of delegatees up to iteration itr:

$$REG(Dl)_{itr} = \frac{\sum_{i=1}^{|Dl|}(REG(\alpha_i, \tau_i))}{|Dl|} \tag{35}$$

$$REG(\alpha, \tau) = MSAT(\alpha, \tau) - SAT(\alpha, \tau) \tag{36}$$

where, the regret of α concerning the execution of τ, $REG(\alpha, \tau)$, is defined as the difference between the satisfaction that could be obtained if the best possible combination of delegatees were selected by α to execute τ, which is denoted as maximum satisfaction ($MSAT$), and the satisfaction obtained by α with its current choice of delegatees. To compute $MSAT$, each partner sends an offer that includes estimated performance values as well as the actual outcome of the task after execution. Although the delegator selects its delegatees based only on the offers, the availability of actual outcomes from all potential partners (including those not selected) allows α to retrospectively identify which combination of delegatees would have yielded the highest satisfaction. The dif-

ference between this optimal satisfaction and the satisfaction resulting from the actual selection corresponds to the regret. A positive regret value indicates that the chosen set of delegatees was suboptimal.

We remark that delegators' success rate average allows us to evaluate our experiments based on the number of tasks successfully completed by the agents, regardless of the quality of the tasks performed by them (*i.e.*, the capability of a delegatee to execute a task according to its performance estimations). Conversely, the delegators' satisfaction and regret averages reflect the delegatees' ability to fulfill their performance estimations. Thus, by analyzing these metrics together, we can identify partners capable of successfully completing their tasks without disregarding the expectations of the delegators concerning the execution of the tasks.

4.3 General Assumptions

The following assumptions are taken during a simulation:

- All tasks have two evaluation criteria: the cost required for an agent to perform the task and the time to complete it. Each partner uses these criteria to estimate their performance regarding a task before sending an offer to their delegator. The partners' performance estimations for the cost and time criteria are generated randomly within predefined numerical intervals. For example, if a delegator intends to have a task completed within a period of 1 to 10 days, its partners will make their performance estimations to fit within this range. However, task execution does not necessarily occur within the estimated time. For instance, a delegatee might take 20 days to complete a task that was estimated to take 10 days. For convenience, we assume that all n task criteria associated with a task are equally relevant for a delegator α (*i.e.*, $p_i = \frac{1}{n} \forall (c_i, p_i) \in P(\alpha)$ and $1 \leq i \leq n$).
- When an agent receives a task from a delegator, it can delegate the task onward through recursive delegation, decompose it into sub-tasks and then delegate them, or execute the task by itself. The first two cases are delegation actions, and the last corresponds to an execution action. The recursive delegations and task decomposition are configured beforehand in the input DS-net.
- An agent executes all its tasks in parallel. This occurs when the agent is simultaneously involved in multiple delegation chains, enabling it to respond to requests from distinct delegators at the same time.
- When a partner receives a task from a delegator, its objective is to complete it through either delegation or execution actions. Therefore, the partner's goal will only be achieved if the task is successfully completed.
- Two agents can only exchange social evaluations about their common partners (*i.e.*, social evaluations can only be shared among delegators who assess the same partner).

- For simulation efficiency, when a partner sends an offer to a delegator (during the offer request phase), the partner includes the task's simulated outcome for the task. It is more efficient for all candidate partners to simulate the task execution than to introduce new simulation cycles to announce the winning partners who should simulate and backpropagate the task results upwards. We remark that outcomes are not considered during partner selection. As discussed previously, a delegator selects its delegatees based on their performance estimations, success history, and competencies. For experimental purposes, by knowing the outcome of each partner, a delegator can calculate the difference between its satisfaction with a delegatee and the satisfaction that could have been obtained if the best partner had been selected (*i.e.*, the partner that achieved the best outcome).
- Generating offers for agents that act as intermediaries, connecting their delegatees to their delegators, requires particular attention, as the offers produced by such agents depend on those received from their delegatees. Specifically, the offers generated by an intermediate agent are built based on a composition process, which is illustrated in Figure 9. Note that, in a simple delegation instance, such as shown in Figure 9 (a), an intermediate agent (β_i) propagates the cost and time from its delegatee's offer (β_{i+1}), with the time being increased by one unit to simulate the communication time. Conversely, in a composed delegation instance (Figure 9 (b)), the cost is calculated as the sum of the estimated costs from β_i's delegatees, while the time is defined based on the estimated time provided by the delegatee, who will take the longest to complete its task.

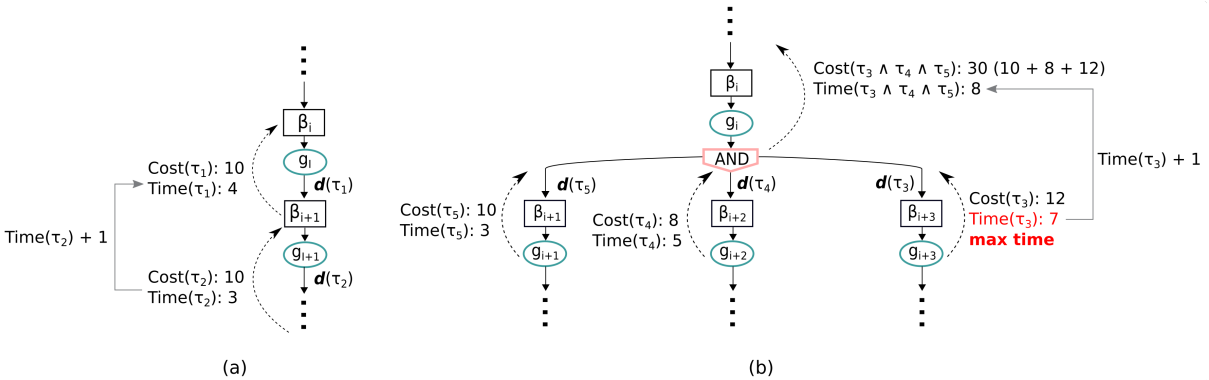


Figure 9 – Offer composition process considering the time and cost of tasks: (a) simple delegation instance and (b) composed delegation instance.
Source: Own authorship (2025).

4.4 Network Topologies

In our experiments, each scenario is represented through a DS-net comprising 85 agents. Each DS-net has a distinct topology that influences how agents delegate tasks, establish connections, communicate with others, and determine how evaluations are shared and propagated.

4.4.1 Network Characteristics

In our experiments, we consider two network topologies:

Tree network topology: the nodes (agents) are organized in a parent-child relationship of precedence (Afanador; Baptista; Oren, 2019). As shown in Figure 10, such a network was designed with four levels, each of size 4. Each agent at level l has only one delegator at level $l-1$, except for the root agent, which does not have a delegator, and four partners at level $l+1$, excluding the terminator agents. Note that such a configuration produces a rigid network structure with a well-defined delegation hierarchy, where the hierarchy levels limit agent communication.

Random network topology: traditionally, a random network is composed of **N** nodes (agents), where each pair of nodes is connected with probability **p** (Barabási; Albert, 1999). In order to adapt the classical Barabási e Albert (1999) network model to the requirements of our delegation scenario, some structural modifications were applied. First, a root agent was added to the generated network to serve as the entry point for delegation chains. This root agent is connected to all agents at level 0 of the original network. Then, to ensure acyclicity and directionality, the network is traversed from the root using a breadth-first search (BFS), and only the reachable edges that preserve a top-down delegation flow are maintained. The resulting structure is a directed acyclic graph (DAG), which is used as the base for building the DS-net, which is shown in Figure 11.

In our case, the level of the agents within the network determines the connection probability. As depicted in Figure 11, the random topology has six levels. The first level contains the root agent, the second level has ten agents, the third, fourth, and fifth levels have 20 agents each, and the last level has 14 agents. Except for the first and last levels, an agent at level l may connect with up to 5 delegators from level $l-1$ and with up to 5 partners at level $l+1$. This network structure allows greater interconnectivity between levels, as the connections among the agents are determined by a probability. The high connectivity among the agents and multiple communication paths facilitate the propagation of information.

Note that in Figure 10 and Figure 11, the agents may perform behavioral changes, adopting different behaviors throughout the iterations. These changes occur progressively over a

simulation. The agents' behavioral changes are introduced to add a dynamic component to our experiments, allowing us to understand how different partner selection strategies are affected by such changes.

The behavioral changes are discussed in more detail in the next chapter. However, they are associated with the agents' ability to accurately estimate their performance concerning a task and their capability to successfully complete a task. Specifically, an agent with good behavior is guaranteed to estimate its performance accurately and successfully complete tasks (nodes in Figure 10 and Figure 11 with green and black borders). On the other hand, agents with bad behavior are unable to make precise performance estimates and tend to fail in their tasks (nodes in Figure 10 and Figure 11 with red borders).

It is important to note that, throughout the iterations, agents with good behavior may adopt bad behavior and vice versa, a feature we refer to as *behavioral inversion*. Despite this inversion, note that in the networks presented in Figure 10 and Figure 11, there is always a path connecting the root agent (agent with ID 0) to the terminator agents (leaf nodes of the network). These paths need to be discovered by the delegators as tasks are delegated and agents interact with each other, incorporating behavioral changes into this learning process.

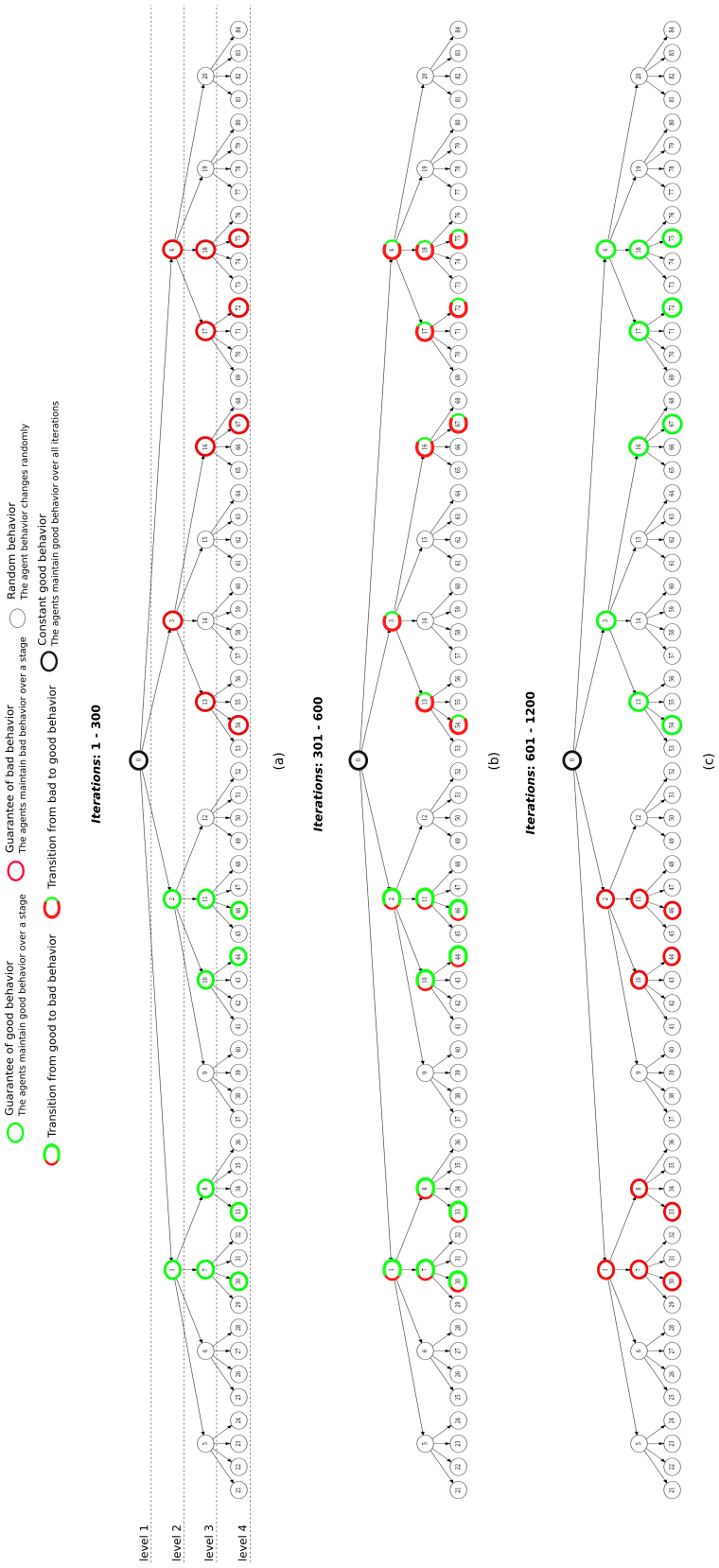


Figure 10 – Agents' behavior changes over simulation stages considering the tree-topology network: (a) agents' behavior configuration for stage 1 of the simulation, (b) Agents' behavior configuration for stage 2 of the simulation, and (c) Agents' behavior configuration for stage 3 of the simulation.

Source: Own authorship (2025).

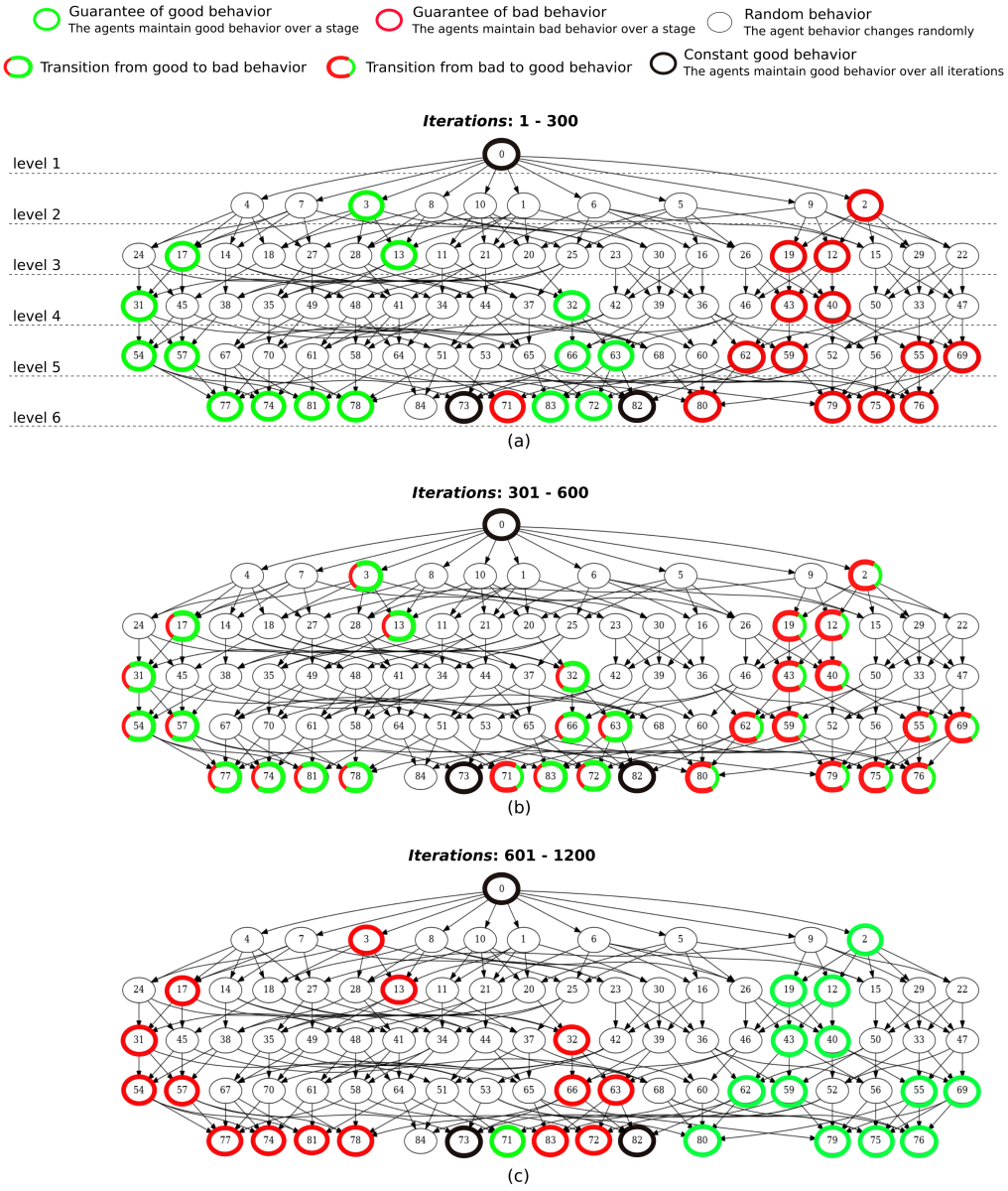


Figure 11 – Agents' behavior changes over simulation stages considering the random-topology network: (a) agents' behavior configuration for stage 1 of the simulation, (b) Agents' behavior configuration for stage 2 of the simulation, and (c) Agents' behavior configuration for stage 3 of the simulation.

Source: Own authorship (2025).

4.4.2 Connectivity Factor

We chose the tree and random topologies to illustrate how our delegation model benefits from the propagation of social evaluations among agents. In particular, the computation of the accumulated success rate and the spreading of social evaluations take advantage of the connectivity among agents (*i.e.*, the number of partners and delegators each agent has). In this sense, we chose the tree topology to represent a task delegation situation characterized by low connectivity among agents, describing the worst-case scenario for information propagation. On the other hand, the random network topology represents a situation in which agents' connections facilitate the exchange and propagation of information among the agents, considering the increase in the number of delegators per agent when compared to the tree network topology.

In Figure 12, the propagation of information is illustrated, showing how it can be influenced by the way agents are connected. The delegation chain formation and evaluation propagation processes are depicted through the DS-nets in Figure 12. Delegation instances are formed sequentially as a delegator selects its delegatees, indicated by the time markers in green. In contrast, the time markers in orange represent when updates of the success rate, accumulated success rate, or impressions are propagated (dotted links).

Specifically, Figure 12 (a) and Figure 12 (b) show the propagation of the success rate (SR) and the accumulated success rate (ASR) through a DS-net, which are triggered by the execution of the task τ_4 in the time instant t_3. Note that the delegation instances selected in both cases are composed of the same agents, ag_3 and ag_5 (t_1), and then ag_5 and ag_6 (t_2). The update of the success rate of the agents in Figure 12 (a) starting from instant t_4, affecting only the agents in the delegation chain (*i.e.*, agents that are part of the delegation instances). The agents in the greyed-out part of Figure 12 (a) are not part of the delegation chain and, hence, do not receive updates. In contrast, when we consider the accumulated success rate, several other updates are performed, as presented in Figure 12 (b). This behavior is a consequence of the recursive property of our approach that accumulates the success rate of the agents along the delegation chains, associating a success probability to each chain as a whole. In this case, as agent ag_6 belongs to three different chains, an update of its success rate affects the success probability of all delegation chains in which it takes part.

On the other hand, Figure 12 (c) and Figure 12 (d) present the mechanisms of propagation of impressions. Figure 12 (c) presents a DS-net designed from a tree network topology. In this case, as each agent has only one delegator, the sharing of impressions among agents is restricted to the agents of a delegation instance (delegator and delegatees). Therefore, the impressions cannot be shared with agents outside the delegation chain (*i.e.*, agents in the greyed-out part of Figure 12 (c)). Conversely, when the number of delegators per agent increases, as presented in the DS-net in Figure 12 (d), the delegators' capabilities for propagating their personal impressions become wider because delegators that assess the same delegatee can exchange their impressions about it (shared impressions), regardless of whether these delegators are part of the delegation chain.

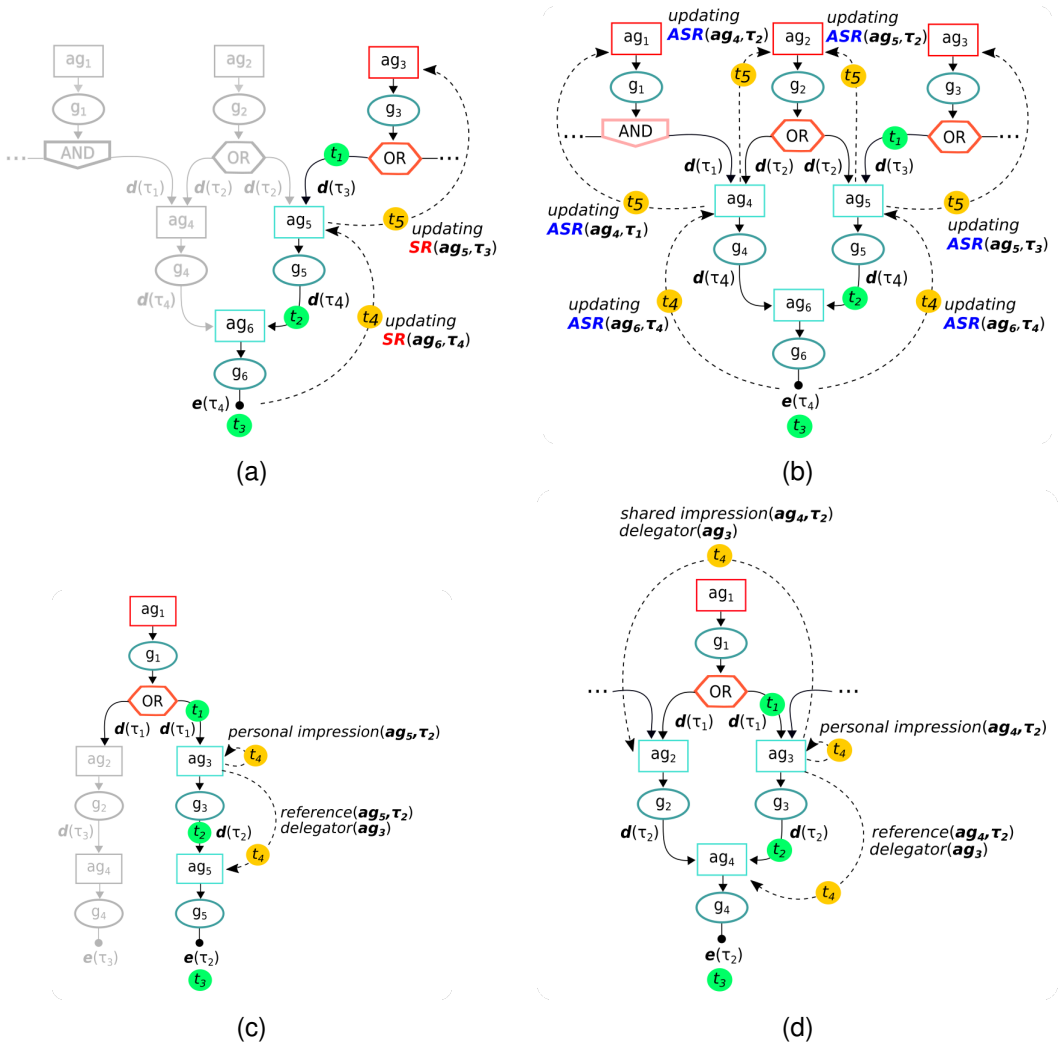


Figure 12 – Mechanisms of information propagation: (a) propagation of the success rate considering only the agents' direct dependencies, (b) propagation of the accumulated success rate considering both direct and transitive dependencies established by the agents, (c) propagation of social evaluations assuming each delegatee has only one assessor (delegator), and (d) propagation of social evaluations assuming each delegatee has more than one assessor (delegators).
Source: Own authorship (2025).

4.5 Experimental Scenarios

For each network topology, we simulated the task delegation process through three distinct experimental scenarios. In Table 3, the weights associated with each evaluation dimension considered in our delegation model are discriminated, taking into account the experimental scenarios. Furthermore, we remark that some dimensions are not considered in certain scenarios according to the purpose of the experiment, as discussed in the specification of scenarios presented hereafter.

A particularity of the experimental scenarios is that the delegators can use their preferences to select partners, taking into account the utility of their partners' offers for a task. The selection of partners based on their offers is useful when there are no other sources of informa-

tion available for a delegator to determine its delegatees, such as social evaluations or historical information about the partners. In particular, the lack of social evaluation and historical information often occurs during the system's initialization, when the delegators have not yet interacted with their partners. Nevertheless, as the number of interactions between the delegators and their partners increases, this approach becomes less reliable since the performance estimations of a partner concerning a task offer no guarantees about its actual performance while executing the task, considering that the partner may commit estimation errors.

In this context, in order to reduce the chance of a partner being selected as a delegatee based on erroneous performance estimates, we assigned a lower weight to the offer's utility dimension UT than to the trust measure dimension (TM) in our experiments (*i.e.*, assigning 0.1 to W_{DE} and 0.9 to W_{TM}). In this way, the delegator's preferences do not significantly affect its decision to trust when other sources of information are available, such as social image, reputation, references, or success history.

A brief description about the experimental scenarios is presented as follows:

Scenario 1 - Mono-episodic selection: in our baseline scenario, we configure our delegation model so that delegators only take into account the success rate of their partners during the decision to trust, disregarding the partners' competencies (*i.e.*, by assigning 1 to W_{SL} and 0 to W_{CM}). Furthermore, all agents along a failure chain are equally penalized in case of failure propagation because, in a mono-episodic delegation instance, a delegator sees its delegatees as terminator agents (agents capable of executing tasks directly). This configuration allows us to evaluate the partner selection process in a scenario where partners are chosen solely based on delegators' direct experiences, disregarding any information about the delegation chains formed through sub-delegations (Griffiths, 2005) (Castelfranchi; Falcone, 2010) (Cho; Chan; Adali, 2015).

Scenario 2 - Delegation chain selection: In this scenario, we replace the success rate mechanism from our baseline scenario with the accumulated success rate. However, we continue to adopt the full-penalization strategy in case of failure propagation. This configuration allows delegators to select their partners based solely on the transitive de-

Table 3 – Configuration of the delegation model parameters, considering the weights (W) of evaluation dimensions and the experimental scenarios, where SI is the partner's social image, RP is the partner's reputation, KH is the partner's know-how, CM is the competence measure, SL is the success likelihood, TM is the trust measure, and UT is the offer's utility.

Weights	Mono-episodic selection	Delegation chain selection	Competence-oriented selection
W_{SI}	-	-	0.33
W_{RP}	-	-	0.33
W_{KH}	-	-	0.33
W_{CM}	0.0	0.0	0.5
W_{SL}	1.0 (using SR)	1.0 (using ASR)	0.5 (using ASR)
W_{TM}	0.9	0.9	0.9
W_{UT}	0.1	0.1	0.1

pendencies established by agents along a delegation chain. This modification takes advantage of the network topology and the agents' connections. By employing the accumulated success rate approach, the agents' success rates are cumulatively aggregated to generate a success probability for a path (delegation chain) connecting a root with a terminal node in the network. Thus, in this approach, instead of using the individual success rate of the partners as a selection metric, delegatees are chosen based on the success probability associated with the delegation chain rooted in them.

Scenario 3 - Competence-oriented selection: in this scenario, agents are evaluated based on their accumulated success rates and competencies. This configuration allows delegators to assess their delegatees from a more qualitative perspective, considering both their partners' accumulated success rate along a chain and their satisfaction with past choices. This is achieved by an equally weighted distribution of the model dimensions associated with the competencies and success rate of the agents (*i.e.*, assigning 0.33 to W_{SI}, 0.33 to W_{RP}, 0.33 to W_{KH}, 0.5 to W_{CM}, and 0.5 to W_{SL}). Additionally, in this scenario, agents are partially penalized in case of failure propagation. This penalization is applied based on the agents' positions in the failure chain.

It is important to remark that we do not consider the agents' competencies during the partner selection for the mono-episodic and delegation chain scenarios (evaluation scenarios 1 and 2, respectively). In these scenarios, we aim to evaluate the agents' performance by using the success rate as a partner selection mechanism compared to its cumulative version. In both scenarios, agents tend to make decisions that maximize their success rate as delegators.

4.6 Conclusion

In this chapter, we presented the simulator developed specifically to evaluate the proposed delegation model, which is built upon the DS-net structure. The simulator supports the execution of complex scenarios involving multiple agents, social evaluation mechanisms, delegation chains, and configurable network topologies. This tool serves as the foundation for our experimental studies.

5 RESULTS

This chapter presents the experimental results and is organized as follows:

- Section 5.1 discusses the characterization of the agents and the factors that affect their performance while executing a task.
- Section 5.2 discusses the dynamic elements considered in our experiments, specifically the behavioral changes of the agents.
- Section 5.3 introduces the MAB policy that was adopted in our experiments.
- Section 5.4 presents the experimental results based on the scenarios described in the previous chapter.
- Section 5.5 discusses how the results validate the hypothesis of this thesis.

5.1 Agents' Characterization

In our experiments, tasks are considered abstract because the specific task type details are irrelevant to a delegator's partner selection. Our delegation model only requires the delegator to know which tasks each partner can perform. For example, a task can be decomposed into sub-tasks and delegated to different partners, with each sub-task treated as distinct. If a delegator sub-delegates a task onward, the task itself remains unchanged. Therefore, we assume all agents have the same capabilities to perform any task. However, an agent's performance depends on its *failure likelihood* and *accuracy estimation level*.

Failure likelihood: Defined as a random value representing the probability of a delegatee failing to perform a task. This value ranges from [0,1], where 0 means no chance of failure and 1 means inevitable failure. Failures can occur in two ways. First, the delegatee fails to complete an execution action, preventing it from producing an outcome for its task. Second, the delegatee cannot communicate the task outcome to its delegator. For instance, in task decomposition, a failure may prevent a delegatee from aggregating sub-task outcomes into a single result. Thus, even if all sub-tasks are successfully completed, the delegatee may still fail to deliver the composed task's outcome to its delegator.

Estimation accuracy: Represents a partner's ability to accurately estimate its performance based on task criteria. We consider six accuracy levels, each corresponding to a distinct error range, as shown in Table 4. The error is a random value that depends on the accuracy level. Higher estimation errors result in lower satisfaction degree that the partner can provide to its delegator.

Accuracy level	Error interval
VERY LOW	[0.8, 1.0[
LOW	[0.6, 0.8[
MIDDLE	[0.4, 0.6[
HIGH	[0.2, 0.4[
VERY HIGH	]0, 0.2[
PERFECT	[0, 0]

Table 4 – Correspondence between accuracy levels and their respective error intervals.

An example of how estimation accuracy and failure likelihood are used in agent characterization is shown in Figure 13. Agents' relationships are represented through a DS-net structure. Except for the root, all agents are characterized by these factors. Each agent has a profile for task delivery and execution. The execution profile includes an estimation accuracy level and a failure likelihood for task execution, while the delivery profile only specifies the failure likelihood for delivering a task outcome. Thus, agent ag_7 has a low execution failure likelihood (5%) and no failure in delivering task outcomes. However, since its estimation accuracy level is *MIDDLE*, it may commit estimation errors ranging from 40% to 60% for each task criterion based on its actual outcome. On the other hand, agent ag_3 has a *LOW* estimation accuracy level, making errors between 60% and 80%. Additionally, it has an intermediate failure likelihood (50%) for both execution and delivery, making it a less reliable partner.

Estimation accuracy and failure likelihood influence the trust a delegator places in their partners over time. Partners with high precision provide more reliable performance estimates, leading to higher satisfaction from delegators regarding task execution. In contrast, low precision and high failure probabilities introduce uncertainty and increase the risk associated with delegation. The characterization of agents is integrated into the DS-net structure using an XML file format, which is discussed in detail in Appendix 6. This XML format enables the definition of a DS-net using a hierarchical structure and allows for customization of the network to incorporate agents' delivery and execution profiles. Moreover, the DS-nets used as input for our experiments follow this XML format.

5.2 Behavioral Changes

Aiming to ensure a dynamic behavior for partners, their failure likelihood and estimation accuracy levels may change during the simulation through *upgrades* and *downgrades*. An upgrade improves a partner's characteristics by decreasing its failure likelihood or increasing its estimation accuracy, whereas a downgrade deteriorates these characteristics (*i.e.*, increasing the probability of failure and decreasing the estimation accuracy level).

Failure likelihood changes: When a partner receives an upgrade or downgrade, the simulator randomly assigns a new failure likelihood based on predefined intervals:

- *Interval 1*: $[0, 0.2[$ (low failure likelihood);
- *Interval 2*: $[0.2, 0.4[$ (moderate failure likelihood);
- *Interval 3*: $[0.4, 0.6]$ (high failure likelihood);

An upgrade moves the partner to a lower interval, reducing its failure probability. For example, if a partner is in *Interval 2* ($[0.2,0.4[$), an upgrade shifts it to *Interval 1* ($]0,0.2[$). In turn, a downgrade moves the partner to a higher interval, increasing its failure probability. Within each interval, the simulator randomly selects a new failure likelihood using a uniform distribution.

Estimation accuracy changes: An upgrade may improve a partner's estimation accuracy level. The new level is determined by a uniform distribution, considering the partner's current level and up to two higher levels. Similarly, a downgrade may decrease accuracy by up to two levels.

Behavioral changes occur gradually between simulation stages. As illustrated in Figure 14 and Figure 15, a simulation run consists of 1200 iterations, divided into three stages: *Stage 1* (1–300), *Stage 2* (301–600), and *Stage 3* (601–1200). These changes occur twice, at iterations 300 and 600, before a new offer request phase begins. In particular, this subdivision aims to introduce behavioral changes over the simulation and to assess the performance of the delegation model in response to both the initial adaptations and the subsequent stabilization of

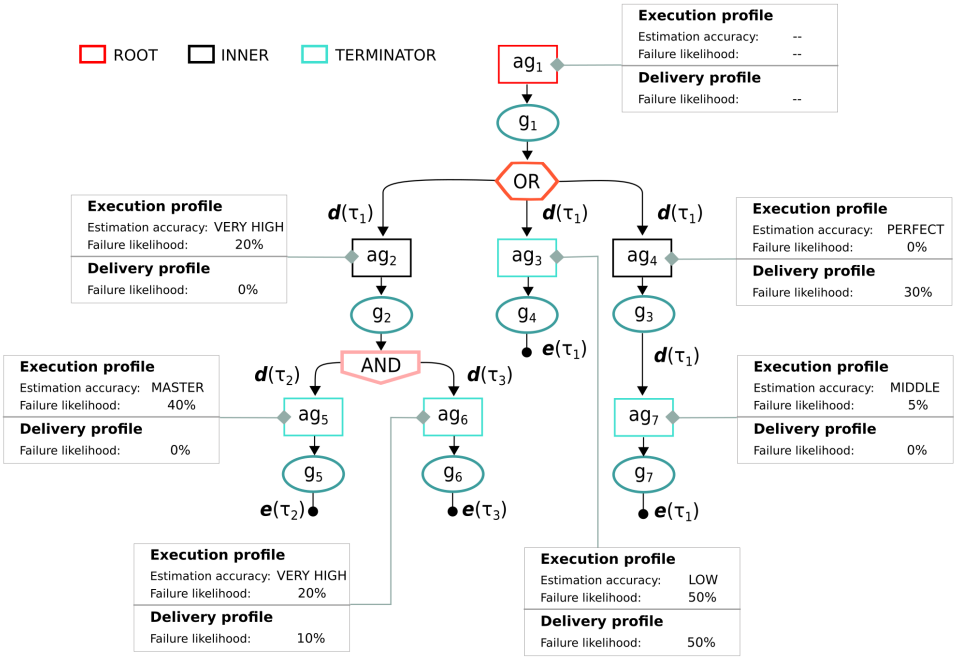


Figure 13 – Assignment of failure likelihoods and estimation accuracy levels for a set of agents through a DS-Net.
Source: Own authorship (2025).

agent behaviors. In each iteration, agents follow the routine described in Algorithm 1 (Section 4), continuously executing and delegating tasks. Although these actions are performed repeatedly, updates on efficiency metrics (*i.e.*, partners' success rates and delegators's satisfaction and regret) are maintained across iterations, allowing delegators to learn about their partners' behavior over time. Furthemore, the partners' behavioral changes affect their delegators. A partner may start with good behavior, successfully completing tasks and meeting performance expectations. However, due to these changes, it may end the simulation performing poorly, failing tasks, and disappointing its delegator concering the task execution. In contrast, an initially unreliable partner may improve over time, becoming more consistent and trustworthy by the end of the simulation.

5.3 MAB Policy

Despite the existence of several other MAB policies in the literature, such as UCB1 (Garivier; Moulines, 2011) and Thompson sampling (Chapelle; Li, 2011), we adopt the ϵ-*greedy* policy due to its simplicity. It provides an effective exploitation/exploration mechanism without the complexity of more advanced policies. Specifically, we use a variation of the ϵ-*greedy* policy, where the parameter ϵ depends on the number of times a delegator α acts on a task τ ($|(\alpha,\tau)|$) (Artemis, 2021), as defined below:

$$\epsilon = \frac{1}{|(\alpha,\tau)| + 1} \tag{37}$$

where the probability of exploration increases as ϵ approaches 1, meaning the delegator is more likely to select a partner at random. Otherwise, α exploits its known options by selecting partners based on their delegation likelihood. The higher a partner's delegation likelihood, the more likely it is to be chosen by α.

5.4 Results

For each experimental scenario, we conducted two experiments, A and B. Experiment A uses the DS-net with a tree network topology, while Experiment B employs the DS-net with a random network topology. The results of each experiment are obtained by averaging the performance indicator values from 5 simulation runs. These performance indicators are based on the evaluation metrics defined in Chapter 4 (*i.e.,* delegators' success rate average ($SR(Dl)itr$), delegators' satisfaction average ($SAT(Dl)itr$), delegators' regret average ($REG(Dl)itr$), and delegators' maximum satisfaction ($SATmax$))[1]. Such an evaluation approach was chosen to account for the random nature of delegators' partner selections over time, balancing the exploitation of known partners and the exploration of new ones. Averaging multiple runs mitigates the randomness associated with partner selection and the performance estimation process.

[1] The results from each of the five simulation runs can be accessed at the following link: https://github.com/jjbaqueta/scdModel/wiki/Experimental-Results.

5.4.1 Behavioral Changes Evaluation

We emphasize that during the simulation, from *stage 2* onwards, agents may adopt a different behavior compared to the previous stage, as shown in Figure 14 and Figure 15, which illustrate the evolution of agents' behavior over iterations for Experiment A and Experiment B, respectively. For each new stage, delegators must relearn which agents are the most promising partners due to the upgrades and downgrades. The time a delegator takes to learn about their partners' behavior is called *learning time*.

The behavioral differences observed in Figure 14 and Figure 15 arise from the presence of optimal paths formed by high-performing agents. Since network topologies differ, delegators may follow different paths when selecting partners. Throughout a stage, they must identify the best partners, finding the most efficient path. However, when a stage transition occurs, this path might change as upgrades or downgrades are performed. Factors like network depth and breadth also influence agent relationships and path formation. Despite these differences, both figures (Figure 14 and Figure 15) show a consistent pattern. Agents that performed well in the first stage tend to degrade over time, while initially poor-performing agents tend to improve. This behavioral inversion leads to a higher concentration of agents with estimation accuracy levels between *LOW* and *HIGH* in *stage 2*.

Regarding failure likelihood, the figures show that, in *stage 1*, agents exhibit a wide range of failure likelihood values, with some reaching high probabilities. As the simulation progresses, failure likelihood stabilizes (*stage 2*). In turn, in the last stage, due to behavioral inversion, the number of agents with lower failure likelihood increases again, which, in general, is concentrated among agents with accuracy levels *VERY HIGH* and *PERFECT*. However, despite this improvement, some agents still exhibit high failure rates, suggesting that behavioral transitions do not always guarantee optimal performance across all agents.

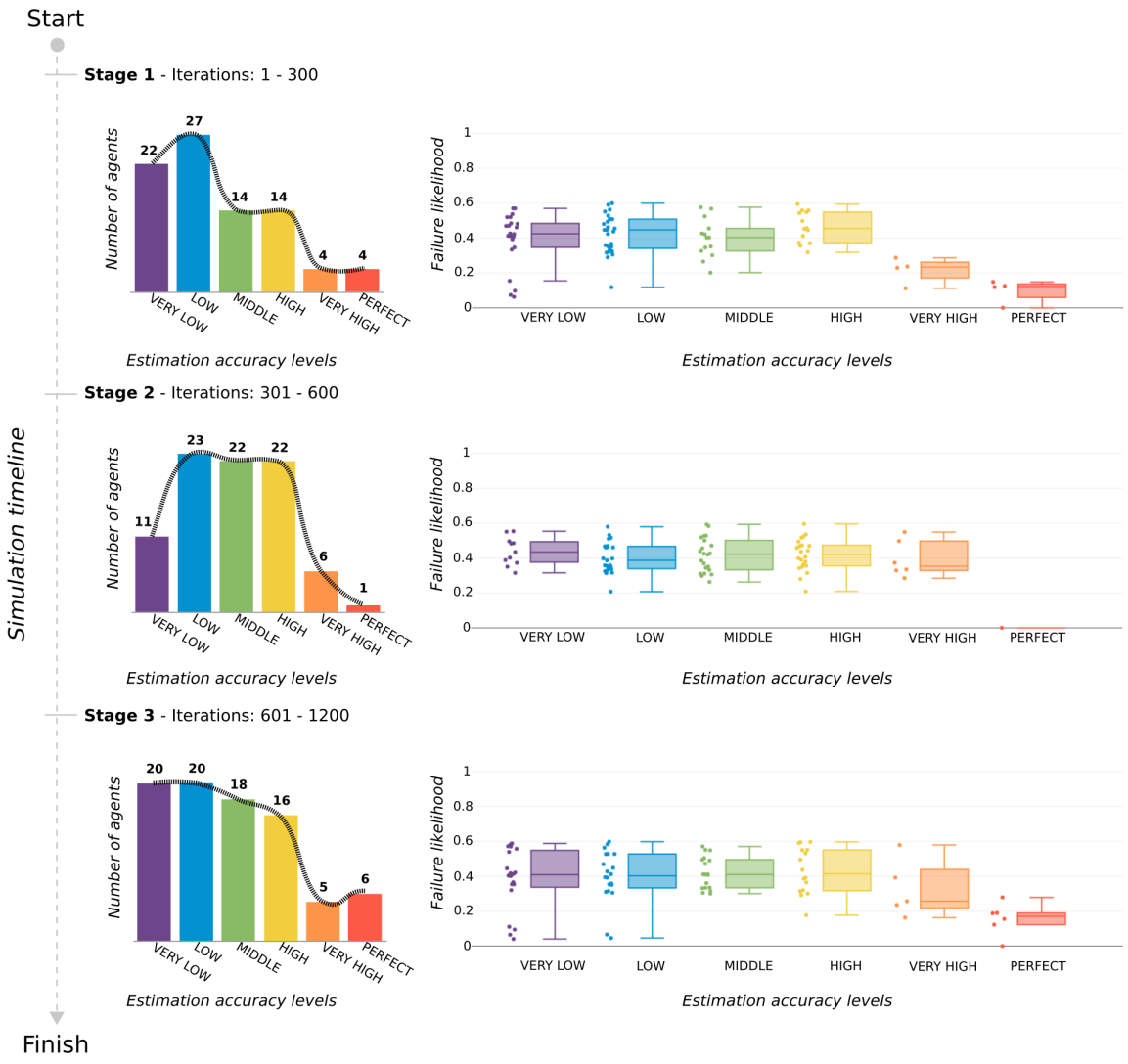


Figure 14 – Estimation accuracy levels and failure likelihood of agents over time in Experiment A (Tree network topology).

Source: Own authorship (2025).

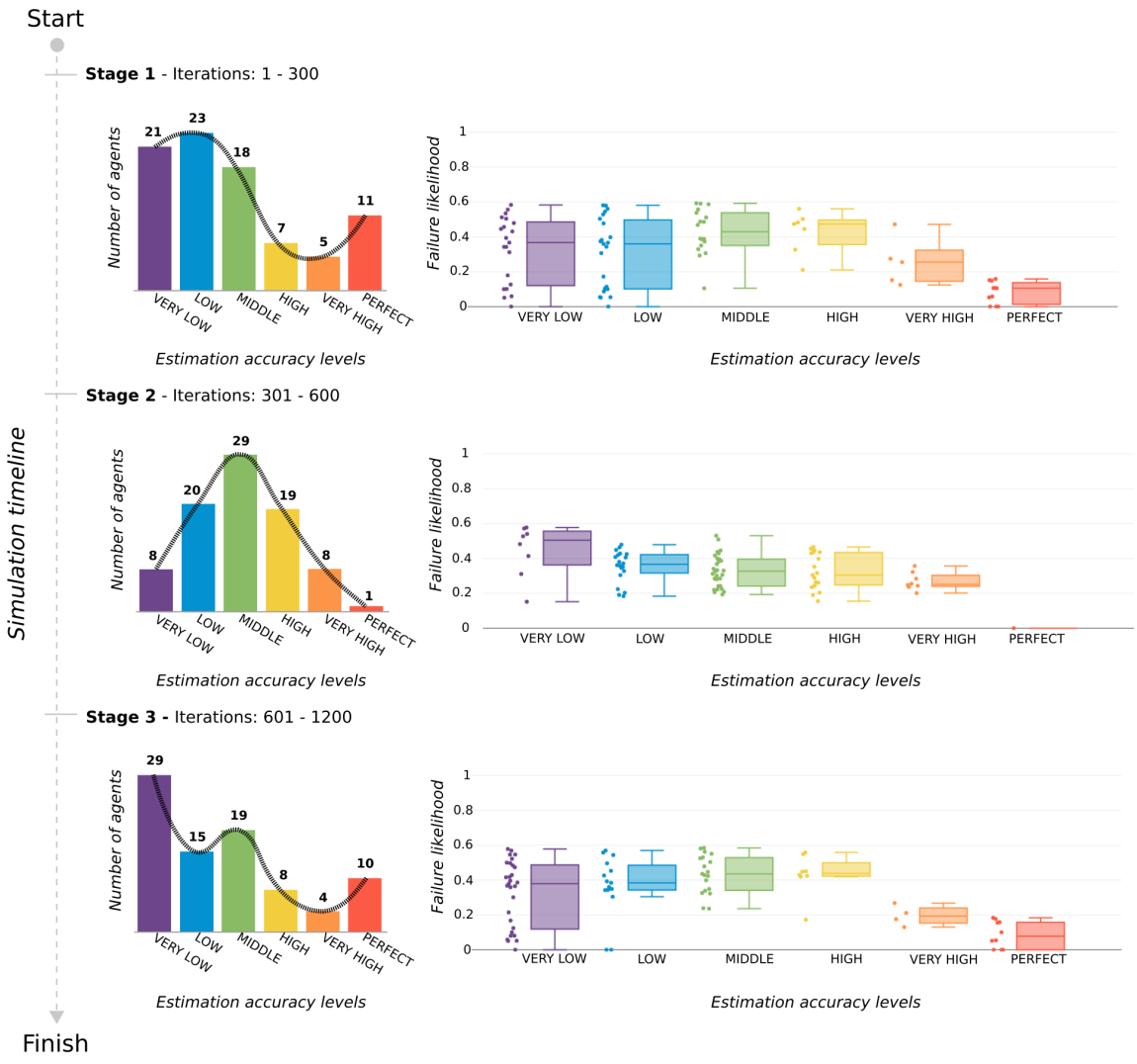


Figure 15 – Estimation accuracy levels and failure likelihood of agents over time in Experiment B (Random network topology).

Source: Own authorship (2025).

5.4.2 Organization of the Experiments

To provide a comprehensive understanding of the results obtained from our simulations, we conducted our analysis from two distinct perspectives, the experimental scenario perspective and the performance indicator perspective, as illustrated in Figure 16 (a) and Figure 16 (b), respectively. The *experimental scenario perspective* (Figure 16 (a)) evaluates all performance indicators within an experimental scenario, allowing us to observe how these indicators interact and influence each other over the iterations. In contrast, the *performance indicator perspective* (Figure 16 (b)) isolates each performance indicator, analyzing its variation across all experimental scenarios. This analysis enables a direct comparison of the delegators' performance considering a given indicator under the different peculiarities and constraints of each scenario.

[2] In order to compare the regret of a delegator across the experimental scenarios, we normalized its regret concering to the maximum satisfaction it could achieve in an iteration itr. Thus, the greater the difference between the delegator's regret and its maximum satisfaction, the lower its normalized regret regarding its choices.

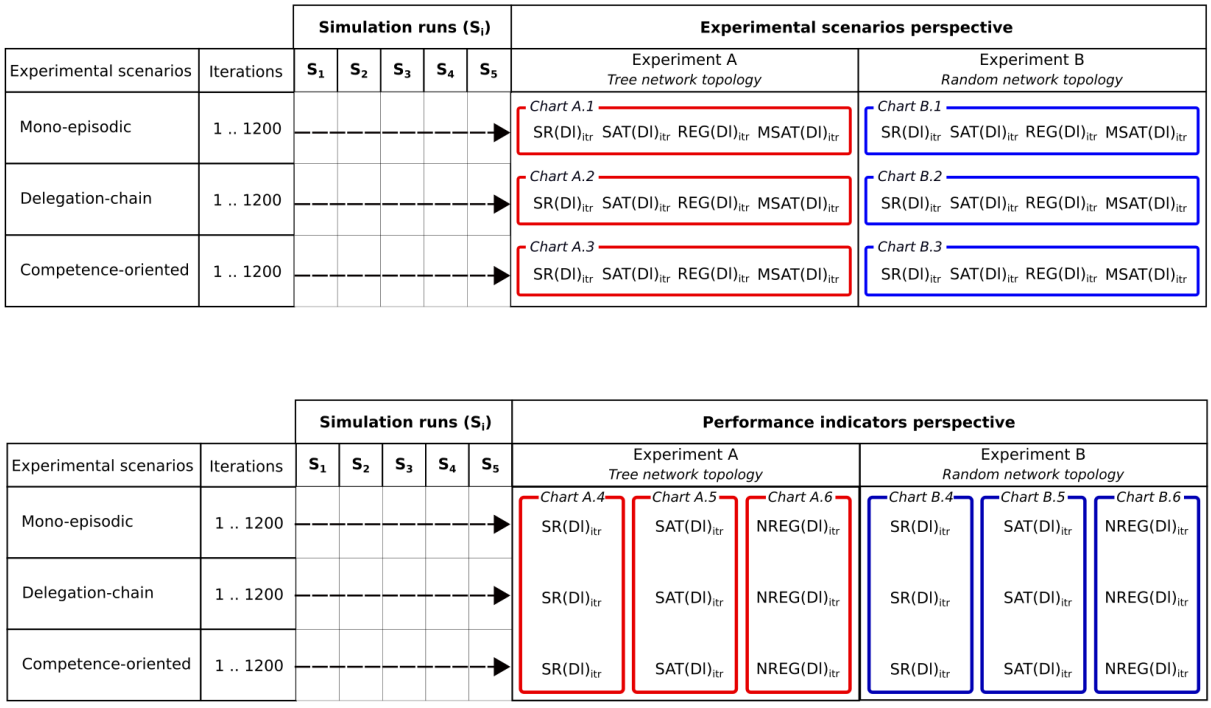


Figure 16 – Organization of experiments, considering the analysis perspectives: (a) experimental scenario perspective and (b) performance indicator perspective. We consider the following performance indicators during the analysis of results, which takes into account each iteration *itr*: $SR(Dl)_{itr}$: delegators' average success rate, $SAT(Dl)_{itr}$: delegators' average satisfaction, $REG(Dl)_{itr}$: delegators' average regret, $MSAT(Dl)_{itr}$: average of delegators' maximum satisfaction, and $NREG(Dl)_{itr}$[2]: average of delegators' normalized regret relative to $MSAT(Dl)_{itr}$.
Source: Own authorship (2025).

5.4.3 Experiment A: Tree Network Topology

This section presents the results of the experiments conducted taking as input the DS-net designed based on the tree network topology. The purpose of this experiment is to evaluate our delegation model in a scenario where social evaluations and success rates (accumulated or not) are propagated through delegation chains with a low connectivity factor between delegators and delegatees. This occurs because, in a tree network topology, each delegatee is connected to only one delegator.

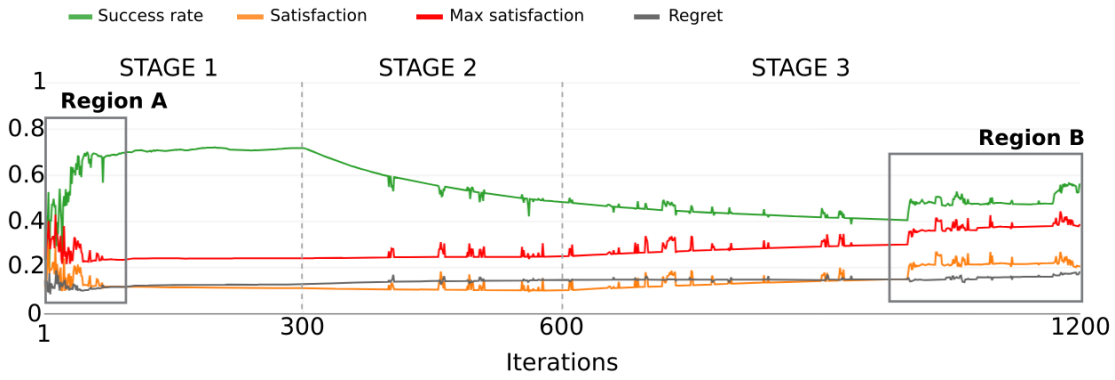
5.4.3.1 Experimental Scenario Perspective

The results in Figure 17 illustrate the delegators' performance over iterations in each experimental scenario, considering the tree network topology. Across all scenarios, the initial learning phase, marked as *Region A*, represents the period when delegators explore different partners to obtain information about their performance and behavior. At this stage, choices are often random due to the lack of prior interactions.

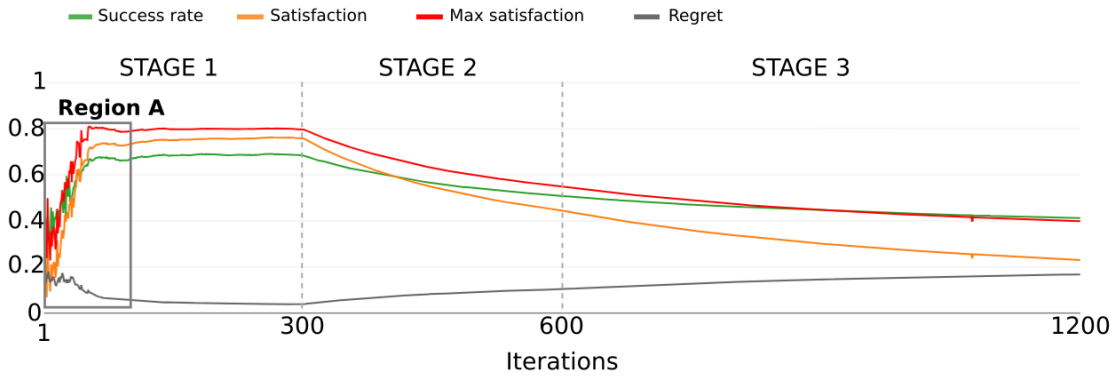
In Figure 17 (a) (Chart A.1 - mono-episodic selection), the success rate initially increases as delegators start identifying suitable partners. However, since each delegation instance is treated independently, delegators take longer to learn about their partners' behavior, as they cannot share any information among themselves. In contrast, in Figure 17 (b) (Chart A.2 - delegation chain selection), particularly in *Region B*, a slight recovery can be observed around iteration 1000. Even though the limited connectivity of the tree topology reduces information propagation across the network, the results indicate that success rate propagation through delegation chains enables delegators to progressively adapt to their partners' behavioral changes, gradually relearning their new behaviors.

Finally, in Figure 17 (c) (Chart A.3 - competence-oriented selection), delegators incorporate partners' competencies as an additional evaluation dimension. Note that during the initial stage (*stage 1*), this approach provides a more stable learning curve, as delegators can better identify partners capable of completing tasks successfully and meeting their expectations regarding task execution. Such a feature results in a more efficient convergence in partner selection compared to the other scenarios, at least for *stage 1*. Nevertheless, the efficiency of this approach is limited by the network topology, as the parent-child relationships among agents prevent a partner from having multiple delegators. Consequently, delegators cannot share impressions since they lack common partners, and their impressions are based only on direct experiences. Besides, a delegatee's know-how is built exclusively on interactions with a single delegator. The impact of the lack of impression propagation becomes evident after iteration 300, when delegators have difficulties identifying suitable partners due to their behavioral changes at each new stage.

(a)



(b)



(c)

Figure 17 – Delegators' average performance across each experimental scenario, taking the DS-net designed based on the tree network topology as input: (a) Chart A.1 - mono-episodic selection; delegations do not consider delegation chains or sub-delegations, (b) Chart A.2 - delegation chain selection; delegations are performed considering the accumulated success rate propagated by agents through delegation chains, and (c) Chart A.3 - competence-oriented selection; in addition to the accumulated success rate, delegators also consider their own and third-party impressions. Region A: initial learning period; Region B: recovery period based on relearning of the partners' behaviors.

Source: Own authorship (2025).

5.4.3.2 Performance Indicators Perspective

The results in Figure 18 provide a complementary perspective on delegators' performance by analyzing each performance indicator separately. In particular, Figure 18 (a) (Chart A.4) compares the average success rate of delegators across experimental scenarios, Figure 18 (b) (Chart A.5) compares the average satisfaction of delegators, and Figure 18 (c) (Chart A.6) analyzes the average normalized regret of delegators, considering their maximum satisfaction. These analyses help clarify how different delegation strategies (*i.e.*, mono-episodic, delegation chain, and competence-oriented selection) influence delegators' decision-making over time.

As shown in Figure 18 (a), regardless of the scenario, delegators' performance tends to follow a pattern, which starts with a learning period (*Region A*), where the learning time varies depending on the selection strategy. After the first stage, the delegators' performance is followed by a long period of decay caused by the behavioral changes of their partners in each new stage. Note that only the delegation chain selection approach allows delegators to gradually relearn partners' new behaviors, whereas the other strategies do not support adaptation beyond the first stage.

One point that should be highlighted is that in the competence-oriented selection approach, as shown in Figure 18 (b) and Figure 18 (c), delegators show the best measures of satisfaction and regret regarding their partner choices, at least in the first stage. From the second stage onwards, as discussed earlier, the tree network constraints regarding the connectivity among agents limit the efficiency of both the delegation chain selection and competence-oriented selection, as these approaches rely on the propagation of information between agents, which, in turn, depends on the network's topology.

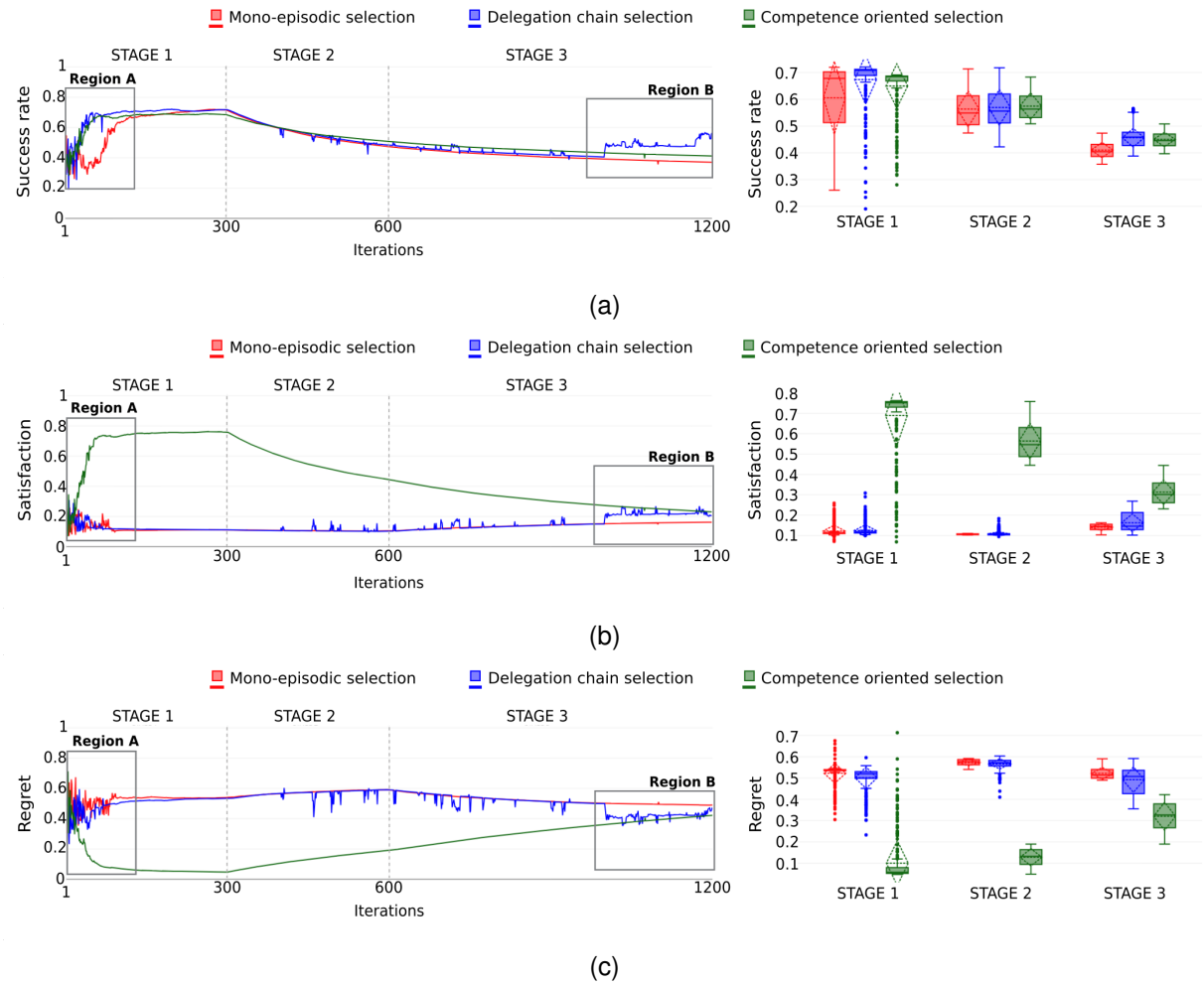


Figure 18 – Delegators' average performance considering each performance indicator for each experimental scenario, taking the DS-net designed based on the tree network topology as input: (a) Chart A.4 - delegators' average success rate, (b) Chart A.5 - delegators' average satisfaction, and (c) Chart A.6 - average of delegators' normalized regret relative to delegators' maximum satisfaction. Region A: initial learning period; Region B: recovery period based on relearning of the partners' behaviors. Source: Own authorship (2025).

5.4.4 Experiment B: Random Network Topology

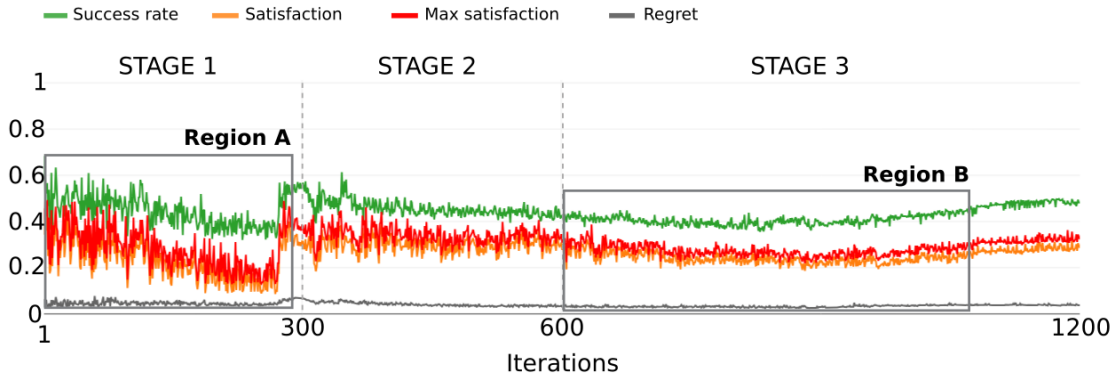
This section presents the average results for the experiments using the DS-net with a random topology as input. Unlike *Experiment A*, the connectivity between agents in this case is not restricted to the parent-child relationship, as in the tree network topology. This experiment aims to evaluate our delegation model based on the performance indicators defined for each experimental scenario, considering an arbitrary connectivity factor.

5.4.4.1 Experimental Scenarios Perspective

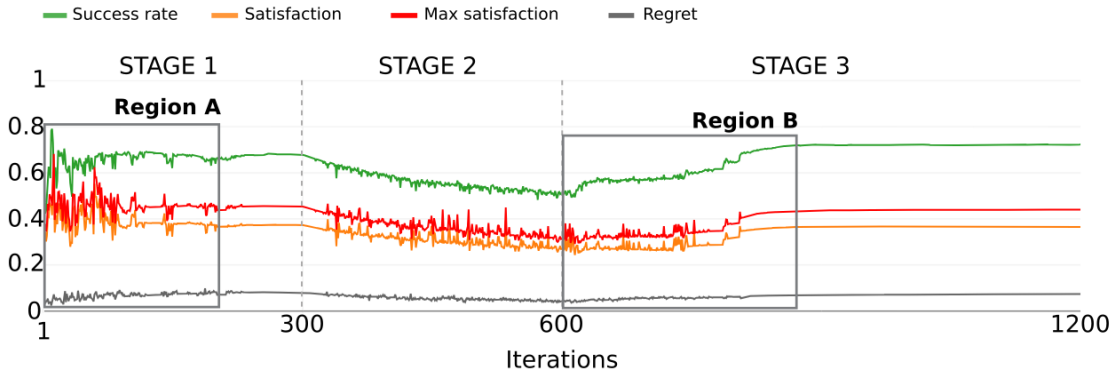
As shown in Figure 19, the random network topology used in *Experiment B* yields more significant results. This topology allows multiple delegators to assess the same delegatee, facilitating the sharing of information about common partners. Compared to the tree topology, a random network provides a higher degree of connectivity among agents (Shirley; Rushton, 2005). This characteristic increases the number of paths in the network (*i.e.*, instances of delegation chains), the number of partners available to a delegator at each network level, and the number of delegators connected to a given partner.

In particular, increasing connectivity among agents also implies to a higher number of explorations performed by delegators. This effect is most evident in mono-episodic selection (Figure 19 (a) (Chart B.1)), where partner selection relies on local information from each delegation instance. In this case, delegators require multiple iterations to identify the best partners, resulting in prolonged learning periods, such as highlighted in *Region A* and *Region B* of Figure 19 (a). In contrast, learning periods are shorter in the other experimental scenarios (*i.e.*, delegation chain selection and competence-oriented selection, Figure 19 (b) (Chart B.2) and Figure 19 (c) (Chart B.3), respectively). This is because, in such scenarios, delegators choose partners based on a set of impressions and the probability of success associated with delegation chains rather than solely on the success rate of an individual delegation instance.

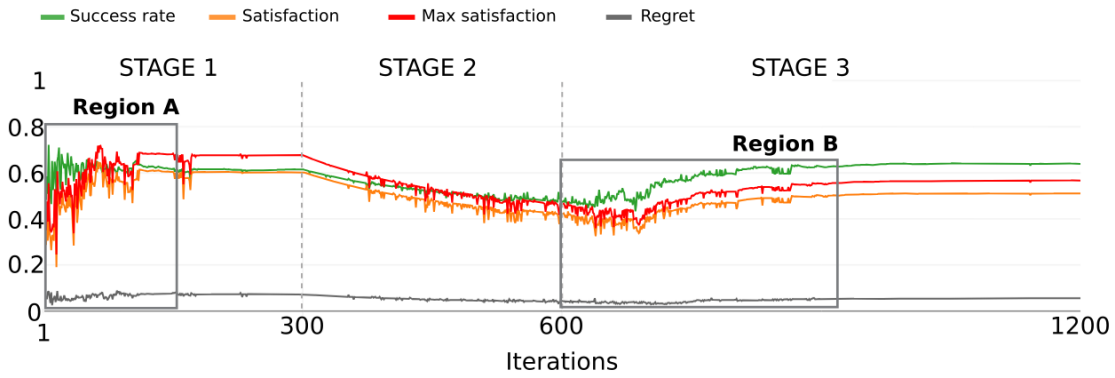
The main advantage of the non-mono-episodic approaches is the faster learning of new behaviors adopted by the agents after behavioral upgrades and downgrades. Even after iteration 600 (the second period of behavioral updates), the delegators are able to learn about the most recent social behaviors of their partners, leading to a faster recovery in delegators' performance than in the mono-episodic approach. An important detail to note is that significant performance recovery over the iterations can only be achieved in scenarios that incorporate some mechanism of information propagation along the delegation chains (*e.g.*, accumulated success rate or social evaluation sharing). Such a performance recovery (*Region B* in Figure 19) may take a long time to become apparent in mono-episodic selection, as delegators rely solely on the information from their own delegation instance to select their partners.



(a)



(b)



(c)

Figure 19 – Delegators' average performance across each experimental scenario, taking the DS-net designed based on the random network topology as input: (a) Chart B.1 - mono-episodic selection; delegations do not consider delegation chains or sub-delegations, (b) Chart B.2 - delegation chain selection; delegations are performed considering the accumulated success rate propagated by agents through delegation chains, and (c) Chart B.3 - competence-oriented selection; in addition to the accumulated success rate, delegators also consider their own and third-party impressions. Region A: initial learning period; Region B: recovery period based on relearning of the partners' behaviors.

Source: Own authorship (2025).

5.4.4.2 Performance Indicators Perspective

The performance of delegators from the point of view of each performance indicator, taking into account the DS-net with the random topology, are summarized in in Figure 20. Specifcally, Figure 20 (a) (Chart B.4) compares the average success rate of delegators across experimental scenarios, Figure 20 (b) (Chart B.5) compares the average satisfaction of delegators, and Figure 20 (c) (Chart B.6) analyzes the average normalized regret of delegators, considering their maximum satisfaction.

Note that, regarding the success rate (Figure 20 (a)), the highest values are achieved with the delegation chain selection strategy. This is because partners are chosen in order to maximize the accumulated success rate along the delegation chains they belong to. Consequently, selected partners are those that make part of chains with the highest probability of task completion, considering the transitive relationships among agents. However, as shown in Figure 20 (b) and Figure 20 (c), this approach fails in terms of delegators' satisfaction and regret regarding their partner choices compared to the competence-oriented selection strategy. Although it identifies partners capable of completing delegated tasks with a high success probability, it cannot distinguish between those who provide accurate performance estimations and those who do not.

Therefore, we remark that the best results, considering a good trade-off among all performance indicators, are achieved with the competence-oriented selection. Although this approach does not guarantee the highest success rate for the delegators, it can achieve satisfactory results for this metric while also providing a selection mechanism that finds partners capable of meeting the delegators' expectations concerning the performance estimations for a task. This tends to increase the delegators' satisfaction average and decrease the regret average regarding their choices of partners. Consequently, this approach can identify partners with a high probability of successfully completing a task and provide accurate estimates of their performance over time. Furthemore, as shown in the charts in Figure 20, successive choices of partners with such a profile tend to lead to a more stable system behavior concerning performance indicators, as the performance curves for competence-oriented selection exhibit fewer peaks and valleys along the iterations compared to other approaches.

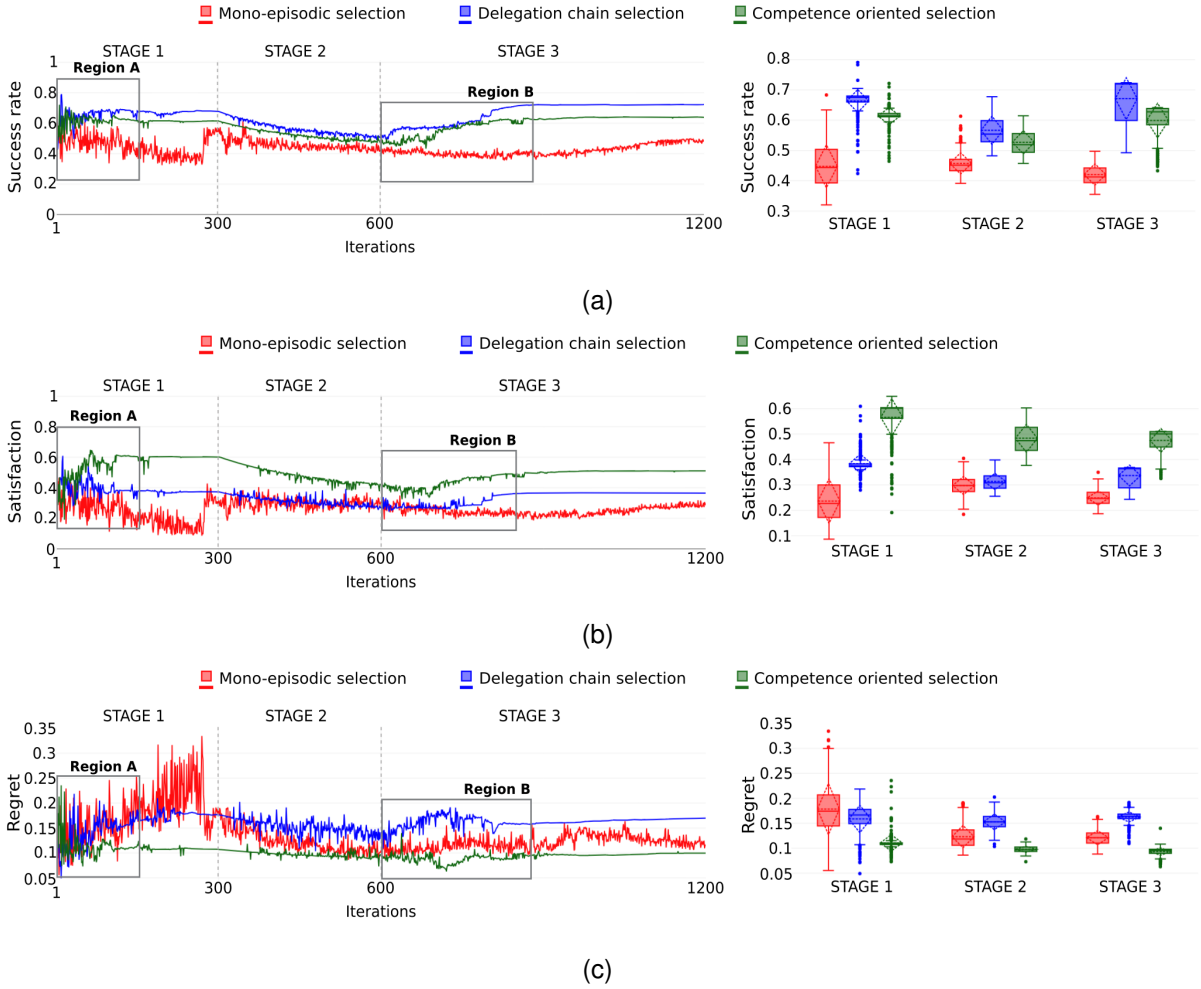


Figure 20 – Delegators' average performance considering each performance indicator for each experimental scenario, taking the DS-net designed based on the random network topology as input: (a) Chart B.4 - delegators' average success rate, (b) Chart B.5 - delegators' average satisfaction, and (c) Chart B.6 - average of delegators' normalized regret relative to delegators' maximum satisfaction. Region A: initial learning period; Region B: recovery period based on relearning of the partners' behaviors.

Source: Own authorship (2025).

5.5 Validation of Hypothesis

Herein, we analyze the hypotheses that underpin the investigation conducted throughout this thesis. This analysis aims to validate or refute each hypothesis based on the results obtained through our experiments. Such an evaluation is essential to confirm the capability of our delegation model, mainly for task delegation scenarios where the delegators need to cope with delegation chains and their decisions are affected by environmental changes, like the agents' behavioral change discussed in this work. Additionally, this assessment provides an important reflection about some relevant aspects that must be considered in a delegation model.

In particular, to evaluate the impact of the different experimental scenarios (mono-episodic selection, delegation chain selection, competence-oriented selection) on the performance metrics (success rate, satisfaction, and regret), appropriate statistical analyses for multi-

ple group comparisons were conducted. Initially, we used the ANOVA test (*Analysis of Variance*) (St; Wold *et al.*, 1989), which allows us to verify whether there are statistically significant differences among the means of the three scenarios. This approach is suitable because each metric was measured across three independent groups, and the objective was to determine whether at least one of these groups exhibited behavior that differed from the others. In cases where the ANOVA test indicated statistical significance ($p < 0.05$), we applied Tukey's HSD (*Honestly Significant Difference*) multiple comparison test (Abdi; Williams, 2010). This test identifies which pairs of groups differ significantly from each other while controlling the risk of incorrectly rejecting a true null hypothesis (*i.e.*, false positive), which can occur due to multiple comparisons.

The results obtained from the tests ANOVA and Tukey's HSD are sumarized in Table 5. Note that the *F-statistic* represents the ratio of between-group variance to within-group variance in the ANOVA test. A small *F-value* (*e.g.*, close to 1) indicates that the variability between group means is similar to the variability within groups, suggesting little to no evidence of significant differences among the means. In contrast, a larger *F-value* (typically greater than 1) suggests that the group means differ more than would be expected by random variation alone, indicating a potential statistically significant difference. The *p-value* quantifies the probability that the observed differences occurred by chance. Results with $p < 0.05$ are considered statistically significant, providing evidence that at least one group differs from the others. When $p < 0.05$, the Tukey HSD post-hoc test is applied to identify which specific pairs of scenarios exhibit significant differences. The Tukey HSD column lists those pairs where the differences were confirmed.

The analyses were conducted separately for each network topology (tree network topology and random network topology), in order to capture the specific effects of each structure on the performance of the selection scenarios. The results of these statistical analyses were obtained based on the data presented in Table 6, Table 7, and Table 8, which reports the mean values of the performance metrics, success rate, satisfaction, and regret, respectively. The data presented in these Tables represent the mean and standard deviation (sd) values calculated from five simulation runs for each evaluation metric, which serve as the basis for the results discussed throughout this section.

Table 5 – Summary of the statistical results obtained for each evaluation metric under the different delegation scenarios (MES: Mono-episodic selection, DCS: Delegation chain selection, COS: Competence-oriented selection), network topologies (tree and random) and evaluation metrics (success rate, satisfaction, and regret).

Network Topology	Metric	F-statistic	p-value	Significant	Tukey HSD
Tree	Success rate	3.4790	0.0643	No	–
	Satisfaction	695.2175	0.0000	Yes	MES-COS, DCS-COS
	Regret	572.3873	0.0000	Yes	MES-COS, DCS-COS
Random	Success rate	113.1988	0.0000	Yes	MES-DCS, MES-COS, DCS-COS
	Satisfaction	90.8899	0.0000	Yes	MES-DCS, MES-COS, DCS-COS
	Regret	13.7824	0.0008	Yes	MES-COS, DCS-COS

Table 6 – Descriptive statistics (mean and standard deviation (sd)) for the *success rate* metric across five runs. Results are presented for two network topologies (tree and random) and three experimental scenarios.

		Run 1		Run 2		Run 3		Run 4		Run 5		Avg (5 runs)	
Network Topology	Scenario	mean	sd	mean	sd	mean	sd	mean	sd	mean	sd	mean	sd
Tree	Mono-episodic selection	0.518	0.126	0.498	0.125	0.474	0.120	0.467	0.102	0.533	0.142	0.498	0.118
	Delegation chain selection	0.539	0.128	0.557	0.117	0.483	0.104	0.541	0.013	0.576	0.130	0.539	0.019
	Competence-oriented selection	0.538	0.108	0.524	0.100	0.530	0.113	0.522	0.104	0.543	0.108	0.531	0.100
Random	Mono-episodic selection	0.455	0.009	0.421	0.072	0.443	0.077	0.433	0.091	0.431	0.085	0.437	0.049
	Delegation chain selection	0.610	0.080	0.655	0.093	0.644	0.086	0.637	0.084	0.669	0.088	0.643	0.072
	Competence-oriented selection	0.604	0.072	0.535	0.069	0.608	0.068	0.591	0.066	0.586	0.071	0.585	0.058

Table 7 – Descriptive statistics (mean and standard deviation (sd)) for the *satisfaction* metric across five runs. Results are presented for two network topologies (tree and random) and three experimental scenarios.

		Run 1		Run 2		Run 3		Run 4		Run 5		Avg (5 runs)	
Network Topology	Scenario	mean	sd	mean	sd	mean	sd	mean	sd	mean	sd	mean	sd
Tree	Mono-episodic selection	0.127	0.035	0.124	0.036	0.127	0.035	0.125	0.026	0.134	0.034	0.127	0.024
	Delegation chain selection	0.126	0.035	0.121	0.033	0.134	0.040	0.129	0.052	0.185	0.148	0.139	0.042
	Competence-oriented selection	0.481	0.195	0.458	0.188	0.469	0.201	0.461	0.191	0.482	0.196	0.470	0.190
Random	Mono-episodic selection	0.288	0.112	0.231	0.097	0.282	0.111	0.251	0.108	0.238	0.090	0.258	0.056
	Delegation chain selection	0.329	0.066	0.331	0.085	0.349	0.049	0.321	0.067	0.384	0.070	0.343	0.046
	Competence-oriented selection	0.511	0.080	0.476	0.089	0.453	0.082	0.521	0.076	0.538	0.104	0.500	0.068

Table 8 – Descriptive statistics (mean and standard deviation (sd)) for the *regret* metric across five runs. Results are presented for two network topologies (tree and random) and three experimental scenarios.

		Run 1		Run 2		Run 3		Run 4		Run 5		Avg (5 runs)	
Network Topology	Scenario	mean	sd	mean	sd	mean	sd	mean	sd	mean	sd	mean	sd
Tree	Mono-episodic selection	0.553	0.059	0.540	0.057	0.532	0.062	0.530	0.089	0.526	0.058	0.534	0.038
	Delegation chain selection	0.532	0.052	0.550	0.068	0.527	0.084	0.532	0.070	0.480	0.131	0.513	0.059
	Competence-oriented selection	0.213	0.135	0.224	0.133	0.222	0.137	0.227	0.131	0.210	0.132	0.217	0.127
Random	Mono-episodic selection	0.171	0.075	0.121	0.051	0.142	0.068	0.146	0.065	0.132	0.048	0.136	0.038
	Delegation chain selection	0.150	0.036	0.177	0.062	0.146	0.033	0.193	0.036	0.139	0.061	0.159	0.018
	Competence-oriented selection	0.109	0.019	0.118	0.027	0.090	0.023	0.080	0.028	0.100	0.029	0.099	0.012

The evaluation of each hypothesis is provided below, considering the statistical results presented in Table 5, Table 6, Table 7, and Table 8. Moreover, to simplify the discussion, we employ the abbreviations adopted in Table 5 to refer to the experimental scenarios. Thus, mono-episodic selection is referred to as MES, delegation chain selection as DCS, and competence-oriented selection as COS.

- **Hypothesis 1**: A delegation model designed to cope with delegation chains can achieve better results than mono-episodic models. *Confirmed.*

  Our delegation model yielded superior results in task delegation simulations when operating in a non-mono-episodic manner. For this hypothesis, we compare MES with DCS and COS, focusing on the success rate metric.

  In MES and DCS, partner selection was configured to maximize the delegators' success rate. Specifically, in DCS, partners are chosen based on the accumulated success rate propagated through transitive dependencies. In contrast, in MES, partners are selected considering only their individual success rates, which are obtained through direct dependencies. Finally, COS extends DCS by also considering partner competencies during selection.

  In our analysis, although differences in success rates under the tree network topology were not statistically significant ($p = 0.0643$), significant differences were observed in the random network topology, where connectivity among agents is higher. As shown in Table 5, success rate differences between MES and DCS, as well as between MES and COS, were statistically significant. Moreover, these differences have been consistently observed across all five runs presented in Table 6.

  Thus, when the connectivity among agents is not severely constrained, as in random topologies, the use of transitive dependencies enables delegators to more effectively learn about partner behaviors based on their shared experiences (*i.e.*, social evaluations and accumulated success rates). This accelerates adaptation to partners' behavioral changes and leads to better performance compared to mono-episodic approaches.

- **Hypothesis 2**: The network's topology, created based on dependence chains, influences how often success rates are updated and impressions are shared by delegators, improving the effectiveness of social evaluation mechanisms and partner selection strategies based on delegation chains. *Confirmed.*

  Our experiments confirm that network topology significantly influences how frequently success rates are updated and how effectively impressions are shared among delegators. In the tree topology, the limited connectivity constrains information flow, resulting in slower propagation of success rates and reduced sharing of social evaluations (impressions). This limitation reduces the effectiveness of accumulated success rates and prevents delegators from sharing their impressions efficiently, leading to lower overall performance of the delegation model in both DCS and COS.

The ANOVA results (Table 5) further support this effect. In the tree topology, no significant difference was observed between the scenarios for the success rate metric ($p = 0.0643$), suggesting stagnation in success rate updates due to limited information exchange. In contrast, the random topology showed statistically significant differences ($p < 0.05$), indicating that richer connectivity facilitates more dynamic success rate updates and impression propagation. These results show that the structural properties of the network directly impact the effectiveness of social evaluation mechanisms and partner selection strategies based on delegation chains.

- **Hypothesis 3**: The use of transitive dependencies as a mechanism for propagating information can speed up the identification of good partners. *Confirmed.*

  The transitive dependencies established among the agents allow them to propagate their social evaluations and accumulated success rates along the delegation chains. Such dependency structures contribute to agents creating a view of their partners' behavior, which reduces the time they take to discover good partners since they can use this information to judge other agents' behavior.

  This acceleration pattern is evident in our results, for example, in the regions labeled as Region A and Region B in Figure 18 and Figure 20. Region A indicates the number of iterations needed for delegators to initially learn about partner behaviors, while Region B reflects the time required to re-learn partner behaviors following behavioral changes. The number of iterations required to learn partner behavior is significantly smaller for DCS and COS than for MES.

- **Hypothesis 4**: Using our multi-goal delegation model, a delegator can choose partners based on various dimensions, not just their success history. *Confirmed.*

  This includes evaluating their competencies concerning the task execution and specific criteria. Our experiments show that delegators can assess delegatees by comparing the actual outcomes of tasks to the performance estimations made by them during the offer phase of the task delegation process, rather than simply checking whether the task was completed.

  This approach leads to better choices, resulting, in general, in good rates of success and satisfaction and low regret. Note that this claim is confirmed by the results presented in Table 5, Table 6, Table 7, and Table 8, where COS consistently outperforms both MES and DCS in satisfaction and regret metrics. In both network topologies, COS exhibits significantly higher satisfaction and significantly lower regret compared to the other approaches. This indicates that competence-oriented selection enables more effective decision-making, leading to greater satisfaction and reduced regret.

In summary, the hypotheses were validated based on the experiments performed, suggesting that a delegation model explicitly designed to handle delegation chains, taking into account the network topology and the types of dependence relations established among agents,

may offer significant performance gains for delegators compared to a mono episodic approach in task delegation scenarios where agents can change their behavior over time.

6 CONCLUSIONS

In this work, we present a delegation model capable of handling sub-delegations and delegation chains, where the task delegation process is modeled as an exploitation/exploration problem from a multi-goal perspective. Our model enables agents to decide which tasks to delegate and to whom, based on direct and transitive dependencies, while also considering various social and cognitive factors, such as social image, reputation, know-how, success history, and the delegators' expectations regarding task execution. Our experiments demonstrated that explicitly addressing delegation chains can be advantageous compared to mono-episodic task delegation, as evidenced by the performance differences in our results concerning the delegators' success rate, satisfaction, and regret. These differences were statistically validated through ANOVA and Tukey's HSD tests, confirming the significance of the observed improvements.

Based on our evaluation, we conclude that several features associated with delegation chains can be explored to refine an agent's partner selection, such as the accumulated success rate of agents along a chain, the mechanisms of penalization and failure propagation, and the impact of connections among agents on information sharing. These features primarily contribute to reducing the learning time required by delegators to find new partners in scenarios where agents' social behaviors change over time. This is particularly evident in competence-oriented selection, where fluctuations in success rate, satisfaction, and regret are minimized, allowing agents to exhibit more stable behavior compared to other approaches. Consequently, this achieves a well-balanced trade-off between exploration and exploitation, enabling delegation chains composed of dynamically behaving agents to function effectively.

As an additional contribution, we introduced the DS-net structure, which allows us to describe and represent delegation chains formed through recursive delegations and task decomposition. This structure, supported by a formal grammar and impression filtering mechanisms, admits the representation of the situational dependencies among the agents, allowing the modeling of different types of dependence relationships, like single-action, multi-action, and multiparty relations. Such dependence relations are especially helpful for activities requiring teamwork, where coordination and collaboration between the agents are essential for accomplishing their goals (Cui; Idota; Ota, 2019) (Dong; Ota; Dong, 2021) (Barker; Whitcomb, 2016).

In addition to the conceptual contributions, we also developed a dedicated simulation tool to support the implementation and evaluation of our model. This simulator incorporates the DS-net structure and enables the execution of complex multi-agent scenarios, including delegation chains, social evaluation mechanisms, and configurable network topologies. It was essential not only for validating our model but also as a contribution in itself, offering a flexible platform for future studies.

In future work, we intend to explore the relation described in (Castelfranchi; Falcone, 2010) concerning the representation of internal and external factors and their effect on the trust decision. In our delegation model, the decision to trust is made considering basically internal factors (*i.e.*, elements associated with agents' capabilities

and relationships). Thus, we intend to incorporate into our delegation model an analysis of the external factors that may affect the agents' behaviors over time, taking into account dynamic environments (*e.g.*, obstacles, adversities, and interferences that might offer some risk or even new opportunities to the agents). A typical application that involves this kind of problem is task delegation in an open multi-agent system (Huynh; Jennings; Shadbolt, 2004) (Pinyol; Sabater-mir, 2013) (Bijani; Robertson, 2014), where agents can join and leave the system over time according to environmental conditions, which might affect the abilities, behaviors, and beliefs of the agents.

Additionally, we also intend to extend the DS-net structure's capabilities, aiming to represent task delegation scenarios where the agents can simultaneously pursue several goals. This extension allows us to incorporate a stage of goal deliberation in our delegation model, in which partners are selected according to the states of the goals pursued by a delegator (Castelfranchi; Paglieri, 2007) (Castelfranchi *et al.*, 2000). In this case, a delegator's choice regarding their partners can be modeled based on its beliefs that represent the internal and external factors capable of causing or inhibiting state changes in its goals (*i.e.*, positive and negative beliefs (Dhurandhar *et al.*, 2018)), allowing the delegator to explain its choice of partners. We remark that such a feature could be employed to build an explainable delegation model capable of generating explanations from causal chains obtained by analyzing the agents' partner selection and goal deliberation processes (Jacovi *et al.*, 2021) (Jasinski; Morveli-espinoza; Tacla, 2020).

Another promising direction is to consider more complex social dynamics, including the formation of alliances, occurrences of betrayal, or abrupt behavioral changes triggered by external events. Addressing such dynamics would increase the realism and adaptability of the model in highly dynamic and uncertain environments. Finally, integrating our model with frameworks for algorithmic transparency and explainability would be essential for systems where human agents also participate in the delegation chain, ensuring trust, accountability, and comprehensibility in human-agent interactions.

REFERENCES

ABDEL-BASSET, M.; MANOGARAN, G.; MOHAMED, M. Internet of things (iot) and its impact on supply chain: A framework for building smart, secure and efficient systems. **Future Generation Computer Systems**, v. 86, n. 9, p. 614–628, 2018.

ABDI, H.; WILLIAMS, L. J. Tukey's honestly significant difference (hsd) test. **Encyclopedia of research design**, Sage Thousand Oaks, CA v. 3, n. 1, p. 1–5, 2010.

AFANADOR, J. **Recursive delegation and a trust-measuring algorithm**. 2019. Tese (Doutorado) — University of Aberdeen 2019.

AFANADOR, J.; BAPTISTA, M. S.; OREN, N. Algorithms for recursive delegation. **AI Communications**, IOS Press v. 32, n. 4, p. 303–317, 2019.

AFANADOR, J.; OREN, N.; BAPTISTA, M. S. A coalitional algorithm for recursive delegation. *In*: SPRINGER. INTERNATIONAL CONFERENCE ON PRINCIPLES AND PRACTICE OF MULTI-AGENT SYSTEMS. 2019. **Anais [...]** [*S.l.*], 2019. p. 405–422.

AGARWAL, M.; AGGARWAL, V.; LAN, T. Multi-objective reinforcement learning with non-linear scalarization. *In*: PROCEEDINGS OF THE 21ST INTERNATIONAL CONFERENCE ON AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS. 2022. **Anais [...]** [*S.l.*: *s.n.*], 2022. p. 9–17.

AN, B.; MIAO, C.; CHENG, D. A coalition formation framework based on transitive dependence. **IEICE transactions on information and systems**, The Institute of Electronics, Information and Communication Engineers v. 88, n. 12, p. 2672–2680, 2005.

AN, B. *et al.* Algorithms for transitive dependence-based coalition formation. **IEEE Transactions on Industrial Informatics**, IEEE v. 3, n. 3, p. 234–245, 2007.

ARTEMIS. **Multi-Armed Bandits: Epsilon-Greedy Algorithm with Python Code**, . 2021. Accessed on 06, 06, 2023. Disponível em: https://medium.com/@eminik355.

ASHTIANI, M.; AZGOMI, M. A. Contextuality, incompatibility and biased inference in a quantum-like formulation of computational trust. **Advances in Complex Systems**, World Scientific v. 17, n. 05, p. 1450020, 2014.

AUER, P.; CESA-BIANCHI, N.; FISCHER, P. Finite-time analysis of the multiarmed bandit problem. **Machine learning**, Springer v. 47, n. 2, p. 235–256, 2002.

AUMANN, R. J.; DREZE, J. H. Cooperative games with coalition structures. **International Journal of game theory**, Springer v. 3„ p. 217–237, 1974.

BARABÁSI, A.-L.; ALBERT, R. Emergence of scaling in random networks. **science**, American Association for the Advancement of Science v. 286, n. 5439, p. 509–512, 1999.

BARKER, L. D.; WHITCOMB, L. L. A preliminary survey of underwater robotic vehicle design and navigation for under-ice operations. *In*: IEEE. 2016 IEEE/RSJ INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS (IROS). 2016. **Anais [...]** [*S.l.*], 2016. p. 2028–2035.

BIJANI, S.; ROBERTSON, D. A review of attacks and security approaches in open multi-agent systems. **Artificial Intelligence Review**, Springer v. 42„ p. 607–636, 2014.

BORDINI, R. H.; HÜBNER, J. F. Bdi agent programming in agentspeak using jason. *In*: SPRINGER. INTERNATIONAL WORKSHOP ON COMPUTATIONAL LOGIC IN MULTI-AGENT SYSTEMS. 2005. **Anais [...]** [*S.l.*], 2005. p. 143–164.

BOTELHO, V. *et al.* Dossier: decentralized trust model towards a decentralized demand. *In*: IEEE. 2018 IEEE 22ND INTERNATIONAL CONFERENCE ON COMPUTER SUPPORTED COOPERATIVE WORK IN DESIGN ((CSCWD)). 2018. **Anais [...]** [*S.l.*], 2018. p. 371–376.

BRANZEI, R.; DIMITROV, D.; TIJS, S. **Models in cooperative game theory**. [*S.l.*]: Springer Science & Business Media, 2008. v. 556.

BUCCAFURRI, F. *et al.* Experimenting with certified reputation in a competitive multi-agent scenario. **IEEE Intelligent Systems**, IEEE v. 31, n. 1, p. 48–55, 2015.

BURNETT, C.; OREN, N. Sub-delegation and trust. *In*: PROCEEDINGS OF THE 11TH INTERNATIONAL CONFERENCE ON AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS-VOLUME 3. 2012. **Anais [...]** [*S.l.*: *s.n.*], 2012. p. 1359–1360.

CAMACHO, E. F. *et al.* Control for renewable energy and smart grids. **The Impact of Control Technology, Control Systems Society**, IEEE Control Systems Society v. 4, n. 8, p. 69–88, 2011.

CANTUCCI, F.; FALCONE, R.; CASTELFRANCHI, C. A computational model for cognitive human-robot interaction: An approach based on theory of delegation. *In*: WOA. 2019. **Anais [...]** [*S.l.*: *s.n.*], 2019. p. 127–133.

CANTUCCI, F.; FALCONE, R.; CASTELFRANCHI, C. Robot's self-trust as precondition for being a good collaborator. *In*: TRUST@ AAMAS. 2021. **Anais [...]** [*S.l.*: *s.n.*], 2021.

CASTELFRANCHI, C. *et al.* Deliberative normative agents: Principles and architecture. *In*: SPRINGER. INTELLIGENT AGENTS VI. AGENT THEORIES, ARCHITECTURES, AND LANGUAGES: 6TH INTERNATIONAL WORKSHOP, ATAL'99, ORLANDO, FLORIDA, USA, JULY 15-17, 1999. PROCEEDINGS 6. 2000. **Anais [...]** [*S.l.*], 2000. p. 364–378.

CASTELFRANCHI, C.; FALCONE, R. **Trust theory: A socio-cognitive and computational model**. [*S.l.*]: John Wiley & Sons, 2010. v. 18.

CASTELFRANCHI, C.; FALCONE, R. Trust: Perspectives in cognitive science. *In*: **The Routledge Handbook of Trust and Philosophy**. [*S.l.*]: Routledge 2020. p. 214–228.

CASTELFRANCHI, C.; GUERINI, M. Is it a promise or a threat? **Pragmatics & Cognition**, John Benjamins v. 15, n. 2, p. 277–311, 2007.

CASTELFRANCHI, C.; MICELI, M.; CESTA, A. Dependence relations among autonomous agents. **Decentralized AI**, v. 3„ p. 215–227, 1992.

CASTELFRANCHI, C.; PAGLIERI, F. The role of beliefs in goal dynamics: Prolegomena to a constructive theory of intentions. **Synthese**, Springer v. 155„ p. 237–263, 2007.

CHAPELLE, O.; LI, L. An empirical evaluation of thompson sampling. **Advances in neural information processing systems**, v. 24„, 2011.

CHO, J.-H.; CHAN, K.; ADALI, S. A survey on trust modeling. **ACM Computing Surveys (CSUR)**, ACM New York, NY, USA v. 48, n. 2, p. 1–40, 2015.

CHUGH, T. Scalarizing functions in bayesian multiobjective optimization. *In*: IEEE. 2020 IEEE CONGRESS ON EVOLUTIONARY COMPUTATION (CEC). 2020. **Anais [...]** [*S.l.*], 2020. p. 1–8.

COHEN, G. Algorithmic trading and financial forecasting using advanced artificial intelligence methodologies. **Mathematics**, MDPI v. 10, n. 18, p. 3302, 2022.

CONTE, R.; PAOLUCCI, M. **Reputation in artificial societies: Social beliefs for social order**. [*S.l.*]: Springer Science & Business Media, 2002. v. 6.

CONTE, R.; PAOLUCCI, M. Social cognitive factors of unfair ratings in reputation reporting systems. *In*: IEEE. PROCEEDINGS IEEE/WIC INTERNATIONAL CONFERENCE ON WEB INTELLIGENCE (WI 2003). 2003. **Anais [...]** [*S.l.*], 2003. p. 316–322.

COSTA, A. C. da R.; DIMURO, G. P. Quantifying degrees of dependence in social dependence relations. *In*: SPRINGER. INTERNATIONAL WORKSHOP ON MULTI-AGENT SYSTEMS AND AGENT-BASED SIMULATION. 2006. **Anais [...]** [*S.l.*], 2006. p. 172–187.

CUI, Y.; IDOTA, H.; OTA, M. Improving supply chain resilience with implementation of new system architecture. *In*: IEEE. 2019 IEEE SOCIAL IMPLICATIONS OF TECHNOLOGY (SIT) AND INFORMATION MANAGEMENT (SITIM). 2019. **Anais [...]** [*S.l.*], 2019. p. 1–6.

DHURANDHAR, A. *et al.* Explanations based on the missing: Towards contrastive explanations with pertinent negatives. **Advances in neural information processing systems**, v. 31„, 2018.

DILEEP, G. A survey on smart grid technologies and applications. **Renewable energy**, Elsevier v. 146„ p. 2589–2625, 2020.

DONG, J.; OTA, K.; DONG, M. Uav-based real-time survivor detection system in post-disaster search and rescue operations. **IEEE Journal on Miniaturization for Air and Space Systems**, IEEE v. 2, n. 4, p. 209–219, 2021.

DRUGAN, M. M.; NOWE, A. Designing multi-objective multi-armed bandits algorithms: A study. *In*: IEEE. THE 2013 INTERNATIONAL JOINT CONFERENCE ON NEURAL NETWORKS (IJCNN). 2013. **Anais [...]** [*S.l.*], 2013. p. 1–8.

FAHEEM, M. *et al.* Smart grid communication and information technologies in the perspective of industry 4.0: Opportunities and challenges. **Computer Science Review**, Elsevier v. 30„ p. 1–30, 2018.

GARIVIER, A.; MOULINES, E. On upper-confidence bound policies for switching bandit problems. *In*: SPRINGER. INTERNATIONAL CONFERENCE ON ALGORITHMIC LEARNING THEORY. 2011. **Anais [...]** [*S.l.*], 2011. p. 174–188.

GRIFFITHS, N. Task delegation using experience-based multi-dimensional trust. *In*: PROCEEDINGS OF THE FOURTH INTERNATIONAL JOINT CONFERENCE ON AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS. 2005. **Anais [...]** [*S.l.*: *s.n.*], 2005. p. 489–496.

HAYES, C. F. *et al.* A practical guide to multi-objective reinforcement learning and planning. **Autonomous Agents and Multi-Agent Systems**, Springer v. 36, n. 1, p. 1–59, 2022.

HOSSAIN, M. *et al.* Role of smart grid in renewable energy: An overview. **Renewable and Sustainable Energy Reviews**, Elsevier v. 60„ p. 1168–1184, 2016.

HUYNH, T. D.; JENNINGS, N. R.; SHADBOLT, N. Fire: An integrated trust and reputation model for open multi-agent systems, „„. 2004.

JACOVI, A. *et al.* Contrastive explanations for model interpretability. **arXiv preprint arXiv:2103.01378**, „„ 2021.

JASINSKI, H. M.; MORVELI-ESPINOZA, M.; TACLA, C. A. Argagent: a simulator of goal processing for argumentative agents. **Computational Models of Argument**, IOS Press„ p. 469–470, 2020.

KARIMADINI, M.; LIN, H. Synchronized task decomposition for two cooperative agents. *In*: IEEE. 2010 IEEE CONFERENCE ON ROBOTICS, AUTOMATION AND MECHATRONICS. 2010. **Anais [...]** [*S.l.*], 2010. p. 368–373.

KOX, E. *et al.* Trust repair in human-agent teams: the effectiveness of explanations and expressing regret. **Autonomous Agents and Multi-Agent Systems**, Springer v. 35, n. 2, p. 1–20, 2021.

LAU, B. P. L.; SINGH, A. K.; TAN, T. P. L. A review on dependence graph in social reasoning mechanism. **Artificial Intelligence Review**, Springer v. 43, n. 2, p. 229–242, 2015.

MANAVALAN, E.; JAYAKRISHNA, K. A review of internet of things (iot) embedded sustainable supply chain for industry 4.0 requirements. **Computers & Industrial Engineering**, Elsevier v. 127„ p. 925–953, 2019.

MARGOLIS, J. T. *et al.* A multi-objective optimization model for designing resilient supply chain networks. **International Journal of Production Economics**, Elsevier v. 204„ p. 174–185, 2018.

MUSHTAQ, A. *et al.* Multi-agent reinforcement learning for traffic flow management of autonomous vehicles. **Sensors**, MDPI v. 23, n. 5, p. 2373, 2023.

NELLORE, K.; HANCKE, G. P. A survey on urban traffic management system using wireless sensor networks. **Sensors**, MDPI v. 16, n. 2, p. 157, 2016.

PETTITT, R.; ELLIOTT, L. R.; SWIECICKI, C. C. Squad-level soldier-robot dynamics: Exploring future concepts involving intelligent autonomous robots. *In*: SPRINGER. INTERNATIONAL CONFERENCE ON VIRTUAL, AUGMENTED AND MIXED REALITY. 2017. **Anais [...]** [*S.l.*], 2017. p. 426–436.

PINYOL, I.; SABATER-MIR, J. Computational trust and reputation models for open multi-agent systems: a review. **Artificial Intelligence Review**, Springer v. 40, n. 1, p. 1–25, 2013.

PINYOL, I. *et al.* Reputation-based decisions for logic-based cognitive agents. **Autonomous Agents and Multi-Agent Systems**, Springer v. 24, n. 1, p. 175–216, 2012.

RAO, A. S. Agentspeak (l): Bdi agents speak out in a logical computable language. *In*: SPRINGER. EUROPEAN WORKSHOP ON MODELLING AUTONOMOUS AGENTS IN A MULTI-AGENT WORLD. 1996. **Anais [...]** [*S.l.*], 1996. p. 42–55.

SABATER, J.; PAOLUCCI, M.; CONTE, R. Repage: Reputation and image among limited autonomous partners. **Journal of artificial societies and social simulation**, v. 9, n. 2„ 2006.

SABATER, J.; SIERRA, C. Regret: reputation in gregarious societies. *In*: PROCEEDINGS OF THE FIFTH INTERNATIONAL CONFERENCE ON AUTONOMOUS AGENTS. 2001. **Anais [...]** [*S.l.*: *s.n.*], 2001. p. 194–195.

SERUGUNDA, C. N. *et al.* Autonomous network traffic classifier agent for autonomic network management system. *In*: IEEE. 2021 IEEE GLOBAL COMMUNICATIONS CONFERENCE (GLOBECOM). 2021. **Anais [...]** [*S.l.*], 2021. p. 1–6.

SHAVANDI, A.; KHEDMATI, M. A multi-agent deep reinforcement learning framework for algorithmic trading in financial markets. **Expert Systems with Applications**, Elsevier v. 208„ p. 118124, 2022.

SHIRLEY, M. D.; RUSHTON, S. P. The impacts of network topology on disease spread. **Ecological Complexity**, Elsevier v. 2, n. 3, p. 287–299, 2005.

SICHMAN, J. S.; CONTE, R. Multi-agent dependence by dependence graphs. *In*: PROCEEDINGS OF THE FIRST INTERNATIONAL JOINT CONFERENCE ON AUTONOMOUS

AGENTS AND MULTIAGENT SYSTEMS: PART 1. 2002. **Anais [...]** [*S.l.*: *s.n.*], 2002. p. 483–490.

SICHMAN, J. S. *et al.* A social reasoning mechanism based on dependence networks. *In*: PROCEEDINGS OF 11TH EUROPEAN CONFERENCE ON ARTIFICIAL INTELLIGENCE. 1998. **Anais [...]** [*S.l.*: *s.n.*], 1998. p. 416–420.

SMITH, R. G. The contract net protocol: High-level communication and control in a distributed problem solver. **IEEE Transactions on computers**, IEEE Computer Society v. 29, n. 12, p. 1104–1113, 1980.

ST, L.; WOLD, S. *et al.* Analysis of variance (anova). **Chemometrics and intelligent laboratory systems**, Elsevier v. 6, n. 4, p. 259–272, 1989.

SUTTON, R. S.; BARTO, A. G. **Reinforcement learning: An introduction**. [*S.l.*]: MIT press, 2018.

TAO, R. *et al.* Robo advisors, algorithmic trading and investment management: wonders of fourth industrial revolution in financial markets. **Technological Forecasting and Social Change**, Elsevier v. 163„ p. 120421, 2021.

TURGAY, E.; ONER, D.; TEKIN, C. Multi-objective contextual bandit problem with similarity information. *In*: PMLR. INTERNATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE AND STATISTICS. 2018. **Anais [...]** [*S.l.*], 2018. p. 1673–1681.

WANG, K. *et al.* A study of fire drone extinguishing system in high-rise buildings. **Fire**, MDPI v. 5, n. 3, p. 75, 2022.

WONG, C. *et al.* An overview of robotics and autonomous systems for harsh environments. *In*: IEEE. 2017 23RD INTERNATIONAL CONFERENCE ON AUTOMATION AND COMPUTING (ICAC). 2017. **Anais [...]** [*S.l.*], 2017. p. 1–6.

YLINIEMI, L.; AGOGINO, A. K.; TUMER, K. Multirobot coordination for space exploration. **AI Magazine**, v. 35, n. 4, p. 61–74, 2014.

GLOSSARY

accuracy estimation level The agent's ability to accurately estimate the values for the criteria associated with a task, considering its execution.. 71

behavioral inversion The time required for a delegator to learn about a partner's behavior. This period occurs at the beginning of a simulation run and also when a new stage starts, as partners may suffer behavioral changes. The learning period depends on the strategy adopted by the delegators to select their partners.. 64

delegatee It is an agent that receives a task from a delegator through delegation and has the capability to execute or further delegate this task through decomposition or recursive delegation.. 16

delegation chains A sequence of sub-delegations created through transitive social dependencies, forming a path that connects the delegation chain's root to one or more terminator nodes (delegatees).. 16

delegator It is an agent that needs to perform a particular task, which may be completed through a delegation action.. 16

failure likelihood The probability of an agent failing to execute or deliver the outcomes of a task.. 71

learning time The time required for a delegator to learn about a partner's behavior. This period occurs at the beginning of a simulation run and also when a new stage starts, as partners may suffer behavioral changes. The learning period depends on the strategy adopted by the delegators to select their partners.. 75

mono-episodic It is a delegation instance where the delegatee can execute its task by itself. It does not require sub-delegations.. 16

recursive delegation A type of delegation action where the task is passed onward until it arrives to a delegatee that can execute it.. 16

root An agent that is at the beginning of the delegation chain. It is the agent that performs the first delegation action, resulting in the formation of the delegation chain.. 16

task decomposition A type of delegation action, where a task is decomposed into subtasks, which can be sub-delegated to different partners.. 16

Appendix A – Formal DS-net XML

This appendix presents the file format used in this work to represent DS-nets, referred to as the *Formal DS-net XML Format*. This format is a direct product of this thesis and was employed in our experiments to represent the DS-net networks used as input for our simulations. In particular, in a DS-net represented using the Formal DS-net XML Format, each agent is assigned a single goal that depends on the execution of one task. A goal is considered achieved only when its associated task is completed. This format simplifies the management of an agent's dependence relations, as tasks and dependencies are directly associated with a goal.

We highlight that, since this format provides a more compact and less expandable DS-net structure, it is better suited for models where fine-grained control over tasks and dependencies is not required. The main elements of the reduced DS-net XML format are defined as follows:

Agents: Agents are listed individually, and each <agent> has:

- <id>: Unique identifier of the agent.
- <name>: Name assigned to the agent.
- <type>: Role assigned to the agent (*ROOT*, *INNER*, and *TERMINATOR*).

Goals: Each agent has a single goal (<goal>), represented by:

- <id>: Unique identifier of the goal.
- <task>: The task associated with the goal, which includes:
  - <id>: Unique identifier of the task.
  - <criteria>: Execution constraints of the task (*i.e.*, the task criteria, indicating, for instance, the time and cost associated with the task).
  - <dependence>: Defines the dependence relations between agents and goals (*e.g.*, $(ag2g2 \wedge ag3g3)$, indicating that the current task is only completed if agent ag_2 achieves its goal g_2 or if agent ag_3 achieves its goal g_3).

Profiles <execution> and <delivery>: A task may include execution and delivery configurations, which define the estimation accuracy and failure likelihood of the agent as the executor of the task:

- <accuracy>: The agent's performance estimation accuracy for a task (*VERY LOW*, *LOW*, *MIDDLE*, *HIGH*, *VERY HIGH*, and *PERFECT*).
- <failure>: A value within the interval $[0..1]$ that represents the agent's probability of failure in executing or delivering the task's outcomes.

To illustrate this format, we present below (Figure 21, Figure 22, Figure 23, and Figure 24) a DS-net composed of four agents and their relationships[1]. Note that ag_1 is the *root* agent of the DS-net. This agent has g_1 as its goal, which can be accomplished through the execution of a task. This task can either be directly delegated to its partner ag_4 or decomposed into two sub-tasks, which are then delegated to its partners ag_2 and ag_3. Therefore, ag_1 will achieve its goal only if both ag_2 and ag_3 complete their tasks or if ag_4 completes its task. Morevoer, we remark that ag_1 will evaluate its delegatees regarding their competencies, considering their accuracy in estimating the time required to execute their tasks, since the tasks are evaluated based on the criterion *TIME*.

On the other hand, agents ag_2, ag_3, and ag_4 are *terminators*, which indicates that they are capable of executing their tasks by themselves. In the case of ag_2, it can estimate the task duration with *PERFECT* accuracy, and its failure likelihood during execution is 10%. Additionally, taking into account a scenario where the task will be completed, the probability of ag_2 failing to deliver the task outcome to its delegator (ag_1) is also 10%. Regarding agents ag_3 and ag_4, both have the same level of estimation accuracy (*HIGH*). However, ag_3 has a lower probability of failure during the task execution (10%) but a significantly higher failure probability in task outcome delivery (50%) compared to ag_4.

[1] A complete example of a DS-net described in the XML format can be found at: https://github.com/jjbaqueta/scdModel/tree/main/DSNetExamples/randomDSNet

ag_1 **(ROOT)**

```
<agent>
    <id>1</id>
    <name>ag1</name>
    <type>ROOT</type>
    <goal>
        <id>1</id>
        <task>
            <criteria>TIME</criteria>
            <dependence>((ag2g2 ∧ ag3g3) ∨ ag4g4)</dependence>
        </task>
    </goal>
</agent>
```

Figure 21 – Representation of agent ag_1 (root) using the DS-Net XML format.
Source: Own authorship (2025).

ag_2 **(TERMINATOR)**

```
<agent>
    <id>2</id>
    <name>ag2</name>
    <type>TERMINATOR</type>
    <goal>
        <id>2</id>
        <task>
            <criteria>TIME</criteria>
            <execution>
                <accuracy>PERFECT</accuracy>
                <failure>0.2</failure>
            </execution>
            <delivery>
                <failure>0.1</failure>
            </delivery>
            <dependence>ag2g2</dependence>
        </task>
    </goal>
</agent>
```

Figure 22 – Representation of agent ag_2 (terminator) using the DS-Net XML format.
Source: Own authorship (2025).

ag_3 **(TERMINATOR)**

```
<agent>
    <id>3</id>
    <name>ag3</name>
    <type>ROOT</type>
    <goal>
        <id>3</id>
        <task>
                <criteria>TIME</criteria>
                <execution>
                        <accuracy>HIGH</accuracy>
                        <failure>0.1</failure>
                </execution>
                <delivery>
                        <failure>0.5</failure>
                </delivery>
                <dependence>ag3g3</dependence>
        </task>
    </goal>
</agent>
```

Figure 23 – Representation of agent ag_3 (terminator) using the DS-Net XML format. Source: Own authorship (2025).

ag_4 **(TERMINATOR)**

```
<agent>
    <id>4</id>
    <name>ag4</name>
    <type>ROOT</type>
    <goal>
        <id>4</id>
        <task>
                <criteria>TIME</criteria>
                <execution>
                        <accuracy>HIGH</accuracy>
                        <failure>0.3</failure>
                </execution>
                <delivery>
                        <failure>0.2</failure>
                </delivery>
                <dependence>ag4g4</dependence>
        </task>
    </goal>
</agent>
```

Figure 24 – Representation of agent ag_4 (terminator) using the DS-Net XML format. Source: Own authorship (2025).