1. Distributed hash table

Distributed hash tables (**DHTs**) are a class of decentralized distributed systems that provide a lookup service similar to a hash table: (*key*, *value*) pairs are stored in the DHT, and any participating node can efficiently retrieve the value associated with a given key. Responsibility for maintaining the mapping from keys to values is distributed among the nodes, in such a way that a change in the set of participants causes a minimal amount of disruption. This allows DHTs to scale to extremely large numbers of nodes and to handle continual node arrivals, departures, and failures.

DHTs form an infrastructure that can be used to build more complex services, such as distributed file systems, **peer-to-peer** file sharing and content distribution systems, cooperative web caching, multicast, any-cast, domain name services, and instant messaging.

2. P2P overlay

A peer-to-peer overlay is a distributed collection of autonomous end-system computing devices called peers that form a set of interconnections called an overlay to share resources of the peers such that peers have symmetric roles in the overlay for both message routing and resource sharing. The P2P overlays have several inherent characteristics that we treat as the principles of the P2P paradigm: self-organization, role symmetry, resource sharing, scalability, peer autonomy, and resiliency.

The overlay geometry refers to the static model of the graph that the routing and maintenance algorithm constructs in the absence of churn. Such models have been widely studied in graph theory, along with selected overlay algorithms Chord, CAN, and Pastry, also DHT types.

Tradeoff between Routing State and Path Distance If each peer had complete information about all other peers in the overlay, each P2P message would take at most one hop. However, all peers would need to maintain an O(N) routing table size, and each join and leave event would need to propagate to all other peers in the overlay, creating a large maintenance load. This maintenance load would grow with both the size of the overlay, N, and the churn rate. On the other hand, if a node knows only the link to its successor on

a ring, routing state and maintenance load would be nominal but routing performance would be O(N).

3. P2P overlay types

- Unstructured (Bittorrent, Gnutella ..): do not provide any algorithm for organization or optimization of network connections. In particular, three models of unstructured architecture are defined. In *pure peer-to-peer* systems the entire network consists solely of equipotent peers. There is only one routing layer, as there are no preferred nodes with any special infrastructure function.
- Structured (Chord, CAN, Tapestry and Pastry): in structured peer-to-peer networks, connections in the overlay are fixed. They typically use distributed hash table-based (DHT) indexing, such as in the Chord system (MIT).

A second category of overlays, called structured overlays, emerged to address limitations of unstructured overlays by combining a specific geometrical structure with appropriate routing and maintenance mechanisms. Here we focus on geometry and routing; the next chapter is devoted to overlay maintenance. A large number of multi-hop designs have been studied, which we organize into prefix routing, ring with logarithmic degree mesh, and constant degree, such as butterfly, cubed-connected cycles

The earliest peer-to-peer systems used unstructured overlays that were easy to implement but had inefficient routing and an inability to locate rare objects. These problems spawned many designs for overlays with routing mechanisms that are deterministic and that can provide guarantees on the ability to locate any object stored in the overlay. The large majority of these designs used overlays with a specific routing geometry and are called structured overlays. At the same time, many unstructured overlays have incorporated some degree of routing structure, such as clustering, near/far links, and semantic links to improve search efficiency. In addition to the structured and unstructured overlay categories, there are hierarchical models.

Another difference between structured and unstructured overlays is that the former support key-based routing such that object identifiers are mapped to the peer identifier address space and an object request is routed to the nearest peer in the peer address space. Peer-to-peer systems using key-based routing are frequently called distributed object location and routing (DOLR) systems. A specific type of DOLR is a distributed hash table (DHT) in which the identifiers are computed using a consistent hash function78 and each peer is responsible for a range of the hash table. Within the category of structured overlays are several dimensions for distinguishing the many designs. These dimensions include:

- Maximum number of hops taken by a request given an overlay of N peers. The primary categories are multi-hop, one-hop, and variable-hop.
- Organization of the peer address space. Typically the identifier space is large and peer address assignments are uniformly distributed. The primary categories of address space structure are flat and hierarchical.
- Next-hop decision criteria. For routing to converge to the correct destination, the distributed routing algorithm needs to ensure that the distance between the route progression and the endpoint is narrowing at each step. The criteria is often referred to as a distance metric, and computation of the distance metric in a routing context is done by a distance function. Distance metrics that converge include prefix matching, XOR metric, Euclidean distance in a d-dimensioned space, linear distance in a ring, and modulo bit shifting in de Bruijn graphs.
- Geometry of the overlay. Important graph properties for search such as the small-world model and power-law graphs were discussed. Another important characteristic is how the node degree changes as the size of the overlay grows. This in turn reflects the growth of the routing table and maintenance traffic. Two important categories are logarithmic degree graphs and constant degree graphs such as butterflies and de Bruijn graphs.
- Overlay maintenance. Overlay membership changes due to peers joining and leaving the overlay require that routing tables be updated. Strategies for overlay maintenance include active versus correct on use.
- Locality and topology awareness. To improve overlay performance, many designs determine peer neighbor relationships according to the proximity and connectivity of the peers in the underlay.

Table 4.1 Structured Overlays Discussed in This Chapter

Type	Systems	Applications	Implementations
Logarithmic mesh with prefix routing	PRR, S-PRR		
	Tapestry	Oceanstore, Bayeaux	Chimera, OverlayWeaver
	Pastry	PAST, Scribe	FreePastry,
	P-Grid	UniStore, PIX-Grid	OverlayWeaver
	Bamboo		P-Grid
	Z-Ring		OpenDHT
Ring with embedded logarithmic degree mesh	Chord	DHash, CFS	Chord, OverlayWeaver
	DKS		DKS
	Chord#		
Constant degree	Ulysses		
	Koorde		OverlayWeaver
	Cycloid		
Logarithmic degree with specialized distance metrics	CAN		
	Kademlia		KAD, OverlayWeaver
O(1)-hop	Kelips		
	OneHop		
	EpiChord		
	D1HT		

4. Geometry and Routing

Geometry defines the idealized static graph model for interconnecting peers. It constrains the available paths for sending a message between two peers, whereas a routing algorithm selects among these paths, depending on the distance metric. During the design of an overlay, the selection of the overlay geometry should consider both static and dynamic conditions. Dynamic peer membership could cause a specific geometry to be unstable or expensive and complicated to maintain. Further, the important properties of the geometry that are realized when the overlay address space is completely populated might be unpredictably altered when the address space is sparsely populated, which would be the usual case in practice. Other geometry-related design goals to consider include reducing the maintenance bandwidth and lowering worst-case and average-case routing delay. For a given geometry there may be multiple possible routing algorithms. The routing state used by an overlay routing algorithm is referred to as the routing table, but the actual organization of the routing state is algorithm dependent. Organization of the routing table for fast selection of the next-hop peers is an important aspect of the algorithm performance. The routing behavior of a peer can be abstracted as a message-forwarding procedure as defined by the following NextHop function. The first step is to find candidate peers for forwarding the message. A special case is if the current peer is the intended destination of the message. Otherwise, there may be several choices, ordered according to their distance from the destination peer, using the distance metric and the current values in the routing table RT. Other elements not shown in this simplified NextHop function include whether the algorithm uses parallel lookups, uses intermediate responses to update the routing table state, or gathers overlay metrics.

- **NextHop(dest)**
    - **if(dest == this_peer) return this_peer**
    - **peer_choices = closest(dest,RT)**
- **return peer_choices**

5. LOGARITHMIC DEGREE WITH PREFIX ROUTING PRR

Prefix routing is used in many structured overlays, including **Tapestry, Pastry,** P-Grid, Cycloid, and Z-Grid. Plaxton, Rajaraman, and Richa (PRR)79 presented the first algorithm for a peer-to-peer object location and routing system in a 1997 paper.
By mapping object identifiers to the address space of peers, PRR enables key-based routing and is able to support read, insert, and delete operations on objects stored in the overlay. This principle is the basis for subsequent DHT designs.
PRR uses suffix-based routing, which is a symmetric case of prefix routing.

Suffix and prefix-based routing match increasing portions of the destination address at each hop along the path until the destination is reached. For example, if the target address is the hexadecimal address 3A9F1, in suffix-based routing matching 4 bits at a time, successive hops along the path match xxxx1, xxxF1, xx9F1, xA9F1, and 3A9F1, where x is a wildcard. Tapestry uses a variation of PRR routing; other overlays, including Pastry and P-Grid, use prefix-based routing. Tapestry, Pastry, and P-Grid are described later in this section. The PRR algorithm doesn't deal with overlay formation or maintenance issues. In PRR, peers are connected in a static overlay mesh network and each peer has a routing table to route messages. The routing table is organized in a fixed number of levels and within each level a fixed number of entries. The number of levels corresponds to the number of suffix match steps performed to route an address, and the number of entries corresponds to the number of digits in an address. So if addresses are 24 bits in length, and each hop matches the next 4 bits of the address, each peer needs 24/4 = 6 routing table levels and $2^4 = 16$ entries at each level. Level 1 matches the suffix position S as in xxxxxS, level 2 matches the suffix SS as in xxxxSS, and so forth. At each level there are 16 possible digit choices to match; hence the 16 entries at each level. In general, suppose that identifiers are represented as a sequence of digits in some base b, with the number of digits equal to $B = 2^b$, and the number of nodes N in the overlay is a power of B. Then suffixes are matched b bits at a time at each hop, and the routing table at each peer has $(\log_2 N)/b$ levels and B entries per level. In addition, in the PRR design, each entry has a cost value for routing between the peer and the corresponding destination. The lowest-cost neighbor at any (level, i) position is the primary neighbor for that entry. If there are other neighbors that meet low-cost criteria for a given (level, i) position, these are stored in the routing table as secondary links. If a node w is a primary neighbor of node x at entry (level, i), node w has a reverse link to x at the (i, j) position in its routing table.

6. RING WITH EMBEDDED LOGARITHMIC DEGREE MESH

Chord organizes peers on a logical ring, and peers maintain neighbor pointers spaced at logarithmic intervals around the ring. In addition, each peer has a link to its predecessor and successor peers on the ring. The Chord routing table is called a finger table. In addition to the finger table, each Chord peer maintains links to its successor in the address space. Chord uses consistent hashing to map keys to nodes. It can be shown that if the hash function has a random distribution, there is a high probability that K keys will be distributed across N nodes such that each node is responsible for, at most, (1+ e)K/N keys. In addition, also with high probability, when a node joins or leaves an overlay of size N, only O(K/N) keys move to or from the joining or leaving node. These characteristics are important for the load distribution and efficiency of the overlay under churn. Chord's routing function uses its successor ring in the last hop and uses the finger table to maximize the size of the step toward the destination, as shown in the following NextHop function:

- **NextHop(dest)**
  // if dest is in range of this peer and its successor, then
  // the successor is responsible
  - **if (dest 2 (this_peer,successor]) return successor**

  // otherwise, search the finger table for
  // highest predecessor of dest
  - **for i = log2 N downto 1**
    - **if (finger[i] 2 (this_peer,dest)) return finger[i]**
  - **return this_peer**

7. Constant-degree structured multihop

Content Addressable Network, or CAN is a constant-degree structured multi-hop DHT that organizes peers in a d-dimensional Cartesian coordinate system.
Like the other systems we discuss, CAN peers and objects have identifiers from the same virtual address space. Each peer's position in the d-dimensional space and the boundaries it shares with other peers determine the extent of the zone of the space for which the peer is responsible. Peers in CAN maintain information about neighboring peers that abut them in each of the d dimensions. When a message is being sent, a neighboring peer that is the closest to the target is selected. Assuming that the space is divided into equal zones, the average routing path length is $(d/4)(n^{1/d})$, where d is the number of dimensions and N is the number of peers in the overlay. To join the overlay, a new peer performs three steps. First, it locates some peer already in the CAN. Second, it randomly selects a peer whose zone will be split to accommodate the new peer, and it sends a join request to it via the first peer. Third, split the existing zone and notify the neighbors of the split zone so that the routing decisions include the zone changes. In a d-dimension space $[d1, d2,. . ., d_n]$, zones are split in a fixed order of the dimensions. Merging of zones when a node subsequently leaves the CAN is done in reverse order. When a zone is split, the original peer retains those key-value pairs that fall within its new subdivided zone. The remaining key-value peers go to the joining peer. Similarly, the joining peer inherits the neighbors of the original peer that abut the edges of the zone for which the new peer is responsible. Both the new peer and the original peer become neighbors, and the original peer prunes its neighbor list and notifies its neighbors accordingly. Neighboring peers exchange heartbeat messages to verify that they are still connected to the overlay. If missing heartbeat messages indicate that the peer is no longer available, the neighbors of the leaving peer need to coordinate to determine which peer will assume ownership of the zone held by that peer. A message exchange is conducted so that the neighbor with the smallest zone assumes ownership of the orphaned zone.