Hindawi
Security and Communication Networks
Volume 2022, Article ID 4865209, 11 pages
https://doi.org/10.1155/2022/4865209

Research Article

Agent-Based Data Extraction in Bioinformatics

Shakir Ullah Shah,[1,2] **Abdul Hameed**,[1] **Abdulwahab Ali Almazroi**,[3] **and Mohammed A. Alqarni**[4]

[1]*Department of Computer Science, Iqra University, Islamabad, Pakistan*
[2]*National University of Computer and Emerging Sciences, Peshawar, Pakistan*
[3]*College of Computing and Information Technology, College of Computer Science and Engineering at Khulais, Department of Information Technology, University of Jeddah, Jeddah, Saudi Arabia*
[4]*College of Computer Science and Engineering at Khulais, Department of Information Technology, University of Jeddah, Jeddah, Saudi Arabia*

Correspondence should be addressed to Shakir Ullah Shah; shahshakir@yahoo.com

Received 5 January 2022; Revised 24 January 2022; Accepted 31 January 2022; Published 26 March 2022

Academic Editor: Thippa Reddy G

Bioinformatics is an active and important research discipline in which molecular data is exponentially growing in complex nature. Because of the substantial research in this field, researchers are faced with critical issues such as bandwidth, storage, and complexity in order to retrieve molecular data. It becomes very difficult to conduct research using low computational devices such as Internet of things and sensors. We are employing migration of the agent technique to decrease network traffic and to mitigate the client's limited resource problem by utilizing server-side resources to perform large-scale computation. Our proposed solution does not necessitate additional storage or processing power on the client's side which makes it cost effective. In the proposed solution, (i) an agent visits service provider containing biological data, say sequences requested by the client, (ii) agent fetches the required data, and on the server side it will manipulate the data, and (iii) returns along with the required results to its source platform. Thus, it solves the bandwidth, storage, and computational issues without involving the low resources of the client. For the proof of concept, Java Agent Development (JADE) framework is used as an implementation tool and the results are compared with Java Remote Method Invocation (RMI). It is important to note that our findings reveal that our strategy saves the user up to 16.25% of average time with respect to bandwidth. On the other hand, our approach takes 46.82% less time than the other with respect to data that the agent carries. In addition to the previous contributions, our approach acts as a mashup, to collect data in different format from several service providers, and converts it in any required format. Thus, it solves the problem of complexity hidden in the nature of the data to increase the researchers' productivity.

1. Introduction

The volume and complexity of data over multiple service providers are generated exponentially. Now the issue is extraction, retrieval, and processing of relevant information which has made obvious the need for a system to facilitate users. Data coming in from multiple domains needs to be integrated together to provide a cohesive view. One technology that massively helps facilitate this goal is the concept of mashups [1, 2]. Mashups help users get an integrated user-oriented view of data and code from multiple heterogeneous sources. Trendsmap, housingmaps, Wheel of Lunch, and InstantWatcher are some examples of mashup. A mashup is a web application that stitches together the contents, presentations, and application functionalities from multiple sources and gives them a new and useful look. In other words, it combines multiple services into a single one [3, 4]. In this paper, we apply the concept of mashup through multiagent paradigm to the domain of bioinformatics.

The bioinformatics [5] and [6] is about understanding biological data and is a growing field of research. With the advance in technology, the amount of biological data is

growing at a tremendous pace. This makes the field of bioinformatics important to the society. They have also made strides in understanding how they interact with other proteins, which is known as protein-protein interactions. For example, data reports [8] and [9] also include both structural data [8] and other data [10], [11] about proteins. A wide range of computational techniques has proposed, primarily for image manipulation and pattern detection. With NGS, analysts are analyzing precious data and doing a lot of intensive work to find patterns. More importantly, it is considerably difficult for them to create complex maps from heterogeneous sources.

There are various experimental approaches [18] which are very expensive and time consuming because they require a lot of resources and time to measure the physical interaction among proteins. They have a high possibility of error because experiments are purely carried out in the lab and are not standardized. The adoption of agent technologies and multiagent systems constitutes an emerging area in bioinformatics where data is quite big (that is, in gigabytes) and complex in nature. Researchers face the problem of data retrieval due to low (1) bandwidth, (2) storage, and (3) computation on their machines. For the processing, analyzing and transportation of multilevel complexity of molecular data require high bandwidth and storage. Thus, it becomes very difficult for researchers to conduct research with low power researches. This study proposed agent migration-based approach [19–21].

The main advantage of an agent-based approach is that the agent will get the request from the requester and visit the service provider, which saves the time for the data provider to make the data available, and agents will return back to the requester. Another advantage of using agent-based approach is that we will be able to transfer data to the machine which has high computational power. The computation is done at the service side. And the results are available in the same format as they were at service side. This will help to reduce the data size and transfer the data to the machine which has high computational power. This will give better efficiency, reduce network congestion, and transfer the data to machine with high computational power. The main theme of this paper is to use agent-based methodology which tackles processing and network issues of multiagent systems. It processes all the necessary steps on the client side with low computational resources.

The rest of the paper is organized into different sections. Literature review about bioinformatics, multiagent systems, and mashup is provided in Section 2. Section 3 provides a list of complete steps of our agent-based solution along with details. In Section 4, as a proof of concept, a reference implementation is listed. Section 5 provides the analysis and discussion of the proposed solution. Section 6 concludes this study along with future directions.

2. Literature Review

In this section, literature about bioinformatics mashup and multiagent systems is presented. In each section, the importance of each domain is provided. We first turn to the target domain, namely, biofoundation.

2.1. Bioinformatics. Bioinformatics is an interdisciplinary field that mostly uses computer as a computational tool for solving issues related to the biological data. Such computing devices are used for the analysis of the internal structure and biological functions of living organisms. The main purpose of computing devices is giving an efficient structure to the data so that it could be interpreted accurately. Mainly it deals with genome and protein. One of the important characteristics of bioinformatics is personalized medicines. It is the application of computer processing techniques to the field of genetics and biochemistry. This is a branch of computer science that deals with the storage, retrieval, and analysis of biological data [11]. It is a classification of data in a standard manner. The data are analyzed in order to determine their structure and content [12]. The bioinformatics includes the research in the field of genetics and genomics. This branch of science is implemented in various fields such as the study of evolution and phylogeny of various species [13]. The data generated by the bioinformatics are stored at various data centers. These data can be used for the diagnosis of the diseases and for the treatment and prevention of the diseases [14].

The protein-protein interactions [22] are of extreme importance because they play a vital role in many biological processes, such as signal transduction and transcription regulation. They also act as protein-based modules that are extensively used by nature to build complex systems. Therefore, they are a primary objective of many bioinformatics algorithms. However, in the field of protein-protein interactions, the problem of interactions between proteins in the context of the entire proteome has received less attention. In this context, a recent study has been carried out by proposing a new protein-protein interaction network. The network is based on the comparison of the entire proteome between two different organisms.

Protein-protein interactions [8, 9, 23] are a key element in the study of molecular biology, as molecular interactions are at the foundation of all biological systems. In this regard, the interactions between proteins have been extensively studied [10, 24]. Protein Interactions by Structural Matching (PRISM) [25–27] is an online web tool for predicting protein-protein interactions with high confidence. It is based on the structural and functional domain similarity of proteins. The first step in the PRISM-2 algorithm is to generate a structural alignment of two proteins. The proteins are compared by hand-determined structural similarity, and this similarity is used to generate a structural alignment.

There is a gigantic development in the organic succession where a huge amount of data is being made and transferred on the sites/servers. Presently to get the information we would have to communicate with the connection point utilizing electronic inquiries [28]. This means that the user has to click on a single link to access all the linked websites. This is very time consuming and tedious to stay online each query. The idea of a mashup is to integrate multiple datasets

into a single system. The paradigm of a mashup combines the data from multiple heterogeneous data sources. It is a great way to combine diverse data sources into a single view. Mashups are a great way to implement the existing data into a new structure. It is used in situations where the data from multiple heterogeneous data sources are required to be combined into a single view. The main idea behind the mashup is to integrate the data from multiple data sources into one system.

2.2. Mashup. Information retrieval is becoming a challenging task due to rapid proliferation of data. It becomes more complex when the required information is scattered on multiple service providers. This complexity demands an efficient system to retrieve the desired results in an appropriate manner. There are different approaches to retrieve the information, combine it, and give a desired look; mashup is one of such approaches [29]. Mashup gives entirely a new and different look or some added value to the existing data for end users. Service providers provide APIs which act as an interface for data and services. Some APIs are free, and some are proprietary in natures that need authentication and authorization. Asynchronous JavaScript and XML (AJAX), Representational State Transfer (REST), and Services-Oriented Access Protocol (SOAP) are some state-of-the-art technologies that have influenced the mashup architecture [30]. REST, screen scraping, and RSS feed/widget are used to retrieve the contents from other websites. It is widely developed for web applications such as social networking, e-government, enterprise resource management, real state, and more [31, 32].

A number of tools exist to create mashup such as Yahoo Pipes [33] or IBM Mashup Center [34]. Traditionally, a mashup runs inside a web browser, but there are also some other environments for it. Two important styles of mashup are server-side mashup and client-side mashup [35]. The difference between server-side and client-side mashups is the way the data is processed. In a server-side mashup, also called a proxy-style mashup, a web server serves the mashup to retrieve all the data from multiple web hosts, and stitching takes place on server side and is rendered on client's web browser. In a client-side mashup, opposite to server-side mashup, stitching of the services and contents takes place on the client, namely, within the web browser. These are also called Rich Internet Applications (RIAs) and have the added advantage of prompt response over server-side mashup. A mashup can be either a consumer or enterprise [36]. A consumer mashup also known as service or client mashup integrates data from multiple public sources inside the browser, for example, iGuide; server-side mashup is the target of this study. Both styles of mashup have their own obvious benefits, as both provide new insight into existing resources. But using such mashup tools, users must trust them. So user data is not secure, since it has to be released to the third parties. We address this issue using the multiagent paradigm.

2.3. Multiagent Systems (MAS). Multiagent system is the collection of multiple software agents [37]. A software agent is a piece of code that works autonomously and communicates with other agent-oriented and non-agent-oriented software [38, 39]. The basic building blocks of an agent consist of code, data, and state. The data part represents the data structure to preserve important information about the expression before and after evaluation. The configuration of the agent is stored in its data and state parts. It contains information about platforms which changes dynamically when it travels from one node to another node within a network. The code part of an agent is the collection of ordered statements that remains nearly constant during the execution though it can change when required. It represents the actual logic of the agent. The state part of an agent represents the current status of the data part. Basically, state is the collection of information of all data structures.

2.4. Significance of Multiagent Systems. Agent-oriented software paradigm has become a promising technology which is widely used in distributed environments such as e-commerce [40], network management [41], data mining [42], robotics [43, 44], and information extraction [45]. Some interesting applications of agent systems can be found in healthcare system [46, 47] for patient scheduling, storing medical records of patients, and sharing them with concerns. Agent-based system, also called multiagent system [48], is the system in which multiple agents interact, cooperate, and coordinate with each other. Such system, loosely coupled, enhances the capabilities of monolithic system to perform different tasks which are beyond the scope of individual agent. It is widely used to share or get resources over the network among agents. The resources might be computational, logic to solve the problem, software or expertise distributed temporally and spatially. Normally, systems are categorized into two categories: client (to make a request) and server (to server) but multiagent systems combine the benefits of both in a social, proactive, and reactive manner.

2.5. Design Issues in Multiagent Systems. The most important design issue for multiagent systems is how they will communicate among each other and with other entities. The starting point is to select any tool or middleware to facilitate developers to get the core benefits of this technology rather than to resolve the basic issues of communication. So some standards are needed prior to deploying such system. The Foundation for Intelligent Physical Agents (FIPA) [49], AGENTLINK [50], and OMG Agent Platform Special Interest Group (PSIG) [51] are the leading standardization bodies to promote agent technology. This study focuses on FIPA for agent reference and development model. There are various tools for agent-based modeling like NOMADS [52, 53], AgentScape [54], Agentcities [55], Aglets [56], Voyager [57], Janus [58], TACOMA [59, 60], Grasshopper [61], JADE [62, 63], JaCaMo [64], Addre Jason [65], and ABLE [66].

JADE [62] is used to launch an agent platform. The most important reason is that it is an open source middleware under the Library Public License (LGPL). It is entirely implemented in the Java language, which makes it more portable and smarter. It is one of the most popular middleware types within the research community. It alleviates the implementation of multiagent systems (MAS). It provides a set of graphical tools which make it very easy to deploy agent platform on a standalone system as well as over a distributed network. This study highly recommends JADE as agent middleware. Its infrastructure is very flexible and agent community is adding different add-ons to enhance its features. It is compliant with the FIPA-IEEE computer society specifications. The core concept of FIPA is to resolve the issue of interoperability and it has extended FIPA's model in multiple areas. According to the JADE specification, the mobility of an agent can be categorized into two types: inter- and intraplatform. In intraplatform mobility, an agent migrates itself between containers of the same platform but cannot move to containers of the different platform. In intraplatform mobility [67], an agent moves among different platforms. In interplatform mobility, the agent leaves its own main container and joins another main container of another platform. The main focus of this study is interplatform mobility; see step 1 for details of each and how an agent can migrate from one platform to another.

2.6. Bioinformatics and Multiagent Systems. This study also explores the area of bioinformatics as a real application of multiagent systems and explores how a mobile agent can operate in a highly dynamic environment for data dissemination. A mobile agent visits different itineraries to collect the required information and stitch it together to provide a new shape. We propose agent migration to mitigate the aforementioned issues by moving the agent to the server side to perform computations [21, 68]. In a nutshell, this study proposes agent migration characteristic to make it a mashup. Hence, it can be used for data dissemination.

3. Solution

We propose mobility characteristic of an agent to find a solution. The solution provides accurate and fine-grained result even though the bandwidth and storage of the client might be low. It also deals with complexity present in the nature of the data as well as in the dynamic environment. In the agent migration approach, an agent is executed in a client machine; the agent migrates to another machine when the original machine is overloaded. To migrate an agent from one machine to another, the agent must be able to traverse the network. The abstract details are in Figure 1.

Java Remote Method Invocation (RMI) [69] is a way to extract data from the server as it allows remote access to Java objects on a remote host. It is light weight communication protocol. The payload of Java RMI is the Java object that contains references to the remote methods. A Java object on the remote host is a Java object that is created on the remote host. A client stub object, which is a proxy object that contains references to the remote object, is used to access remote object. The complete steps which were used in this study are listed in Figure 2.

Mobile agents use the resources of the system to complete the tasks and then get back to the system where they started their execution [10, 11]. Agent-based systems are a powerful and effective way to develop intelligent systems because of their simplicity and extensibility. They are more effective in handling with the problems related to distributed, parallel, and autonomous systems.

It is also effective in handling with the problem of complex systems. The reason behind it is that they are not heavy in computation and they can be used on multiple systems at the same time. This makes it easy to design and implement these systems. The steps which are carried out in this study are mentioned in Figure 3.

4. Reference Implementation

To deploy agents, various agent frameworks are available [70, 71]. Java Agent Development (JADE) framework is FIPA Agent Markup Language (FAM), a language designed to be used to model agent systems. It is compliant with the FIPA Agent Communication Language (ACL) specification and with the FIPA Agent Communication Framework (ACF). We provide the source of our own reference implementation at https://github.com/BioAgent.

For the testbed configuration, two personal computers were used: one as a server and the other one as a client. Both systems were connected through the 4G Huawei E5573s-320 which is a pocket WiFi router. Table 1 shows both hardware and software details of both personal computers.

5. Results and Discussion

This section presents the study carried out on the performance of mobile agents and Java RMI. A detailed discussion of the results is carried out in this section. Java RMI and agent migration approached are compared. Due to the fact that mobile agents are not dependent on the host application and can be independently transferred to another host, an effective approach to large-scale agent migration has been proposed.

Table 2 shows the amount of network load made by client using Java RMI and our agent-based approach. The agent approach is more efficient than Java RMI approach because it decreases the number of network calls made by the client. The agent approach is more efficient because the agent is migrated to the server only when there is a need for it. As a result, the client makes fewer network calls, and the overhead of the network calls is reduced. Therefore, it is clear that the Java RMI approach has more network load than the agent approach. In the agent approach, the number of network calls is reduced by migrating the agent. It is important to note that, in the agent approach, the agent is only migrated if there is an urgent need for it.

The Java RMI approach is a lot more mature compared to the agent-based approach. Table 2 provides a summary of the results of Java RMI and agent approaches based on

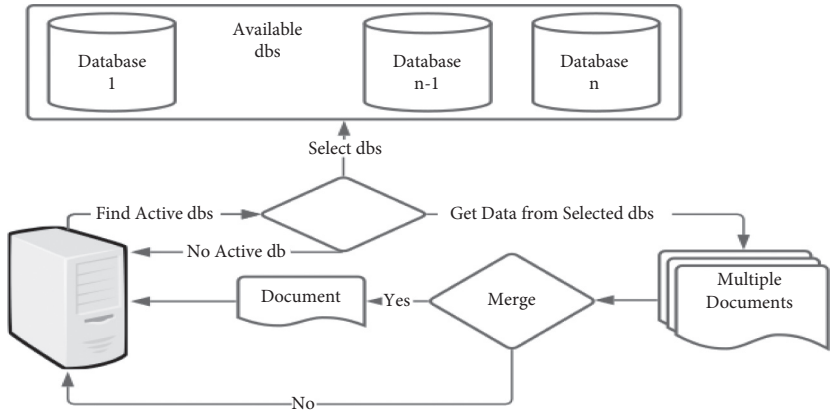

Figure 1: System architecture.

Algorithm 1 P2P Interactions using JAVA RMI

Require: $len(Avail_{dbs}) \geq 1$
Ensure: $Download_{dbs} = Selected_{dbs}$
  $Download_{dbs} \leftarrow [\ ]$
  $Avail_{dbs} \leftarrow fetch_{dbs}(URL, Protein_{id})$
  while $Avail_{dbs} \neq 0$ **do**
    $Selected_{dbs} \leftarrow select_{dbs}(Avail_{dbs})$
    $i \leftarrow 1$
    $current_{dbs} \leftarrow Selected_{dbs[i]}$
    while $i \neq len(Selected_{dbs})$ **do**
      $inter_s \leftarrow interactions(current_{dbs})$
      while $inter_s \neq 0$ **do**
        $downd_{db} \leftarrow interactions(inter_s, current_{dbs})$
        $Download_{dbs}.append \leftarrow downd_{db}$
        $inter_s \leftarrow inter_s - 1$
      end while
      $i \leftarrow i + 1$
      $current_{dbs} \leftarrow Selected_{dbs}[i]$
    end while
  end while

Figure 2: Protein-protein interactions using Java RMI.

network load. The agent migration approach does not have any network load as it needs only interconnectivity.

In Figure 4, the x-axis represents the size of the extracted data from multiple databases, while the y-axis shows the network load consumed by both approaches. According to Figure 4, we conclude that Java RMI is showing an increasing trend in result size. But, at the same time, the curve shows an increasing trend in network load. Thus, while achieving high results, network load also increases with time. That is why the curve has been shown in the graph. It shows a direct relationship between result size and network load. The result size is dependent on network load. On the other hand, an agent-based system shows high results with a constant value of network load. That is why the graph of an agent-based system is a straight line.

In Figure 5, we can see that the agent-based approach gives better results as compared to Java RMI when the result size increases from 2 kB. Similarly, the response time is only 10 seconds at result size of 5 kB. Furthermore, 10 kB result size is achieved at a response time of only 15 seconds.

From Figure 5, which is based on Table 3, we can conclude that, in the Java RMI system, result size is directly proportional to response time. As the size of the result increases, the response time also increases. It will take more response time to achieve a high volume of results. On the other hand, an agent-based system shows a high return size with 46.82% less response time than the other with respect to data that the agent carries. The average of Java RMI approach is 20.21785714 while the average time of our approach is 10.75047619. The difference of both approaches is 9.467380952.

In Figure 6, the blue line shows the agent graph and the red line shows the graph of Java RMI. We can clearly see that if we decrease the bandwidth, our agent is computing faster as compared to Java RMI.

According to Table 4, the average responses of both approaches are 20.21785714 and 10.75047619. It is important to note that our findings reveal that our strategy saves the user up to 16.25% of the average time with respect to bandwidth.

Algorithm 2 P2P Interactions using Agent Migration

Require: $Avail_{dbs} \neq 0, AgentPlatform, MobilityService_{(client+dbs)} \leftarrow True$
Ensure: $Download_{dbs} = Selected_{dbs}$
  $bioAgent \leftarrow Agent$
  $bioAgent.address \leftarrow client$
  $Download_{dbs} \leftarrow [\ ]$
  $Avail_{dbs} \leftarrow fetch_{dbs}(URL, Proten_{id})$
  while $Avail_{dbs} \neq 0$ **do**
    $Selected_{dbs} \leftarrow select_{dbs}(Avail_{dbs})$
    $i \leftarrow 1$
    $current_{db} \leftarrow Selected_{dbs[i]}$
    while $i \neq len(Selected_{dbs})$ **do**
      $AMS_r \leftarrow AID(\text{"}ams@current_{db}/JADE\text{"}, AID.ISGUID)$
      $destination \leftarrow AMS_r.addAddresses(\text{"}http://current_{db}:7778/acc\text{"})$
      $bioAgent.doMove(destinationdb)$
      $down_{db} \leftarrow interactions(inters, current_{db})$
      $inter_s \leftarrow interactions(current_{db})$
      while $inter_s \neq 0$ **do**
        $downd_{db} \leftarrow interactions(inter_s, current_{db})$
        $Download_{dbs}.append \leftarrow downd_{db}$
        $inter_s \leftarrow inter_s - 1$
      end while
      $i \leftarrow i + 1$
      $current_{dbs} \leftarrow Selected_{dbs}[i]$
    end while
    $bioAgent.doMove(client)$
  end while

Figure 3: Protein-protein interactions using agent migration.

Table 1: Testbed configuration.

Feature	Server	Client
Operating system	Windows 10 (64 bits)	Windows 10 (64 bits)
Model	Toshiba satellite L50-B1380 core i5 6 GB RAM 1 TB HDD	Dell inspiron 15 5570 core i5 4 GB RAM 1 TB HDD
JADE version	4.5.0	4.5.0

Table 2: Comparison of network load between Java RMI and agent-based approach.

Result size (byte)	Java RMI Network load (byte)	Agent-based approach Network load (byte)
0	0	0
50	0.0625	0
100	0.125	0
150	0.25	0
200	0.5	0
250	0.75	0
300	1.00	0
350	1.0625	0
400	1.25	0
450	1.75	0
500	2.00	0
550	2.25	0
600	2.65	0
650	3.01	0
700	3.50	0
750	3.96	0
800	4.25	0
850	4.98	0
900	5.25	0
950	5.97	0
1000	6.125	0

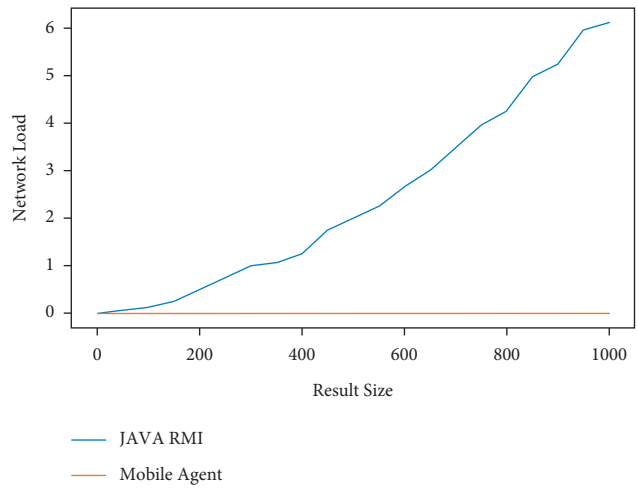

Figure 4: Client-side network load and result size.

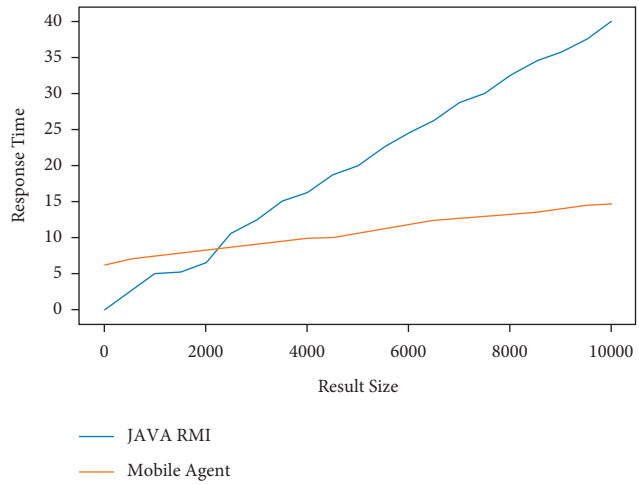

Figure 5: Response time according to agent size.

Table 3: Java RMI vs. agent, based on response time.

Result size (byte)	Java RMI Response time (seconds)	Agent-based approach Response time (seconds)
0	0	6.25
500	2.5	7.00
1000	5.0	7.42
1500	5.25	7.84
2000	6.5	8.26
2500	10.625	8.68
3000	12.5	9.10
3500	15.0	9.52
4000	16.25	9.94
4500	18.75	10.00
5000	20.0	10.60
5500	22.5	11.20
6000	24.5	11.80
6500	26.25	12.40
7000	28.75	12.75
7500	30.0	13.00
8000	32.5	13.25

Table 3: Continued.

Result size (byte)	Java RMI Response time (seconds)	Agent-based approach Response time (seconds)
8500	34.5	13.50
9000	35.7	14.00
9500	37.5	14.50
10000	40.0	14.75

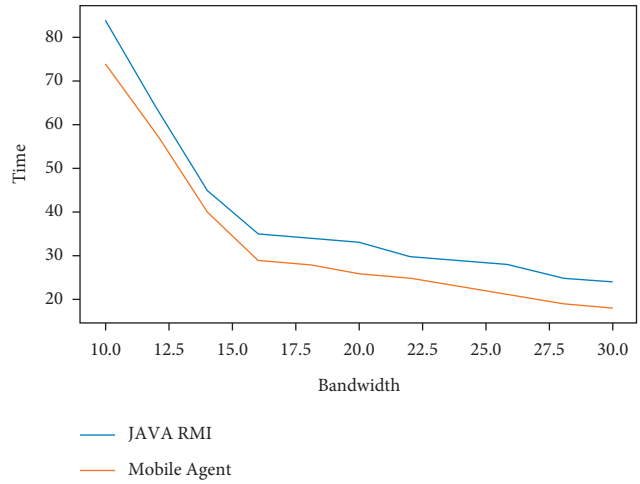

Figure 6: Response time according to agent size.

Table 4: JAVA RMI vs. agent based on bandwidth.

Bandwidth (kbs)	Java RMI Time (seconds)	Agent-based approach Time (seconds)
10	84	74
12	64	58
14	45	40
16	35	29
18	34	28
20	33	26
22	30	25
24	29	23
26	28	21
28	25	19
30	24	18

6. Conclusions and Future Work

In this study, we have designed an agent migration approach for transferring the information between the clients. The agents migrate from client to client to collect the data and transfer it to a central server. The client uses the agent's services. Feedback service of the agent is used to ask the client for any information required by the agent. The client can provide information to the agent to ask the server for any service the client requires. The agent can migrate between the client and the server. The client can also request the agent to migrate to any other client. This approach has many advantages. The agents are intelligent, and they can even work well in low network areas. They can be used for many generic purposes. The agents can be used to find out the interactions between proteins. This approach can be used for many bioinformatics problems like finding out the similarity of sequences, or even finding the missing sequence in known sequences. The findings also show that mobile agent technology leverages network load and storage on the client side and heterogeneous data can be converted into homogeneous format. The main limitation of this study is the deployment of agent environment on client and service side. This approach does not demand the availability of the user online for a full time period. Our research can be modified to make it work on different bioinformatics problem, like viewing the interaction of sequences. It can also be used to find out the similarity of sequences. By modifying the approach, one can

also find out the similarity of proteins, or even find the missing sequence in known sequences. It is also possible to find out the similarities between different organisms.

Data Availability

All relevant code samples can be found at GitHub-shahshakir/BioAgent (https://github.com/shahshakir/BioAgent/).

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] A. Koschmider, V. Torres, and V. Pelechano, "Elucidating the mashup hype: definition, challenges, methodical guide and tools for mashups," in *Proceedings of the 2nd Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web at WWW2009*, pp. 1–9, Madrid, Spain, January 2009.

[2] A. Jhingran, "Enterprise information mashups: integrating information, simply," in *Proceedings of the The 32nd International Conference on Very Large Data Bases*, pp. 3-4, Seoul, Korea, September 2006.

[3] B. Hartmann, L. Wu, K. Collins, and S. R. Klemmer, "Programming by a sample: rapidly creating web applications with d. mix," in *Proceedings of the ACM Proceedings of the 20th annual ACM symposium on User interface software and technology*, pp. 241–250, Newport Rhode Island, USA, October 2007.

[4] N. Zang, M. B. Rosson, and V. Nasser, "Mashups: who? What? Why," in *Proceedings of the ACM CHI'08 Extended Abstracts on Human Factors in Computing Systems*, pp. 3171–3176, Florence Italy, April 2008.

[5] A. D. Baxevanis, G. D. Bader, and D. S. Wishart, *Bioinformatics*, John Wiley & Sons, New Jersey, NY, USA, 2020.

[6] R. Stevens, C. A. Goble, and S. Bechhofer, "Ontology-based knowledge representation for bioinformatics," *Briefings in Bioinformatics*, vol. 1, no. 4, pp. 398–414, 2000.

[7] A. Amadei, A. B. M. Linssen, and H. J. C. Berendsen, "Essential dynamics of proteins," *Proteins: Structure, Function, and Genetics*, vol. 17, no. 4, pp. 412–425, 1993.

[8] D. F. Waugh, "Protein-protein interactions," *Advances in Protein Chemistry*, vol. 9, pp. 325–437, 1954.

[9] T. Berggård, S Linse, and P. James, "Methods for the detection and analysis of protein–protein interactions," *Proteomics*, vol. 7, no. 16, pp. 2833–2842, 2007.

[10] Q. C. Zhang, D. Petrey, L. Deng et al., "Structure-based prediction of protein-protein interactions on a genome-wide scale," *Nature*, vol. 490, no. 7421, pp. 556–560, 2012.

[11] S. Das and S. Chakrabarti, "Classification and prediction of protein–protein interaction interface using machine learning algorithm," *Scientific Reports*, vol. 11, no. 1, pp. 1–12, 2021.

[12] K. A. Theofilatos, C. M. Dimitrakopoulos, A. D. Likothanassis, T Papadimitriou, and P Mavroudi, "Computational approaches for the prediction of protein-protein interactions: a survey," *Current Bioinformatics*, vol. 6, no. 4, pp. 398–414, 2011.

[13] R. B. Russell, F. Alber, P. Aloy et al., "A structural perspective on protein-protein interactions," *Current Opinion in Structural Biology*, vol. 14, no. 3, pp. 313–324, 2004.

[14] B. Suter, X. Zhang, C. G. Pesce, A. R. Mendelsohn, S. P. Dinesh-Kumar, and J. H. Mao, "Next-generation sequencing for binary protein-protein interactions," *Frontiers in Genetics*, vol. 6, Article ID 346, 2015.

[15] R. Carter, A. Luchini, L. Liotta, and A. Haymond, "Next-generation techniques for determination of protein-protein interactions: beyond the crystal structure," *Current pathobiology reports*, vol. 7, no. 3, pp. 61–71, 2019.

[16] D. S. Horner, G. Pavesi, T. Castrignano et al., "Bioinformatics approaches for genomics and post genomics applications of next-generation sequencing," *Briefings in Bioinformatics*, vol. 11, no. 2, pp. 181–197, 2010.

[17] J. K. Kulski, "Next-generation sequencing—an overview of the history, tools, and "omic" applications," *Next generation sequencing-advances, applications and challenges*, vol. 10, pp. 3–60, 2016.

[18] N. Safari-Alighiarloo, M. Taghizadeh, M. Rezaei-Tavirani, B. Goliaei, and A. A. Peyvandi, "Protein-protein interaction networks (ppi) and complex diseases," *Gastroenterology and Hepatology from bed to bench*, vol. 7, no. 1, 2014.

[19] M. Wooldridge and N. R. Jennings, "Intelligent agents: theory and practice," *The Knowledge Engineering Review*, vol. 10, no. 2, pp. 115–152, 1995.

[20] A. Omicini, A. Ricci, and M. Viroli, "Artifacts in the A&A meta-model for multi-agent systems," *Autonomous Agents and Multi-Agent Systems*, vol. 17, no. 3, pp. 432–456, 2008.

[21] L. Gao, H. Dai, T. L. Zhang, and K. C. Chou, "Remote data retrieval for bioinformatics applications: an agent migration approach," *PLoS One*, vol. 6, no. 6, Article ID e20949, 2011.

[22] T. Simonson, "Electrostatics and dynamics of proteins," *Reports on Progress in Physics*, vol. 66, no. 5, pp. 737–787, 2003.

[23] M. Shatsky, R. Nussinov, and H. J. Wolfson, "A method for simultaneous alignment of multiple protein structures," *Proteins: Structure, Function, and Bioinformatics*, vol. 56, no. 1, pp. 143–156, 2004.

[24] J. Zahiri, J. Bozorgmehr, and A. Masoudi-Nejad, "Computational prediction of protein-protein interaction networks: algorithms and resources," *Current Genomics*, vol. 14, no. 6, pp. 397–414, 2013.

[25] U. Ogmen, O. Keskin, A. S. Aytuna, R. Nussinov, and A. Gursoy, "Prism: protein interactions by structural matching," *Nucleic Acids Research*, vol. 33, pp. W331–W336, 2005.

[26] N. Tuncbag, A. Gursoy, R. Nussinov, and O. Keskin, "Predicting protein-protein interactions on a proteome scale by matching evolutionary and structural similarities at interfaces using prism," *Nature Protocols*, vol. 6, no. 9, pp. 1341–1354, 2011.

[27] Y. Ding and L. Gao, "Macrodynamics analysis of migration behaviors in large-scale mobile agent systems for the future internet," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 41, no. 5, pp. 1032–1036, 2011.

[28] K.-C. Chou, "Some remarks on protein attribute prediction and pseudo amino acid composition," *Journal of Theoretical Biology*, vol. 273, no. 1, pp. 236–247, 2011.

[29] A. Ranganathan, A. Riabov, and O. Udrea, "Mashup-based information retrieval for domain experts," in *Proceedings of the ACM Proceedings of the 18th ACM Conference on Information and Knowledge Management*, pp. 711–720, Hong Kong, China, November 2009.

[30] D. Merrill, *Mashups: The New Breed of Web App*, pp. 1–13, IBM Web Architecture Technical Library, 2006, http://scholar.google.com/scholar_lookup?hl=en&publication_year=2006&pages=1-13&journal=IBM+Web+Architecture+Technical+Library&author=Duane+Merrill&title=Mashups%3A+The+new+breed+of+Web+app.

[31] G. Nachouki and M. Quafafou, "Mashup web data sources and services based on semantic queries," *Information Systems*, vol. 36, no. 2, pp. 151–173, 2011.

[32] P. de Vrieze, L. Xu, A. Bouguettaya, J. Yang, and J. Chen, "Building enterprise mashups," *Future Generation Computer Systems*, vol. 27, no. 5, pp. 637–642, 2011.

[33] Y Incorporation, Yahoo Pipes, https://pipes.yahoo.com/.

[34] IBM. Corporation, IBM Mashup Center, https://www.ibm.com/software/info/mashup-center/.

[35] J. Yu, B. Benatallah, F. Casati, and F. Daniel, "Understanding mashup development," *IEEE Internet Computing*, vol. 12, no. 5, pp. 44–52, 2008.

[36] V. Hoyer and M. Fischer, "Market overview of enterprise mashup tools," *Service-Oriented Computing-ICSOC 2007*, vol. 5364, pp. 708–721, 2008.

[37] M. Luck, P. McBurney, and C. Preist, *Agent Technology: Enabling Next Generation Computing (A Roadmap for Agent Based Computing)*, AgentLink, Louisville, USA, 2003.

[38] N. R. Jennings, K Sycara, and M. Wooldridge, "A roadmap of agent research and development," *Autonomous Agents and Multi-Agent Systems*, vol. 1, no. 1, 1998.

[39] M. El Fissaoui, A. Beni-hssane, S. Ouhmad, and K. El Makkaoui, "A survey on mobile agent itinerary planning for information fusion in wireless sensor networks," *Archives of Computational Methods in Engineering*, vol. 28, no. 3, pp. 1323–1334, 2021.

[40] M. B. Hasan and P. W. C. Prasad, "A review of security implications and possible solutions for mobile agents in e-commerce," in *Proceedings of the Innovative Technologies in Intelligent Systems and Industrial Applications, CITISIA 2009*, Kuala Lumpur, Malaysia, July 2009.

[41] A. Koliousis and J. Sventek, "A trustworthy mobile agent infrastructure for network management," in *Proceedings of the 2007 10th IFIP/IEEE International Symposium on Integrated Network Management*, pp. 383–390, USA, May 2007.

[42] M. Yubao and D. Renyuan, "Mobile agent technology and its application in distributed data mining," in *Proceedings of the 2009 First International Workshop on Database Technology and Applications*, pp. 151–155, Wuhan, China, April 2009.

[43] S. C. Banik, K. Watanabe, M. K. Habib, and K. Izumi, "An emotion-based task sharing approach for a cooperative multiagent robotic system," in *Proceedings of the Mechatronics and Automation*, pp. 77–82, Takamatsu, Japan, Auguest 2008.

[44] A. S. Gazafroudi, T. Pinto, F. Prieto-Castrillo et al., "Energy flexibility assessment of a multi agent-based smart home energy system," in *Proceedings of the Ubiquitous Wireless Broadband (ICUWB)*, Salamanca, Spain, September 2017.

[45] G. S. Narula, "An approach for information extraction using jade: a case study," *Journal of Global Research in Computer Science*, vol. 4, no. 4, pp. 186–191, 2013.

[46] F. Bergenti, A. Poggi, and M. Tomaiuolo, *Handbook of Research on ICTs for Human-Centered Healthcare and Social Care Services*, IGI Global, USA, pp. 549–567, 2013.

[47] N. Benhajji, D. Roy, and D. Anciaux, "Patient-centered multi agent system for health care," *IFAC-PapersOnLine*, vol. 48, no. 3, pp. 710–714, 2015.

[48] V. Julian and V. Botti, "Multi-agent systems," *Applied Sciences*, vol. 1402, 2019.

[49] O. James and N. Marian, "The foundation for intelligent physical agents," 2003, https://www.fipa.org/docs/input/f-in-00085/f-in-00085.pdf.

[50] M. Luck, P. McBurney, O. Shehory, and S. Willmott, *Agent Technology: Computing as Interaction (A Roadmap for Agent Based Computing)*, AgentLink, Louisville, USA, 2005.

[51] APSIG, Agent platform special interest group, https://agent.omg.org/.

[52] N. Suri, J. M. Bradshaw, M. R. Breedy et al., "Nomads: toward a strong and safe mobile agent system," in *Proceedings of the Fourth International Conference on Autonomous Agents*, pp. 163-164, Barcelona Spain, June 2000.

[53] N. Suri, J. M. Bradshaw, M. R. Breedy et al., "An overview of the nomads mobile agent system," in *Proceedings of the Workshop On Mobile Object Systems in association with the 14th European Conference on Object-Oriented Programming (ECOOP 2000)*, Cannes, France, June 2000.

[54] F. M. T. Brazier, D. G. A. Mobach, B. J. Overeinder, S. van Splunter, M. van Steen, and N. Wijngaards, "Agentscape: middleware, resource management, and services," in *Proceedings of the 3rd international SANE Conference*, pp. 1–3, Maastricht, The Netherlands, May 2002.

[55] S. Willmott, J. Dale, B. Burg, P. Charlton, and P. O'Brien, *Agentcities: A Worldwide Open Agent Network*, Agentlink News, Louisville, USA, 2001.

[56] D. B. Lange and O. Mitsuru, *Programming and Deploying Java Mobile Agents Aglets"*, Addison-Wesley Longman Publishing Co., Inc., MA, USA, 1998, https://en.wikipedia.org/wiki/Boston.

[57] G. Glass, "ObjectSpace voyager-the agent ORB for Java," in *Proceedings of the International Conference on Worldwide Computing and Its Applications*, pp. 38–55, Tsukuba, Japan, March 1998.

[58] S. Galland, N. Gaud, S. Rodriguez, and V. Hilaire, *Janus: Another yet General-Purpose Multiagent Platform*Seventh AOSE Technical Forum, Paris, France, 2010.

[59] N. P. Sudmann, *Tacoma–fundamental Abstractions Supporting Agent Computing in a Distributed Environment*, pp. 33–59, Department of Computer Science, University of Tromso, Norway, 1996.

[60] D. Johansen, F. B. Schneider, and R. V. Renesse, "What tacoma taught us," *Mobility, Mobile Agents and Process Migration–An Edited Collection*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 1999.

[61] C. Bäumer and T. Magedanz, *Grasshopper?a mobile Agent Platform for Active Telecommunication Networks*, pp. 19–32, Springer International Workshop on Intelligent Agents for Telecommunication Applications, Paris, France, 1999.

[62] F. L. Bellifemine, G. Caire, and D. Greenwood, *Developing Multi-Agent Systems with JADE*, Wiley, Hoboken, NY, USA, 2007.

[63] Y. El-Gamal, K. El-Gazzar, and M. Saeb, "A comparative performance evaluation model of mobile agent versus remote method invocation for information retrieval," *Proceedings of World Academy of Science*, pp. 1–6, 2007.

[64] M. Cossentino, S. Lopes, A. Nuzzo, G. Renda, and L. Sabatucci, "A comparison of the basic principles and behavioural aspects of akka, jacamo and jade development frameworks," in *Proceedings of the 19th Workshop from Objects to Agents (WOA)*, pp. 133–141, New Jersey, USA, April 2018.

[65] R. H. Bordini and J. F. Hübner, "Bdi agent programming in agentspeak using jason," in *Proceedings of the International Workshop on Computational Logic in Multi-Agent Systems*, pp. 143–164, London, UK, June 2005.

[66] J. P. Bigus, D. A. Schlosnagle, J. R. Pilgrim, W. N. Mills III, and Y. Diao, "Able: a toolkit for building multiagent autonomic systems," *IBM Systems Journal*, vol. 41, no. 3, pp. 350–371, 2002.

[67] F. B. G. Caire, A. Poggi, and G. Rimassa, "Jade. a white paper," *Telecom Italia Lab*, vol. 3, pp. 1–14, 2003.

[68] K. Miller, G. Mansingh, and G. Mansingh, "Comparing the use of mobile intelligent agents vs. client server approach in a distributed mobile health application," *Journal of Computers*, vol. 10, no. 6, pp. 365–373, 2015.

[69] G. A. Aderounmu, B. O. Oyatokun, and M. O. Adigun, "Remote method invocation and mobil agent: a comparative analysis," *Issues in Informing Science and Information Technology*, vol. 3, 2006.

[70] R. H. Bordini, L. Braubach, M. Dastani, A. E. F. Seghrouchni, and J. J. Gomez-Sanz, "A survey of programming languages and platforms for multi-agent systems," *Informatica*, vol. 30, no. 1, pp. 33–44, 2006.

[71] K. Kravari and N. Bassiliades, "A survey of agent platforms," *The Journal of Artificial Societies and Social Simulation*, vol. 18, no. 1, Article ID 11, 2015.