# A Self-Adaptive Module for Cross-Understanding in Heterogeneous MultiAgent Systems

Guilhem Marcillaud, Valérie Camps, Stéphanie Combettes, Marie-Pierre Gleizes, Elsy Kaddoum

# A Self-adaptive Module for Cross-understanding in Heterogeneous MultiAgent Systems

Guilhem Marcillaud[1], Valérie Camps[1], Stéphanie Combettes[1], Marie-Pierre Gleizes[1] and Elsy Kaddoum[2]

[1]*Institut de Recherche en Informatique de Toulouse, Université Toulouse III Paul Sabatier, Toulouse, France*

[2]*Institut de Recherche en Informatique de Toulouse, Université Toulouse II Jean Jaures, Toulouse, France*

Keywords: Cross-understanding, Data Imputation, Multi-referential Information, MultiAgent System, Heterogeneous Agents.

Abstract: We propose a self-adaptive module, called LUDA (Learning Usefulness of DAta) to tackle the problem of cross-understanding in heterogeneous multiagent systems. In this work heterogeneity concerns the agents usage of information available under different reference frames. Our goal is to enable an agent to understand other agents information. To do this, we have built the LUDA module analysing redundant information to improve their accuracy. The closest domains addressing this problem are feature selection and data imputation. Our module is based on the relevant characteristics of these two domains, such as selecting a subset of relevant information and estimating the missing data value. Experiments are conducted using a large variety of synthetic datasets and a smart city real dataset to show the feasibility in a real scenario. The results show an accurate transformation of other information, an improvement of the information use and relevant computation time for agents decision making.

## 1 INTRODUCTION

This paper tackles the problem of cross-understanding in a heterogeneous MultiAgent System (MAS) (Wooldridge, 2009). In such a system each agent has a local view of its environment and its own understanding of this view i.e. its reference frame (Hoc and Carlier, 2002). To enable an agent to use its perception, the agent decision-making process is adapted to its reference frame. In this work, we define two agents as heterogeneous when their reference frame is different. As information communicated by an agent is shared according to its reference frame, different agents will have trouble to understand each other. In systems where cooperation is needed, understanding problems may lead to non-cooperative situations (Georgé et al., 2011).

The cross-understanding problem can be encountered in several problems. For example, let consider the Internal Decision Layer of an Autonomous Vehicle (Loke, 2019). It is designed to take a decision with information from a set of sensors perceiving information in a reference frame (speed 30 km/h). To enrich its world understanding the vehicle may have access to other vehicles information through communication (speed 18.64 mph). However, a reference frame difference is an obstacle to the correct use of this information.

The studied problem possesses characteristics from the Feature Selection (Gibbons et al., 1979) and Data Imputation (Efron, 1994) fields. The Feature Selection one consists in learning the feature from the set of communicated information that is related to the missing feature (Li et al., 2017). The Data Imputation field focuses on estimating missing data from past observations (Van Buuren, 2018).

This paper addresses the problem of enriching the understanding of an agent in a MAS using information from neighbour agents having their own reference frame. This can be decomposed into two sub-problems: 1) enable an agent to understand communication from heterogeneous agents and 2) improve the information value using multiple sources (Zhao and Liu, 2008).

For the first sub-problem, the module we have developed, called LUDA (Learning Usefulness of DAta), supposes the existence of a linear relationship (Montgomery et al., 2012) between two data representing the same information in different reference frames. As LUDA aims at being added to an already



Marcillaud, G., Camps, V., Combettes, S., Gleizes, M. and Kaddoum, E.
A Self-adaptive Module for Cross-understanding in Heterogeneous MultiAgent Systems.
DOI: 10.5220/0010298503530360
In *Proceedings of the 13th International Conference on Agents and Artificial Intelligence (ICAART 2021) - Volume 1*, pages 353-360
ISBN: 978-989-758-484-8

functioning agent decision-making process the computation impact must be limited. Thus, we strongly emphasise the requirement for a limited number of histories necessary to learn reference frame transformation.

Once the agent is able to transform different reference frames, it has access to multiple sources of the same information and focuses on the second subproblem. This information redundancy can be used to improve the accuracy in a noisy environment (Hall and Llinas, 1997). To enable that, we extended LUDA with a novel agent behaviour that combines multiple information instances of the same information.

This paper is organised as follows: in section 2 that starts with an overview of Feature Selection and Data Imputation, we propose a description of the cross-understanding problem presented in this paper. From this description, section 3 describes the LUDA system. Section 4 discusses several experiments assessing the effectiveness of LUDA. Finally, section 5 concludes the paper and presents some perspectives.

# 2 STATE OF THE ART AND PROBLEM FORMALISATION

Cross-Understanding in MAS is a problem that has not been studied much even if the problem characteristics are not new. From existing works, we have identified two domains, **feature selection** and **data imputation**, that possess similar characteristics with the cross-understanding problem. Indeed, in the studied problem, an agent has access to its own perceptions and to those of its neighbours agents through communications. Thus, the agent with missing perceptions can complete them from perceptions of neighbour agents. This can be seen as a feature selection problem. However, in our case, the heterogeneity of agents reference frame makes the feature selection unsatisfactory because selected perceptions are not always usable and an estimation of their relationship with the known perception is needed. This issue can be seen as a data imputation problem. Thus, the cross-understanding problem in a heterogeneous MAS consists for an agent to be able to select, from its neighbouring agents perceptions, missing information while translating them into its own reference frame.

## 2.1 Feature Selection

The issue of variable and feature selection has been extensively studied in recent years. This problem consists in finding the most relevant variables to use (Gibbons et al., 1979) in different contexts, such as data mining, to reduce the computation time (Duda et al., 2006), biology to avoid a genetic study, medicine to understand the cause of a disease (Niel and Sinoquet, 2018), simulation to select the best experiments conditions (Ryzhov et al., 2012) or statistical studies related to economics.

In an intelligent and complex environment such as a connected house or an intelligent vehicle, the observation of the environment comes from a multitude of sensors. The noise and the redundancy of the perceived data have to be taken into account. To avoid redundancy as much as possible, the multi-source methods proposed by (Zhao and Liu, 2008) and Multi-view methods (Wang et al., 2013) are used.

**Difference with LUDA.** More and more complex feature selection problems are solved thanks to recent advancements. One still remaining limitation concerns the storage space needed to achieve accurate results. Methods usually use a large amount of data either received or continuously perceived (Li et al., 2017). This can be problematic for some real case problems when available computation time is limited.

As feature selection considers a problem with multiple features for one result. Its objective is to reduce the features number to only keep the most relevant ones without changing their value. As the decision process is a black box, the decision about which data to choose follows the same issue as ours. Our stated problem differs as it consists in replacing an already known feature by another expressed in a different reference frame.

## 2.2 Data Imputation

Data imputation is the field that focuses on estimating the value of missing data using multiple histories. This field addresses the issue of incorrect or missing data (due to non-functioning or non-available sensors) and improving data quality in noisy environments (Marwala, 2009). Usually, data imputation methods complete the existing dataset (containing enough observation) or estimate information in real-time from the previous observation (Marwala, 2009).

Several methods are available for estimating missing data, which are classified into two categories: mathematical (Van Buuren, 2018) and artificial intelligence ones (Kumar et al., 2013). The most used mathematical technique, the regression analysis (Liu et al., 2018) enables to describe the relationships between information.

In the field of artificial intelligence, neural networks are highly used (Heaton, 2008). Radial Basis Function Network, a particular family of neural net-

works, is effective for regressions and functions approximation (Zhang and Suganthan, 2016) but has a cost of computation and storage.

In data imputation field, systems learn to estimate new data values from previously observed data. In contrast, in the cross-understanding problem, the missing data is available but in a different reference frame. Moreover, the correct data is lost among the massive amount of available data. Using data imputation method for every available data is time consuming and the cost of observation history would not be adapted to real-time constraint applications.

### 2.3 Cross-understanding Formalisation

In the cross-understanding problem, an agent among the agents set perceives from its environment several information that we have classified into two subsets. The first subset called **principal data** ($P_D = \{p_{d_i}\}$) refers to information directly linked to the agent sensors ($i$ being one of the agent sensors) and which this agent can understand. The second subset called **extero data** ($E_D = \{e_{d_j}\}$) refers to information communicated by its neighbouring agents ($j$ being one of the agent neighbours), each having its own reference frame. The cross-understanding problem in heterogeneous MAS consists for an agent in being able to select from $E_D$ missing information of $P_D$ while translating them into its own reference frame.

The main objective of cross-understanding is to enable an agent to use $e_{d_j}$ communicated by a neighbour agent $j$ to replace missing $p_{d_i}$ one for its decision process. The decision process uses a function $f$ and computes a result $r$ to solve a situation. $f$ is considered as a black box taking a fixed number of principal data $p_{d_i}$ as inputs. The figure 1 illustrates an example with 6 inputs, 3 $p_{d_i}$ ($p_{d_1}, p_{d_3}, p_{d_5}$) and 3 $e_{d_j}$ ($e_{d_2}, e_{d_4}, e_{d_6}$).

When all principal data are available the computed result is considered as the ideal one noted $r_{ideal}$. When the principal data are missing, the objective is to find the right extero data $e_{d_j}$ to replace them while minimising the difference between $r_{ideal}$ and $r$ obtained by $f$ using $e_{d_j}$.

For an agent, the problem can be formalised as follow.

**Given** $P_D = \{p_{d_i}\}$ and $E_D = \{e_{d_j}\}$

**For Each** $p_{d_i} \in \{P_D\}$**:**

1. select $e_{d_j} \in \{E_D\}$ that can be linked to $p_{d_i}$
2. determine the two coefficients $x$ and $y$ such as $p_{d_i} = x \times e_{d_j} + y$

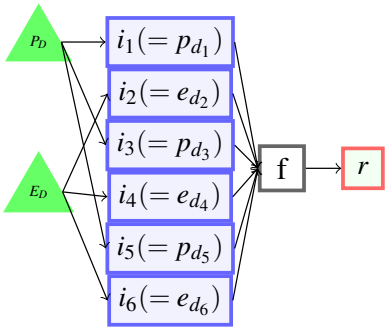


Figure 1: The function $f$ computes the result $r$ using principal data for $i_1, i_3, i_5$ and extero data for $i_2, i_4, i_6$.

**While Minimising:**

$$r = \sum_{k=0}^{nbInput} |p_{d_i} - e_{d_k}| \quad (1)$$

As different and several $e_{d_j}$ can be available for one $p_{d_i}$ the LUDA's second objective is to take advantage of this multi-source to improve the accuracy of the proposed values for $p_{d_i}$, especially in a noisy environment.

## 3 THE ADAPTIVE MULTIAGENT SYSTEM LUDA

This section presents LUDA a *MultiAgent System* (MAS) enabling an agent to translate any data communicated into its reference frame.

### 3.1 MAS Overview

A MAS is composed of multiple interacting and autonomous entities known as *agents*. An agent is able to perceive a local part of its environment and acts with this partial knowledge to autonomously solve its own goals (Wooldridge, 2009). The MAS paradigm is highly efficient to solve complex problems thanks to the local and distributed computation between agents (Serugendo et al., 2003).

An agent in LUDA respects the following properties: (*i*) it is autonomous, it decides alone its actions, (*ii*) it interacts with neighbour agents to achieve its goal, *(iii)* it has a partial view of the environment, (*iv*) it has negotiating skills, and (*v*) it acts in a dynamic and continuously evolving environment. In addition, each agent possesses a characteristic named **criticality**. Criticality is a numerical value that represents the degree of non-satisfaction of the agent goals, which impacts the agent behaviour.

### 3.2 LUDA Agents

We propose a MAS-based module to enable an agent to understand communications from heterogeneous agents. Based on the problem formalisation in section 2.3, we identified three types of agents: **data agents**, **morph agents** and **group agents** as well as one entity, the Decision Process (DP). The decision process is a black box only able to compute a result with values given by agents (it corresponds to the $f$ function in fig. 1).

**Data Agents.** For each information (belonging either to $E_D$ or $P_D$) a *data agent* is created and associated with it. The objective of each *data agent* is to find the other *data agents* representing the same information in a different reference frame. Let remind that in this work, we assume that two different data representing the same information are linked by a linear transformation: $d_1 = x \times d_2 + y$. To achieve this objective, a *data agent* creates as many *morph agents* as other *data agents* in the environment.

**Morph Agents.** Once created a *morph agent* is associated with two *data agents*: its creator $d_c$ and the objective one ($d_o$) which is one of the *data agents* of the environment. The *morph agent* goal is to find values of linear coefficients that help its $d_c$ to approximate the best linear transformation towards the value of its $d_o$. To modify the coefficient values, the *morph agent* uses an adaptation strategy exploiting the criticality. The criticality of a *morph agent* is computed from the error of the worst situation encountered during its operation (i.e. a situation where the agent has decided to accelerate instead of brake). From a situation, the *morph agent* stores as an observation the $d_c$ value and the $d_o$ value. The observation is evaluated by comparing the $d_c$ value transformed by the *morph agent* with the $d_o$ value acquired during this observation (cf. equation 2). An observation is evaluated each time the coefficients are modified. As the scale of each reference frame is different, the difference value needs to be normalised using the principal data range enabling the *morph agents* to interact with each other.

$$crit(ma) = argmax(\frac{|(x \times d_c + y) - d_o|}{range(d_o)}) \quad (2)$$

Instead of using a linear regression, *morph agents* use an adaptation process to compute coefficients. This process aims at reducing the memory required for all *morph agents*. Indeed, the more available observations, the more accurate the linear transformation. However, the amount of memory increases as the number of observations increases. As the required memory has to stay at a "reasonable" level, we propose to store in memory only the observations with the highest error to adapt the coefficients $x$ and $y$. Observations selection relies on the observation relevance. Relevance is the difference between the principal data value and the value transformed by the *morph agent*. From the three worst observations that it keeps in memory, the agent computes the best coefficients to satisfy them. Then, it computes its new coefficients using a weighted sum of the old coefficients and the best ones. The weight used is $\omega \in [0.1, 0.9]$. The higher $\omega$, the higher the impact on new coefficients. Using its new coefficients a *morph agent* modifies the relevance of observations it has in memory.

This behaviour enables to observe two trends: either the criticality decreases until it converges around a low value with the augmentation of observed situations, either no convergence is observed. The first trend occurs when a linear relation exists between both the data objective and the *data agent*. In other words, a low criticality means that the *morph agent* is able to transform the reference frame of its $d_c$ into its $d_o$. A *morph agent* unable to reduce its criticality is considered useless and consequently, it destroys itself. However, to avoid inappropriate destruction because of a slow adaptation, a *morph agent* considers itself useless only if its criticality is high and if another *morph agent* from the same *data agent* has a low criticality.

**Group Agents.** As different $e_{d_j}$ represent the same $d_{p_i}$, we introduced a third agent type called **group agent**. A *group agent* is always linked to at least one *data agent*. A *group agent* aims at having the most reliable value to give to the Decision Process *DP* and to reach this goal, it tries to regroup several *data agents* together.

The environment of a *group agent* is composed of all the other *group agents* of the environment, the *data agents* linked to it and the *DP*. A *group agent* can execute four types of action: 1) **propose one of its *data agent* to another *group agent***, 2) **propose a merging with another *group agent***, 3) **exclude a *data agent*** and 4) **send a value to a DP**. A *group agent* evaluates its criticality using formula 3 according to the adequacy of all its *data agents*. The adequacy of the *data agent* $da_i$ with the *data agent* $da_j$ is the criticality computed by the *morph agent* of $da_j$. A *group agent* tries to have as much *data agents* as possible. The parameter $\delta \in [0.0, 2.0]$ models the *group agent* will to link a new *data agent*. The highest $\delta$ is, more easy is to link a new *data agent*.

$$crit_{ga} = \frac{\sum_{(i=0, j=1, i \neq j)} (crit(ma_{da_i, da_j}) - \delta)}{nbDataAgentInGA} \quad (3)$$

From this formula, the criticality of a *group agent* strongly depends on which *data agent* is inside the

group. To lower its criticality, a *group agent* wants to exclude high critical *data agents*, and enables those *data agents* to be available to join another group. Each *group agent* evaluates the impact of the integration of the proposed *data agent* from a criticality point of view. If the integration reduces the *group agent* criticality, the *data agent* is accepted. When no group is willing to host the *data agent*, it is expelled and associated with a new *group agent*. This new *group agent* contains this unique *data agent*. However, as adding a new low critical, *data agent* decreases the overall criticality of the *group agent*, this latter is willing to be linked to the most *data agents* possible in order to reduce the impact of noisy information.

A *group agent* also has the possibility to merge with another *group agent* if they both possess *data agents* representing the same information. The merging decreases the adequacy of its *data agents*. However, it must not lead to the expulsion of one *data agent* of the group. In the same way as the proposition of hosting a *data agent*, two *group agents* can evaluate if the resulting merging is less critical than remaining separated.

Finally, a *group agent* aims to **propose a value** to the *DP*. A *group agent* sends a proposition for an input *i* of the *DP*. To increase the accuracy, the *group agent* adapts the proposed value of each available *data agent* for the input. A *group agent* associates a weight $w_j$ with each *data agent* to which it is linked. Those weights enable it to compute the value sent to the *DP* according to equation 4.

$$value_{ga} = \frac{\sum_{i=0} w_i * da_i}{\sum_{i=0} w_i} \tag{4}$$

Once the value proposed to the DP, the *group agent* receives the DP feedback which is the ideal value. Then, the *group agent* adapts the weights to improve its accuracy for the next time. Each *data agent* has a noisy value. Some sensors overestimate, others underestimate the same information and some sensors are more accurate than others. A *group agent* adapts the weight of each *data agent* according to the value they have given. The most accurate *data agents* are rewarded with an increase in their weight, while other weights are reduced. In addition, a *group agent* increases the weight of *data agents* whose value would impact positively the *group agent* value if its weight was higher. Equation 5 is used by a *group agent* to modify the weight of its *data agents*. With $\alpha$, $\beta$ and $\gamma$ the parameters used to modify the evolution speed of the weight and $V$ the value of the entity (i for the *data agent*, DP for the decision process and ga the *group agent*). After several experiments exploring those parameters, the values $\alpha = 0.1$, $\beta = 0.05$ and $\gamma = 1/30$ have been chosen.

$$w_i = \begin{cases} w_i + \alpha \text{ if } |V_i - V_{DP}| < V_{ga} - V_{DP}| \\ w_i - \gamma \text{ if } |V_i - V_{DP}| > |V_{ga} - V_{DP}| \\ w_i + \beta \text{ if } V_{DP} \in ]V_{ga}, V_i[ \text{ or } V_{DP} \in ]V_i, V_{ga}[ \end{cases} \tag{5}$$

### 3.3 Resolving Objectives through Agents Interactions

In MAS, interaction between agents can lead to a global phenomenon called emergence. This section describes the emergence of the objectives from the interactions of agents.

**First Objective: Replacing Missing Data.** This first objective is achieved thanks to interactions of *data* and *morph agents*. The $e_{d_{i,j}}$ replacing a missing $p_d$ is selected according to its adequacy with $p_d$. After several resolution cycles, we observe a reduction of the *morph agents* number linked to a *data agent*. The remaining *morph agents* achieve an accurate transformation of the reference frame. As a consequence, a *data agent* is able to replace a missing $p_d$ still linked to the *morph agent*.

**Second Objective: Improving the Reliability of Information.** This second objective enables to overcome the noisy data problem. As the *data agent* can now translate its value into a different reference frame, it is interesting to group together related $e_d$ and $p_d$ to extract the most accurate value from the group so formed, thus taking advantage of the multi-source information to reduce the noise. At the beginning, *group agents* are, at the most, as many as *data agents*. After several resolution cycles (cf section 4.1), related *data agents* have a strong adequacy and the *group agents* linked together merge. By consequence, the number of *group agents* decreases and the remaining *group agents* have more related *data agents*. The value sent by a *group agent* to the *DP* is the combination of all the linked *data agents* values.

## 4 EXPERIMENTATION

This section aims at evaluating the LUDA system with a large number of experiments. Most used datasets are synthetically generated to verify the linear transformation hypothesis and have a significant amount of different data. A real smart city dataset has been modified using real translations to experiment on real data.

### 4.1 Metrics

The efficiency of the LUDA method is evaluated using several metrics:

1. the accuracy of *morph agents* adaptation (the transformed value is compared to the correct one);
2. the resulting group: complete or incomplete group, misplaced or excluded *data agent*;
3. the impact of the values sent to the *DP* on the result of equation 1;
4. the time required to complete an adaptation cycle;
5. the number of remaining agents.

The different experiments consider several environments with various number of i) *principal data* and of ii) *extero data* per *principal data*.

All the experiments last 200 cycles of resolution. A resolution cycle consists in : 1) each *group agent* computes a value to give to the *DP*; 2) each *morph agent* receives the DP feedback and adapts its coefficients; 3) each *group agent* receives the feedback and adapts the *data agents* weight and 4) group agents reorganise themselves.

### 4.2 Experimentation with Noisy Data

Noise is added to each dataset using one of the three Gaussian-based strategies : 1) every data is altered using a Gaussian noise with identical distribution; 2) the Gaussian distribution is different for each data and 3) the data value is always over or below the true value.

**Similarities of Translation Functions.** This section assesses the effectiveness of the *morph agent* learning. We compare our results to state-of-the-art methods that are linear regression (LR), Random forest regression (RDM) and Gradient Boosting Regression (GRAD). Figure 2 shows the value difference between the true data and data estimated with imputation methods. For visualisation purposes, only 100 cycles are displayed. We observe that even if the error is lower using LUDA, some peaks have arisen, which has a negative impact on the overall result. A peak represents a situation where a *morph agent* was not efficient to transform the value of its *data agent*.

**Formed Groups.** Table 1 shows the resulting groups at different resolution cycles. Data is well-placed when it shares the same group as their related $p_d$. A group is completed when all the *data agents* representing the same information in various reference frames are grouped together. In an environment with no noise or with same noise, group accuracy is high. In case of different noise values, the noisiest observations are stored in memory to be used for the adaptation. Noise degrades the efficiency of the adequacy computation, leading to less efficient agent interactions and fusion.

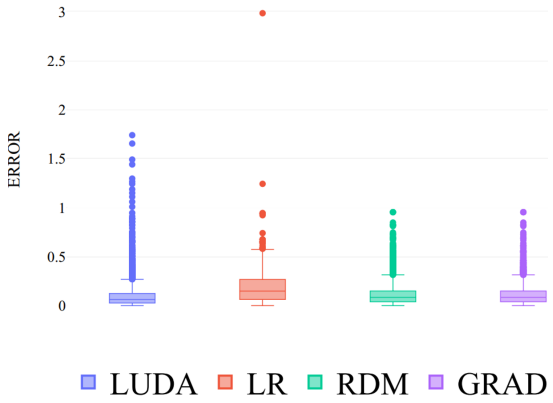


Figure 2: Difference between the true data and data estimated with LUDA, LR, RDM and GRAD.

Table 1: Formed Groups result according to the percentage of Completed Groups (CG) and Well-Placed Data (WPD).

| Cycle | 30 | | 90 | | 150 | | 180 | |
|---|---|---|---|---|---|---|---|---|
| Metric | CG | WPD | CG | WPD | CG | WPD | CG | WPD |
| 68 data No noise | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 |
| 68 data 5% noise | 88.24 | 92.65 | 88.26 | 95.59 | 94.11 | 95.59 | 94.11 | 98.53 |
| 68 data Diff noise | 70,59 | 85,29 | 88,24 | 97,06 | 64,71 | 88,24 | 64,71 | 85,29 |
| 192 data Diff noise | 75,51 | 90,10 | 73,47 | 94,27 | 73,47 | 91,15 | 71,43 | 90,63 |
| 392 data Diff noise | 73,74 | 89,54 | 75,76 | 92,86 | 76,77 | 92,09 | 78,79 | 93,62 |

**Decision Process Result Difference.** This section explores the advantages of using several data using the LUDA system in a noisy environment. Figure 3 shows the value difference between the principal data and the one adapted by a *group agent* following equation 1. The effectiveness of LUDA compared to using only known data depends on the noise nature (similar or different Gaussian, overestimation and underestimation). LUDA is more effective when different sources underestimate or overestimate. The peaks observed in figure 2 have a direct negative impact on the *DP* result. A LUDA improvement would be to detect when the value of a *data agent* is really far from those proposed by the other *data agents* in the same group.

### 4.3 Exploration of the Computation Time

To explore the computation time, experiments consider the number of remaining agents compared to the objective (metric 5 in 4.1), and the time to achieve a resolution cycle.

Figures 4 and 5 show impact of the data number on computation time. In the worst-case scenario, all

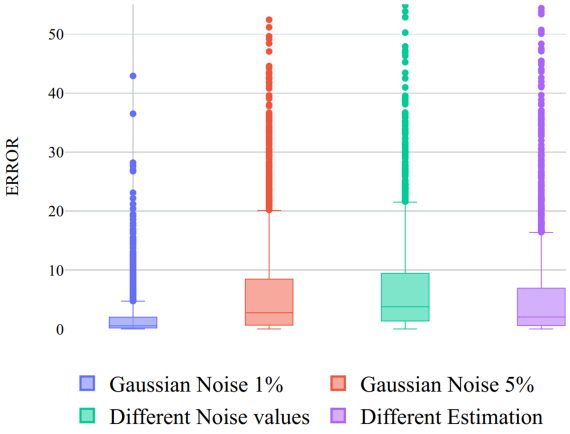


Figure 3: Effectiveness of LUDA on different noise values.

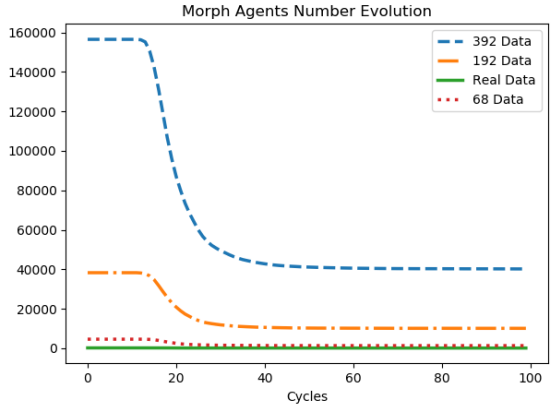


Figure 5: Evolution of the *morph agents* number.

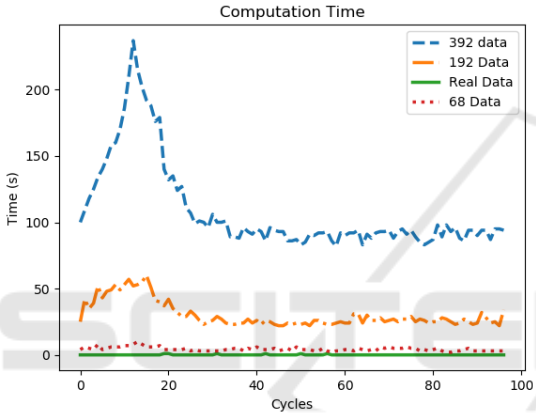


Figure 4: Evolution of the *morph agents* coefficients adaptation.

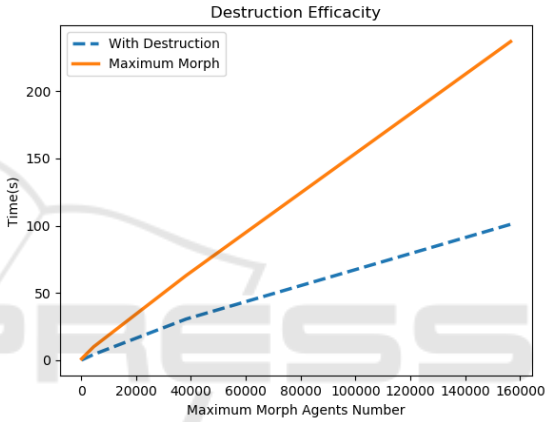


Figure 6: Difference between keeping *morph agents* and enabling them to destroy themselves.

data are available and each *data agent* has created a *morph agent* to link it with a *data agent*. During the coefficients adaptation of *morph agents*, the destruction of useless *morph agents* decreases the computation time as seen in figure 4. A peak has arisen at cycle 17 and then the number of remaining *morph agents* reduces until the 35$^{th}$ cycle from which it is stabilised.

In a real case with computation time constraints, the number of data to process in one cycle should be limited. A solution would be to give priority to critical *morph agents* while non-critical ones would not adapt. Figure 6 assesses the effectiveness of destroying useless *morph agents*, on the computation time.

## 5 CONCLUSION AND PERSPECTIVES

In this paper, we presented the MAS-based LUDA module to address the cross-understanding in MAS. Heterogeneity of agents reference frame can be an obstacle to the cooperation resolution of a MAS in multi-sources context. Thanks to the interactions of three types of agents, the LUDA module achieves two objectives: 1) translate information from a reference frame to another one and 2) reduce the noise by grouping similar information from different reference frames. The low number of observations enables to add the LUDA module to an agent in operation.

We are currently working on using LUDA in the real context of Connected and Autonomous Vehicles (CAVs). Using LUDA in the Internal Decision Layer of a CAV should enable it to understand any communication. The different reference frames used by each vehicle constructor may impact communications between CAVs. In this context, LUDA could translate a value communicated by a CAV and enable cooperative strategies.

We plan to improve the *morph agent* learning algorithm to enable it to learn non-linear function with a small loss of effectiveness. The approximation of a

non-linear function in addition to several linear functions is currently under study. A further study will be conducted with three objectives: 1) improving the adaptation of *morph agents* to include non-linear transformations; 2) reducing the computation time of the *data agent* displacement between groups by reducing the necessity for a group to know every other group; 3) reducing initial computation time by giving a cooperative strategy to *morph agents* to select only the one that needs adaptation the most. Furthermore, group values adaptation can be improved by estimating the noise of each *data agent*.

## ACKNOWLEDGEMENTS

This work is part of the neOCampus operation of the University Toulouse III Paul Sabatier. www.neocampus.org

## REFERENCES

Duda, R. O., Hart, P. E., et al. (2006). *Pattern classification*. John Wiley & Sons.

Efron, B. (1994). Missing data, imputation, and the bootstrap. *Journal of the American Statistical Association*, 89(426):463–475.

Georgé, J.-P., Gleizes, M.-P., and Camps, V. (2011). Cooperation. In Di Marzo Serugendo, G., Gleizes, M.-P., and Karageorgos, A., editors, *Self-organising Software: From Natural to Artificial Adaptation*, pages 193–226. Springer, Berlin Heidelberg.

Gibbons, J. D., Olkin, I., and Sobel, M. (1979). An introduction to ranking and selection. *The American Statistician*, 33(4):185–195.

Hall, D. L. and Llinas, J. (1997). An introduction to multisensor data fusion. *Proceedings of the IEEE*, 85(1):6–23.

Heaton, J. (2008). *Introduction to Neural Networks for Java, 2nd Edition*. Heaton Research, Inc., 2nd edition.

Hoc, J.-M. and Carlier, X. (2002). Role of a common frame of reference in cognitive cooperation: sharing tasks between agents in air traffic control. *Cognition, Technology & Work*, 4(1):37–47.

Kumar, K., Parida, M., and Katiyar, V. (2013). Short term traffic flow prediction for a non urban highway using artificial neural network. *Procedia - Social and Behavioral Sciences*, 104.

Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., and Liu, H. (2017). Feature selection: A data perspective. *ACM Computing Surveys*, 50(6):1–45.

Liu, T., Wei, H., and Zhang, K. (2018). Wind power prediction with missing data using gaussian process regression and multiple imputation. *Applied Soft Computing*, 71:905 – 916.

Loke, S. W. (2019). Cooperative automated vehicles: A review of opportunities and challenges in socially intelligent vehicles beyond networking. *IEEE Transactions on Intelligent Vehicles*, 4(4):509–518.

Marwala, T. (2009). *Computational Intelligence for Missing Data Imputation, Estimation, and Management: Knowledge Optimization Techniques*. IGI Global, 1st edition.

Montgomery, D. C., Peck, E. A., and Vining, G. G. (2012). *Introduction to linear regression analysis*. John Wiley & Sons.

Niel, C. and Sinoquet, C. (2018). Optimisation par colonie de fourmis pour la sélection de variables par construction stochastique de couverture de Markov - Application pour la médecine de précision. In *19th edition of the annual congress of the French Society for Operation Research and Decision Assistance, ROADEF2018*, Lorient, France.

Ryzhov, I. O., Defourny, B., and Powell, W. B. (2012). Ranking and selection meets robust optimization. In *Proceedings of the 2012 Winter Simulation Conference (WSC)*, pages 1–11.

Serugendo, G. D. M., Foukia, N., Hassas, S., Karageorgos, A., Mostéfaoui, S. K., Rana, O. F., Ulieru, M., Valckenaers, P., and Van Aart, C. (2003). Self-organisation: Paradigms and applications. In *International Workshop on Engineering Self-Organising Applications*, pages 1–19. Springer.

Van Buuren, S. (2018). *Flexible imputation of missing data*. CRC press.

Wang, H., Nie, F., and Huang, H. (2013). Multi-view clustering and feature learning via structured sparsity. In Dasgupta, S. and McAllester, D., editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28, pages 352–360, Atlanta, Georgia, USA.

Wooldridge, M. (2009). *An introduction to multiagent systems*. John Wiley & Sons.

Zhang, L. and Suganthan, P. (2016). A survey of randomized algorithms for training neural networks. *Information Sciences*, 364-365:146 – 155.

Zhao, Z. and Liu, H. (2008). Multi-source feature selection via geometry-dependent covariance analysis. In Saeys, Y., Liu, H., Inza, I., Wehenkel, L., and de Pee, Y. V., editors, *Proceedings of the Workshop on New Challenges for Feature Selection in Data Mining and Knowledge Discovery at ECML/PKDD 2008*, volume 4, pages 36–47, Antwerp, Belgium. PMLR.