Peer-to-Peer Networking and Applications (2021) 14:729–746
https://doi.org/10.1007/s12083-020-01027-9

Resource discovery mechanisms in pure unstructured peer-to-peer systems: a comprehensive survey

Elahe Khatibi[1] · Mohsen Sharifi[1]

Received: 14 May 2020 / Accepted: 5 November 2020 / Published online: 26 November 2020

Abstract
The concept of unstructured Peer-to-Peer (P2P) systems—setting free from any structural constraints—has put forward an appropriate paradigm for sharing a wide assortment of resources in a distributed-sharing manner efficiently. More importantly, unstructured P2P systems' architecture and concepts have permeated diverse spheres of today's successful and world-famous computer science areas, including NoSQL databases for excellently sharing data. However, pinpointing any given appropriate resource in such massive systems, namely unstructured P2P systems, is a challenging task; the two of the most dispensable rationales behind this proclaim are large scales of such systems and unstructured nature of overlay networks. Finding a decent resource with low response time, low bandwidth consumption, and high success-rate has played a crucial role in both the overall system performance and the functionality of P2P systems. Briefly, an efficacious resource discovery mechanism is the lifeblood of any productive P2P system. Given these points, in this study, we present an exhaustive survey on state-of-the-art resource discovery mechanisms employed in file-sharing pure unstructured P2P systems; we offer a new resource discovery categorization accordingly. Furthermore, we deeply delve into a plethora of resource searching methods and their merits as well as demerits to furnish the paper with an in-depth evaluation.

Keywords Unstructured peer-to-peer systems · Resource discovery mechanism · Taxonomy · File-sharing

1 Introduction

Distributed systems are introduced to transparently connect resources and users in a scalable manner. This is because the traditional solution—client/server—is no longer proper for large-scale distributed applications. Peer-to-Peer (P2P) systems as a kind of distributed systems enable scalable and high-throughput resource sharing among peers. Hence, today's distributed applications can rely on the P2P paradigm to become both flexible and efficient [1].

Peer-to-Peer (P2P) systems are comprised of a network of connected nodes, having comparable capabilities and roles, sharing their resources in a distributed way [2–6]. P2P systems provide access to a massive pool of shared self-organized resources—this is the main philosophy behind the P2P concept. In other words, the key idea behind the P2P paradigm is to utilize idle resources to do useful things like content sharing or cycle sharing [1]. There is an increasingly perpetuating interest in exploiting the P2P concepts and its architecture in a plethora of different modern commercial applications and technologies. Popular applications and technologies emanating from P2P systems include File-Sharing Applications, Video as well as Audio Streaming, NoSQL Databases, Content Distribution Networks, Cloud Storage Systems, and Blockchain protocol [2, 3]. These applications enjoy multiple benefits from P2P systems such as reliability, scalability, robustness, shared computing, shared storage platform, autonomy, fault-tolerance, and applicability [7–11].

P2P systems assume a logical overlay network for the arrangement of nodes and are classified into two classes: *unstructured* P2P systems and *structured* P2P systems.

Highlights
• We put forth a novel taxonomy for resource discovery mechanisms in pure unstructured Peer-to-Peer systems.
• We thoroughly investigate contributing factors in devising any given resource discovery mechanism from a myriad of perspectives.
• We present a complete evaluation of each proposed class of resource discovery mechanisms.

✉ Mohsen Sharifi
msharifi@iust.ac.ir

Elahe Khatibi
elahe.khatibi@iust.ac.ir; elahe.khatibi@gmail.com

1 School of Computer Engineering, Iran University of Science and Technology, University Road, Narmak, Tehran 1684613114, Iran

Overlay networks are formed at application layer on top of the underlying physical networks; nodes are connected through virtual/logical links, and each path in the overlay network may be equivalent to many physical links in the underlying network [5]. Figure 1. illustrates the architecture of an overlay network. Nodes intending to join an overlay network should find at least one node that is already part of the overlay network—this operation is known as bootstrapping [4, 5].

Structured P2P systems impose a strict role on forming a well-structured overlay network for the sake of ensuring high performance of search mechanisms; some examples of structured P2P systems include *Chord*, *CAN*, and *Tapestry*, to name but a handful [7, 12]. Maintaining as well as updating such a rigid structure with a high peer churn-rate has a mounting overhead on the whole system performance—high overlay maintenance cost. One of the distinct features of the P2P systems that distinguishes them from other types of distributed systems is peer churn—peer arrivals and departures; different studies show that most nodes stay in the systems for less than 10 min [6].

In this paper, we focus on pure unstructured P2P systems putting no strain on the network structure—organize nodes randomly to form a network graph [3]. In this regard, without any single shadow of a doubt, one of the indispensable challenges in such P2P systems is finding an appropriate resource for a resource request. Resource discovery is also crucial for scaling unstructured P2P networks.

Both distributed nature and dynamicity of P2P systems profoundly affect the resource searching process. Other contributing factors, exerting an influence over searching process, include skewed user query patterns, uneven distribution of workloads, unreliability of nodes, and node heterogeneity [12–16].

In this study, in comparison to other existing surveys in this scope [1, 5, 6, 17–28] we illustrate an exhaustive literature review of resource discovery issue in pure unstructured P2P systems; more importantly, our survey encompasses a new class for resource discovery mechanisms—bio-inspired

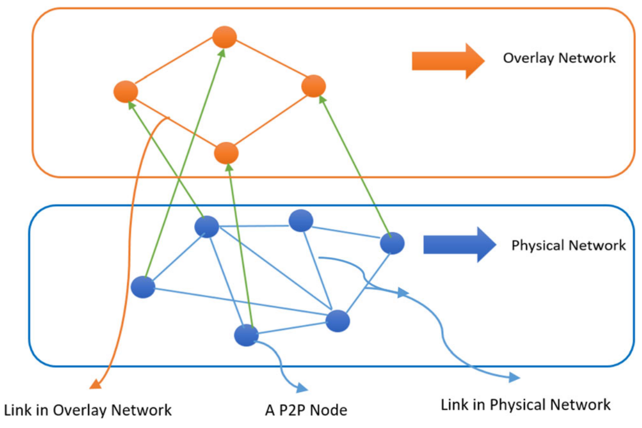


Fig. 1 An Example of a P2P Network

mechanisms—being omitted from other proposed surveys. Furthermore, a new categorization for blind search mechanisms are suggested. Section 2 presents an introduction to the essential concepts of the P2P paradigm and its models. Section 3 focuses on introducing resource discovery concepts and contributing factors with particular emphasis on designing efficient resource discovery mechanisms. Section 4 proposes a novel classification of resource discovery mechanisms in pure unstructured P2P systems and reviews some important search mechanisms in each group alongside their comprehensive appraisals. Section 5 concludes the paper.

2 Peer-to-peer systems and their underlying concepts

2.1 Peer-to-peer systems

P2P technologies are classified into two main groups, namely P2P applications and P2P infrastructure (Fig. 2.). The P2P application group entails the content distribution service, while P2P infrastructure merely provides some frameworks, and services such as routing, and reputation management services. The P2P application group is in turn divided into two classes: file exchange systems and content publishing as well as storage systems, wherein a content distribution medium is established as well [7–14].

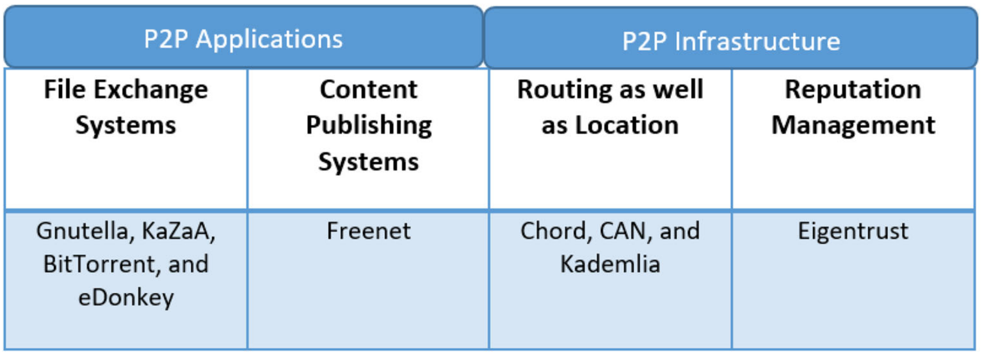
2.2 The architecture of a P2P node in a file-sharing P2P system

Different units of every P2P node can be classified into three parts (Fig. 3.) [7–16]:

1. A user interface whose main thrust is to record a received query.
2. A data management layer that oversees the actions, including query processing and management of metadata information.
3. A P2P infrastructure that encompasses both a network of P2P nodes and a relevant overlay network.

When the *user interface unit* receives a query, it passes the query to the components in the data management layer. The *query manager unit* processes the query; it uses the rewards of both stored *metadata* and *semantic mapping units* to find the nodes having the requested resource. The query manager then invokes the related services offered by the underlying infrastructure to contact with the relevant nodes.

Moreover, in super-node/super-peer systems, only super-nodes process the query. In other systems, the outcomes of query processing are stored in some intermediate nodes for the



P2P Applications		P2P Infrastructure	
File Exchange Systems	**Content Publishing Systems**	**Routing as well as Location**	**Reputation Management**
Gnutella, KaZaA, BitTorrent, and eDonkey	Freenet	Chord, CAN, and Kademlia	Eigentrust

Fig. 2 Classification of P2P Technologies

sake of faster response to similar forthcoming queries; the *data cache manager unit* stores the above results to expedite response to future queries. When it comes to the updating of data, the *update manager* has the responsibility of updating and keeping diverse versions of replicated data consistent.

The *P2P network sub-layer unit* can be implemented as either an unstructured or a structured network to provide communication service for the data management layer accordingly [7–22, 24, 29, 30]. Due to the importance of data management layer, we briefly explain some of its functionalities hereunder.

2.2.1 Data management layer

P2P systems have resources such as files, CPU cycles, bandwidth, and storage space; system nodes can perform operations such as searching, inserting, deleting, and finding files. Consequently, any data/resource management unit as a key component in a P2P system should be able to carry out the following operations:

- Content Deletion and Update: Although this is not a common function of a resource management unit in P2P systems, it calls for an effective synchronization process. These operations can be implemented by some sequence of immutable files as in the MojoNation system [22].
- Content Versioning: This is a complicated function and is implemented in the OceanStore system [22] via a version-based archival storage system.
- Directory Structure: The Mnemosyne system [22] has this structure and introduces a group of files with a defined name and key.

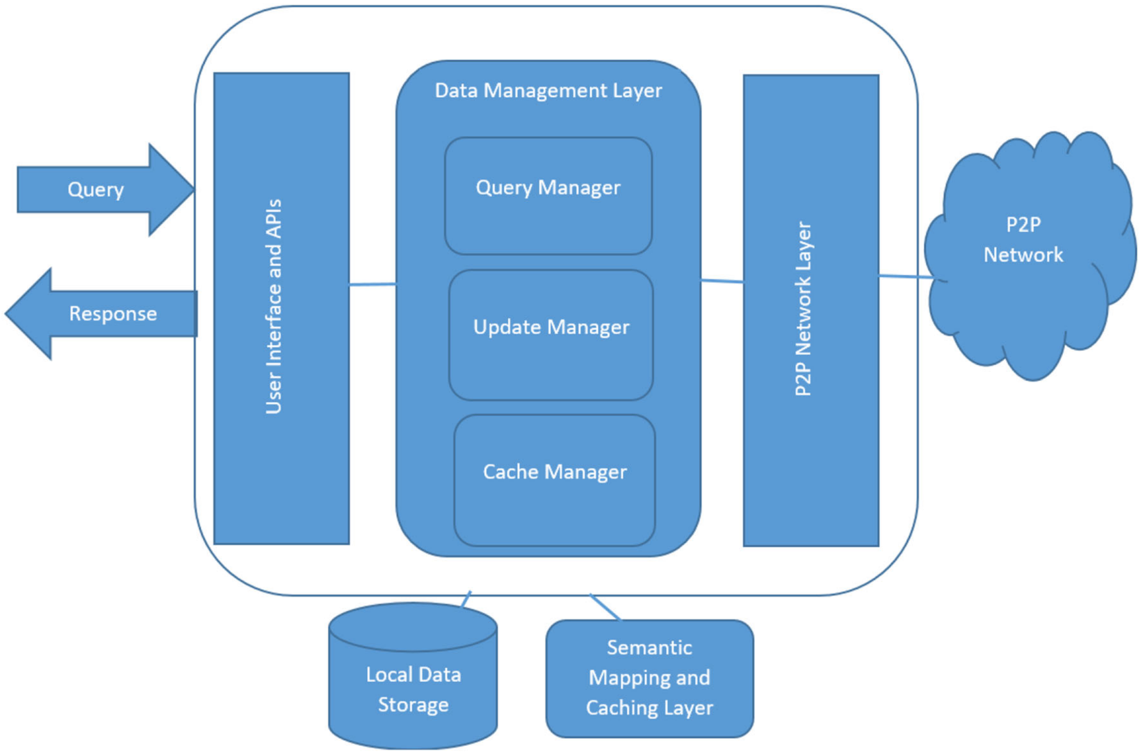


Fig. 3 A Typical Architecture of a P2P Node

- Content Searching: Resource searching in the unstructured P2P systems is done via keyword searching, though convenient, it suffers from low-performance resource searching mechanisms. In contrast, structured P2P systems have an efficient search mechanism, but these structural-based mechanisms impose additional overheads on systems; this overhead is due to the maintenance and update of the structured P2P overlay under churn. Moreover, structured P2P systems fail to conduct keyword search—they are solely proper for an exact match.

  Any search mechanism encircles two units:
- Query Processing: This defines a node's ability in terms of processing of any given query.
- Data Location: This determines the node's capability in terms of finding the related nodes having the requested resource.
- Storage and Bandwidth Management: This unit is put into practice differently in various systems; for instance, the MojoNation system has utilized economic factors to share hard disk space among nodes.
- Content Expiration: This is implemented by using a set of well-defined timestamps, e.g., as in the FreeHaven system [23].
- Metadata Management: The use of metadata facilitates the search process by including information such as file name, file size and file description.
- Replication and Data Consistency: The resource management unit should take the responsibility of striking consistency among different versions of data; this is because data is disseminated and cached in different nodes.
- Data Integration: When data is shared among system nodes based on various schemas and tree structure models, all nodes should be able to find any requested data despite these heterogeneities.
- Load Balancing: Load balancing function tries to bring the idle nodes into play and assign them some of the workloads, thereby reducing the overloading issue in the whole system [7–16].

2.3 Classification of resource management mechanisms in P2P systems

P2P systems can be grouped into three classes (Centralized, Pure, and Super-Node) with regard to the location of both index and information of resources; these information are used in the processing of any given query, routing, and searching for preferred resources. Figure 4. shows these three classes.

1) Centralized Model: In this model (Fig. 4. a)), all indexes of system files are stored in one server, or a cluster of servers, and these servers will be in charge of directing the searching process. Although this model ensures a high-speed searching process, it is not scalable. This is because of being dependent on a group of servers—causing single-point-of-failure problem. Napster, eMule system [7, 12] and the BitTorrent system [7, 12] follow this model [16, 24, 25, 30, 31].
2) Pure Model: In this model (Fig. 4. b)), every node acts both as server and client. Furthermore, each node stores the information and the indexes of their files; overall, all system nodes are equal in terms of function. Gnutella version 1.0 [7, 12, 32] belongs to this model [23, 33, 34].
3) Super-Node Model: In this model (Fig. 4. c)), there are a special group of nodes, namely super-nodes, which, in turn, keep supervision of a small group of nodes. They imitate the function of a central server for just a handful number of nodes; hence, they store the file indexes of the nodes under their memberships; the super-nodes themselves connect in a decentralized manner. Gnutella version 2.0 [7, 12] and KaZaA [7, 12] belong to this model [26, 35–38].

Table 1 summarizes a comparison of the above-mentioned models. Although the centralized model has higher search performance, it suffers from lack of both scalability and fault-tolerance due to dependency on a single server. The pure model enjoys higher scalability and is more fault-tolerant in comparison to the centralized model; however, its main issue is low search performance; in the pure model, there are more workloads on nodes that are involved in any given search process. The super-node model trades off scalability for better search performance. Its scalability is somehow between the centralized and pure models.

3 Resource discovery

3.1 Search mechanism definition

When a (sender) node wants a resource, it creates a query. Then, based on the defined search mechanism, the query will be sent to some nodes; if the node receiving the query has the requested resource, it sends a response message to the sender from the reverse of the same path passed by the query message. It is worth mentioning that a suitable search mechanism should strike a balance between cost—bandwidth consumption in terms of the number of generated messages per query—and benefit in terms of the number of finding resources; an efficient search mechanism should also find the requested resource in low response-time. These factors should be weighted according to the

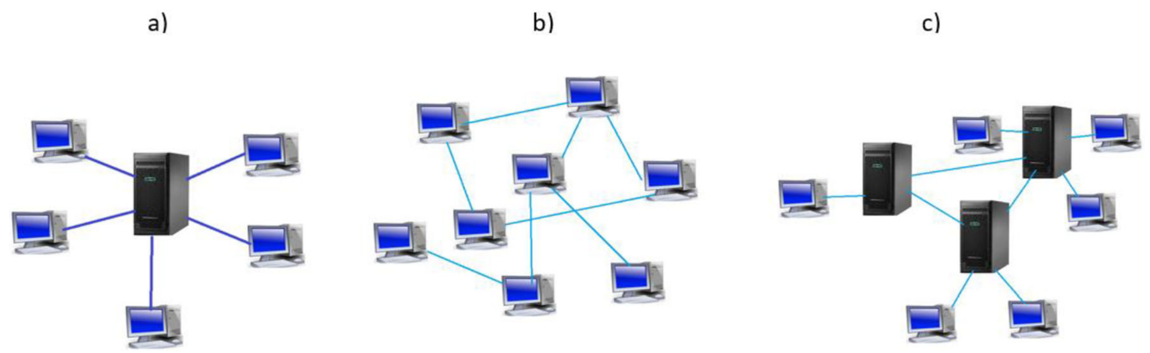


Fig. 4 **a** Centralized, **b** Pure, and **c** Super-Node Models

purposes and requirements of any application [14, 34, 39, 40].

3.2 Summary of steps in a search mechanism

- Query sending mechanism
- Selection of candidate nodes and the number of candidate nodes that are going to process the query
- Query message format
- Query processing algorithm in each node
- Both indexes and information locally stored and updated in each node [29]

3.3 Estimating the performance of a search mechanism

- Success-rate: the number of response messages is a quality service considered by a user; in fact, success-rate is presented as a fraction of successful searches for a requested resource [41].
- Response time: an overhead imposed on the network, defined as an interval between the initiation of a search and the time when the resource is found [41].
- Bandwidth consumption: the number of total generated and propagated messages, an overhead imposed on the network [42].

4 Taxonomy of resource discovery mechanisms in pure unstructured P2P systems

We categorize the resource discovery mechanisms in pure unstructured P2P systems into five groups: *Blind*, *Informed*, *Group-based*, *Hybrid*, and *Bio-Inspired Meta-Heuristic* (Fig. 5.).

4.1 Blind search mechanisms

Blind mechanisms store no information regarding resources to direct the search process; thus, the search process fails to reach the nodes that are more capable to respond positively to the query in an informed way [43]. These mechanisms force many nodes to process the query in a search process to achieve the result; in this regard, they bring about additional overhead and bandwidth consumption [42, 44].

Because blind mechanisms have failed to store any relevant information concerning the resources, they send a query message as much as possible to a large number of nodes. Accordingly, this trend generates traffic jams and eats away network resources. Two fundamental search mechanisms lay the basis for many search mechanisms in both informed and blind groups: *flooding* and *random walk* [14–16, 29]. We have proposed a new category for blind search mechanisms, which are grouped into three major classes: flooding, random walk, and popularity-aware. To put it simply, the flooding class uses flooding, the random walk class utilizes random

Table 1 Comparison of Centralized, Pure, and Super-Node Resource Management Models

Performance Metrics Resource Management Architecture	Search Performance	Scalability	Resilience to Single-Point-of-Failure
Centralized Model	High	Low	Low
Pure Model	Low	High	High
Super-Node Model	Medium	Medium	Medium

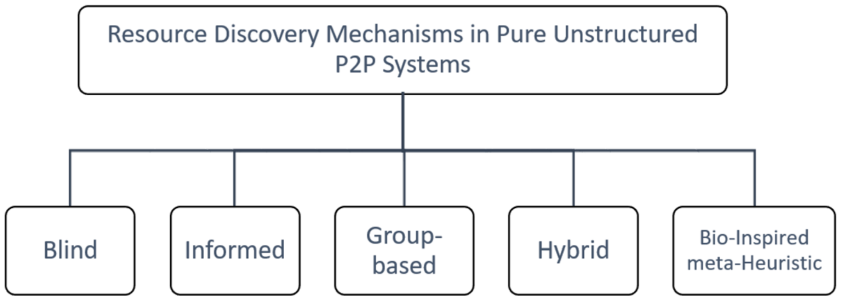

Resource Discovery Mechanisms in Pure Unstructured P2P Systems
Blind
Informed
Group-based
Hybrid
Bio-Inspired meta-Heuristic

Fig. 5 Taxonomy of Resource Discovery Mechanisms in Pure Unstructured P2P Systems

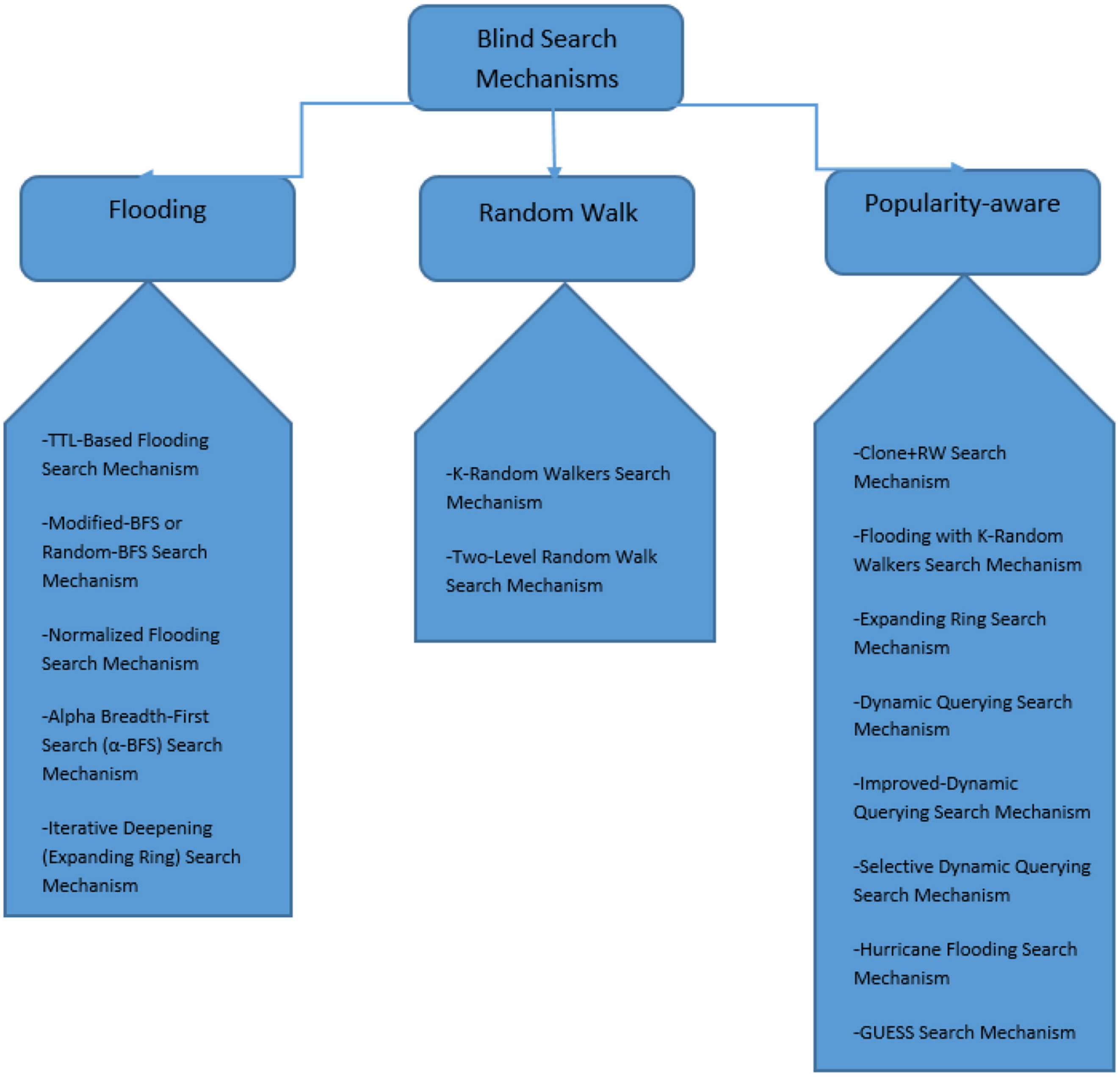

Blind Search Mechanisms
Flooding
Random Walk
Popularity-aware
-TTL-Based Flooding Search Mechanism
-Modified-BFS or Random-BFS Search Mechanism
-Normalized Flooding Search Mechanism
-Alpha Breadth-First Search (α-BFS) Search Mechanism
-Iterative Deepening (Expanding Ring) Search Mechanism
-K-Random Walkers Search Mechanism
-Two-Level Random Walk Search Mechanism
-Clone+RW Search Mechanism
-Flooding with K-Random Walkers Search Mechanism
-Expanding Ring Search Mechanism
-Dynamic Querying Search Mechanism
-Improved-Dynamic Querying Search Mechanism
-Selective Dynamic Querying Search Mechanism
-Hurricane Flooding Search Mechanism
-GUESS Search Mechanism

Fig. 6 Classification of Blind Search Mechanisms

```
For each node x receiving the query
If x has the requested resource then
        Send a response message in reverse path direction
End if
If (--TTL>0) then
        Send the query message to all neighbors in a parallel fashion
End if
```

Fig. 7 Pseudocode of the Flooding Mechanism

walk, and the popularity-aware class considers popularity of resources to orient the search process as its main goal. In other words, the popularity-aware group tries to put into practice the best searching method according to estimated popularity of files. Figure 6. shows this classification.

4.1.1 Flooding

TTL-based flooding search mechanism This mechanism utilizes the breadth-first mechanism for search. The search depth or the number of hops that should be passed by the query message is defined in the message as the TTL value. Figure 7. shows the pseudocode of the flooding mechanism. Since the breadth-first search or flooding mechanism continues the search until the TTL value equals to zero, it leads to high traffic. The flooding search squanders network resources drastically, and it causes overshooting; overshooting means finding many response messages, i.e. more than required. The flooding mechanism is beneficial for searching rare resources; however, when it comes to finding a popular one, it consumes network resources [27, 44].

Modified-BFS or random-BFS search mechanism Modified-BFS (MBFS) has been introduced to resolve the flaws of the flooding mechanism to dwindle the generated traffic and lessen the number of nodes involved in query processing. To this end, a query will be sent only to a sub-group of neighbors opted randomly [27, 44]. Although MBFS reduces the traffic, it still creates traffic overhead.

Normalized flooding search mechanism This mechanism, whose main thrust is traffic reduction, sends the query to a subset of neighbors randomly. If the minimum degree of a node in the system—that is, the number of neighbors of each node—is d, the query will be sent to only d nodes at each hop. This mechanism fails to consider node heterogeneity. Furthermore, the value of d parameter is an important value for search performance, as it should be able to cover all useful nodes [45].

Alpha breadth-first search (α-BFS) search mechanism A newer version of BFS like Alpha Breadth-First Search (α-BFS) mechanism [15] tries to diminish the network resources wastage. Even though this mechanism reduces the average message traffic compared to the flooding scheme, it still generates too much overhead and suffers from uninformed query forwarding. Hence, it causes performance degeneration.

Iterative deepening (expanding ring) search mechanism This mechanism intends to reduce the number of returned response messages, which are more than the required ones—overshooting issue. This mechanism is efficient for applications emphasizing the number of returned response messages. However, this mechanism brings about prolonged response time due mainly to waiting time between each search step. It also squanders network resources by sending duplicate query message every time. Precisely speaking, the Expanding Ring (ER) mechanism defines a set of depths, namely TTLs, by which the query should be sent to all of the neighboring nodes by the flooding method. In the first depth, ER waits until the requester node receives their responses. If the number of responses is adequate, the search ceases; otherwise, the search will continue a second depth value; all of the nodes on the previous depth will receive the query again, but they will drop it.

To solve the defects of the ER, some Dynamic Query mechanisms [46–50] have been suggested to stop sending a query to repetitive nodes at each stage. They also pay attention to both overloading issue and resource popularity, so they dynamically put an end to the search, when it is required.

4.1.2 Random walk

Random walk search mechanism Random walk (RW) solves the traffic issue resulting from flooding by sending a query at each hop to just one neighbor that is elected randomly. When the result is found, the search process stops. Although RW leads to traffic reduction and load balancing, it causes prolonged response time and lower success-rate after all. This mechanism fails to take into consideration resource popularity, node heterogeneity, and storing of metadata to guide the search process [45].

K-random walkers search mechanism To resolve the prolonged delay and low success-rate of the random walker technique, the K-Random Walkers technique has been proposed. In this technique, at each search step, instead of sending one random walker, K random walkers are sent simultaneously—each of these K queries is called a random walker. Although the K-Random Walkers technique improves the time-delay by K factor, the network traffic increases. Termination of search process can be implemented by either checking the TTL value or contacting at each step with the query sender to be informed whether the result has been received or not [45, 50, 51].

Two-level random walk search mechanism In the K-Random Walker technique, it is plausible that random walkers collide with the search scopes of each other, since they may send the same query message to identical nodes many times. In light of this, a two-level random walk mechanism starts sending a query with a small value for both K and TTL. Shortly afterward, when it is assured that walkers do not have overlapping search scopes, the query is sent with higher values for both K (K = 2 K) and TTL. It is worth mentioning that widening of the search scopes after the first hops imposes a delay on the search process due mainly to the fact that walkers take a long path from the sender [45, 46].

4.1.3 Comparison between two major blind search classes–flooding and random walk

We consider TTL = T and D = d, i. e. the average degree of nodes. As we can see in Table 2., the overhead of flooding explodes exponentially in terms of T, whereas in random walk, it grows linearly with T and K. Researchers who want to devise an efficient resource discovery algorithm should adaptively set TTL and K in order to satisfy a desired level of performance [41].

4.1.4 Popularity-aware blind search mechanism

The popularity of each resource has a profound impact on the performance of any given search mechanism. However, estimating resource popularity in a dynamic P2P system is a difficult task; a query initiator node cannot determine the popularity of resources confidently without any feedbacks from previous searches. Moreover, popularity of resources can vary due to insert/delete/replication of resources, nodes join/leave, or any other dynamicity in the P2P systems.

Furthermore, when a satisfactory number of resources is found, the search process should be ceased to reduce the traffic and the overshooting issue—an excessive number of returned response messages exceeding the required number. In this regard, the following mechanisms are proposed:

- Mechanisms that control the search scope via TTL value.
- Checking mechanisms that contact the sender after a certain number of hops to determine whether the search should be stopped.
- Chasing mechanisms in which the resource owner sends some packets over the network to stop other random walkers, when the requested resource is found [16, 51, 52].

Table 2 Comparison between Flooding and Random Walker Algorithms

Search Algorithm	Order of Generated Messages
Flooding	O((d-1)^T)
Random Walk	O(K*T)

Clone+RW search mechanism Flooding is so robust in finding rare resources, and random walk is fruitful in pinpointing the popular ones, so both lead to less traffic and high probability of success for finding the rare and popular resources, respectively. In this regard, Clone+RW tries to combine the above-mentioned facts and start by random walk via small TTL. If the resource is rare and not found by RW, random walkers turn to flooding to find the resource by creating a dominating set. Dominating set contains nodes with high degrees and many resources to ensure finding rare resources; dominating set also eliminates sending repetitive messages resulting from the flooding mechanism [43, 53].

Flooding with K-random walkers search mechanism This method combines the benefits of both flooding and random walker mechanisms. Accordingly, first, the query initiator floods a query by TTL = 1 to pinpoint *K* nodes, and if during flooding the requested resource is found, the search will cease; otherwise, each of the *K* nodes starts a search using the RW method. In light of these, not only does network traffic from flooding reduces, but also random walkers can find rare resources in farther distances from the sender. Furthermore, the overlapping of search scopes is avoided [45].

Expanding ring search mechanism A query is flooded via a low TTL. If an adequate number of resources is not found, the search continues with a higher value for TTL. This mechanism suffers from delays and sending repetitive messages to previous nodes.

Dynamic querying search mechanism The rationale behind devising this method is that the search process should be different for finding rare and popular resources to reduce network costs. This method first floods a query with low TTL to estimate the resource popularity based on returned results. Then, it tries to come up with the right value for TTL, and then sends the query to merely one node at each step by calculated TTL. As a result, this method leads to a decline in traffic, eliminates the situation in which nodes receive repetitive messages, and takes into account resource popularity. Nonetheless, it causes latency; moreover, it presumes that network topology should have super-nodes whose degrees are more than 15 by which the search mechanism can be able to work efficiently [32, 54].

Improved-dynamic querying search mechanism This mechanism solves the delay of the Dynamic Querying mechanism by exploiting one confident function to assign a higher value to TTL at each step. Although the response time is improved in comparison to the Dynamic Querying search mechanism, the Improved-Dynamic Querying search mechanism still suffers from delay [54].

Selective dynamic querying search mechanism This mechanism takes advantage of the benefits of low response time of the Expanding Ring and low traffic of the Dynamic Querying mechanisms; to achieve this, it sends a query at each hop to more than one node by considering diversity in nodes' degrees [55].

Hurricane flooding search mechanism This method is similar to Expanding Ring; it, however, expands the search scope gradually rather than sharply. Firstly, the sender splits their neighbors into *r* groups and sends a query to the first group with low TTL, which, in turn, is flooded to neighbors of nodes of the first groups as well. If the result is not found, the query is sent to the second group with a rise in TTL value, and so forth in a spiral pattern. If the result is not found, the query is cyclically sent to the first group again. Generally, this mechanism reduces response time compared to ER [56].

GUESS search mechanism Search mechanisms are categorized into *Non-Forwarding* and *Forwarding* groups (Fig. 8.). This classification is grounded on whether or not the sender is able to control the search scope; controlling the search scope is defined in terms of both selecting specific candidate nodes and the number of candidate nodes.

If the search mechanism tries to control the search scope, it will belong to a non-forwarding group; although the non-forwarding group is more efficient, it eats away system resources due to the requirement to store information concerning other nodes. The forwarding group can be classified based on probabilistic or certainty aspect; the forwarding group encompasses search mechanisms like Expanding Ring, Adaptive Probabilistic Search, Directed-BFS, Local Indices, Routing Indices, BFS, Intelligent Search Mechanism, and K-Random Walker.

GUESS can control the number of candidate nodes and node selection process—search scope—by considering the resource popularity. In this regard, GUESS belongs to the non-forwarding group. At first, the query with low TTL is sent to one super-node—unicast process. After some time, if the required number of resources is not received, the query is sent to the next super-node. It is obvious that each node should be aware of many nodes whereby we have a network topology that ensures a successful search, but it causes overhead. The performance of GUESS is better than both Gnutella and Iterative Deepening, as it sends a parallel query at each hop [57–60].

4.2 Informed search mechanisms

Informed methods use some information to direct the search process to the nodes more probable to respond to the query. This group takes advantage of feedback from previous searches, thus storing some information to guide future searches. However, this leads to the overhead of updating such information and indexes, thereby overloading some nodes [16]. In summary, although the informed group enjoys lower response time compared to the blind group, it culminates in more traffic due to updating of indexes.

4.2.1 Classification of informed search mechanisms based on information gathering methods

Informed search mechanisms can be grouped in two classes (Fig. 9.), *proactive* and *reactive*, with regard to the methods they use for storing information.

- Proactive Methods: Information regarding resources are distributed among nodes before the search process starts, like the Routing Indices mechanism [16].
- Reactive Methods: This group stores information based on the feedback from returned results of previous searches. Therefore, in reactive methods in comparison

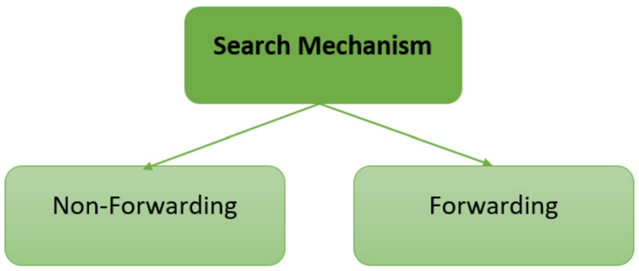


Fig. 8 Classification of Search Mechanisms based on Controlling Search Scope

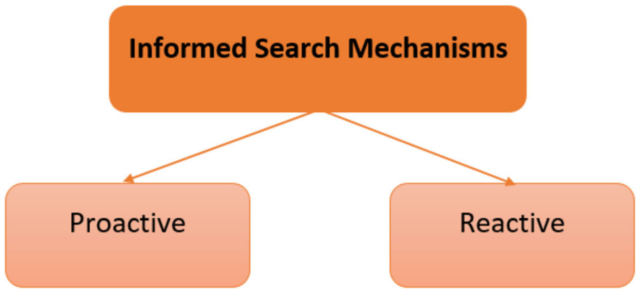


Fig. 9 Classification of Informed Search Mechanisms based on Gathered Information

to proactive ones, system resources are not wasted by storing information on rare resources; rare resources may not ever be searched at all. However, the reactive group suffers from not updating information to boost search performance over time. The Intelligent Search Mechanism (ISM) [16] belongs to this group. The information employed by informed search mechanisms—reactive group—are as follows:

- The number of returned responses from previous searches
- A previous search has the smallest response time
- Smallest query message queue
- Recent searches result in shortest response time
- Nodes with highest degree
- Overall maximum returned responses
- Overall minimum response time
- Similarity of requested resources

The main disadvantage of informed search mechanisms is that all of them use merely one of the factors mentioned above to guide the search process while in real-world implementation prototypes, all of the factors should be considered simultaneously to lead to an acceptable performance/result.

4.2.2 Ant search mechanism

The free-riding problem consists of two types; firstly, it is mentioned that 63% percent of users are reluctant to share any files, and in the second type, users share unpopular files subject to no query anymore. In this regard, some specific nodes in the system having popular files are prone to become overloaded.

The ant search mechanism aims to solve the free-riding issue. In the ant search mechanism, all nodes store a list of recent success rates of their neighbors by which they try to send their queries solely to K% of neighbors who have recently won the most success-rate. This mechanism is analogous to the Dynamic Querying method. Although it reduces sending repetitive query messages by resolving the free-riding issue, it still suffers from latency and overshooting issues [61, 62].

4.2.3 Intelligent search mechanism

The Intelligent Search Mechanism (ISM) intends to sort out the lack of scalability issue of the flooding mechanism. Hence, in ISM, nodes learn from previous responses of their neighbors and try to send their queries to their best neighbors, having a higher probability of owning the resource. The pivotal issue of ISM is that it is possible that each time some specific nodes are selected as candidates, and they elect each other repeatedly. However, none of the selected candidates possesses the resource; to solve this flaw, at each step, some nodes should be selected arbitrarily [63, 64].

4.2.4 Ranked neighbor caching search mechanism

The Gnutella network is a power-law network; in power-law networks (scale-free networks), there are only a few nodes whose degrees are much higher than other nodes in the network, i.e. power-law distribution—this is a common feature of real-world networks. Moreover, in Gnutella, 63% of users are reluctant to share any files. These features impose extra workload on only the high-degree nodes, thus overloading the network.

The Ranked Neighbor Caching method solves this issue by sending the query to one neighbor at each hop. It selects the neighbor that is more likely to respond based on their previous responses. Furthermore, the ranked neighbor caching mechanism utilizes a caching facility to redirect the query to the nodes recently downloading the requested resource, for the sake of load-balancing [65].

4.2.5 Differentiated search mechanism

Although the main goal of the P2P concept is to treat all nodes equally by eliminating server/client roles, nodes have different capabilities; for instance, only small groups of nodes share resources. Therefore, the Differentiated search method tries to consider another factor, namely the number of files for the super-node selection. Thus, each query is flooded by considering this factor among super-nodes [66].

4.2.6 Local indices search mechanism

In this mechanism, each node stores the information about the resources of the nodes that are in *r*-hop distance from them. Besides, there is some well-defined policy to determine the depth in which nodes that receive a query should process the query or not. Therefore, the local indices mechanism searches more nodes while preserving the quality of service [46].

4.2.7 Directed BFS search mechanism

Since the iterative deepening method—belonging to the blind group—causes prolonged delay due mainly to waiting time between each search step, Directed BFS (D-BFS) strives to reduce the response time by sending a query only to the best neighbor at each step [46].

4.2.8 Percolation search algorithm

This algorithm tries to solve the traffic issue, which is generated by the flooding search mechanism. The Percolation search algorithm reaps the rewards of high-degree nodes like a bridge to the best nodes of the system—having many resources. However, this method, comprising of three steps, is practical only in the power-law network.

In the Percolation search algorithm, when a new node joins the system, it broadcasts information regarding its resources to its neighbors by RW of TTL = LogN (*N* is the total number of system nodes). In this way, at least one high-degree node stores the resource information of that new-join node. When it comes to searching, the requester node starts a search process by RW of TTL = LogN to implant its query in at least one high-degree node. Ultimately, all nodes storing the query message initiate searching in parallel. The time complexity order of this algorithm is of the order of log [46].

4.2.9 HPF search mechanism

Partial coverage problem Informed search mechanisms reduce network traffic, and based on simulation results, they lead to a decline in both bandwidth consumption and processing power by 28% and 38%, respectively. However, they do not cover many nodes, as they either select a handful of nodes or opt for best neighbors with regard to merely one factor. These result in partial coverage issues, meaning that many nodes are not contacted.

In this regard, HPF uses some function to select some nodes by considering some parameters; these parameters include the number of resources, link latency, processing power, response time, and so forth. In addition, HPF assigns some weights to the parameters above as well [28]. HPF utilizes the termination policy used in the iterative deepening method.

4.2.10 Routing indices search mechanism

The purpose of the Routing Indices (RI) mechanism is to reduce the network traffic and encircles three groups: *compound (CRI)*, *hop-count (HRI)*, and *exponential (ERI)*. In CRI, the number of documents of each neighbor is considered as a selection factor. However, it fails to consider the cost of searching in terms of the number of hops. In HRI, the number of hops to the destination is deemed as a selection factor leading to more storage consumption. ERI considers only the rank of the neighbor in selecting the candidate node, resulting in some imprecision. Overall, RI leads to a rise in response time and a fall in traffic compared to flooding [67].

4.2.11 Adaptive probabilistic search mechanism

Adaptive Probabilistic Search (APS) is an informed as well as probabilistic mechanism—forwarding group— wherein each node stores a table. The table contains information on received requests and the responding probability of each of the node's neighbors per that requested resource. In this regard, each query receiver node makes up its mind to send the query to the *k* best neighbors via the k-random walker method. In addition, APS updates the ranks of neighbors based on the received feedback. APS has shown better performance than both flooding and GUESS [50, 68].

4.2.12 MP-ISRL and ISRL search mechanism

MP-ISRL and ISRL mechanisms have higher performance than the random walk. ISRL searches for only one copy of the requested resource, and MP-ISRL searches for multiple ones. These mechanisms are the first ones exploiting reinforcement learning to learn about the best search path in terms of the minimum number of hops. These mechanisms send the query to the best neighbors via a k-random walk by probability P_q, and also they send the query to new paths by probability $(1-P_q)$ [69].

4.2.13 Gia search mechanism

The workload of any given node in the Gnutella network is increased when the number of received queries increase—culminating in a system that is not scalable. The Gia network has been proposed to solve the scalability problem of Gnutella. It improves scalability four or five times by exploiting biased random walk rather than random walk to direct search process to high-degree nodes. Gia considers node heterogeneity in terms of processing power, disk latency, and access bandwidth, similar to KaZaA. Moreover, Gia has a topology adaptation mechanism to compel nodes to attach to high-degree nodes in a short distance to ensure search success [70].

4.2.14 Dynamic multi-level feedback-based search mechanism

Dynamic Multi-Level Feedback-Based Search (DMFS) tries to enhance the overall search performance by using some mathematical formulas; these formulas calculate the ranks of neighbors, when it comes to the search selection process. Moreover, it takes into account the dynamic popularity of resources. The superiority of DMFS is in the introduction of some temporal parameters as compared to other informed search mechanisms in order to adapt to dynamic nature of P2P systems dynamically. Meanwhile, DMFS updates the ranks of neighbors consistently based on the most recent feedback [29].

4.2.15 Hybrid search mechanism

This hybrid method combines both non-forwarding and forwarding methods to expand the search scope to two hops under the supervision of query originator node; therefore, this mechanism improves performance and reduces search cost. However, this method is not suitable for large-scale networks; it is more appropriate for super-node systems [71].

4.2.16 Dominating set-based search mechanism

This mechanism tries to boost the success-rate by producing a Connected Dominating-Set (CDS) of nodes in which the search process is accomplished by using a random walk method. CDS is a network of high-degree nodes by which the rest of the network nodes can be accessible by only one hop; molding a minimum CDS in a graph is an NP-complete problem [72].

4.2.17 Location-aware topology search mechanism

The main issue regarding tremendous traffic of flooding has its root cause in incompatibility between physical and overlay networks, in which nodes are far-off from each other in reality; this gives rise to both latency and traffic. In this regard, the Location-Aware Topology search mechanism (LAT) tries to construct an improved overlay network topology based on the physical network to enhance search performance, and it considers the dynamicity of P2P systems as well [73].

4.2.18 Local minima search mechanism

The Local Minima Search Mechanism (LSM) uses DHT [74] to assign an ID to both nodes and resources; each node replicates its resources on their neighbors until hop = 2. As a result, LSM can only search a small surrounding to ensure the success of resource finding due to the lack of global knowledge [75]. The order of searching nodes with random walk method in a local neighborhood whose ID is closest to ID of the requested resource is higher than *O(logN)*.

4.2.19 Scalable query routing search mechanism

In the Scalable Query Routing search mechanism (SQR), each node stores information about the resources of their neighbors in a data structure called Exponentially Decaying Bloom Filter (EDBF); in this way, the routing information is compressed. SQR uses this information to orient the search process via a random walk method [75].

4.2.20 Flooding with random walk with neighbors table search mechanism

Some other new resource discovery mechanisms like Flooding with Random Walk with Neighbors Table (FRWNT) leverage a combination of the blind as well as the informed search mechanisms in resource searching [16]. However, FRWNT suffers from being negligent on updating information, the popularity of resources, and the distinct capacity of each node.

4.2.21 Dynamic popularity-aware search mechanism

The Dynamic Popularity-Aware search mechanism (DPAS), as an informed search mechanism, strives to improve search performance in comparison to its counterparts. To do this, it considers the dynamic popularity of resources, dynamic responsiveness status of nodes, heterogeneity of nodes, and dynamicity of P2P systems by devising some mathematical formulas. Hence, DPAS guides and controls the search process dynamically at each step [76].

Table 3. tabulates a comparison among flooding, random walk, and informed search groups in terms of eight performance metrics. As a matter of fact, any search mechanism should be deployed according to the application's requirements. Overall, informed search mechanisms overshadow the blind ones due to storing information as well as learning process; however, informed mechanisms incur overhead and costs in terms of storage and processing. In other words, in informed mechanisms, nodes have to store some indexes/information and also process some data before redirecting the search request or making any decision in this regard. Given these points, in real-world applications, an end user should make a trade-off among diverse performance metrics and use a proper search mechanism that meets his/her needs.

4.3 Group-based search mechanisms

In this group, nodes having similar resources form a group [77]. As a result, the query will be sent to the group whose resources are more analogous to the requested resource. Although in this mechanism both precision and quality of resources are guaranteed, in a network comprising of many diversified resources, and each node owning a few resources, this mechanism ends up in failure; this is because distinguishing similarity among nodes is difficult.

4.3.1 Interest-based shortcut search mechanism

This mechanism intends to resolve the scalability issue of Gnutella by allowing the nodes with similar interests to connect above the Gnutella network. Precisely speaking, interest-based locality means that if node x wants a specific resource responded by node y previously, it will be highly likely that node y has other resources that will be requested by node x in the future as well. This mechanism reduces Gnutella traffic by factor of three to seven [77].

4.3.2 Semantic overlay network search mechanism

This mechanism forms some groups, each of which consists of nodes with similar contents. Therefore, any given query by virtue of a central database is forwarded via the flooding

Table 3 Comparison among Flooding, Random Walk, and Informed Search Mechanisms

Performance Metrics Mechanism	Delay Time	Success-Rate	Traffic-Generation	Storage Cost	Precision	Load-Balancing	Restriction on Search Scope	Considering Resource Popularity
Flooding Mechanism	Good	Good	Bad	No	No	No	No	No
Random Walk	Bad	Bad	Good	No	No	Yes	No	No
Informed Mechanism	Medium	Medium	Medium	Yes	Yes	No	Yes	Yes

mechanism to the group entailing resources similar to the requested one [78].

4.3.3 SETS search mechanism

In this method, sites containing similar documents form a topic-segment, as documents relevant to a specific topic can respond to similar requests. SETS uses flooding for searching in each topic-segment [79].

4.3.4 GES search mechanism

GES is the first search mechanism that uses the concepts of information retrieval area, page ranking, and vector-space model. Thus, a vector represents each node's resource, and a node vector shows a brief representation of the node's contents. Accordingly, similar nodes based on their node-vectors form a semantic group, and a query is forwarded to a similar group by a biased-random walk. If that group contains the desired resources, the query is flooded in the group; otherwise, it is forwarded to the next candidate group via biased-random walk—GES reduces the number of nodes involved in query processing [80].

4.3.5 Comparison between GES, SETS, and random walk

In terms of search cost and success-rate, both SETS and GES are superior to random walk; however, GES due mainly to its central architecture is better than SETS.

4.3.6 Flaws of previously mentioned group-based search mechanisms

In group-based mechanisms—taking into consideration the similarity of nodes' contents to form logical groups—determining the similarity among nodes will be a difficult task. For example, when there are various types of resources in the network, and each node merely stores a few resources, identifying the similarity is challenging [34].

Other group-based mechanisms exploit locality-interest or the Sripanidkulchai method [77]; that is, if node *p* is capable of responding to query of node *q*, *p* probably owns other resources that will be requested by *q* in the future. Thus, a short path is established between *p* and *q*. These mechanisms also suffer from the previous issue—difficulty in identifying similarity among nodes' contents.

Some mechanisms use the benefits of semantic vectors to exhibit resources and exploit Euclidean distance to estimate the similarity between vectors. However, in these mechanisms, it is difficult to represent every property by numbers to calculate the Euclidean distance.

4.3.7 UIM search mechanism

The UIM search mechanism tries to use symbols to solve the issues mentioned above. UIM uses Log-Linear Conditional Probability, $\Pr(fj|fi)$ to find similarity between files; in fact, $\Pr(fj|fi)$ is the probability that user *p* has also file *fj*, given that *p* has already had file *fi*. Hence, using statistical patterns to describe resources is more flexible and efficient. In UIM, every node has a list of nodes that have previously responded to its requests successfully and to those the node can direct its future queries [81–84].

4.3.8 State-based search mechanism

In the State-Based Search mechanism (SBS), a query is sent to the most similar group by considering another vital factor—node's status. Node's condition is regarded since if the query receiver node is overloaded, it will fail to respond. SBS puts into practice both a fuzzy controller and the Grey theory [83, 84] to estimate a node's status in terms of its static and dynamic states ahead of time [42].

4.3.9 State-based versus other group-based search mechanisms

SBS enjoys both better response time and success-rate, in that it tries to avoid sending queries to the failed nodes, overloaded ones, or some having a long query queue.

4.3.10 Class-based search mechanism

The GES search mechanism—using the VSM model to present node's resources—is only efficient when nodes have even distribution of resources; however, if there is a high diversity in the types of node's resources, the resulting vector will be inaccurate. Given these points, the CSS search mechanism has been put forth to solve this imprecision by introducing class-vector rather than node-vector for each resource. Therefore, CSS clusters resources of a node in a granular-scale, and each cluster has a virtual node. Ultimately, all virtual nodes connect. Overall, CSS is more efficient in terms of search cost, response-time, and success-rate as compared to GES [85].

4.4 Hybrid search mechanisms

Although it is announced that most queries target popular resources in the Gnutella network, searching for rare resources should also be considered, which produce some traffic in the network. In light of this requirement, hybrid methods are introduced to combine advantages of both unstructured and structured systems to find both rare and popular resources with high success-rates [86–90]. If a resource is rare, it will be searched by DHT, and if it is popular, it will be found by flooding. Note that maintaining the structured network for finding rare resources is costly.

Hybrid search mechanisms are categorized into two groups based on the method they use to estimate popularity of resources:

- *SimplyHybrid* or *Detection-Based* group in which the search process starts by flooding. If an adequate number of resources is not found, it will estimate that the resource is rare, and try to find it by DHT. Finding rare resources has latency, and flooding at the first phase is resource consuming.
- The *Gossip-Based* group in which the search process uses the gossip technique to calculate the resource popularity. Then, the search mechanism can decide to exploit flooding or DHT for resource searching. This group is superior to the previous one in terms of performance. The gossip method is used to gain some global knowledge regarding both nodes and resources. It stands to reason that gossip-based techniques are of paramount importance to P2P systems due to their simplicity, scalability, and fault-tolerance.

4.4.1 SimplyHybrid search mechanism

This mechanism uses DHT for searching for rare resources and flooding for popular ones. Firstly, flooding with a small TTL occurs to find a requested resource, and if it is not successful, DHT will be used. Updating as well as maintaining the DHT structure are expensive, especially by considering the content publishing phase in DHT [86].

4.4.2 GAB search mechanism

The SimplyHybrid mechanism is not efficient due to its first-stage flooding to calculate the popularity of resources, causing network traffic. As a result, GAB uses gossiping to estimate the popularity of requested resources to apply either DHT or flooding. GAB employs the super-node architecture [87], produces less traffic and has lower response time compared to SimpleHybrid [88].

4.4.3 QRank search mechanism

Previous mechanisms misestimate resource popularity, as they assume that if a query receives many responses, it means that many nodes store that requested resource. However, it may be wrong, and maybe a few nodes store many similar resources. The rationale behind this proclamation is that in P2P systems, there is a locality interest concept. In other words, any given user is inclined to store identical resources in their storage. As a result, this user can respond to the forthcoming queries of a user downloading one of its resources. Overall, QRank tries to map some weights to keywords in a query string based on their frequencies. Hence, QRank can estimate the number of nodes receiving a query—giving rise to better performance [89].

4.4.4 PASH search mechanism

PASH is superior to the previous search mechanisms in the hybrid search group. This is because it tries to consider a concept, namely the dynamic popularity of any given resource to its calculation. Consequently, PASH enhances search performance in terms of lower response time, less traffic, and better success-rate. It is worth mentioning that the dynamicity of P2P systems in terms of insertion/deletion of nodes or their resources has a profound impact on resource popularity. Thus, PASH deems the dynamicity of P2P systems by using both the gossip method and some sensor nodes to estimate the dynamic popularity of resources [90]. Table 4. tabulates a comparison of the informed, group-based, and hybrid search groups. Although the group-based mechanisms in comparison to other search groups in Table 4. guarantee precision, they suffer from additional traffic due to formation of sub-groups and their update under churn. Furthermore, the hybrid search group tries to satisfy the success-rate for both popular and rare resources.

Table 4 Comparison of Informed, Group-Based, and Hybrid Search Groups

Performance Metrics Mechanism	Latency	Success-Rate	Traffic	Storage cost	Precision	Load-balancing	Partial Coverage	Considering Resource Popularity
Informed	Medium	Medium	Medium	Yes, but inefficient	Medium	No	Yes	Yes
Group-based	Good	Good	Very Bad	No	Good	No	Yes	Yes
Hybrid	Suitable for popular resources and Bad for rare ones	Very Good	Bad	Yes, and it is also efficacious	Bad	Yes	No	Yes

4.5 Bio-inspired meta-heuristic search mechanism

We have considered a fifth group—called bio-inspired meta-heuristic group. Researchers strive to involve organic and biological concepts—bio-inspired techniques—to deal with issues in computer science domains, including the search mechanism area [6, 91–95]; this search group is termed bio-inspired meta-heuristic search mechanisms.

In this regard, some search mechanisms [85, 91–94] have been inspired by the humoral immune system (IBS) [92], structure of fungi, or cellular slime mold life cycles (Dictyostelium Discoidem) (SMP2P) [91, 92]. However, this search group has some pitfalls too. For instance, the SMP2P class suffers from considering some rules/structures for peer position and overlay network to map slime mold bio-mechanism to P2P lookup operation; this brings about extra overhead.

Other mechanisms use the food-foraging behavior of either bees or ants—whose focuses are on using swarm intelligence to resolve complex problems [94, 95]. For example, food-forager ants—similar to query process—try to search for a desired resource by depositing chemical substances, namely pheromone as trials. The IACO search mechanism [6] utilizes inverted ant colony optimization to consider load-imbalance issues as compared to ACO [95]; this is done by diminishing the pheromone impact on the used paths. However, IACO fails to adapt well to frequently changing conditions of P2P systems. Furthermore, IACO does not consider the candidate node status before sending queries. These flaws lead to a situation in which the system suffers from additional network traffic, overloaded nodes, low success-rate, and long response time.

5 Conclusion

Resource discovery is of paramount importance in performance and productivity of any given distributed system, including Peer-to-Peer (P2P) systems. Therefore, it is challenging to come up with an efficient resource discovery mechanism. In this regard, we have expounded on fundamental concepts, underlying parameters, and attendant models in resource discovery issue in unstructured P2P systems. Any resource discovery mechanism should be selected according to the application's requirements and purposes; in fact, the application considers as well as assigns diverse weights for the search performance metrics whereby a suitable search mechanism will be identified. In this paper, we have also offered a novel taxonomy for classification of search mechanisms in the pure unstructured P2P system. Meanwhile, some conclusive evaluations between various search classes have been incorporated as well. This literature review can also help researchers to devise more efficient resource discovery mechanisms.

References

1. Amoretti M (2009) A survey of P2P overlay schemes: effectiveness, efficiency and security. Recent Patents on Computer Science 2(3):195–213
2. Tushar W, Saha T, Yuen C, Smith D, Poor H (2020) Peer-to-peer trading in electricity networks: an overview. IEEE Transactions on Smart Grid 11:3185–3200
3. Ashraf F, Naseer A, Iqbal S (2019) Comparative analysis of unstructured P2P file sharing network. In: ICISDM 2019: Proceedings of the 2019 3rd International Conference on Information System and Data Mining, Houston
4. Amoretti M, Zanichelli F (2018) P2P-PL: a pattern language to design efficient and robust P2P systems. P2P Networking and Applications 11(3):518–547
5. Shah N, Abid S, Qian D, Mehmood W (2017) A survey of P2P content sharing in MANETs. Comput Electr Eng 1(57):55–68
6. Zhang X, Hassanein H (2012) A survey of P2P live video streaming schemes–an algorithmic perspective. Comput Netw 56(15):3548–3579
7. Masood S, Shahid MA, Sharif M, Yasmin M (2018) Comparative analysis of P2P networks. International Journal of Advanced Networking and Applications 9(4):3477–3491
8. Shamshirband S, Soleimani H (2018) LAAPS: an efficient file-based search in unstructured P2P networks using reinforcement algorithm. Int J Comput Appl:1–8
9. Ogino N, Kitahara T (2017) An efficient content search method based on local link replacement in unstructured P2P networks. IEICE Trans Commun:1–11
10. Asghari S, Navimipour NJ (2019) Resource discovery in the P2P networks using an inverted ant Colony optimization algorithm. P2P Networking and Applications 12(1):129–142
11. Ed-daoui I, El Hami A, Itmi M, Hmina N, Mazri T (2018) Unstructured P2P systems: towards swift routing. Int J Eng Technol 7(2.3):33–36
12. Zarrin J, Aguiar RL, Barraca JP (2018) Resource discovery for distributed computing systems: a comprehensive survey. J Parallel Distr Com 1(113):127–166
13. Palmieri F (2017) Bayesian resource discovery in infrastructure-less networks. Inf Sci 376:95–109
14. Li Z (2017) A hybrid P2P framework for supply chain visibility. Doctoral dissertation
15. Jamal AA, Teahan WJ (2017) Alpha multipliers breadth-first search technique for resource discovery in unstructured P2P networks. Int J Adv Sci Eng Inf Technol 7(4):1403–1412
16. Bashmal L, Almulifi A, Kurdi H (2017) Hybrid resource discovery algorithms for unstructured P2P networks. Procedia Comput Sci 109(1):289–296
17. Risson J, Moors T (2006) Survey of research towards robust P2P networks: search methods. Comput Netw 50(17):3485–3521
18. Sarmady S (2010) A survey on P2P and DHT. arXiv: 1006.4708
19. Buford J, Yu H (2010) P2P Networking and applications: Synopsis and research directions. Handbook of P2P Networking, pp. 3–45
20. Androutsellis-Theotokis S, Spinellis D (2004) A survey of P2P content distribution technologies. ACM computing surveys (CSUR) 36(4):335–371
21. Lua E, Crowcroft J, Pias M, Sharma R, Lim S (2005) A survey and comparison of P2P overlay network schemes. IEEE Commun Surv Tutor 7(2):72–93
22. Suryanarayana G, Taylor RN (2004) A survey of trust management and resource discovery technologies in P2P applications. Citeseer
23. Androutsellis-Theotokis S, Spinellis D (2004) A survey of P2P content distribution technologies. ACM computing surveys (CSUR) 36(4):335–371
24. Pourqasem J (2018) Toward the optimization resource discovery Service in Grid Systems: a survey. Journal of Applied Research on Industrial Engineering 5(4):346–355
25. Meshkova E, Riihijärvi J, Petrova M, Mähönen P (2008) A survey on resource discovery mechanisms, P2P and service discovery frameworks. Comput Netw 52(11):2097–2128
26. Sakaryan G, Wulff M, Unger H (2004) Search methods in P2P networks: A survey. In: International Workshop on Innovative Internet Community Systems, Berlin
27. Thampi SM (2010) Survey of search and replication schemes in unstructured P2P Networks. arXiv Prepr. arXiv1008.1629
28. Prakash A (2006) A survey of advanced search in P2P networks. Department of Computer Science, Kent State University
29. Khatibi E, Mirtaheri SL, Khaneghah EM, and Sharifi M (2012) Dynamic multilevel feedback-based searching strategy in unstructured P2P systems. In: 2012 IEEE International Conference on Green Computing and Communications, Besancon
30. Prasad TRK, Jayakumar P, Sajeev GP (2018) A K-Clique based clustering protocol for resource discovery in P2P Networks, in 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Bangalore
31. Tsoumakos D, Roussopoulos N (2003) A Comparison of P2P Search Methods. In: WebDB, pp. 61–66
32. Fisk A (May 2003) Gnutella dynamic query protocol v0. 1. http://www9.limewire.com/developer/dynamic query.html
33. Castro M, Costa M, Rowstron A (2004) P2P overlays: structured, unstructured, or both?. Microsoft Research, Tech. Rep
34. Zeinalipour-Yazti D, Kalogeraki V, Gunopulos D (2005) Exploiting locality for scalable information retrieval in P2P networks. Inf Syst 30(4):277–298
35. Bawa M, Cooper B, Crespo A, Daswani N, Ganesan P, Garcia-Molina H, Kamvar S, Marti S, Schlosser M, Sun Q, Vinograd P (2003) P2P research at Stanford. ACM SIGMOD Rec 32(3):23–28
36. Chen Z, Liu J, Li J (2010) An adaptive expanding antbudget search algorithm for unstructured P2P networks. In: 2010 2nd International Conference on Future Computer and Communication, Wuha
37. Zhiwei S, Shaowu M, Jianan W, Xiongyan T (2009) Analyzing the technologies of search algorithm based on P2P. In: 2009 2nd IEEE International Conference on Broadband Network & Multimedia Technology, Beijing, 2009
38. Al-Aaridhi R, Dlikman I, Masinde N, Graffi K (2018) Search Algorithms for distributed data structures. In: P2P Networks, in 2018 International Symposium on Networks, Computers and Communications (ISNCC), Rome
39. Bosunia MR, Jeong S-H (2019) Machine-to-machine content retrieval in wireless networks. Wirel Pers Commun 107(3):1465–1490
40. Boulfekhar S, Benmohammed M (2013) A novel energy efficient and lifetime maximization routing protocol in wireless sensor networks. Wirel Pers Commun 72(2):1333–1349
41. Bisnik N, Abouzeid A (2005) Modeling and Analysis of Random Walk Search Algorithms in P2P Networks. Proceedings - Second International Workshop on Hot Topics in P2P Systems, HOT-P2P 2005, San Diego,
42. Wu K, Wu C (2013) State-based search strategy in unstructured P2P. Futur Gener Comput Syst 29(1):381–386
43. Jiang S, Guo L, Zhang X, Wang H (2008) Lightflood: minimizing redundant messages and maximizing scope of P2P search. IEEE Transactions on Parallel and Distributed Systems 19(5):601–614
44. Fletcher GHL, Sheth HA, Börner K (2004) Unstructured P2P networks: Topological properties and search performance. In: International Workshop on Agents and P2P Computing, Berlin
45. Dorrigiv R, Lopez-Ortiz A, Pralat P (2007) Search algorithms for unstructured P2P Networks. In: 32nd IEEE Conference on Local Computer Networks (LCN 2007), Dublin

46. Yang B, Garcia-Molina H (2002) Improving search in P2P networks. In: Proceedings 22nd International Conference on Distributed Computing Systems, Vienna
47. Li X, Wu J (2006) Searching techniques in P2P networks. In: Handbook of Theoretical and Algorithmic Aspects of Ad Hoc, Sensor, and P2P Networks, pp. 613–642
48. Singla A, Rohrs C (2002) Ultrapeers: another step towards Gnutella scalability. http://rfc-gnutella.sourceforge.net/Proposals/Ultrapeer
49. Yang B, Garcia-Molina H (2002) Efficient search in P2P networks. In: Proceedings of the International Conference on Distributed Computing Systems (ICDCS), New York
50. Tsoumakos D, Roussopoulos N (2006) Analysis and comparison of P2P search methods. In: Proceedings of the 1st International Conference on Scalable information systems, New York
51. Gkantsidis C, Mihail M, Saberi A (2006) Random walks in P2P networks: algorithms and evaluation. Perform Eval 63(3):241–263
52. Lv Q, Cao P, Cohen E, Li K, and Shenker S (2002) Search and Replication in Unstructured P2P Networks. In: Proceedings of the 16th international conference on Supercomputing, New York
53. Leu J-S, Tsai C-W, Lin W-H (2011) Resource searching in an unstructured P2P network based on cloning random Walker assisted by dominating set. Comput Netw 55(3):722–733
54. Jiang H, Jin S (2005) Exploiting dynamic querying like flooding techniques in unstructured P2P networks. In: 13TH IEEE International Conference on Network Protocols (ICNP'05), Boston
55. Tian C, Jiang H, Liu X, Liu W, Wang Y (2008) Towards minimum traffic cost and minimum response latency: a novel dynamic query protocol in unstructured P2P networks. In: 2008 37th International Conference on Parallel Processing, Portland
56. Jin S, Jiang H (2007) Novel approaches to efficient flooding search in P2P networks. Comput Netw 51(10):2818–2832
57. Daswani A, Fisk S (n.d.) Gnutella UDP Extension for Scalable Searches (GUESS) v0.1 (0). [Online]. Available: http://130.203.136.95/showciting;jsessionid=C91B0BE31D86A6763AEAB81A4695B27B?cid=929229
58. Yang B, Vinograd P, Garcia-Molina H (2004) Evaluating GUESS and non-forwarding P2P search. In: 24th International Conference on Distributed Computing Systems, Tokyo
59. Daswani N, Garcia-Molina H (2004) Pong-cache Poisoning in GUESS. In: Proceedings of the 11th ACM Conference on Computer and Communications Security, Washington DC, 2004
60. Zhuge H, Chen X, Sun X (2005) Preferential walk: towards efficient and scalable search in unstructured P2P networks. In: Special interest tracks and posters of the 14th international conference on World Wide Web, New York
61. Ramaswamy L, Liu L (2003) Free riding: a new challenge to P2P file sharing systems. In: 36th Annual Hawaii International Conference on System Sciences, Hawaii
62. Yang K-H, Wu C-J, Ho J-M (2006) Antsearch: an ant search algorithm in unstructured P2P networks. IEICE Trans Commun 89(9): 2300–2308
63. Kalogeraki V, Gunopulos D, Zeinalipour-Yazti D (2002) A local search mechanism for P2P networks. In: Proceedings of the Eleventh International Conference on Information and Knowledge Management, New York
64. Zeinalipour-Yazti D, Kalogeraki V, Gunopulos D (2004) Information retrieval techniques for P2P networks. In: Comput Sci Eng 6(4):20–26
65. Yuan F, Liu J, Yin C (2007) A scalable search algorithm on unstructured p2p networks. In: Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD 2007), Qingdao
66. Wang C, Xiao L (2007) An effective P2P search scheme to exploit file sharing heterogeneity. IEEE Transactions on Parallel and Distributed Systems 18(2):145–157
67. Crespo A, Garcia-Molina H (2002) Routing indices for P2P systems. In: Proceedings 22nd International Conference on Distributed Computing Systems, Vienna
68. Tsoumakos D, Rossopoulos N (2003) Probabilistic knowledge discovery and management for P2P networks. P2P Journal 12(2):129–136
69. Li X and Wu J (2006) Improve searching by reinforcement learning in unstructured P2Ps. In: 26th IEEE International Conference on Distributed Computing Systems Workshops (ICDCSW'06), Lisboa
70. Chawathe Y, Ratnasamy S, Breslau L, Lanham N, Shenker S (2003) Making gnutella-like P2P systems scalable. In: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications, Karlsruhe
71. Li X, Wu J (2007) A hybrid searching scheme in unstructured P2P networks. The International Journal of Parallel, Emergent and Distributed Systems 22(1):15–38
72. Yang C, Wu J (2003) Dominating-Set-based Searching in P2P Networks. In: International Conference on Grid and Cooperative Computing, Berlin
73. Liu Y, Liu X, Xiao L, Ni LM, Zhang X (2004) Location-aware Topology Matching in P2P Systems. In: IEEE INFOCOM 2004, vol. 4, pp. 2220–2230
74. Kamel MBM, Crispo B, Ligeti P (2019) A decentralized and scalable model for resource discovery in IoT network. In: 2019 International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Barcelona
75. Morselli R, Bhattacharjee B, Marsh MA, Srinivasan A (2007) Efficient lookup on unstructured topologies. IEEE Journal on Selected Areas in Communications 25(1):62–72
76. Khatibi E, Sharifi M, Mirtaheri SL (2019) DPAS: a dynamic popularity-aware search mechanism for unstructured P2P systems. P2P Networking and Applications 13(3):1–25
77. Sripanidkulchai K, Maggs B, Zhang H (2003) Efficient content location using interest-based locality in P2P systems. In: IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No. 03CH37428), San Francisco
78. Crespo A, Garcia-Molina H (2004) Semantic overlay networks for P2P Systems. In: International Workshop on Agents and P2P Computing, Berlin
79. Bawa M, Manku GS, Raghavan P (2003) SETS: Search enhanced by topic segmentation. In: Proceedings of the 26th Annual International ACM SIGIR conference on Research and Development in Informaion Retrieval, Toronto
80. Zhu Y, Hu Y (2006) Enhancing search performance on Gnutella-like P2P systems. IEEE Transactions on Parallel and Distributed Systems 17(12):1482–1495
81. Lafferty J, McCallum A, Pereira FCN (2001) Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Proceedings of the 18th International Conference on Machine Learning 2001 (ICML 2001), San Francisco
82. Chen G, Low CP, Yang Z (2008) Enhancing search performance in unstructured P2P networks bBased on users' common interest. IEEE Transactions on Parallel and Distributed Systems 19(6): 821–836
83. Kayacan E, Ulutas B, Kaynak O (2010) Grey system theory-based models in time series prediction. Expert Syst Appl 37(2):1784–1789
84. Lin Y, Liu S (2004) A historical introduction to grey systems theory. In: 2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583), The Hague
85. Huang J, Li X, Wu J (2007) A class-based search system in unstructured P2P Networks. In: 21st International Conference on

Advanced Information Networking and Applications (AINA’07), Niagara Falls, ON

86. Loo BT, Huebsch R, Stoica I, Hellerstein JM (2004) The case for a hybrid P2P search infrastructure. In: International workshop on P2P Systems, Berlin
87. Tun W, Pourqasem J, Edalatpanah SA (2020) Optimizing resource discovery technique in the P2P grid systems. Wireless Communications and Mobile Computing, vol 2020
88. Zaharia M, Keshav S (2008) Gossip-based search selection in hybrid peer-to-peer networks. Concurrency and Computation: Practice and Experience 20(2):139–153
89. Chen H, Jin H, Liu Y, Ni LM (2008) Difficulty-aware hybrid search in P2P networks. IEEE Transactions on Parallel and Distributed Systems 20(1):71–82
90. Shi X, Han J, Liu Y, Ni LM (2009) Popularity adaptive search in hybrid P2P systems. J Parallel Distrib Comput 69(2):125–134
91. Šešum-Čavić V, Kühn E, Kanev D (2016) Bio-inspired search algorithms for unstructured P2P overlay networks. Swarm and Evolutionary Computation 29:73–93
92. Šešum-Čavić V, Kuehn E, Zischka S (2018) Swarm-inspired routing algorithms for unstructured P2P networks. International Journal of Swarm Intelligence Research (IJSIR) 9(3):23–63
93. Guan Z, Cao Y, Hou X, Zhu D (2007) A Novel efficient search algorithm in unstructured P2P networks. In: Second Workshop on Digital Media and its Application in Museum & Heritages (DMAMH 2007), Chongqing
94. Krynicki K, Jaén Martínez FJ, Mocholí Agües JA (2014) Ant Colony optimisation for resource searching in dynamic P2P grids. International Journal of Bio-Inspired Computation 6(3):153–165
95. Krynicki K, Jaen J, Mocholi JA (2013) On the performance of ACO-based methods in P2P resource discovery. Appl Soft Comput 13(12):4813–4831

Elahe Khatibi She has received her M.Sc. degree in Computer Software Engineering from the School of Computer Engineering of Iran University of Science and Technology; her research interests are in the areas of resource discovery, and peer-to-peer systems.

Mohsen Sharifi is a Full-Professor of System Software Engineering in the School of Computer Engineering of Iran University of Science and Technology. He directs a distributed system software research group and laboratory. His main research interest is in the development of distributed systems, solutions, and applications, particularly for use in various fields of science. The development of a true distributed operating system is on top of his wish list. He received his B.Sc., M.Sc. and Ph.D. in Computer Science from the Victoria University of Manchester in the United Kingdom in 1982, 1986, and 1990, respectively.