EASR

Engineering and Applied Science Research

https://www.tci-thaijo.org/index.php/easr/index

Published by the Faculty of Engineering, Khon Kaen University, Thailand

Analysis of temporal consistency management protocols for resource-constrained multi-agent systems

Mohamed Limame*[1)], Julien Henriet[2)], Christophe Lang[2)] and Nicolas Marilleau[1)]

[1)]UMMISCO, IRD (Institut of Research for the Development), Sorbonne University, Bondy, 93143, France
[2)]FEMTO-ST Institute, Franche-Comté University, Besançon, CNRS, 25000, France

Received 18 March 2021
Revised 26 April 2021
Accepted 18 May 2021

Abstract

Multi-Agent Systems (MAS) are distributed system composed of a set of connected nodes, that communicate and share data. Since nodes do not have access to a central clock, the consistency of shared data is one of the important issues raised by this type of system. Approaches to establish temporal consistency of these shared data have been designed and implemented in the past. This paper presents and analyses existing protocols for managing temporal consistency in order to identify those that are in line with the specificities of weakly adjoining MAS with limited resources and connectivity. We propose a classification and a comparative analysis of these protocols for use within this category of MAS.

Keywords: Consistency management, Temporal consistency, Shared data, Synchronization, Multi-agent systems, Multi-agent systems with limited resources and connectivity

1. Introduction

In Multi-Agent Systems (MAS), time is of much importance in the interpretation of collected and exchanged knowledge, and in the determination of the agents' behavior, especially during data exchanges. In this article, we are interested in approaches derived from distributed systems ensuring temporal consistency of shared data. Thus, we propose a classification and a comparative analysis of protocols from the literature on distributed systems that can be used in constrained MAS with limited computing resources, data storage, energy and connectivity. We are interested in this category because some domains as the monitoring domain (the application domain of our research) are considered as sensitive domains in which resources of the used machines have to be preserved as much as possible, especially the energy which can be decisive or even vital for the achievement of the desired application objectives.

According to [1], a MAS is a distributed system composed of autonomous entities (called agents). They share knowledge and interact with each other to reach a collective goal beyond the scope of a single agent as illustrated in the Figure 1 below. However, it raises many scientific challenges such as temporal consistency management. Indeed, data separately collected, stored as distributed agent knowledges must be consistent to safe their gathering and to build a collective knowledge.

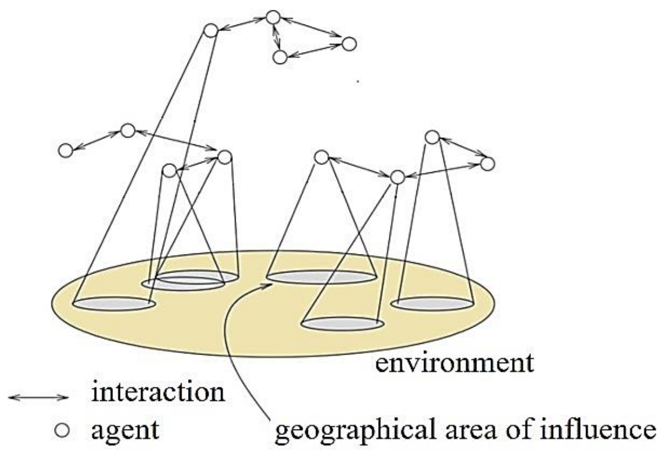

Figure 1 Agents' interactions and knowledge's share

*Corresponding author.
Email address: mohamed.limame@etu.sorbonne-universite.fr
doi: 10.14456/easr.2022.7

According to [2], a distributed system is a set of autonomous computers connected by a computer network that appears to its users as a single coherent system. These computers share information and resources over a wide geographical area, which in some applications has given rise to problems of temporal consistency, particularly in real-time applications used in factories, aeronautics, space vehicles or military applications: more broadly in version management and control of concurrent access in distributed database systems.

In distributed systems, several approaches have been identified to address this issue. It is imperative to understand the characteristics of the agents in order to distinguish among the approaches used in distributed systems those that are in adequacy with their needs and constraints.

The main contribution of this article is a classification of the most used synchronization approaches and algorithms in distributed systems taking into account the specificities of multi-agent systems. This classification work will help researchers in the choice of the most appropriate synchronization technique to meet their MAS limited resources and connectivity constraints.

In what follows, we will define consistency and introduce the temporal consistency. We will then present synchronization approaches that we consider interesting to establish temporal consistency within a MAS. Then we define the classification criteria before presenting the qualification of the selected approaches according to these criteria. A comparative analysis will synthesize the whole of our study.

2. Consistency: concept and methods

In this part, we describe what the consistency is and we give some ways in order to maintain this consistency.

2.1 Consistency definitions

There are several definitions of consistency in the literature as confirmed by [3, 4] defines consistency as a means of measuring the absence of the concept of negative affectivity which is similar to neurosism. In linguistics, according to [5, 6], a discourse (text or dialogue) can be qualified as coherent when its parts "go together" and thus structure the discourse. [7, 8] brings an epistemological vision by defining consistency as a cognitive theory translating a level of constraints satisfaction contained in individual knowledge in its different forms: explanatory, analogical, deductive, perceptual and conceptual.

In distributed systems, according to [9] the concept of consistency can be seen as the conservation of an explanatory or conceptual logic ensured by an algorithm and constraints. Its management is essential to avoid calculation errors on the one hand and anomalies of algorithm execution on the other hand.

In distributed systems, consistency is usually global and associated with the whole system. In multi-agent systems, for their part, according to [10], the system's consistency is defined by the compatibility of collected and stored information by each agent. It is, therefore, defined at the individual agent level. In this case, MAS consistency is an emergent consistency that results from the individual consistencies.

Thomesse [11] introduces temporal consistency in a distributed system by the original concept of "time window". Indeed, the notion of temporal consistency that is applied to a set of information aims at ensuring that the information belonging to this set has been produced in the same time window. According to [12], two types of temporal consistency can be distinguished in distributed systems with respect to the accuracy requirement level of the time notion. Applied to SMAs, it is possible to define a strict temporal consistency in which all the elements and knowledge held by all the agents have exactly the same time stamp and a non-strict temporal consistency in which differences in the stamping of elements and knowledge may exist between agents. Temporal consistency at the global level is obtained when the temporal consistency of each element of the multi-agent system is achieved.

2.2 Methods

Among all the approaches that can be applied to establish temporal consistency, there is the approach that uses a notion of time synchronization between all agents through the help of a synchronization protocol on the one hand, and the correction approach applied by pairs of agents following the use of a conflict resolution technique on the other hand. In this case, one of the two involved agents re-orders the values of the shared data.

It is possible to distinguish different tools and methods for solving conflicts related to the consistency of shared data. According to [13] arbitration is a means used by MAS to solve conflicts and leads to the definition of behavioral rules that act as constraints on all agents. Ferber [13] also proposes the elaboration of a priority agreement. The definition of priority rules avoids the appearance of conflicts. However, in case of conflict, the system will use the unequal weight of the agents. It is also possible to establish a voting and election procedure. According to [14], votes in MAS are methods allowing each agent to express its preferences among possible decisions. Finally, the negotiation is a communication process to reach a mutually accepted agreement according to [15].

Different synchronization protocols are based on physical and logical clocks. A physical clock is a physical process coupled with a time measurement method. Most physical clocks are based on cyclic processes based on an oscillator and a counter. A logical clock is a time-stamping method that records the chronological and causal relationships in a distributed system to establish events global classification of the system actions and processes.

2.2.1 Physical clock synchronization algorithms

The concept of synchronization is based on a physical clock source whose data is propagated in a network. The NTP protocol, Cristian's and Berkeley's algorithms are based on physical clocks.

The NTP protocol uses a semi-stratified time sources hierarchical system based on UTC (Coordinated Universal Time) as time reference. As described in [16] each level of this hierarchy is called a "stratum" and is assigned a number. A server synchronized on a server of stratum *n* runs at stratum *n+1*.

Cristian's algorithm [17] is a clock based algorithm used to synchronize the local clock of a distributed system element with a remote external time server. A system element sends a request to the time server in order to receive the updated time.

The Berkeley algorithm is described in [18]. Unlike Cristian's algorithm, the time server is active. It periodically interrogates each element of the distributed system on its local clock. On the basis of the responses, it calculates an average time and asks all elements of the system in order to advance or slow down their clocks to the new calculated clock.

The implementation of the NTP protocol in a MAS requires the setting up of an inter-agent connection respecting the NTP architecture to send time data and requires continuous connectivity of the stratum n°1 agents to a reference time (Coordinated Universal Time). For a use of Berkley and Cristian algorithms in a MAS, it is necessary to define one agent as the server agent and to ensure continuous connectivity of this agent with all the other agents for the synchronization process.

2.2.2 Logic clock synchronization algorithms

The concept of a logical clock was initially introduced in [19]. Several authors [20-22] have then proposed the use of logical clocks to detect the causal precedence relationship between events.

Lamport's clock [19] shows that synchronization does not necessarily have to be absolute and that it can be deduced from relationships between events. The proposed scheduling allows to assign a logical clock or stamp to all events in a distributed system.

With **Mattern's clock** [20], each component *p* of the distributed system has an integer vector called *stamp* in which each component *stamp[i]* is the estimation by *p* of the Lamport clock value of process *i*.

With **Matrix clocks** [23], each process *p* of a distributed system of *n* processes has an *n x n* matrix of *stamps* in which each component *stamp[i]* is the estimation by *p* of the Mattern clock value of process *i*.

For an implementation of these approaches in a MAS, it is necessary to set up an exchange between agents according to the described principles of exchange between processes. The stamp (s) must be sent during inter-agent interactions to ensure synchronization.

2.2.3 Clock synchronization in wireless sensor networks

Various applications using wireless sensor networks, especially in monitoring and data fusion, require that all the nodes have synchronized clocks. However, the synchronization methods traditionally used in industry are not necessarily suitable for a use in sensor networks due to problems related to energy consumption. For example, the NTP protocol, although widely used for clock synchronization over the Internet, is not suitable for a use in wireless networks because it is too power-hungry. New approaches adapted for a use in this type of networks have thus emerged in recent years.

Reference Broadcast Synchronization Protocol (RBS) is based on a receiver-to-receiver synchronization scheme. As described in [24], each node synchronizes its local clock with all the other clocks of the nodes within its transmission range. The nodes in the transmitter transmission range record the local clock that receives its message. Then the receivers exchange the registered reception clocks with each other. Thus, each receiver can estimate the offset of its clock by eliminating transmission latencies.

Time Synchronization Protocol for Sensor Networks (TPSN) is based on a transmitter-receiver synchronization scheme. The algorithm described by [25] works in two steps. In the first step, a hierarchical structure is established in the network, assigning each node a level in a synchronization tree. Then comes a second synchronization step during which the *i* level nodes are synchronized with the *i-1* level nodes pair by pair.

Flood Time Synchronization Protocol (FTSP) [26] implements a synchronization in which the root node periodically transmits a unique synchronization message to the nodes that are within transmission range. From the timestamp value at transmission and the one at reception, a node can determine its clock deviation from the root node. If a node does not receive synchronization messages for a certain period of time, it declares itself the new root.

Delay measurement time synchronization for wireless sensor networks (DMTS) is based on transmitter-receiver synchronization in which the sender and several receivers are synchronized at the same time. In this protocol [27], a master node is chosen as time server and broadcasts its clock. All receivers within range measure the transmission delay and set their local clock by assigning to it the received master clock plus the estimated transfer delay.

Consensus Time Synchronization (CCS) technique aims at reduce clock gaps between nearby nodes and converges all nodes to a common gap. The main idea of this algorithm [28] is to compensate over several iterations the clock gaps between the nodes of the system using the average time synchronization algorithm described in [29]. The nodes broadcast their local clock in the network so that each node can estimate the rate of clock deviation. Then, the nodes broadcast their estimated rate of virtual clock deviation, allowing the receiver nodes to combine this with their relative deviation estimates to adjust their own virtual clock. Thus, all nodes then converge asymptotically to the same clock.

Gradient Time Synchronization Protocol (GTSP) was introduced in [30]. In this protocol, network nodes periodically broadcast a synchronization beacon with their neighboring nodes. Using a simple update algorithm, they agree on a common logical clock with their neighbors. There is no reference node. Clock synchronization is done by pairs of nodes, both parties must agree on a common logical clock frequency and an absolute value of the logical clock. By applying this approach, the logical clock of each node converges to a common logical clock.

Reachback Firefly Algorithm (RFA) is inspired by synchronization in large biological swarms where individuals follow simple coordination strategies. The canonical example is the synchrony of fireflies observed in parts of Southeast Asia. The behavior of these systems can be modeled as an array of pulse-coupled oscillators where each node is an oscillator that periodically emits a self-generated pulse. An example of modeling is presented in [31]. By observing the pulses of the other oscillators, a node slightly adjusts the phase of its own oscillator. This simple feedback process allows nodes to closely align their phases and achieve synchronization.

Since wireless sensor networks can be considered as particular MAS, for the implementation of these protocols in a MAS, we just need to apply the same processing performed by wireless sensor nodes to the MAS agents level.

2.2.4 Synchronization using data

Synchronization of multi-agent systems based on data evolution (SMASDEV) [32] relies on the content of an event and its data evolution to sort events. The aim of this new approach is to establish overall consistency at the level of a MAS by enabling each agent to re-establish a chronological order of the data it receives from other agents, based on its knowledge and without using a clock (physical or logical). In SMASDEV, each agent records in memory its personal perception of the data evolution that it has collected with a certain frequency in relation to his local clock. The agent can thus predict the data future evolution. Thus, the agent must have a model of evolution in adequacy with the monitored data. In order to move from a personal perception to a global perception, each agent of the system must first ask the other agents to transmit it the collected data and then place them in relation to its personal perception. In this way, the agent will can identify the positioning of all received data in relation to his clock. As a result, the agents will be in phase with the data evolution model over time.

Optimistic Pilgrim Protocol [33] is a consistency management protocol of share data in a distributed system with replicated data. It works on the basis of a token circulating on a unidirectional logical ring and containing a data structure with updates of the shared data. These data are transported by the token called *Pilgrim*. As it passes over one node, the Pilgrim brings the updates from the other nodes and retrieves the values that have been locally modified to disseminate them through the the cooperating sites. In this protocol, only the owner of a shared data can deposit the modifications of this data on the token. For the implementation of this approach in an SMA, the agents represent the nodes and the connectivity must be continuous between the agents and the token resource, otherwise the synchronization cannot take place.

3. Methods classification with evaluation criteria

3.1 Criteria definition

For an effective relevance measurement of each synchronization approach, the chosen classification criteria must sufficiently characterizing constraints in a MAS with limited resources and connectivity. We distinguish two families of criteria:

The "synchronization details" criteria family qualifies the synchronization approaches and the obtained synchronization level. It concerns the synchronization principle, the type of consistency, the synchronization object, the type of synchronization and the fault tolerance.

The "energy" criteria family allows to qualify the synchronization approaches needs especially in terms of memory and power. This family concerns the performed calculation, the volume exchanged, the number of required messages to synchronize data between 2 agents and the number of required messages to synchronize data between *N* agents. These indicators allow to estimate the required energy by the synchronization approach.

3.1.1 The "synchronization details" criteria family

The synchronization principle specifies the synchronization mechanism used by the approach.

The consistency type allows to know the synchronization level in the network between agents. Among consistency's types found in literature [34, 35], we consider the four following types:

The atomic consistency level reflects a strict global synchronization degree among all the agents in the network. At the end of a synchronization cycle, when each agent has received the shared data, all agents in the network must have the same system state. Synchronization at the atomic level is quite complex to deploy in a distributed system.

The sequential consistency level reflects L. Lamport's vision of synchronization and focuses on the events order, not the occurrence time of the event. At the end of a synchronization cycle, when each agent has received the shared data, all the agents on the network must present the same shared events order. This is therefore a less strict degree of synchronization than the atomic level.

The causal consistency level is based on the causal relationship between the events occurring in the network of agents. At the opposite to the sequential consistency level, at the end of a synchronization cycle, when each agent has received the shared data, all the agents in the network do not necessarily have the same events order because the events synchronization is only based on the events linked by a causal relationship. The remaining events are therefore not known by all the agents.

The released coherence level reflects the lowest degree of synchronization compared to those mentioned above.

The synchronization object specifies the nature of the synchronized element(s). The obtained consistency through synchronization is a temporal consistency if the synchronized element is a "time" data.

The synchronization type qualifies the synchronization process in relation to its triggering: whether it is systematic or non-systematic.

The fault tolerance criterion indicates whether the synchronization approach supports the presence of one or more failures in the agent fleet.

3.1.2 The "energy" family of criteria

The realized calculation criterion characterizes the operations needed to launch the synchronization approach.

The exchanged volume characterizes the required volume exchanged between two agents to ensure the synchronization.

The number of required messages to synchronize a data item between 2 agents is the number of exchanged messages required to ensure the synchronization between two agents.

The number of required messages to synchronize data between *N* agents is the number of exchanged messages required in order to ensure the synchronization between *N* agents. The number of exchanged messages allows to quantify the agent energy consumption of the execution of the synchronization approach.

3.2 Classification of the studied synchronization approaches

Our methods' classification is presented in the Table 1 below. It is also summarized in Figures 2 and 3 below. Five of the fifteen approaches considered in this paper support fault tolerance by default. Even if the rest of the approaches do not have this feature, it can be deployed at the time of their implementation.

For the studied synchronization approaches based on clocks or data exchange, the network topology does not usually constitute an obstacle for these approaches. Indeed, except for the NTP and TPSN approaches which need a hierarchical structure, the agents in the rest of studied approaches must be able to communicate with each other independently of the topology.

All the studied approaches, based on physical or logical clocks, ensure synchronization between two elements by sending one to three messages. In the approaches used in wireless sensor networks, a synchronization between two nodes can be achieved with one or two messages.

We find that for the RFA and DMTS approaches, the number of sent messages needed for the MAS global synchronization does not vary by increasing or decreasing the number of agents. In fact, the sender sends a single broadcast message to all the agents.

In wireless sensor networks, the studied approaches require a limited amount of memory space in order to achieve temporal consistency.

Table 1 Classification of the studied synchronization approaches

	The « synchronization's details » criteria's family						**The «energy» criteria's family**		
	Synchr. principle	**Consi. type**	**Synchr. object**	**Syst. sync**	**Fault toler.**	**Realized calculation**	**Exchange volume**	**Nbr msg to synchr. 2 agents**	**Number of messages to synchr. N agents**
Physical clock synchronization algorithms									
NTP	sharing one stamp	R	clock	x	x	op subtraction + op division	stamp	2	n-m+l n receiver level, m transmitter level
Cristian's Algorithm	through a server time	S	clock	x		op addition + op division	Round-trip + server clock time	2	2n
Berkeley's Algorithm	through a server time	R	Offset	x	x	op addition + op division	server: n clocks	3	3n
Logic clock synchronization algorithms									
Lamport clocks	sharing one stamp	C	Data + stamp	x		op comparison + op increment	local stamp and synch	1	n-l
Mattern clocks	share a vector of stamps	S	vector of stamps	x		n-l op comparison + op increment stamp	vector local stamps and synch	1	n-l
Matrix clocks	share a matrix. Stamps	S	Matrix of stamps	x		n-l op comparison +op increment stamp	matrix local stamps and synch	1	n-l
Clock synchronization in wireless sensor networks									
RBS	clock sharing	R	transmitter: data; receiver: clock			Add. + div. + offset calcul.	Shared data + k clocks (k neighbors)	2	1+(n-1)*k; k: nbr neighboring nodes to (n-1) nodes
TPSN	sharing one stamp	R	Synchronization packet.			Tree-treeing + new clock calculation	a stamp	2	n-m+l stamp with n receiver level and m transmitter level
FTSP	clock sharing	R	ID + root stamp + data	x	x	Offset calculation	stamp + shared data	2	1+k : 1 msg transmitter + k msg receivers to neighborhood
DMTS	sharing one stamp	R	Root stamp			calculation new clock	stamp	1	1 msg from sender to receiver
CCS	sharing one stamp	R	stamp and deviation rate	x	x	calculation rate of deviation	stamp + rate of deviation	2 * iterations number	2*(n-1) * number of iterations
GTSP	clock sharing	R	synch beacon and clock freq.	x	x	update clock frequency + calculate. New clock	clock + clock frequency	2	2*(n-1)
RFA	sharing a state	S	state vector	x		state vector and objective fct calculate.	Data synchr. Status vector	1	1 msg from sender to receivers
Synchronization using data									
Smasdev	data assimilation	C	Data with evolution model			data manip. & simul. Of evol.	Shared data tuplet	1	n-l
Optimistic pilgrim	resources reservation and distribution via token	A	Shared Data			jeton manipulation	Shared Data	6	3n+1

The values in the column "Consi. type" are presented with 'A' for atomic consistency, 'S' for sequential consistency, 'C' for causal consistency and 'R' for relaxed consistency. The criterion "synchronization type" is represented by the column "Syst. sync". The value 'X' is filled in only if the consistency is systematic. In the same way for the column "Fault toler.", the value 'X' is only filled in if the fault tolerance is supported by the synchronization approach. In the column «Realized Calculation", 'op' designates operation and 'n' refers to the agents' number in the MAS.

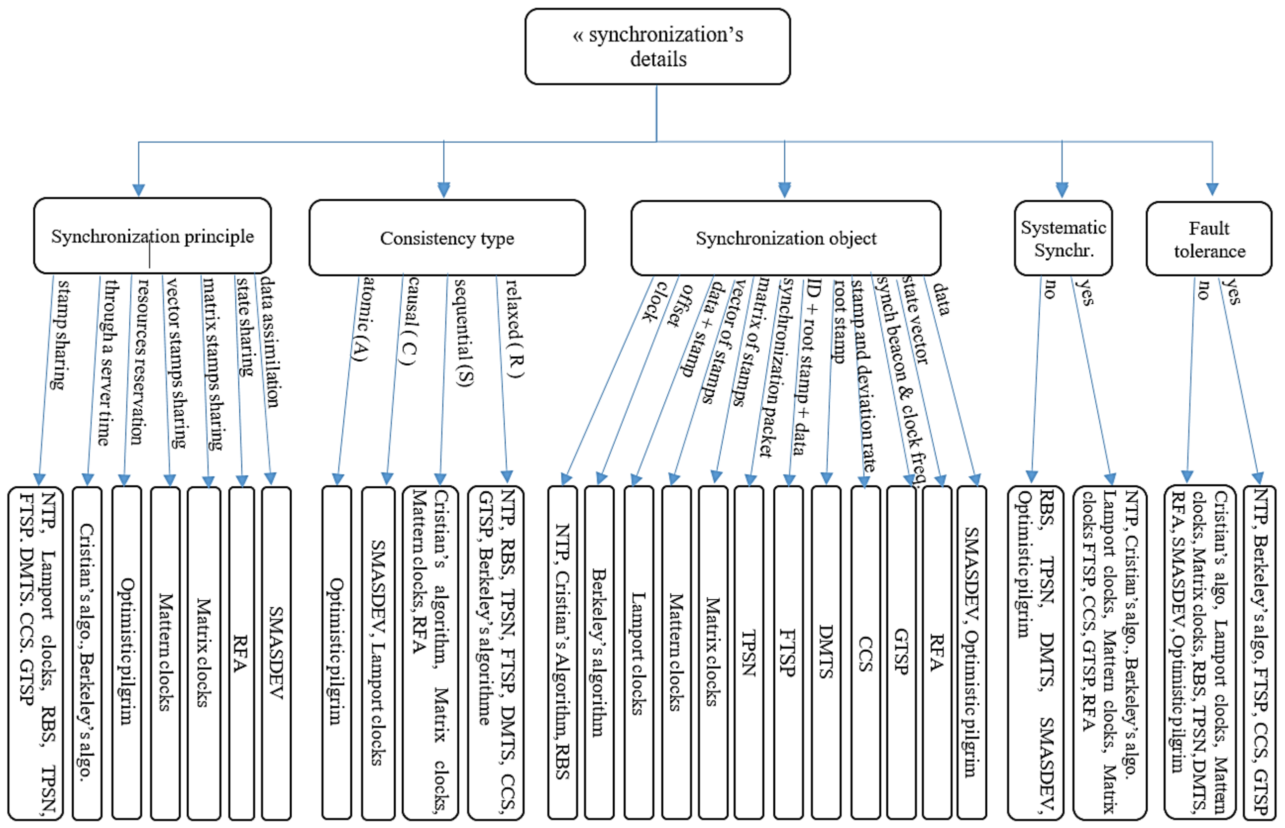
« synchronization's details
Synchronization principle
stamp sharing
through a server time
resources reservation
vector stamps sharing
matrix stamps sharing
state sharing
data assimilation
NTP, Lamport clocks, RBS, TPSN, FTSP, DMTS, CCS, GTSP
Cristian's algo., Berkeley's algo.
Optimistic pilgrim
Mattern clocks
Matrix clocks
RFA
SMASDEV
Consistency type
atomic (A)
causal (C)
sequential (S)
relaxed (R)
Optimistic pilgrim
SMASDEV, Lamport clocks
Cristian's algorithm, Matrix clocks, Mattern clocks, RFA
NTP, RBS, TPSN, FTSP, DMTS, CCS, GTSP, Berkeley's algorithme
Synchronization object
clock
offset
data + stamp
vector of stamps
matrix of stamps
synchronization packet
ID + root stamp + data
root stamp
stamp and deviation rate
synch beacon & clock freq
state vector
data
NTP, Cristian's Algorithm, RBS
Berkeley's algorithm
Lamport clocks
Mattern clocks
Matrix clocks
TPSN
FTSP
DMTS
CCS
GTSP
RFA
SMASDEV, Optimistic pilgrim
Systematic Synchr.
no
yes
RBS, TPSN, DMTS, SMASDEV, Optimistic pilgrim
NTP, Cristian's algo., Berkeley's algo. Lamport clocks, Mattern clocks, Matrix clocks FTSP, CCS, GTSP, RFA
Fault tolerance
no
yes
Cristian's algo, Lamport clocks, Mattern clocks, Matrix clocks, RBS, TPSN, DMTS, RFA, SMASDEV, Optimistic pilgrim
NTP, Berkeley's algo, FTSP, CCS, GTSP

Figure 2 Methods' classification's summary for the "synchronization details" family of criteria.

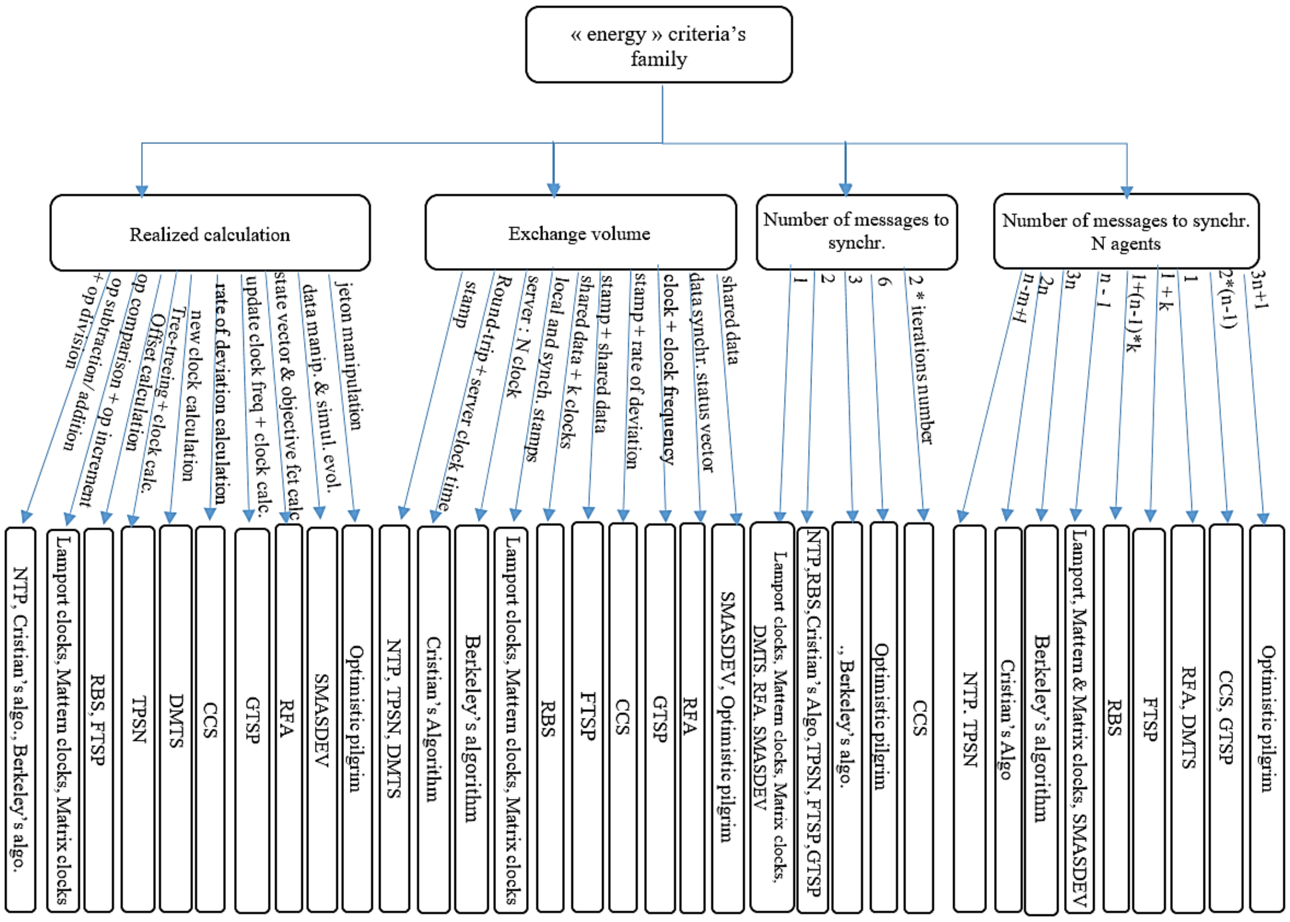
« energy » criteria's family
Realized calculation
+ op division
op subtraction/ addition
op comparison + op increment
Offset calculation
Tree-treeing + clock calc.
new clock calculation
rate of deviation calculation
update clock freq + clock calc.
state vector & objective fct calc
data manip. & simul. evol.
jeton manipulation
NTP, Cristian's algo., Berkeley's algo.
Lamport clocks, Mattern clocks, Matrix clocks
RBS, FTSP
TPSN
DMTS
CCS
GTSP
RFA
SMASDEV
Optimistic pilgrim
Exchange volume
stamp
Round-trip + server clock time
server : N clock
local and synch. stamps
shared data + k clocks
stamp + shared data
stamp + rate of deviation
clock + clock frequency
data synchr. status vector
shared data
NTP, TPSN, DMTS
Cristian's Algorithm
Berkeley's algorithm
Lamport clocks, Mattern clocks, Matrix clocks
RBS
FTSP
CCS
GTSP
RFA
SMASDEV, Optimistic pilgrim
Number of messages to synchr.
1
2
3
6
2 * iterations number
Lamport clocks, Mattern clocks, Matrix clocks, DMTS, RFA, SMASDEV
NTP,RBS,Cristian's Algo,TPSN,FTSP,GTSP
., Berkeley's algo.
Optimistic pilgrim
CCS
Number of messages to synchr. N agents
n-m+1
2n
3n
n - 1
1+(n-1)*k
1 + k
1
2*(n-1)
3n+1
NTP, TPSN
Cristian's Algo
Berkeley's algorithm
Lamport, Mattern & Matrix clocks, SMASDEV
RBS
FTSP
RFA, DMTS
CCS, GTSP
Optimistic pilgrim

Figure 3 Methods' classification's summary for the "energy" family of criteria.

The storage requirement for data-based approaches depends on the synchronization object size and the result of the function used for synchronization.

Among the studied approaches, only the Optimistic Pilgrim allows atomic consistency.

Nine of the fifteen studied approaches, independently of their fields of application, allow to perform data synchronization in addition to the time notion of synchronization (stamp, clock,..).

All of the studied synchronization approaches based on physical and logical clock exchanges systematically trigger on the occurrence of events. Whereas some of the studied approaches on wireless sensor networks such as RBS, TPSN and DMTS aren't systematicaly launched when events occur.

4. Results and discussion

This classification lets one determine which synchronization technique is best suited to his/her application constraints in MAS with limited resources and connectivity. In the case of agents with limited resources, their computing capacity may also be limited, memory space may be reduced, and message exchanges may be restricted. The more the synchronization approach needs are reduced, the longer the approach can be applied. This point can be crucial, especially in cases of application on routes and paths, such as during mapping or monitoring missions. This interpretation is detailed in the following section.

Concerning memory space, the synchronization approach based on matrix clocks is characterized by a large memory space requirement. Indeed, the storage of all the matrices associated with each event (local or received) is costly. Similarly, with a lesser degree, the synchronization using Mattern clocks has a relatively expensive memory space requirement. The use of these approaches can have a negative impact on the robustness of the system. On the other hand, DMTS, NTP, TPSN and Cristian’s algorithm are characterized by a relatively low memory space requirement. Their need consists essentially in storing, at the receiving agent level, the clock or time stamp to be synchronized.

The Lamport’s clock, the Berkeley Algorithm, RBS, FTSP, CCS and the Optimistic Pilgrim protocol require a small computing capacity reduced to the power needed to compute time averaging, the offset or writing on a token. The Cristian and Berkeley algorithms are two approaches adapted to MAS in which it is possible to manage time centrally.

In terms of energy consumption, it is the CCS synchronization approach that requires the largest number of messages, since messages are sent to all the agents in the network over several iterations until shared data gaps are eliminated. This can be a direct factor of agent energy exhaustion. Moreover, the CCS synchronization type is systematic, making it costly in terms of energy consumption.

Similarly, the NTP, FTSP, Cristian and Berkeley algorithms being executed in a regular and periodic way, are likely to consume a lot of energy if the launch frequency is important. Concerning the studied approaches that are launched at the event occurrence (RFA or Mattern’s clocks for example), the energy consumption increases slightly at each launch, which is suitable for the use in a multi-agent system with limited resources and connectivity.

Energy consumption for SMASDEV approach for which synchronization is not systematic, remains relatively low and increases proportionally with the complexity of the shared data evolution model and the distribution density of the agent network.

Although the seven studied approaches used in wireless sensor networks can be applied in a MAS, it would be preferable to limit as much as possible the frequency of their execution in an environment in which agents have limited resources and connectivity.

Based on Lamport's approach, the comparison of clock values does not allow to deduce a causal relationship between two *a* and *b* events. Indeed, if $H(a)<H(b)$ does not necessarily mean that $a \rightarrow b$. Moreover, in case of a high traffic, messages may not be received. With these limitations, we propose not to use it in a MAS with limited resources and connectivity.

The choice of synchronization approach can also be determined according to the nature of the shared data. For example, if the shared data only concerns the notion of time (stamp, time matrix, ...), the choice can be one of the TPSN, DMTS, CCS or GTSP approaches designed to establish temporal synchronization and in adequacy with the studied MAS (limited ressources and connectivity). On the other hand, if the shared data does not concern the notion of time, we propose the use of the RFA, Smasdev, RBS or FTSP approaches which are in adequacy with the studied MAS and which are sorted according to the order of degree of coherence from the strictest to the least strict. We discard from this list the Optimistic Pilgrim requiring that all agents can communicate continuously and therefore not very compatible with an MAS made up of weakly accointing agents.

Protocols that require the least resources have to be privileged for MAS with limited resources and connectivity. Beyond these considerations, the synchronization approach to use depends strongly on the synchronization context and the required level of synchronization.

5. Conclusions

This article presents a study of different possible solutions to the problem of temporal synchronization of shared data between agents amont a MAS with limited resources and connectivity. We have presented and compared several approaches used in distributed systems to solve the lack of a centralized clock. We analyzed them for the use in a MAS with limited resources and connectivity. The approach has consisted in starting from existing solutions in the universe of distributed systems in order to identify those which are in adequacy with the constraints of the studied multi-agent system, in particular the limitation of resources.

In order to carry out this work, we first identified evaluation criteria that characterize the various constraints of a MAS in terms of resources, architecture but also quality of synchronization. Then, we classified the identified synchronization approaches according to these criteria to end with an analysis of the obtained results. In addition to the used criteria to qualify the resource requirements of an approach and to answer the question of the suitability of the approach to the studied multi-agent system, the rest of the evaluation criteria provide a precise assessment of the synchronization approach quality.

In the light of this study, we now plan to test and compare performances of the approaches that appeared most relevant and adapted to this type of MAS. Thus, we plan to develop a multi-agent simulator for different application areas such as territory surveillance using a fleet of UAVs.

6. Acknowledgements

We are grateful to all of those with whom we have had the pleasure to work during this and other related projects.

7. References

[1] Chaib-Draa B. Industrial applications of distributed AI. Commun ACM. 1995;38(1):49-53.
[2] van Steen M, Tanenbaum AS. Distributed systems. 3rd ed. London: Pearson Education; 2017.
[3] Martin B, Kjell H, Ann-Marie L, Marianne C. Sense of coherence: definition and explanation. Int J Soc Welfare. 2006;15(3): 219-29.
[4] Strumpfer D, Gouws J, Viviers M. Antonovsky's Sense of coherence scale related to negative and positive affectivity. Eur J Pers.1998;12(6):457-80.
[5] Grosz B, Sidner C. Attention, intentions, and the structure of discourse. Comput Ling. 1986;12(3):175-204.
[6] Carbaugh D, Craig RT, Tracy K. Conversational coherence: form, structure, and strategy. Lang Soc. 1989;18(1):103-9.
[7] Thagard P. Coherence in thought and action. Cambridge: MIT press; 2000.
[8] Thagard P. Hot thought: mechanisms and applications of emotional cognition. Cambridge: MIT press; 2008.
[9] Grottke S, Sablatnig J, Chen J, Seiler R, Wolisz A. Consistency in distributed systems. Berlin: Technische Universitat Berlin; 2007.
[10] Vu QN, Gaudou B, Canal R, Hassas S. Coherence and robustness in a disturbed MAS. IEEE-RIVF International Conference on Computing and Communication Technologies; 2009 Jul 13-17; Danang, Vietnam. New York: IEEE; 2009. p. 1-4.
[11] Thomesse JP. The fieldbuses. IFAC Proc Volume. 1997;30(7):13-23.
[12] Cauffriez L, Defrenne J. An experimental platform for the reliability-availability evaluation of intelligent distributed systems. IFAC Proc Volume. 1997;30(7):545-50.
[13] Ferber J. Multi-agent systems: an introduction to distributed artificial intelligence. Boston: Addison-Wesley; 1999.
[14] Lang J. Logical preference representation and combinatorial vote. Ann Math Artif Intell. 2004;42:37-71.
[15] Bussmann S, Muller J. A negotiation framework for co-operating agents. Proceeding of CKBS-SIG; 1992 Sep 23-25; Dake Center, University of Keele. Newcastle: University of Keele; 1992. p. 1-17.
[16] Minar N. A survey of the NTP network [Internet]. 1999 [cited 2021 Mar 10]. Available from: http://www.media.mit.edu/ ~nelson/research/ntp-survey99/.
[17] Cristian F. Probabilistic clock synchronization. Distr Comput. 1989;3:146-58.
[18] Gusella R, Zatti S. The accuracy of the clock synchronization achieved by TEMPO in Berkeley UNIX 4.3 BSD. IEEE Trans Software Eng. 1989;15(7):847-53.
[19] Lamport L. Time, clocks, and the ordering of events in a distributed system. In: Malkhi D, editor. Concurrency: the works of Leslie Lamport. New York: Association for Computing Machinery; 2019. p. 179-96.
[20] Mattern F. Virtual time and global states of distributed systems. Proceedings of the International Workshop on Parallel and Distributed Algorithms; 1988 Oct 3-6; Chateau de Bonas, France. Amsterdam: North-Holland; 1989. p. 215-66.
[21] Fidge CJ. Timestamps in message-passing systems that preserve the partial ordering. Aust Comput Sci Commun. 1988;10(1): 55-6.
[22] Schwarz R, Mattern F. Detecting causal relationships in distributed computations: in search of the Holy Grail. Distr Comput. 1994;7:149-74.
[23] Raynal M, Singhal M. Logical time: capturing causality in distributed systems. Comput. 1996;29(2):49-56.
[24] Elson J, Girod L, Estrin D. Fine-grained network time synchronization using reference broadcasts. ACM SIGOPS Oper Syst Rev. 2002;36:147-63.
[25] Ganeriwal S, Kumar R, Srivastava MS. Timing-sync protocol for sensor networks. Proceedings of the 1st International Conference on Embedded Networked Sensor Systems; 2003 Nov 5-7; Los Angeles, USA. New York: Association for Computing Machinery; 2003. p. 138-49.
[26] Maroti M, Kusy B, Simon G, Ledeczi A. The flooding time synchronization protocol. Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems; 2004 Nov 3-5; Baltimore, USA. New York: Association for Computing Machinery; 2004. p. 39-49.
[27] Ping S. Delay measurement time synchronization for wireless sensor networks. IRB-TR-03-013: Intel Research Berkeley Lab. Berkley, USA; 2003.
[28] Maggs MK, O'Keefe SG, Thiel DV. Consensus clock synchronization for wireless sensor networks. IEEE Sensor J. 2012;12(6):2269-77.
[29] Schenato L, Fiorentin F. Average TimeSynch: a consensus-based protocol for clock synchronization in wireless sensor networks. Automatica. 2011;47(9):1878-86.
[30] Sommer P, Wattenhofer R. Gradient clock synchronization in wireless sensor networks. International Conference on Information Processing in Sensor Networks; 2009 Apr 13-16; San Francisco, USA. New York: IEEE; 2009. p. 37-48.
[31] Ramirez-Avila GM, Kurths J, Depickere S, Deneubourg JL. Modeling fireflies synchronization. In: Macau EEN, editor. A mathematical modeling approach from nonlinear dynamics to complex systems. Berlin: Springer; 2019. p. 131-56.
[32] Limame M, Henriet J, Lang C. Synchronisation d'horloge dans un systeme multi-agents. France: APIA; 2019. (In France)
[33] Garcia E, Guyennet H, Henriet J. Towards an optimistic management of concurrency: a probabilistic study of the pilgrim protocol. International Conference on Computer Supported Cooperative Work in Design; 2005 May 24; Coventry, UK. Berlin: Springer; 2005. p. 51-60.
[34] Brzezinski J, Sobaniec C, Wawrzyniak D. From session causality to causal consistency. 12th Euromicro Conference on Parallel, Distributed and Network-Based Processing; 2004 Feb 11-13; Coruna, Spain. New York: IEEE; 2004. p. 152-8.
[35] Raynal M, Schiper A. A suite of formal definitions for consistency criteria in distributed shared memories. Proceedings International Conference on Parallel and Distributed Computing (PDCS'96); 1996 Sep 25-27; Dijon, France. Lausanne: Infoscience; 1996. p. 125-30.