A Topic-Based Data Distribution Management for HLA

Alberto Falcone[a] and Alfredo Garro[b]
Department of Informatics, Modeling, Electronics and Systems Engineering, University of Calabria, Via P. Bucci 41/C, Rende, Italy

Keywords: Distributed Simulation, High Level Architecture (HLA), Data Distribution Management (DDM), Data Reduction, Topic-Based Publish-Subscribe.

Abstract: Modeling and Simulation (M&S) represents a fundamental technology for designing and studying complex systems in various industrial and scientific domains, when real-world testing is too costly to perform in terms of safety, time, and other resources. To promote the reusability and interoperability of simulation models allowing them to interoperate without geographic constraints, distributed simulation has been introduced. One of the most widely adopted standards for distributed simulation is IEEE 1516-2010 - High Level Architecture (HLA). Among the services provided by HLA, a key one is the Data Distribution Management (DDM) that allows to reduce the transmission and reception of unnecessary data in order to improve communication effectiveness among simulation models. Although many matching algorithms have been proposed in the literature, the upcoming HLA 4.0 standard defines a DDM that still relies on performing matching verification by calculating the overlap between regions using their dimensions. In this paper, a novel topic-based publish-subscribe messaging system is proposed to improve the performance, reliability, and scalability of DDM services. Experiments show that the proposed topic-based approach achieves better performance than the standard one.

1 INTRODUCTION

In the last years, the complexity of Cyber-Physical Systems (CPSs) has increased exponentially, mainly due to the heterogeneity of the involved components and related interactions that connect the cyber world, through a network of interconnected computational resources (e.g., sensors, actuators, and processing units), to the physical one (Lee, 2008; Falcone et al., 2020; Falcone et al., 2022). CPSs are characterized by being highly automated, intelligent, and collaborative. The design and implementation of CPSs represent a challenging task due to the vast network and computing resources connected to the physical environment involving multiple domains such as controls, network protocols, and software engineering. To capture the structure and behavior of such systems, researchers rely on Modeling and Simulation (M&S) techniques (Bouskela et al., 2021; Derler et al., 2012; Falcone et al., 2014). M&S represents a pillar technology for designing and studying complex systems in various industrial and scientific domains, when real-world testing is too costly to perform in terms of safety, time, and other resources. To promote the reusability and interoperability of simulation models allowing them to interoperate without geographic constraints, Distributed Simulation (DS) has been introduced (Fujimoto, 2000). One of the most widely adopted standards for distributed simulation is IEEE 1516-2010 - High Level Architecture (IEEE Std. 1516-2010, 2010).

The HLA standard defines a generic architecture to support reusability and interoperability across simulation models. The standard was developed in 1996 under the guidance of the United States Department of Defense (DoD) Modeling and Simulation Coordination Office (M&S CO). After its definition, HLA caught the interest in the industrial and scientific domains, so it was later moved into an IEEE international standard with the official name “IEEE 1516”. In the HLA terminology, a distributed simulation is called *Federation*, which is composed of many HLA simulation applications called *Federates*. Federates interact with each other, in the same Federation, using the services provided by the *Run-Time Infrastructure (RTI)* that represents the communication middleware that implements the HLA interface specifications and rules. The structure and semantics of the data exchanged are delineated following the *Object Model*

[a] https://orcid.org/0000-0002-2660-1432
[b] https://orcid.org/0000-0003-0351-0869

Falcone, A. and Garro, A.
A Topic-Based Data Distribution Management for HLA.
DOI: 10.5220/0012051500003546
In *Proceedings of the 13th International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH 2023)*, pages 186-193
ISBN: 978-989-758-668-2; ISSN: 2184-2841

Template (OMT) specifications. The RTI provides six service groups to handle the distributed simulation execution: *Federation Management*, *Declaration Management*, *Object Management*, *Ownership Management*, *Time Management*, and *Data Distribution Management (DDM)*.

DDM provides a set of services to minimize the transmission and reception of unnecessary data and then maximize communication effectiveness among Federates, letting them specify the data of interest. Specifically, the interest in specific information is expressed as bounded portions of a n-dimensional space of user-defined dimensions. Both consumer and producer federates specify the upper and lower bounds for a specific portion, thus creating a so-called *region*. Federates that produce data define *update regions*; whereas, federates that consume data specify *subscription regions*.

At the core level, DDM services use an algorithm that scans all n-dimensional regions to find the pairs of regions $(update, subscription)$ that generate overlap. The RTI routes data from producer federates to consumer ones if and only if there is an overlap, i.e., a *match* between their *update* and *subscription* regions. According to the HLA standard, a match must be reported to the RTI exactly once by the DDM services. This problem is well-known in theoretical computer science and can be solved using suitable algorithms with ad-hoc spatial data structures. However, many DDM implementations tend to rely on less efficient algorithms that adopt complex spatial data structures whose manipulation may have a significant impact on computational resource utilization (Marzolla and D'angelo, 2020).

In this paper, the novel *Topic-Based Matching Algorithm (TBMA)* is proposed to improve the performance, reliability, and scalability of DDM services. TBMA defines the concept of *topic* to handle a high number of regions, where the *match* operation is performed by using topics instead of calculating overlaps between regions' coordinates.

The paper is structured as follows. Section 2 provides the problem statement along with key definitions. Section 3 discusses related work on the HLA DDM services and existing matching algorithms. Section 4 describes the proposed topic-based DDM approach. Section 5 presents experiments carried out to evaluate the performance of the proposed topic-based matching algorithm, where the simulation results have been compared with the standard one. Finally, conclusions are discussed in Section 6.

2 PROBLEM STATEMENT

The DDM services are based on the following definitions (IEEE Std. 1516-2010, 2010):

Definition 1: *Dimension*. It is a non-negative interval with an associated label. The interval is defined by an ordered pair of values $[d_{lb}, d_{ub}] := \{x \in \mathbb{R} \mid d_{lb} \leq x \leq d_{ub}\}$, with $d_{lb} \leq d_{ub}$. The interval lower bound d_{lb} is 0 for every dimension, while the upper bound d_{ub} can vary for each dimension.

Definition 2: *Range*. It is a continuous semi-open interval defined on a dimension d. It is defined by an ordered pair of values $[r_{lb}, r_{ub}) := \{x \in \mathbb{R} \mid r_{lb} \leq x < r_{ub}\}$, with $r_{ub} - r_{lb} \geq 1$. The component of the range r_{lb} and r_{ub} are known as *range lower bound* and *range upper bound*, respectively.

Definition 3: *Region Specification*. It is defined as a set of ranges, named *RS*. *RS* must contain at most one range r for any given dimension $d \in D$, where the dimension set D is derived starting from the ranges that constitute *RS*. Each range $r \in RS$ is defined in terms of bounds $[0, d_{ub})$, where d_{ub} is the corresponding dimension's upper bound.

Definition 4: *Region Template*. It is defined as an incomplete region specification in which one or more dimensions have not been assigned ranges.

Definition 5: *Region Realization*. It is defined as a region specification that is associated with an instance attribute for update, a sent interaction, or a class attribute or interaction class for subscription. The term *region* may be used in cases where a region specification, a region realization, or both apply.

The DDM process goes through four phases: (i) *declaration*, (ii) *match*, (iii) *connect*, and (iv) *forward*, which repeatedly occur throughout the federation execution (IEEE Std. 1516-2010, 2010; Zhu and Wang, 2022). In the *declaration* step, federates create regions with a specific set of dimensions, and declare the *HLAObjectClass* and/or *HLAInteractionClass* data that they intend to publish and/or subscribe, in terms of *update* and *subscription* regions, respectively. In the *match* phase, the overlap between each pair of update and subscription regions is calculated, and obtained pairs are reported to the RTI. After that, in the *connect* phase, the RTI establishes connections between the sending federate and receiving ones. Finally, In the *forward* phase, data are forwarded through the created connections.

The region matching problem can be defined as follow. Given an update region set $U = \{u_1, u_2, ..., u_n\}$ and a subscription region set $S = \{s_1, s_2, ..., s_m\}$, such that $U \bigcap S \neq \{\emptyset\}$, and $R = U \bigcup S$. An update u_i and a subscription s_j region overlap if and only if all ranges of dimensions that are contained in both regions overlap pairwise. If the regions do not have any dimensions in common, they do not overlap. Algorithm 1 delineates the steps to check for overlap between two ranges $a = [a_{lb}, a_{ub})$ and $b = [b_{lb}, b_{ub})$ for a given a dimension d.

Algorithm 1: Calculation of an overlap between ranges for a given a dimension d.

Require: d ▷ dimension
Require: a, b ▷ ranges defined both on d
1: $overlap \leftarrow \{(a_{lb} = b_{lb}) \vee (a_{lb} < b_{ub} \wedge b_{lb} < a_{ub})\}$
2: return $overlap$

Figure 1 depicts an example of a two-dimensional region matching problem composed of two subscription regions $S = \{s_1, s_2\}$ and one update region $U = \{u_1\}$ both specified on the dimensions X and Y, where, for example, X and Y can represent latitude and longitude, respectively (Möller et al., 2016). It is easy to identify that the pair $\{(s_2, u_1)\}$ is the solution to the region matching problem, since their ranges overlap on both dimensions. The pair $\{(s_1, u_1)\}$ is not part of the solution; therefore, s_2 will receive data from u_1 while s_1 will not.

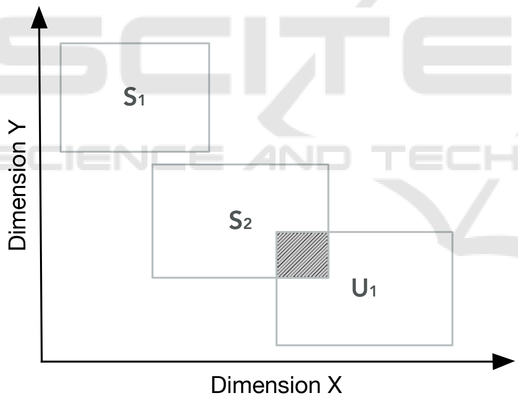


Figure 1: An example of the region matching problem in a two-dimensional space.

The correct definition of the region matching algorithm is crucial for the network and processing cost (Raczy et al., 2005). This problem can be reduced to the rectangles intersection problem. To find the intersections between two-dimensional rectangles, in (Bentley and Wood, 1980), the authors developed an algorithm with computational complexity equal to $O(N \cdot log(N) + K)$, where N is the number of two-dimensional rectangles and K is the number of intersecting pairs discovered. However, the proposed algorithm can only be applied to two-dimensional rectangles and does not offer a generic solution for n-dimensional rectangles.

The following section reviews the existing literature contributions dealing with both the DDM services and region matching algorithms.

3 RELATED WORK

Four relevant DDM matching algorithms have been developed, i.e., *Region-Based Matching (RBM)*, *Grid-Based Matching (GBM)*, *Hybrid-Based Matching (HBM)*, and *Sort-Based Matching (SBM)*.

3.1 Region-Based Matching

The Region-Based Matching algorithm is also known as *Brute-Force Matching* algorithm. Given a dimension d, it finds overlaps by comparing each update region with all the subscription ones. The RBM matching algorithm is simple to implement and allows to derive exact overlapping information. The computational complexity is $O(n \cdot m)$, where n and m are the number of update and subscription regions, respectively. However, this computational complexity is affected by the number of dimensions; therefore, in the generic d-dimensional case, i.e., running the RBM algorithm for each dimension d and calculating intersections between regions, the computational complexity is $O(d \cdot (n \cdot m))$.

The main advantage of this algorithm is its simplicity, but it has scalability issues as regions and dimensions grow. RBM was adopted in the first version of the Defense Modeling and Simulation Office (DMSO) RTI implementation of the HLA 1.3 specifications, and in the MÄK High-Performance RTI (Wood, 2002; Pan et al., 2011).

3.2 Grid-Based Matching

The Grid-Based Matching algorithm works by partitioning the Multi-Dimensional Coordinate Space (MDCS) into a grid of cells. Each region r is then mapped, through a function $f(r)$, to the corresponding cells of the grid. In GBM, an overlap between an update and a subscription region happens if and only if they have at least one cell of the grid in common (Boukerche et al., 2005).

While the GBM algorithm has a lower computational overhead than the RBM one, the overlapping information is not exactly determined. Therefore, irrelevant data may be received by a subscribing federate f_s even if it has no region overlaps with the updating federate f_u, but only because f_s and f_u have at least one cell in the grid in common. Obviously, this leads to unnecessary consumption of network resources, and f_s has to discard irrelevant data it receives. To overcome this issue, many RTI implementations with GBM-based DDM services adopt an additional data filter on the receiving side.

The computational complexity is $O(c+n\cdot m/c)$, where c is the number of cells, n and m are the number of update and subscription regions, respectively. In the generic d-dimensional case, i.e., running the GBM algorithm for each dimension d, the computational complexity is $O(d\cdot(c+n\cdot m/c))$ (Marzolla and D'angelo, 2020).

Like RBM, this algorithm is simple to implement but is more scalable than the first one. The performance of the GBM algorithm depends not only on the grid size but also on the size of the individual cells that compose it. Indeed, the larger the cell size, the greater the amount of irrelevant data transferred, but the shorter the time the GBM algorithm takes to detect overlaps between regions. On the other hand, the smaller the cell size, the less irrelevant data will be transferred, but the GBM algorithm needs much more computational resources to determine overlaps. For this reason, the choice of the cell size represents a crucial aspect as it impacts the performance of the DDM services based on GBM (Ayani et al., 2000; Tan et al., 2000a).

3.3 Hybrid-Based Matching

The Hybrid-Based Matching algorithm combines the GBM and RBM algorithms. Specifically, GBM is used to partition the MDCS into a grid of cells, then map regions to grid cells; while RBM is used to perform exact matching between update and subscription regions that overlap the same cells in the grid (Tan et al., 2000b).

This approach has two advantages. On the one hand, it has a lower computational cost than the RBM algorithm in determining the overlap information. On the other hand, it overcomes the issue of the GBM algorithm since it allows obtaining exact overlapping information. Nevertheless, the main issue of the HBM algorithm is that it has the same drawbacks as GBM, i.e., the performance depends on the size of both the grid and cells.

3.4 Sort-Based Matching

The Sort-Based Matching algorithm improves matching performance by sorting the boundaries of regions before evaluating their overlap (Pan et al., 2007; Raczy et al., 2005).

The algorithm uses a bit-matrix $M \in \mathbb{R}^{nxm}$, where n and m are the number of update and subscription regions, respectively. Semantically, a row n_i, with $i = 0,...,|n|$ indicates the update region, while a column m_j, with $j = 0,...,|m|$ the subscription region. Each element $M_{i,j}$ is defined as follow:

$$\begin{cases} 1, & \text{if the regions i, j overlap,} \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Two sets of subscription regions ($SubSetBefore$ and $SubSetAfter$) are defined as n-bit vectors to track regions. The algorithm operates as follows. Given a dimension d, it begins by assuming that each update region overlaps each subscriber region; as a consequence, each matrix element is initialized to 1. Then, it inserts the ranges' bounds of all regions into an ordered list ord_list, initializes $SubSetBefore = \{\emptyset\}$, and adds all subscription regions into $SubSetAfter$. Upon the initialization is completed, ord_list is scanned from bottom to top and operates, for each element l_i, with $i = |ord\_list|,...,0$, as follows. If l_i is:

- a lower bound of a subscription region R, then $SubSetAfter \setminus \{R\}$;
- an upper bound of a subscription region R, then $SubSetBefore \cup \{R\}$;
- a lower bound of an update region R, then all regions in $SubSetBefore$ do not overlap with R, then M is updated;
- an upper bound of an update region R, all regions in $SubSetAfter$ do not overlap with R, then M is updated.

The computational complexity is $O(n\cdot m)$, where n and m are the number of update and subscription regions, respectively. In the generic d-dimensional case, i.e., running the SBM algorithm for each dimension d, the computational complexity becomes $O(d\cdot(n\cdot m))$ (Raczy et al., 2005).

4 A TOPIC-BASED PUBLISH-SUBSCRIBE APPROACH FOR DDM

Despite research progress, most of the existing region-matching algorithms need to scan every region to find overlaps with others, resulting in a waste of computing resources. To face these needs and shortcomings, the *Topic-based publish-subscribe messaging system (TBMS)*, of which the *Topic-Based Matching algorithm (TBM)* is part, has been defined. The idea behind this new messaging system is to have an

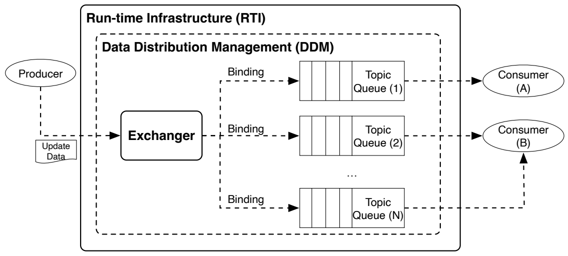


Figure 2: Architecture of the Topic-based publish-subscribe messaging system.

"exchanger" with a queuing system that mediates the communication between federates, minimizing mutual awareness, i.e., what federates should have of each other to be able to exchange messages, effectively implementing *decoupling*. A producer federate sends a message to the exchanger, which in turn forwards it, by using the TBM algorithm, to the corresponding queue that consumer federates use to get the message. A key advantage of TBMS is that the exchanger forwards messages to queues without needing to know consumer federates.

TBMS changes the way with which DDM services manage regions and find overlaps by introducing the concept of *"topic"*. A topic is a well-structured string defined using a dot-delimited format, and it is used to filter and deliver HLAObjectClass and HLAInteractionClass messages. The structure of a topic is composed of three parts:

$$region\_id.\{object|interaction\}.instance\_id \quad (2)$$

where:

- $region\_id$ represents the identifier of the region;
- $\{object|interaction\}$ specifies the supported datatype: *object*, for managing HLAObjectClass and *interaction* for handling HLAInteractionClass (IEEE Std. 1516-2010, 2010);
- $instance\_id$ represents the identifier of the instance.

Figure 2 depicts the architecture of TBMS along with the key parts: *Producer Federates*, *Exchanger*, *Bindings*, *Queues*, and *Consumer Federates*.

Producer Federates (see, Federate (A) in Figure 2) are responsible for creation regions by using the *createRegion()* method, with a specific set of dimensions. For every dimension, the lower and upper bounds of the range of that region are defined through the *setRangeLowerBound()* and *setRangeUpperBound()* methods, respectively (IEEE Std. 1516-2010, 2010).

A Producer Federate sends HLAObjectClass and HLAInteractionClass messages to the *Exchanger* component. When the *Exchanger* receives the message, it is responsible for routing it to different *Queues* by using the message's topic information and *Bindings*, which connect the exchange and queues.

The wildcard "*" has been defined to identify *all elements* in a specific position of the topic structure. This wildcard may be used to define a *binding* over a *topic* following the same topic's structure and Rule 1:

Rule 1. *If the second part of the binding uses the wildcard, then the last part must have the wildcard.*

For example, $< region\_id > .*.*$, and $< region\_id > .object.*$ are valid bindings; whereas, $< region\_id > .*. < instance\_id >$ is invalid.

The *Exchanger* component forwards messages to queues depending on wildcard matches between the message's topic and the queue binding's routing pattern. It is important to note that once the Producer Federates sends a message to the *Exchanger*, it does not wait for a response, i.e., it is not blocked.

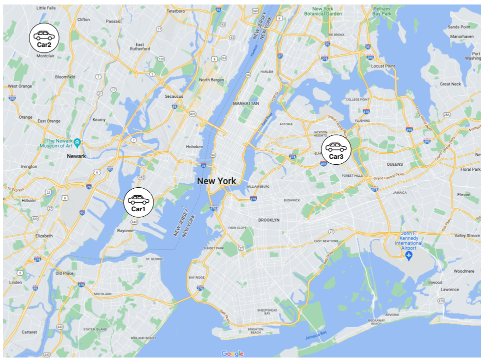
Consumer Federates (see, Federates (B) and (C) in Figure 2) defines one or more *bindings* and subscribe to the related queues in order to receive messages from *Producer Federates*, and then process them.

In order to explain the *Topic-based publish-subscribe messaging system* more easily, a scenario related to a drone patrol system is examined, as reported in Figure 3. In this scenario, Federate (A) simulates three cars in the New York area, and related data are published/updated on the RTI using the DDM services extended with the proposed solution. Federates (B) and (C) simulate two drones used to patrol the New York area, with the difference that Federates (B) is interested in tracking and monitoring all cars, whereas Federates (C) wants to track only "car1".

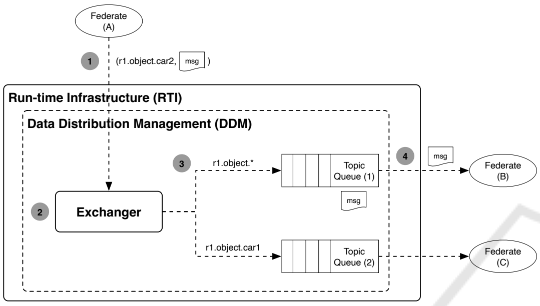
When the simulation starts, Federate (A) creates the region r_1 and simulates the movement of the three cars (car_1, car_2, and car_3) in the New York area. Federates (B) and (C) define the bindings $r_1.object.*$ and $r_1.object.car_1$, respectively, and subscribe to the related queues. Every time Federate (A) updates data related to a car and publishes it on the RTI, the federate sends a message consisting of two parts: (i) the reference topic; and (ii) the car's data (see Figure 3b - step 1). Upon the *Exchanger* receives the message (see Figure 3b - step 2), it uses the TBM algorithm to forward the message to the corresponding queues (see Figure 3b - step 3), and then to the subscriber Federate(s) (see Figure 3b - step 4).

4.1 Topic-Based Matching Algorithm

The Topic-Based Matching algorithm allows the *Exchanger* component to deliver messages to queues based on wildcard matches between the topic and the binding patterns, which is specified by queues. All



(a)



(b)

Figure 3: A scenario related to a drone patrol system: (a) depicts the positions of car1, car2, and car3 on the map; whereas, (b) reports the configuration of TBMS.

messages with a topic that matches a binding's pattern are routed to the queue, and then forwarded to the consumer federates. Algorithm 2 presents the steps performed by TBM to find overlaps between the topic and binding patterns. It uses a *key* − *value* data

Algorithm 2: The Topic-Based Matching algorithm.

Require: t, m ▷ t - topic, m - message
1: $bindings = Map < binding\_pattern, queue >$
2: **for** *entry* in *bindings* **do**
3: $bind \leftarrow entry.binding\_pattern$
4: **if** wildcardMatch($bind, t$) **then**
5: $q \leftarrow entry.queue$
6: $q.enqueue(m)$
7: **else**
8: discard m
9: **end if**
10: **end for**

structure to manage the associations between binding_patterns and queues. For each entry in *bindings* (see line 2), the algorithm checks whether the wildcard *bind* matches with the topic t or not (Golan et al., 2019). If yes, the queue q is retrieved, and then the message m is enqueued into q in order to be forwarded to all subscribed federates (see lines 5- 6). Otherwise, the algorithm continues with the next entry.

The "wildcardMatch" method has a computational complexity of $O(m)$, where m is the length of the topic (Hajiaghayi et al., 2021). Since it is called for each entry in *bindings*, the computational complexity of TBM algorithm is $O(n \cdot m)$.

The following section presents the experiments carried out to evaluate the performance of the Topic-Based Matching algorithm. The gathered results have been compared with the matching algorithm adopted in HLA 4.0 (see, Algorithm 1).

5 EXPERIMENTAL EVALUATION

This section presents the experiments carried out to evaluate the performance of the TBM algorithm. The experiments have been written in the Java language and performed on a MacBook Pro, equipped with MacOS Catalina 10.15, 16GB of RAM, and 1TB of HD. To promote comparability of the simulation results with those available in the literature, the *two − dimensional* case has been considered.

To perform the experiments a RabbitMQ infrastructure has been set up (Rostanski et al., 2014; Ayanoglu et al., 2016). RabbitMQ is a *message-oriented* middleware, also known as *message-broker*, that implements the *Advanced Message Queuing Protocol (AMQP)*. It offers a common platform for sending/receiving messages, ensuring the security of communications. RabbitMQ acts as an intermediary between message consumers and producers, this peculiarity makes it easy to decouple the involved parts. RabbitMQ guarantees the delivery of messages, provides non-blocking features, and can be configured to push notifications to producers. Moreover, it provides support to the publish/subscribe mechanism, asynchronous processing, and message queues.

To conduct the experiments, three factors have been considered: *Total number of regions*, *Overlap rate*, and *Update rate*. The first one is the most obvious factor. If there are more regions, the matching algorithm will take longer to determine overlaps. The total number of regions, used in the experiments, varies from 1000 to 10000. Each region has a size of 20x20. All regions are evenly distributed in a 5000x5000 routing space.

The *Overlap rate* represents, as defined by Equation 3, the number of subscribed regions over the total number of regions. The higher this rate, the longer it takes for the algorithm to check whether or not two regions intersect. In the experiments, three different overlap rates were used: 0.25 (low), 0.50 (medium), and 0.75 (high). Finally, the *Update rate* represents the generation rate of update events on regions defined by using a probabilistic model. It is used by the producer federate to rate the generation of messages to subscriber federates. In the experiments, the Poisson distribution has been chosen with mean number of events per time interval $\lambda = 0.8$ in order to avoid an excessive generation of events.

$$Overlap\,rate = \frac{|subscribed\ regions|}{|total\ regions|} \quad (3)$$

Figure 4 shows the time performance comparison between *TBM* and *RBM* matching algorithms with a low overlap rate, i.e., 0.25. In this situation, since there is a low number of intersections between subscription and update regions, the computational cost for finding overlaps does not vary much between the two considered algorithms as long as the total number of regions remains relatively low, i.e., less than 6000 regions. While, with a large number of regions, more significant than 6000, the *TBM* algorithm achieves much better performance than *RBM*.

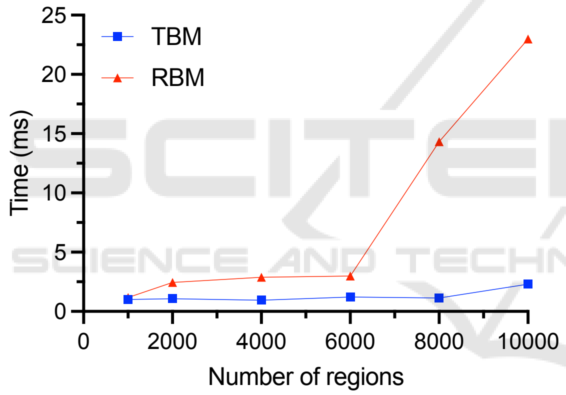


Figure 4: Performance comparison between the TBM and RBM matching algorithms, with low overlap rate 0.25.

Figure 5 depicts the time performance comparison between the *TBM* and *RBM* matching algorithms with an overlap rate of 0.50. In this situation, the computational cost of both matching algorithms clearly increases with respect to the previous case, since there are more regions to evaluate. However, similar to the previous scenario, the performance trend remains, and the performance of the *RBM* algorithm is very poor, while the proposed *TBM* algorithm has a better processing time, especially when the total number of regions is greater than 6000.

Concerning the scenario with a high overlap rate, i.e., 0.75, there is a high degree of intersections between regions. The computational cost of both matching algorithms still increases compared to the previous cases, as both directly depend on the total number of regions. Similarly to the previous scenarios,

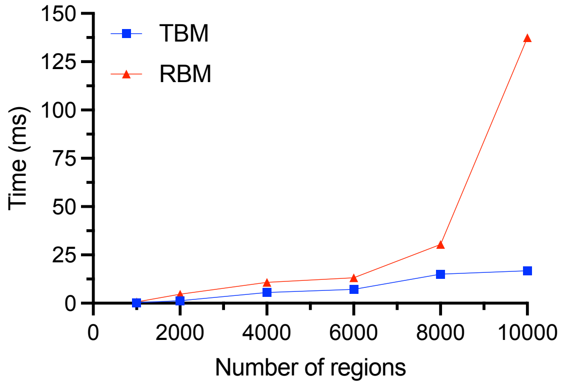


Figure 5: Performance comparison between the TBM and RBM matching algorithms, with medium overlap rate 0.50.

the computational cost shows the same trend, and the *RBM* algorithm keeps achieving a lower performance than the proposed *TBM* one. Figure 6 shows the time performance comparison between the *TBM* and *RBM* matching algorithms with a high overlap rate.

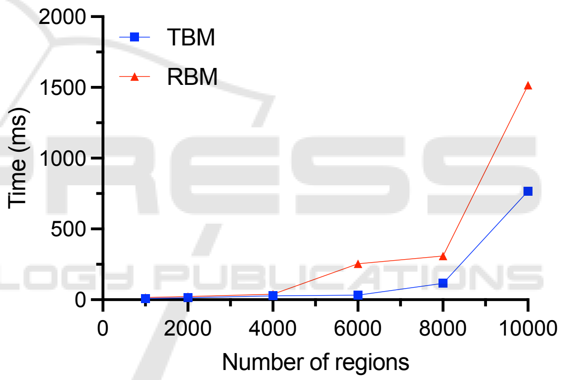


Figure 6: Performance comparison between the TBM and RBM matching algorithms, with high overlap rate 0.75.

6 CONCLUSIONS

In this paper, the novel *Topic-based publish-subscribe messaging system (TBMS)*, of which the *Topic-Based Matching algorithm (TBM)* is part, has been defined to improve the performance, reliability, and scalability of DDM services in HLA. To evaluate the performance of *TBM*, a set of experiments has been carried out by considering different overlap rates. The experiment results were compared with the ones obtained with the standard *Region-Based Matching (RBM)* algorithm. Results highlight the fact that the proposed *TBM* algorithm achieves better performance than the *RBM* one in the all considered overlap rates. Further investigation will focus on the reliability and scalability of the proposal.

REFERENCES

Ayani, R., Moradi, F., and Tan, G. (2000). Optimizing cell-size in grid-based ddm. In *Proceedings Fourteenth Workshop on Parallel and Distributed Simulation*, pages 93–100. IEEE.

Ayanoglu, E., Aytas, Y., and Nahum, D. (2016). *Mastering rabbitmq*. Packt Publishing Ltd.

Bentley, J. L. and Wood, D. (1980). An optimal worst case algorithm for reporting intersections of rectangles. *IEEE Transactions on Computers*, 29(07):571–577.

Boukerche, A., McGraw, N. J., Dzermajko, C., and Lu, K. (2005). Grid-filtered region-based data distribution management in large-scale distributed simulation systems. In *38th Annual Simulation Symposium*, pages 259–266. IEEE.

Bouskela, D., Falcone, A., Garro, A., Jardin, A., Otter, M., Thuy, N., and Tundis, A. (2021). Formal Requirements Modeling for Cyber-Physical Systems Engineering: an integrated solution based on FORM-L and Modelica. *Requirements Engineering*, 27(1):1–30.

Derler, P., Lee, E. A., and Sangiovanni Vincentelli, A. (2012). Modeling cyber–physical systems. *Proceedings of the IEEE*, 100(1):13–28.

Falcone, A., Garro, A., Mukhametzhanov, M. S., and Sergeyev, Y. D. (2020). Representation of Grossone-based Arithmetic in Simulink for Scientific Computing. *Soft Computing*, 24(23):17525–17539.

Falcone, A., Garro, A., Mukhametzhanov, M. S., and Sergeyev, Y. D. (2022). Simulation of Hybrid Systems Under Zeno Behavior Using Numerical Infinitesimals. *Communications in Nonlinear Science and Numerical Simulation*, 111:106443.

Falcone, A., Garro, A., and Tundis, A. (2014). Modeling and simulation for the performance evaluation of the on-board communication system of a metro train. In *the 13th International Conference on Modeling and Applied Simulation, MAS 2014, Held at the International Multidisciplinary Modeling and Simulation Multiconference, I3M 2014, Bordeaux, France, September 10-12, 2014*, pages 20–29. Dime University of Genoa.

Fujimoto, R. M. (2000). *Parallel and distributed simulation systems*, volume 300. Citeseer.

Golan, S., Kopelowitz, T., and Porat, E. (2019). Streaming pattern matching with d wildcards. *Algorithmica*, 81(5):1988–2015.

Hajiaghayi, M., Saleh, H., Seddighin, S., and Sun, X. (2021). String matching with wildcards in the massively parallel computation model. In *Proceedings of the 33rd ACM Symposium on Parallelism in Algorithms and Architectures*, pages 275–284.

IEEE Std. 1516-2010 (2010). IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA): 1516-2010 (Framework and Rules); 1516.1-2010 (Federate Interface Specification); 1516.2-2010 (Object Model Template (OMT) Specification).

Lee, E. A. (2008). Cyber physical systems: Design challenges. In *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, pages 363–369.

Marzolla, M. and D'angelo, G. (2020). Parallel data distribution management on shared-memory multiprocessors. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 30(1):1–25.

Möller, B., Antelius, F., Johansson, M., and Karlsson, M. (2016). Building scalable distributed simulations: Design patterns for hla ddm. In *Proc. of Fall Simulation Interoperability Workshop, 2016-SIW*, volume 3.

Pan, K., Turner, S. J., Cai, W., and Li, Z. (2007). An efficient sort-based ddm matching algorithm for hla applications with a large spatial environment. In *21st International Workshop on Principles of Advanced and Distributed Simulation (PADS'07)*, pages 70–82. IEEE.

Pan, K., Turner, S. J., Cai, W., and Li, Z. (2011). A dynamic sort-based ddm matching algorithm for hla applications. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 21(3):1–17.

Raczy, C., Tan, G., and Yu, J. (2005). A sort-based ddm matching algorithm for hla. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 15(1):14–38.

Rostanski, M., Grochla, K., and Seman, A. (2014). Evaluation of highly available and fault-tolerant middleware clustered architectures using rabbitmq. In *2014 federated conference on computer science and information systems*, pages 879–884. IEEE.

Tan, G., Ayani, R., Zhang, Y., and Moradi, F. (2000a). Grid-based data management in distributed simulation. In *Proceedings 33rd Annual Simulation Symposium (SS 2000)*, pages 7–13. IEEE.

Tan, G., Zhang, Y., and Ayani, R. (2000b). A hybrid approach to data distribution management. In *Proceedings Fourth IEEE International Workshop on Distributed Simulation and Real-Time Applications (DS-RT 2000)*, pages 55–61. IEEE.

Wood, D. D. (2002). Implementation of ddm in the mak high performance rti. In *Proceedings of the simulation interoperability workshop*. Citeseer.

Zhu, G. and Wang, H. (2022). Empirical study of large-scale hla simulation of parallel region-matching knowledge recognition algorithm based on region matching. *Computational Intelligence and Neuroscience*, 2022.