THIRD QUARTER 2003, VOLUME 5, NO. 1

IEEE COMMUNICATIONS SURVEYS
The Electronic Magazine of Original Peer-Reviewed Survey Articles

www.comsoc.org/pubs/surveys

CONSTRAINT-BASED ROUTING IN THE INTERNET: BASIC PRINCIPLES AND RECENT RESEARCH

OSSAMA YOUNIS AND SONIA FAHMY, PURDUE UNIVERSITY

ABSTRACT

Novel routing paradigms based on policies, quality of service (QoS) requirements, and packet content have been proposed for the Internet over the last decade. Constraint-based routing algorithms select a routing path satisfying constraints that are either administrative-oriented (policy routing) or service-oriented (QoS routing). The routes, in addition to satisfying constraints, are selected to reduce costs, balance network load, or increase security. In this article, we discuss several constraint-based routing approaches and explain their requirements, complexity, and recent research proposals. In addition, we illustrate how these approaches can be integrated with Internet label switching and QoS architectures. We also discuss examples of application-level routing techniques used in today's Internet.

With the proliferation of Web and multimedia services, and virtual private networks (VPNs) connecting corporate sites, more-versatile Internet routing protocols have become critical. Current Internet packet forwarding is based on the destination address. Simple routing algorithms that determine forwarding paths based on the minimum number of hops or delay to a specified destination are no longer sufficient. The need to support diverse traffic types and applications with quality demands (Fig. 1) imposes new requirements on routing in the (now commercial) Internet. New routing paradigms are also required to handle service agreements among service providers and users.

Administrative policies, performance requirements, load balancing, and scalability are thus becoming increasingly significant factors in Internet routing. Intelligent path selection based on *multiple constraints* or on *packet content* takes these factors into consideration. Constraint-based routing (CBR) denotes a class of routing algorithms that base path selection decisions on a set of requirements or constraints, in addition to the destination. These constraints may be imposed by administrative policies, or by Quality of Service (QoS) requirements. Constraints imposed by policies are referred to as policy constraints, and the associated routing is referred to as *policy routing* (or policy-based routing). Constraints imposed by QoS requirements, such as bandwidth, delay, or loss, are referred to as QoS constraints, and the associated routing is referred to as *QoS routing* [1].

CBR reduces the manual configuration and intervention required for realizing traffic engineering objectives [2]. The ultimate objective of CBR is to enable a new routing paradigm with special properties, such as being resource reservation-aware and demand-driven, to be merged with current routing protocols. The resource availability information required to make CBR decisions is exchanged via routing protocol extensions. Signaling protocols (such as RSVP) can set up the required state along the computed paths. Finally, explicit routing on the computed path can be performed via switching technologies, such as Asynchronous Transfer Mode (ATM) or Multi-protocol Label Switching (MPLS) [2, 3], or using paths stored in packet headers. CBR typically considers flow aggregates (also known as macro-flows or trunks), rather than individual micro-flows (e.g., a single HTTP (hypertext transfer protocol) connection). Some of the QoS routing approaches discussed in this article, however, were proposed for micro-flows, while others consider aggregate flows.

CBR protocols can be viewed as follows. Consider a flow between a source and a destination. Using network connectivity and resource availability as inputs, a number of routes are viable. If certain constraints are imposed, then some (or all) of these routes may not remain viable. In CBR, two overlapping sets of routes are determined: policy routes and QoS routes. This overlap is due to the fact that some routes may conform to the underlying routing policies and also satisfy QoS requirements.

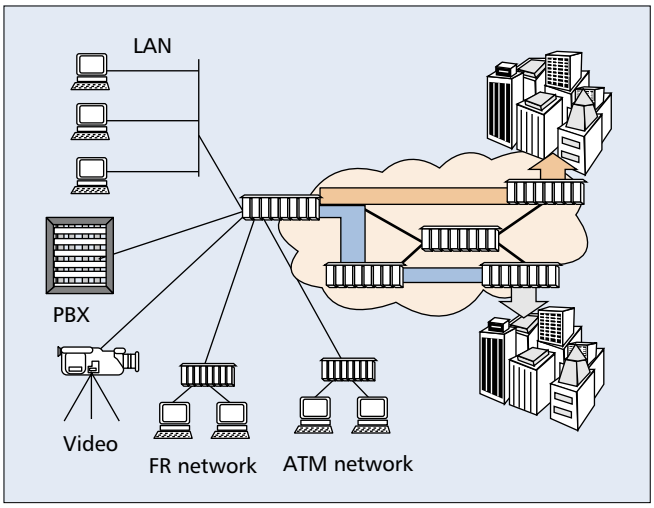

FIGURE 1. *Networks with diverse traffic and diverse requirements.*

Policy routing selects paths that conform to administrative rules and service-level agreements (SLAs). With policy routing, administrators can base routing decisions not only on the destination location, but also on factors such as applications and protocols used, size of packets, or identity of both source and destination end systems. In contrast, QoS routing attempts to satisfy multiple QoS requirements, such as bandwidth and delay bounds. Recent work emphasizes the need to identify paths that not only satisfy QoS requirements, but also consider the global utilization of network resources [4]. Tractability is the primary challenge with QoS routing. Most proposed techniques use simple heuristics to avoid intractability. Stability and scalability are also important concerns. QoS routing performance is sensitive to the accuracy of used information, the network topology, and the network traffic characteristics.

With the evolution of Web caching, *content routing* (or content-based routing) has recently gained significant attention. Content routing operates at the application level (proxy server level), not at the router level. In a Web caching system, proxy servers containing replicated information are placed "between" a Web server and clients. Routing in this case is based upon the content of a given request, e.g., an HTTP URL (uniform resource locator). Content routing balances load among proxy servers and distributes traffic to avoid network bottlenecks. In addition, requests may be directed to the nearest server based upon factors such as measured response times, bandwidth, or network topology. We briefly describe content routing and some related protocols in this article, so as to paint a complete picture of ongoing routing research at different layers of the protocol stack. The primary focus of this article, however, is on CBR (constraint-based routing).

The remainder of this article is organized as follows. We classify routing techniques. We discuss the primary goals and requirements of constraint-based routing (CBR). We examine policy routing in more depth. We discuss QoS routing operation and optimizations, and compare a set of QoS routing algorithms. We examine stability, robustness, and scalability challenges, and some proposed solutions. Finally, we briefly discuss higher-level routing, such as content routing, and give some examples from recent work in this area. We conclude with a brief discussion of future directions.

ROUTING TECHNIQUES

Internet routing can be classified as either intra-autonomous-system (intra-AS) routing (or simply intra-domain routing), or inter-domain routing. Table 1 summarizes the features and challenges of both routing types [1, 5, 6].

As previously discussed, constraint-based routing can be viewed as a generalization of today's single-constraint routing (where the constraint is the destination address). Before we address the particulars of constraint-based routing, we classify routing strategies according to the mechanisms for triggering a search for feasible paths (satisfying constraints), and the amount of state maintained [5, 7]. Mechanisms for triggering a search for feasible paths can be categorized into:

Pro-Active (Pre-Computation) Routing: In this approach, routes to various destinations are maintained at all times, whether they are required or not. Pre-computation approaches (also referred to as path caching approaches) are highly responsive, since the overall average path setup time is significantly reduced [4, 8]. Pre-computation approaches, however,

	Intra-domain routing	Inter-domain routing
Definition	Routing within an AS (protocols known as Interior Gateway Protocols (IGPs)).	Routing across ASs (protocols known as Border Gateway Protocols (BGPs)).
Protocols	Routing Information Protocol (RIP), Open Shortest Path First (OSPF), Intermediate System-Intermediate System (IS-IS).	Border Gateway Protocol (BGP).
Policy	Since routers and hosts are all under the same administrative control, policies are easy to enforce.	Policy is an important concern since crossing AS boundaries imposes restrictions.
Scalability	Depends on the number of nodes and the number edges in the AS. Scalability can be achieved by splitting an AS (i.e., imposing hierarchy).	Depends on the number of exchanged routes, and policies across ASs. Scalability is crucial since the number of Internet ASs is large.
Primary challenges	• Routing a flow along a path that can satisfy constraints or indicating that the flow cannot be admitted. • Indicating disruptions to the current route of a flow. • Accommodating best-effort flows without any resource reservation. • Scaling for large numbers of flows and nodes. • Achieving stability and fast convergence.	• Determining reachability to various destinations. • Providing loop-free routes. • Supporting aggregation. • Determining multiple paths to a given destination (optional multi-path routing). • Storing and processing large numbers of routes and policies. • Expressing, coordinating, and exchanging route policies. • Achieving stability and fast convergence.

Table 1. *Intra-domain and inter-domain routing.*

CBR type	Description	Advantages	Disadvantages
Off-line	Path computation is performed outside the network, for a known traffic demand matrix and network topology	Optimizes network resource usage and planning.	Exhibits high computational complexity and cannot adapt to network changes once a path has been chosen.
Online	Path selection is performed at network elements (routers and hosts), without knowledge of the new requested traffic a priori.	Can adapt to network changes and state updates.	Suffers from strict operational requirements (e.g., computational complexity and convergence time).

Table 2. *CBR process types.*

incur high processing and storage overhead. This is further complicated by the need for frequent route re-computation. To increase the probability of cache hits (for requested paths), multiple alternative paths can be computed and stored for each destination, according to each expected request (e.g., delay or bandwidth requirement). Checking multiple paths simultaneously and providing backups in case of path failure is referred to as *multi-path routing* [9, 10]. The problem with storing multiple paths is that routing tables grow dramatically. This necessitates using efficient mechanisms for storage and retrieval [11, 12].

Reactive (On-Demand) Routing: In this approach, routes to destinations are computed when they are needed. This approach reduces overhead at the expense of slower response times. Examples of this approach include flooding protocols, most of the QoS routing protocols discussed later, and Ad-hoc On-demand Distance Vector (AODV) routing used in mobile ad-hoc networks.

We can also classify all routing techniques according to the amount of global state maintained as follows:

Distance Vector Routing (sometimes referred to as hop-by-hop routing): In this approach, path computation is distributed among the nodes in the network. Each node periodically exchanges distance vector information (distance and *next hop* from itself to *all* destinations) with its *neighbors*. Each node uses distance vector information to compute the paths (e.g., using the Bellman-Ford algorithm). The main problem with this approach is the lack of global knowledge, leading to problems such as slow convergence and routing loops. RIP is an example distance vector protocol. BGP inter-domain routing uses path vectors instead of distance vectors. A path vector generalizes a distance vector (distances and next hops) by specifying the *path* (sequence of autonomous systems) to each destination. Path vectors are included in BGP update advertisements.

Link State Routing: In this approach, the state of all *local* links is periodically broadcast to *all* network nodes. Based on this state, the required feasible path is *locally* determined (e.g., using Dijkstra's algorithm). Link state routing has the advantages of simplicity, accuracy, and avoidance of loops, but it suffers from three problems:

- High storage overhead.
- High path computation overhead.
- High link-state update overhead.

OSPF is an example link-state protocol.

Note that all four combinations (pro-active/reactive x distance vector/link state) are theoretically possible. For example, RIP is a pro-active, distance vector protocol, while AODV is a reactive, distance vector protocol. Due to the introduction of efficient mechanisms for maintaining, updating, and searching routing tables, pro-active routing has been the most appealing approach in the Internet. Among the four combinations, reactive link state routing has been the least popular. This is attributed to its high cost in terms of delay and amount of maintained state. Recent proposals, however, attempt to alleviate some of these problems, e.g., [13].

A routing and forwarding technique sometimes applied in the Internet is source routing. **Source** routing is a technique in which the source specifies the whole path to the destination in the packet header. Source routing requires global knowledge of link-state information for the source to select an *efficient* path. Path selection can, however, be performed blindly (i.e., without global knowledge) if path efficiency is not a major concern.

Hierarchical techniques improve scalability of all routing approaches as a network grows. Typically, any routing strategy is employed within a domain, but across domain boundaries, only aggregate information is advertised. Aggregation can reduce the traffic used for updates by at least an order of magnitude (assuming a well designed hierarchy and an intelligent protocol for propagating routing updates). Figure 2 depicts an example of topology aggregation. Each domain is represented by one logical node in the aggregate structure. Each logical node is typically represented as either a star, where border routers (referred to as ports) in a domain are logically connected to a central (nucleus) node, or a full-mesh, where border routers in a domain are logically connected to each other. An example hierarchical protocol is the ATM Private Network-Network Interface (PNNI) protocol [9].

Constraint-Based Routing (CBR)

Interest in constraint-based routing has been steadily growing in the Internet community, spurred by ATM PNNI [9] and, more recently, MPLS [2]. With MPLS, fixed-length labels are attached to packets at an ingress (entry) router, and forwarding decisions are based on these labels in the interior routers of the label-switched path. MPLS traffic engineering (TE) allows overriding the default routing protocol (e.g., OSPF), thus forwarding over paths not normally considered. MPLS TE can increase path availability, reduce jitter, and reduce loss rate. Global deployment of MPLS, however, is still uncertain. This can be attributed to the complexity and cost associ-

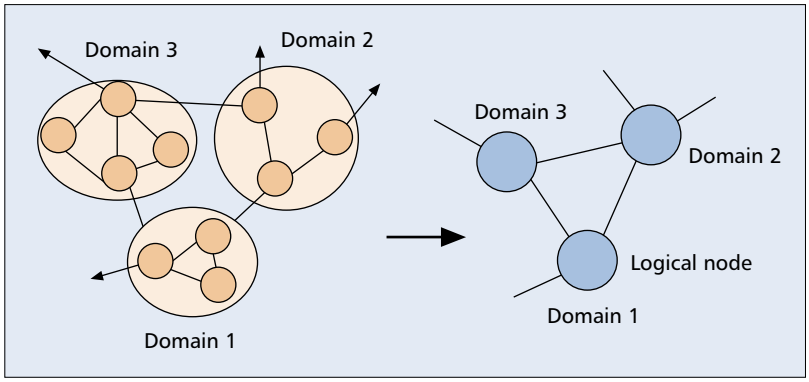

Figure 2. *Topology aggregation example.*

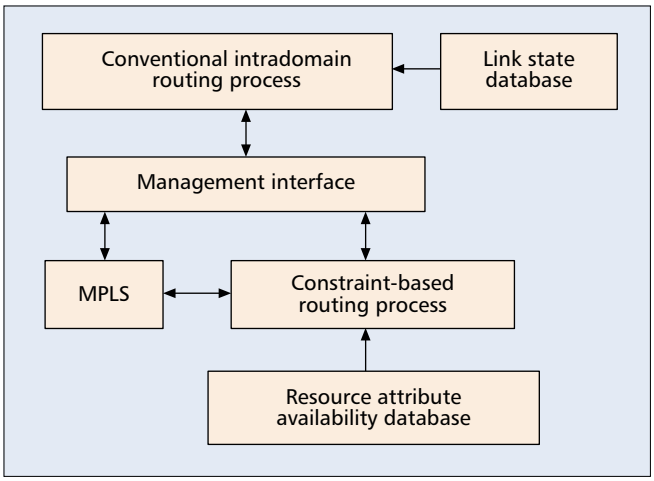

FIGURE 3. *The constraint-based routing process interfaces.*

ated with MPLS deployment. A number of recent studies [14] argue that in large IP domains, traditional routing approaches (e.g., OSPF) can be tuned to engineer traffic flows.

CBR requires mechanisms for:

- Exchanging state information (e.g., resource availability) among CBR processes.
- Maintaining this state information.
- Interacting with the current intra-domain routing protocols.
- Accommodating traffic requirements.

Inputs to a CBR process include traffic trunk attributes, traffic specifications, resource specifications, and policy specifications. A CBR process is incorporated into layer 3 (the network layer). CBR interaction with MPLS and the current intra-domain routing protocols is depicted in Fig. 3. A CBR process can be incorporated into each router and co-exist with the conventional intra-domain routing protocol processes. Routing processes obtain information through dependent or independent databases, as discussed in [2]. A CBR process can be classified as offline or online based on where path computation is performed. Table 2 provides definitions of offline and online CBR processes and their pros and cons [15].

It is important to emphasize that CBR only determines a path, and does *not* reserve any resources on that path. A resource reservation protocol such as RSVP must be employed to reserve the required resources. Classic RSVP is the setup (signaling) protocol originally designed for the integrated services (IntServ) architecture (see [16] and the references therein for a detailed description of RSVP and IntServ). Figure 4 depicts how RSVP establishes interfaces with admission and policy control (which determine whether new flows are accepted or rejected) on a network element (typically a router). RSVP also stores reservation information in a table, which is consulted when processing flow packets, e.g., to determine buffer allocation and scheduling decisions. Alternatively, routing and resource reservation requests can be combined in a single multi-path message from source to destination [10]. In both cases, after a path is selected and reserved, all packets of the flow should be forwarded on that same path. This means that the path should be fixed throughout the lifetime of the flow, or what is referred to as "route pinning." A pinned path means that CBR need not be frequently queried [6, 11]. One way of ensuring that flow packets follow an explicitly specified path is via **source routing**, where the packets themselves carry the computed path they should follow as they are forwarded to their destinations.

It is clear that classic RSVP and CBR are independent. CBR mostly runs on routers, and not on hosts. In contrast, a host or an application typically initiates RSVP reservation setup messages. RSVP messages follow the path computed by the routing protocol, so as not to introduce any dependencies on a particular routing protocol. If the routing protocol computes a new path, however, the reservation over the old path will eventually time out. The reservation must be re-instantiated over the new path. If a new path is selected due to the failure of the original path, then the connection is dropped if a reservation on the new path also fails. If the new path is suggested merely because of its better quality, then the old reservation is maintained (refreshed) until the new reservation succeeds, and transition from the old path is completed. New packets (including RSVP messages) will then be forwarded over the new path. Recently, RSVP-TE [17] was designed to run on routers (mostly), as CBR does. Its main goal is to instantiate label-switched paths that can be automatically routed away from network bottlenecks. RSVP-TE interfaces with a CBR path selection algorithm whose output is a route vector to be used with source routing. This vector is included in RSVP messages for setting up explicit route forwarding state (e.g., labels) along a path that meets the constraints.

Another architecture proposed for providing Internet QoS is the Differentiated Services (DiffServ) architecture [18]. DiffServ scales well by pushing complexity to network domain boundaries. Figure 5 illustrates that edge routers mark packets by setting six DiffServ bits in the IP header. This marking is based on bilateral SLAs between adjacent domains. The six DiffServ bits (referred to as Differentiated Services Code Point (DSCP)) determine how core routers inside a domain will forward packets (i.e., they determine packet dropping and scheduling decisions).

The six DiffServ bits are, in some sense, similar to an MPLS label. DiffServ, of course, fails to prevent congestion inside a domain if provisioning is not carefully performed.

The CBR problem can be formulated as follows. A network graph G is defined as $G = (V,E)$,where V is the set of vertices/nodes (routers or end systems), and E is the set of edges (links). Let n denote the number of constraints. Let $C = (c_1, \ldots, c_n)$ denote the ordered set of constraints, where c_i is the constraint on resource i. Table 3 describes Boolean versus quantitative (path optimization) constraints. The CBR

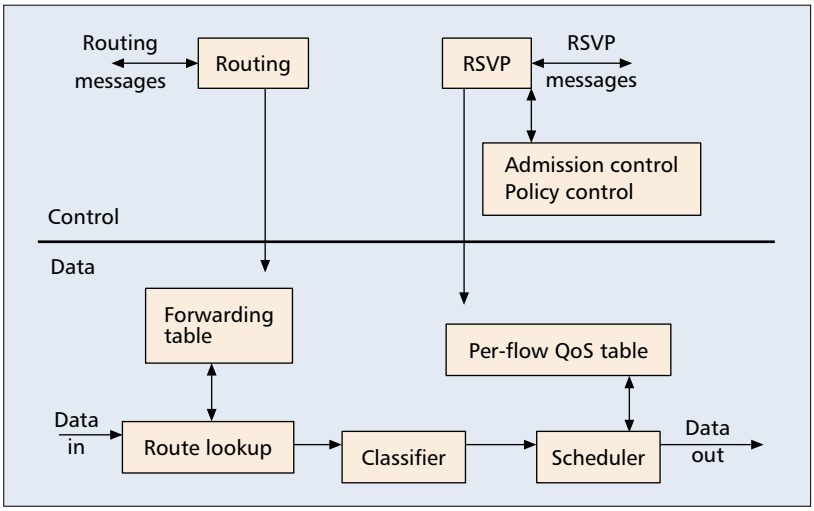

FIGURE 4. *A router with integrated services (IntServ) capabilities and RSVP.*

Constraint type	Definition	Example
Boolean (path-constrained or bounded)	Indicates path feasibility based on resource availability. Boolean constraints include administrative constraints, bandwidth availability, and delay bounds.	End-to-end delay must be less than or equal to 40 ms.
Quantitative (path optimization)	Assigns numerical values to paths so the algorithm can select among them.	The selected path must have the highest bottleneck bandwidth among all feasible paths.

Table 3. *Constraint types.*

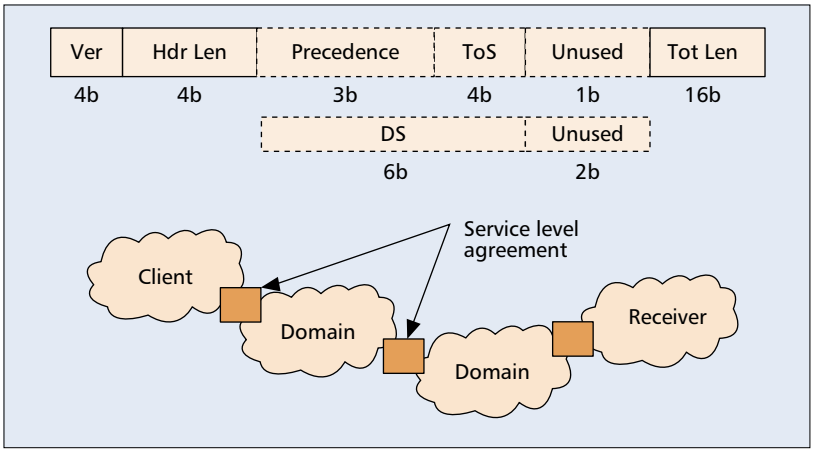

FIGURE 5. *Differentiated services (DiffServ) networks.*

objective is to find a path p between a source and a destination, such that the constraints c_i are all satisfied. Figure 6 gives an example network where $n = 3$ (e.g., bandwidth, delay, and loss-rate constraints), the number of links = 8, and each link is labeled with three $r_{i,j}$ values, where each $r_{i,j}$ denotes the value of resource i on link j. For example, $r_{2,3} = 30$ means that the delay of link 2 is 30 ms. In later sections, we use this same example network with actual resource and constraint values. Note that the available resources on a path q being explored (the $r_{i,j}$s for all links j on the path), and the constraints c_i are counterparts in this context. This means that as paths to a certain destination are being explored, the constraints are compared to the resources to ensure that constraints are still being satisfied. Table 4 defines three types of resources: configurable, dynamic, and topological.

A key problem with CBR, especially when constraints are QoS constraints, is tractability. A typical CBR scenario involves resources that are independent and allowed to take real or unbounded integer values [19]. In such scenarios, satisfying two Boolean constraints, or a Boolean constraint and a quantitative (optimization) constraint, is NP-complete. If all resources except one take bounded integer values, or if resources are dependent, then the problems can be solved in polynomial time [7]. Most proposed algorithms apply simple heuristics to reduce complexity. The time complexity of CBR algorithms is typically a function of the number of nodes, $|V|$, and/or the number of edges, $|E|$, in the network graph. Most hop-by-hop routing approaches have linear time complexity. Source routing approaches have zero forwarding complexity, but their path computation time complexity usually depends on both $|V|$ and $|E|$, e.g., Wang and Crowcroft [20] and Guerin and Orda [21] present algorithms that are $O(|V| \log |V| + |E|)$. Many current proposals have worst-case polynomial time complexity. Most proposals maintain global state information, but some maintain local state (a few others maintain aggregate information [22], [9]). In the following sections, we provide an overview of the two special cases of CBR: policy and QoS routing. For each type, we give its goals and challenges, and discuss recent research results.

Policy Routing

As new Internet services are introduced, more stringent administrative constraints need to be placed on traffic flows. Policy constraints can ensure adequate service provisioning and safety from malicious users attempting to obtain services that do not conform to their SLAs or profiles, without paying for such services. Since policy is used to implement services, services can be viewed at a higher level than policy. The policy routing problem can be viewed as a resource allocation problem that incorporates business decisions [23]. Policy routing provides many benefits, including cost savings, load balancing, and basic QoS [24]. In policy routing, routing decisions are based upon several criteria beyond the destination address, such as packet size, application, protocol used, and identity of the end systems. This makes policy routing ideal for VPN support.

Policy constraints are applied before applying QoS constraints, since they are more restrictive (especially when crossing autonomous system boundaries). Policy constraints may be exchanged by routing protocols while updating route information. They can also be provided manually during network configuration. Policy routing must tackle the following difficult questions:

- How is a policy management strategy selected? Centralized approaches are easier to deploy, but scale poorly. Distributed approaches require higher degrees of cooperation, although they scale better.

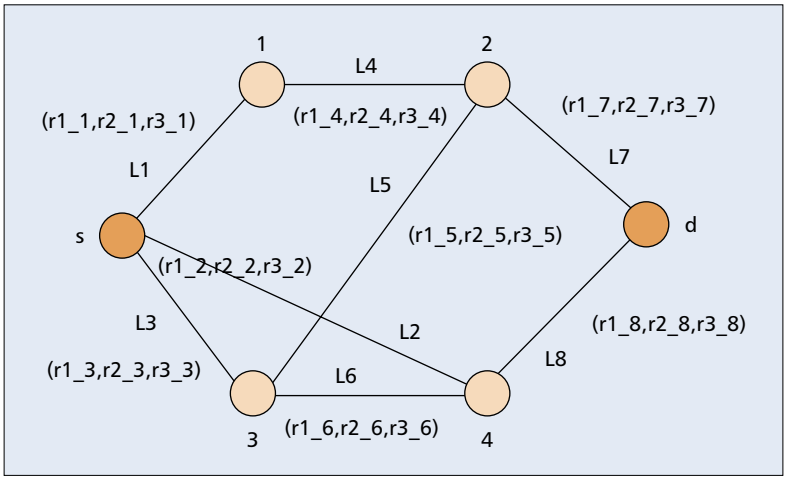

FIGURE 6. *A network graph with three resource values on each link.*

- At which point(s) in a network domain are policy constraints checked and enforced? The choice of these points affects the quality of *policy coverage* in the entire domain.
- How are policy constraints exchanged within a domain? Efficient exchange protocols are required to reduce the overhead of policy flooding, and ensure consistency.
- How is policy data stored, refreshed, and retrieved from policy repositories?
- How are policy rule conflicts and route oscillations avoided?

Figure 7 gives a simple example of policy routing. Assume that two traffic flows — a real-time streaming protocol (RTSP) flow from A, and a best-effort FTP flow from B — arrive at router *s*. Source A here is subscribed to a higher service class than B (e.g., "gold" versus "silver" classes). The flows are destined to the same end system *d*. An example policy that can be used in this case is "RTSP traffic should be routed through router 4." Based on this policy, source A traffic is routed over the path $s \rightarrow 4 \rightarrow d$, which has high bandwidth (10 Mb/s). Source B traffic is routed over the default path $s \rightarrow 3 \rightarrow 2 \rightarrow d$, which has a bandwidth of 1.5 Mb/s. This is sufficient for the best-effort FTP flow.

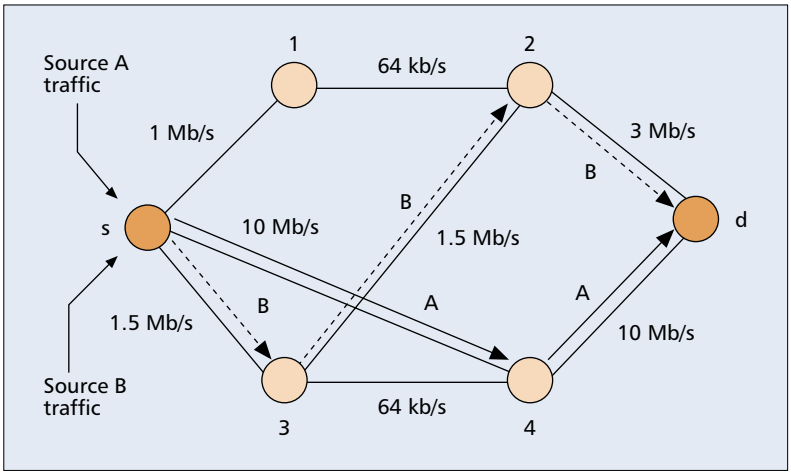

FIGURE 7. *Policy routing example for two users with different traffic types.*

Resource type	Definition	Example
Configurable	Assigned by the administrator	Link propagation delay
Dynamic	Network state dependent	Available link bandwidth
Topological	Enforced by the topology of the network	Path length

Table 4. *Resource types.*

This basic example applies to both intra-domain and inter-domain routing. In inter-domain routing, however, a node in the graph represents an AS. An important problem with inter-domain policy routing is policy rule conflicts. To illustrate this, consider inter-domain routing on the network depicted in Fig. 7. At node *s*, a policy rule of the form "traffic from AS A should be directed to AS 4," will route all traffic originating from A to AS 4. Another rule at AS 4 of the form "traffic from A should be directed to AS s," will create a routing loop (in more realistic examples, there would be a longer chain of rules). This type of oscillation in which each AS in a cycle of ASs repeatedly selects the same sequence of routers is referred to as *persistent route oscillations* [25]. Policy conflicts can generally be avoided by reducing manual configuration, and using routing policy registry servers or repositories. Such servers contain databases of registered domain policies.

The Border Gateway Protocol (BGP), the standard inter-domain routing protocol in today's Internet, provides a mechanism for distributing path information among domains without revealing private internal information. BGP uses policy routing. Analysis of BGP behavior is currently one of the most important problems being addressed by the networking research community. The importance of BGP analysis stems from its effect on route stability. Recent studies show that BGP misconfigurations can be caused by software bugs, outdated configurations, invalid routing summaries, or conflicting policy rules [26]. BGP has two operational modes: external BGP (E-BGP), which exchanges reachability (path vector) information among ASs, and internal BGP (I-BGP), which exchanges *external* reachability information within an AS (not to be confused with intra-domain routing). For I-BGP, messages are routed within an AS using connectivity information provided by the intra-domain routing protocol of this AS. Unlike E-BGP, signaling messages and forwarded traffic do not follow the same paths in both directions. This phenomenon is referred to as *path asymmetry* [27]. Path asymmetry can cause two problems for I-BGP. The first problem is *routing divergence*, which occurs when a number of routers continuously exchange routing information without reaching a stable set of routes. The other problem is *path deflection*. This occurs when a BGP router chooses the best route for an external destination, and along the path from this router to the egress (exit) point of this AS, another BGP router has chosen another egress point to the same destination. Path deflection causes inconsistent forwarding paths, and may, in the worst case, create routing loops [27].

To manage policies, several recent proposals provide a common policy framework [28, 29], as illustrated by Fig. 8. The Common Open Policy Service (COPS) is a protocol used for policy rule exchange between a policy server, referred to as a policy decision point (PDP), and a network device, referred to as policy enforcement point (PEP) [30]. The Lightweight Directory Access Protocol (LDAP) is used for policy rule retrieval from a policy repository, which is the server dedicated to the storage and retrieval of policy rules. The policy management console is the coordinator of the entire process.

An example policy routing implementation is incorporated into the Cisco IOS software. In Cisco IOS, Cisco Express Forwarding (CEF) uses a forwarding information base

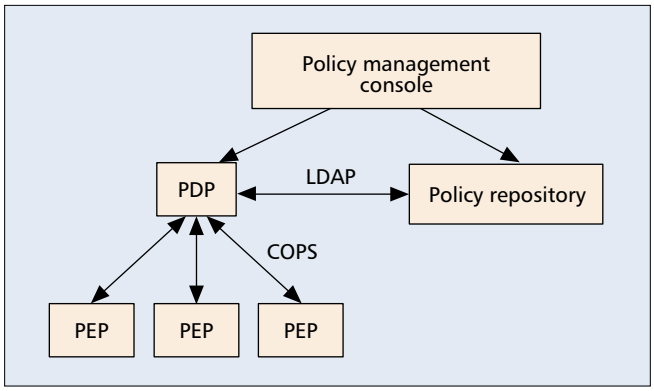

FIGURE 8. *Policy management and enforcement framework.*

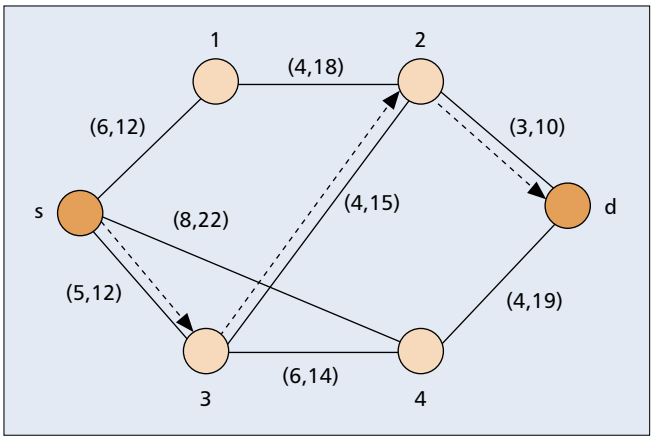

FIGURE 9. *Path selection from* s *to* d *with (cost, delay) values indicated on each link, and (cost minimization, delay ≤ 40) objective.*

(FIB) instead of a routing table when switching packets. Another component, the Distributed CEF (dCEF), addresses the scalability and maintenance problems of caching. A third component, NetFlow, enables accounting, capacity planning, traffic monitoring, and accelerating specific applications. NetFlow policy routing leverages all these components [31]. Policy routing Linux implementations are also available, e.g., [32].

A number of challenges remain in the definition of policy routing frameworks [23]. Distribution and consistency of policy rules among a number of repositories remains an important challenge. Exchanging and updating policies requires reliable authentication (COPS is a step in that direction). Enforcing policies, such as limiting certain traffic types within a domain or among domains, requires efficient mechanisms for traffic identification. Further, current architectures do not consider mobile clients. Finally, a common standard is required for policy frameworks that would allow for interoperable implementations.

QoS Routing

QoS routing selects routes based on flow QoS requirements and network resource availability. QoS routing determines feasible paths satisfying QoS requirements, while optimizing resource usage and degrading gracefully during periods of heavy load [1]. Selected routes are typically "pinned" (i.e., flows are connection-oriented [33]). An example of QoS-constrained path selection is illustrated in Fig. 9. The resources indicated on the links correspond to the cost and delay of each link. The QoS routing objective is cost minimization, subject to path delay ≤ 40 ms, i.e., $C = (delay \leq 40, min\ cost)$. The selected path p from s to d is $s \rightarrow 3 \rightarrow 2 \rightarrow d$ (with cost = 12 and delay = 37 ms). If the cost objective is changed to "cost ≤ 13," then another path, $s \rightarrow 1 \rightarrow 2 \rightarrow d$, becomes a viable choice (see [7] for similar examples). The QoS resources that are most often used as path selection constraints [34] are listed in Table 5.

The order of the two constraints is important in the case of multiple optimization (quantitative) constraints. For example, assume the two optimization constraints are the hop count and available bandwidth. The algorithm may give higher priority to selecting paths with minimum hop counts, or to selecting paths with maximum bandwidth. Choosing the path with maximum bandwidth among equal (with minimum) hop count paths provides basic load balancing on network paths. This is referred to as the "widest shortest path" approach. Alternatively, the shortest widest path selects the shortest path among paths of "equivalent" (highest) bandwidth level [20].

Recent research on QoS routing has proceeded in two directions. Initially, the main focus was on solving the routing problem for different QoS constraints, and combinations of constraints (multi-constrained QoS routing). Recently, the focus has shifted to optimizations and practical problems. We examine both research directions in this section and the next section.

Sample Unicast QoS Routing Proposals — Table 6 classifies the main unicast QoS routing proposals, and gives sample solutions that specifically address stability, robustness, and scalability concerns (which will be discussed in detail later).

Bandwidth-bounded routing: Several solutions have been proposed to this problem [21, 35, 36]. An interesting approach proposed in [35] exploits dependencies among resources, e.g., available bandwidth, delay, and buffer space, to simplify the problem. A modified Bellman-Ford algorithm can then be used.

Delay-bounded routing: This problem is often formulated as finding a path with the highest probability of satisfying a delay bound. The problem is reduced from finding a global

QoS resource	Type	Description
Available link bandwidth	Concave (min/max)	This denotes that some percentage of bandwidth will be available (reserved) for QoS flows. This resource is min/max because the minimum (bottleneck) bandwidth on the path is the available end-to-end bandwidth. Routing goals often include finding the path with the maximum bandwidth.
Link propagation delay	Additive	This denotes the latency encountered on the network links. For delay-sensitive requests, some of the links can be pruned from the graph before selecting the path.
Delay jitter	Additive	This denotes the delay variation on the network path.
Hop count	Additive	This denotes the number of hops. The minimum hop count path is used by most algorithms to designate the shortest path (least-cost path). Hop count is the only resource that is not typically included in SLAs.
Cost	Additive	This denotes an abstract measure of network resource usage. Cost can be defined in dollars, or as a function of the buffer or bandwidth utilization, for example.
Loss probability (or error rate)	Multiplicative	This denotes the acceptable loss rates, which are guaranteed through reservation of the appropriate bandwidth, provided that severe congestion does not occur in the network.

Table 5. *Typical QoS resources.*

Unicast		
Constraints		
Problem	**Technique**	**Examples**
Bandwidth-bounded	Source	Modified Bellman-Ford [35]
	Hop-by-hop	QoS routing for best-effort flows [36]
	Hierarchical	Most Reliable Path (MRP) [21]
Delay-bounded	Hop-by-hop	Distributed route selection [37]
	Hierarchical	Quantized Probabilities (QP) [21]
Bandwidth-bounded, delay-bounded	Source	Bandwidth-delay constrained path [20]
Bandwidth-optimized, delay-optimized	Hop-by-hop	Shortest widest path, widest shortest path [20]
Bandwidth-bounded, cost-bounded	Source	Extended Bellman-Ford (EBF) [7], Extended Dijkstra Shortest Path (EDSP) algorithm [7]
Delay-bounded, cost-optimized	Source	Lagrange Relaxation-based Aggregation Cost (LARAC) [38]
	Hop-by-hop	Delay-Constrained Unicast Routing (DCUR) [39]
Multi-constrained	Source	Modified Bellman-Ford [35], Heuristic Multi-Constrained Optimal Path (H_MCOP) [40]
Performance		
Problem	**Technique**	**Examples**
Stability and path availability	Source	Nash equilibrium [41]
	Hop-by-hop	Routing and resource reservation [10]
	Hierarchical	Advance reservation [8], pre-computation [4]
Robustness	Source	Network graph reduction [42], safety routing [43]
	Hop-by-hop	Adaptive proportional routing [22], dynamic routing [44], premium class routing [45], ticket-based probing [7]
	Hierarchical	Most Reliable Path (MRP) [21]
Multiple traffic classes	Hop-by-hop	QoS routing for best-effort flows [36], optimal premium class routing [45]
Scalability	General	Topology aggregation [46]
	Hop-by-hop	Selective/Ticket-based probing [7]
	Hierarchical	Partitioning QoS requirements [47], PNNI [9]

Table 6. *Summary of unicast QoS routing proposals.*

solution to a local one, in [21]. The end-to-end delay constraint is split among intermediate links, such that every link on the path has an equal probability of satisfying its local constraint. The path with the highest multiplicative probability over all links is then selected. In [37], a distributed route selection scheme is proposed in which all possible routes are searched in parallel and infeasible routes are pruned quickly to maintain linear complexity in the number of links in the network.

Bandwidth-bounded, delay-bounded routing: One approach to satisfy both bandwidth and delay bounds is to first prune all links not satisfying the bandwidth requirement. Dijkstra's shortest path algorithm is then applied to find a feasible path, if any, satisfying delay requirement [20].

Bandwidth-optimized, delay-optimized routing: As previously discussed, this problem can be either solved as a *widest shortest path problem* or a *shortest widest path* problem [20].

Bandwidth-bounded, cost-bounded routing: Solutions to this problem typically map the cost or the bandwidth to a bounded integer value, and then solve the problem in polynomial time using an Extended Bellman-Ford (EBF) or Extended Dijkstra Shortest Path (EDSP) algorithm [7].

Delay-bounded, cost-optimized routing: This problem has witnessed significant interest [38, 39, 48]. Ergun *et al.* [48] propose a number of approximation algorithms that provide performance guarantees. Lagrange Relaxation based Aggregation Cost (LARAC) [38] uses aggregated cost and Lagrange relaxation, which provide a means for controlling the tradeoff between the running time of the algorithm and the quality of computed paths.

Multi-constrained routing: The objective of multi-constrained routing is to simultaneously satisfy a set of constraints [35, 40. Korkmaz *et al.* [40] propose a heuristic approach for the multi-constrained optimal path problem (H MCOP), which optimizes a non-linear function (for feasibility) and a primary function (for optimality). Although this outperforms some linear approximation approaches, such as [7], performance with inaccurate information is still under study.

Sample Multicast QoS Routing Proposals — Multicast QoS routing is generally more complex than unicast QoS routing. The additional complexity stems from the need to support shared and heterogeneous reservation styles and global admission control, in addition to the typical QoS routing requirements, such as scalability and robustness. Most current multicast QoS routing algorithms are designed to satisfy bandwidth, delay, jitter, and cost constraints. The time complexity

Multicast		
Constraints		
Problem	**Technique**	**Examples**
Delay-optimized	Source	MOSPF [49], Steiner tree [50]
Delay-bounded, cost-optimized	Source	Constrained Steiner tree solutions (constrained adaptive ordering [51–53])
	Hop-by-hop	Distributed constrained Steiner tree [54], QoS-aware Multicast Routing Protocol (QMRP) [55]
Delay-bounded, jitter-bounded	Source	Delay Variation Multicast Algorithm (DVMA) [56]
Performance		
Problem	**Technique**	**Examples**
Path availability	Source	Network graph reduction [42]
	Hop-by-hop	Ticket-based probing [7]
Scalability	Hop-by-hop	QoS-aware Multicast Routing Protocol (QMRP) [55]
	Hierarchical	Partitioning QoS requirements [47]

Table 7. *Summary of multicast QoS routing proposals.*

of current proposals is generally polynomial. Table 7 classifies current proposals, which include:

Delay-optimized (or bandwidth-optimized) routing: Algorithms proposed for this optimization problem use source routing and maintain global state. For example, MOSPF [49] is the multicast version of OSPF. Other approaches, e.g., [50], use a Steiner tree formulation.

Delay-bounded, cost-optimized routing: This problem can be formulated as a constrained Steiner tree problem. An interesting approach, QoS-aware Multicast Routing Protocol (QMRP) [55], monitors network conditions and adaptively switches between single-path routing and multi-path routing.

Delay-bounded, jitter-bounded routing: Delay jitter-bounded multicast QoS routing can be solved using a constrained Steiner tree approach. In the Delay Variation Multicast Algorithm (DVMA), a multicast tree with bounded delay and bounded delay-jitter is constructed [56].

Several recent studies aim at increasing robustness of QoS multicast routing, e.g., [42]. In the next section, we address stability, robustness, and scalability issues in more depth.

Routing Challenges

In this section, we address the primary challenges with CBR, especially with QoS constraints. These include stability, robustness, and scalability.

Stability

If path selection is based on resource availability, every node in a network must maintain local state information. This local state typically includes available bandwidth, and queuing and propagation delays of the outgoing links. A node can also maintain global state (the collective local states of all nodes), and use a protocol for exchanging link state [49]. An important decision in a routing protocol is the frequency of link updates. A high frequency of updates (e.g., whenever the link bandwidth changes) increases traffic and routing overhead, and thus does not scale to large networks. Minimizing link update frequency, however, makes information inaccurate. To balance this tradeoff, updates can be advertised whenever there is a significant change in the value of the resources used in the constraints [11]. There are two ways of measuring significance:

- Absolute scale, which entails dividing the range of values into equivalence classes and triggering the update accordingly.
- Relative scale, which entails triggering the update when the percentage of change since the last advertisement exceeds a certain threshold.

This may result in a high update frequency, when multiple resources are allocated or deallocated. In addition to resource-based updates, updates are triggered whenever a certain time period expires. Of course, a certain minimum period is enforced between successive updates.

State updates increase when dynamic routing is used. Dynamic routing is a technique in which a static route is not pre-reserved; rather, the route is determined dynamically in order to bypass congested links and balance the load. Dynamic routing can thus be considered as a *reactive* routing approach that may be periodically invoked for better performance and load balancing. The main problems with dynamic routing are instability, route flapping, and high frequency of state updates. If dynamic routing can be augmented with a technique to keep the state update frequency moderate, it balances the load and scales well. An example load-sensitive routing algorithm is presented in [44], in which short-lived flows are routed statically, while dynamic routing is applied to long-lived flows. Another approach to address stability is [41], where user flows compete (in a game-theoretic sense), and the system attempts to reach a Nash equilibrium. Combinations of conventional routing and QoS routing are used in [36, 45] to increase stability. Significant research is still required to analyze route stability in various contexts.

Robustness

The process of determining a route to accommodate a new incoming request relies on the accuracy of available state information. If resource information is inaccurate, some flows that can be accommodated may be rejected and vice versa. These inaccuracies might occur due to the following reasons [21]:

- Limitations on the frequency at which updates are performed.
- Limitations on the number of nodes (or links) generating update information.
- Aggregation of state information for scalability.

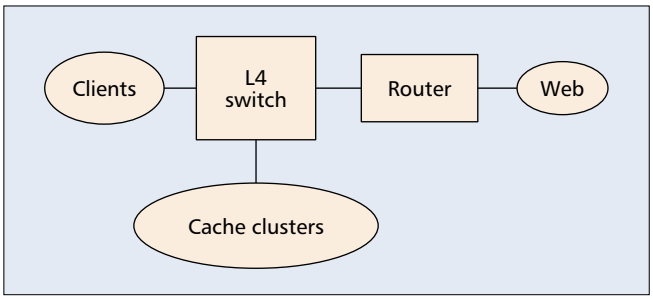

FIGURE 10. *Transparent caching with L4 switch.*

A number of studies analyze the impact of inaccurate information on routing [43, 57]. These studies reveal three interesting results. First, the effect of inaccuracies is minimal if only bandwidth requirements are given. Second, inaccuracies can have a high impact on end-to-end delay requirements, and the path selection problem then becomes intractable. Two models are studied for end-to-end delay: the rate-based model and the delay-based model. In the former, the problem is intractable. In the latter, the problem can be tractable using some heuristics, such as splitting the end-to-end constraints into local constraints. Third, triggering policies for updates (that do not result in pruning of links and advertised values) and randomization approaches for path selection can perform well in periods of high inaccuracy in the perceived link information.

Another technique to overcome inaccuracies is safety routing [43]. Given the requested amount of bandwidth, the available bandwidth last advertised, and the triggering policy, a range of values for the available bandwidth on the link can be determined. Assuming a distribution function for the available bandwidth on a link, and using the range of values obtained, the availability of the required amount of bandwidth can be probabilistically determined. Computation of the distribution functions that realistically model information inaccuracies remains an open problem.

In addition to operating with inaccurate information, CBR must gracefully degrade in the events of link failures or congestion. Dynamic routing [54] is one approach to adapt to such conditions. Alternatively, congested links can be pruned before making routing decisions [42]. A third alternative is using self-adaptive proportional routing techniques, such as Proportional Sticky Routing (PSR) [22]. In PSR, the only route-level information assumed to be available is the number of blocked flows. The technique distributes the load from a source to a destination among multiple paths according to the observed flow blocking probability.

Scalability and Routing Cost

Implementation and deployment costs of CBR include both computational cost and protocol overhead [6, 57]. The computational cost can be offset by the evolution of technology, i.e., faster processors and larger storage. In contrast, protocol overhead is not easily contained, because it affects several parameters, such as available link bandwidth and storage. The main factors affecting the computational cost are the path selection criteria, the trigger for path selection computations, and flexibility in supporting alternate path selection choices. The main factors affecting the protocol overhead are the update frequency, and the update message size. A study of QoS routing extensions to OSPF [11] reveals that the processing cost of applying QoS routing is within the capabilities of medium-range processors, and that link-state generation and reception cost is tolerable, even in the case of large networks [6]. In addition, the fraction of bandwidth usage for updates is small (less than 1 percent).

In order to scale better, only aggregate information is advertised outside a domain (recall Fig. 2). Topology aggregation mechanisms may not always negatively impact routing performance, depending on the QoS routing algorithm, network topology, and state update frequency [46]. An example of aggregation with widest-shortest path routing is provided in [46]. In this work, hop count information is advertised infrequently but in detail, while the available bandwidth is advertised more frequently but in less detail. An alternative approach limits QoS routing decisions to edge routers, thus reducing overhead on core routers [22]. A third approach employs a "divide and conquer" strategy to partition QoS requirements into smaller components (sub-paths or subtrees), and later combines the results for the entire structure [47]. Finally, ticket-based probing [7] limits probe flooding on QoS paths according to the network contention level.

Application-Layer Routing Example: Content Routing

Performance, scalability, and availability concerns for Web services have led to the introduction of proxies and cluster-based server architectures. Boomerang (from Cisco), Squid, and Akamai are example architectures. Proxies and clusters provide a means for rapid information access, especially during periods of high demand for some particular data [58, 59]. To balance the load among servers, the content of the HTTP request can be considered in making the request routing decision. DNS-based approaches are not as powerful, since they resolve the destination server address without examining the HTTP request. The request content can be used to redirect each incoming request to the most appropriate proxy or cache.

It is important to distinguish between local load balancing and global load balancing. Local load balancing improves availability and scalability by intercepting and redirecting incoming requests to one member of a group of proxy servers in a common cluster. Global load balancing has similar goals, but requests are not distributed among local members of a group. Instead, they are distributed among servers or caches that may be near the client, which reduces latency and increases performance.

Content routing is an application-level routing approach used in such transparent caching architectures. A dedicated switch can be used to carry out the load balancing task, along with the content routing. For example, a Layer 4 switch (L4 switch) can intercept requests and redirect them to one of the caches according to the TCP or UDP (transport) headers (as illustrated in Fig. 10). Examples of L4 switches are the Alteon ACE-Director and the CACHE-Director. The main problem with L4 switches is that they require that the entire data be replicated on the caches (which can share the same file system). This imposes a large storage and update overhead.

An alternative architecture employs an L5 switch, which uses session-level information (mainly the URL), together with information provided from lower layers, to make the routing decision. This switch can be used anywhere in the network. An example of this switch is the Arrowpoint Content SmartSwitch (CSS). The usefulness of the L5 system as a front-end to a server cluster is studied in [60]. Performance analysis reveals that when the L5 system partitions the URL space among all the cluster members, performance significantly improves. With secure HTTP connections, which use the Secure Socket Layer (SSL) for authentication, the L5 system can improve overall throughput. Figure 11 shows an L5 system in which the L5 switch is a front end to a server cluster.

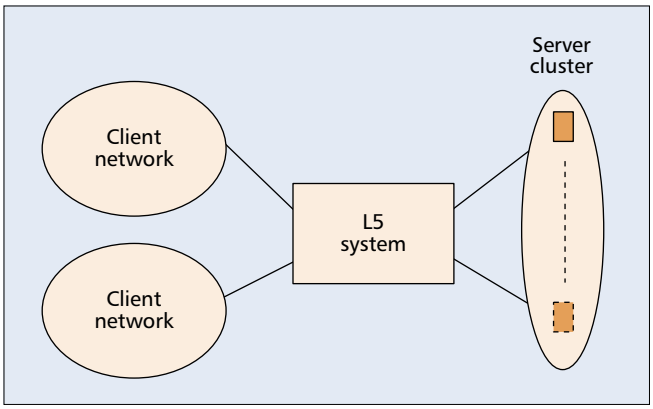

FIGURE 11. *Transparent caching with L5 switch.*

The primary problem with the L5 switch is that its protocols run on top of the TCP layer, and have to establish a TCP connection with the source and migrate this live connection to the destination. This process is not easy without changing the TCP message format and TCP state machine [60].

Several new protocols are being introduced for load balancing. A content-aware distributor is introduced between the client and the server in [58]. The distributor manages connection establishment, and uses a dispatcher to parse the URL of the incoming request. After looking up the URL in cluster tables, the distributor selects the server with the lowest load to handle the request. Using hashing techniques can speed up the search in the URL tables. Another example are Cisco content routing protocols that allow communication about content state among Cisco networking products [61]. The Dynamic Feedback Protocol (DFP) enables load balancing devices to take advantage of the available information on servers and network appliances to increase availability. The Director Response Protocol (DRP) performs global load balancing by allowing the devices to share routing information for optimal performance. The Web Cache Communication Protocol (WCCP) offers transparent Web cache redirection similar to that provided by Layer 4-7 switches. Recently, Cisco and Akamai have joined forces to improve such protocols for Internet content routing and service delivery. More discussions on challenges in content routing can be found in [62].

CONCLUSIONS

Constraint-based routing comprises both policy and QoS routing. Policy routing is important for providing better and more flexible services. We have discussed the general policy framework, policy routing problems in BGP, and some of the recent work at the Internet Engineering Task Force (IETF) working groups. QoS routing has been studied in the literature more extensively than policy routing. Recent studies show the possibility of performing QoS routing with inaccurate information without suffering significant performance losses. In addition, applying aggregation techniques for scalability does not always negatively impact performance. Intelligent tuning of QoS routing algorithm parameters used in state updates can improve performance in terms of stability and load balancing.

Future work in this area hinges upon resolving concerns related to complexity, scalability, and ease of deployment. A number of projects such as RON [63], Detour [64], and peer-to-peer (P2P) systems provide load balancing, content routing, or dynamic selection among multiple paths. These approaches move the route selection functionality to the application or transport layer, through the use of overlay networks of cooperating end systems. Application-layer approaches are also useful in the case of multicasting when multicast cannot be easily supported at the router-level [65, 66]. Balancing the scalability versus adaptivity tradeoff in such approaches, however, remains an important challenge.

ACKNOWLEDGMENTS

We would like to thank Professor Martin Reisslein, Professor John N. Daigle, and the anonymous reviewers for their valuable comments that helped improve the article. This work has been sponsored in part by the Schlumberger Foundation technical merit award.

REFERENCES

[1] E. Crawley *et al.*, "A Framework for QoS-based Routing in the Internet," RFC 2386, Aug. 1998.
[2] D. Awduche *et al.*, "Requirements for Traffic Engineering over MPLS," RFC 2702, Sept. 1999.
[3] D. Awduche *et al.*, "Overview and Principles of Internet Traffic Engineering," RFC 3272, May 2002.
[4] A. Orda and A. Sprintson, "QoS Routing: The Precomputation Perspective," *Proc. Conf. Comp. Commun.*, *(IEEE INFOCOM'00)*, Mar. 2000.
[5] J. Kurose and K. Ross, *Computer Networking: A Top Down Approach Featuring the Internet*, Addison Wesley, 2001.
[6] G. Apostolopoulos *et al.*, "Intra-Domain QoS Routing in IP Networks: A Feasibility and Cost/Benefit Analysis," *IEEE Network*, vol. 13, no. 5, Sept./Oct. 1999.
[7] S. Chen and K. Nahrstedt, "An Overview of Quality of Service Routing for Next-Generation High-Speed Networks: Problems and Solutions," *IEEE Network*, vol. 12, no. 6, Nov./Dec. 1998.
[8] R. Guerin and A. Orda, "Networks with Advance Reservations: The Routing Perspective," *Proc. Conf. Comp. Commun.*, *(IEEE INFOCOM'00)*, Mar. 2000.
[9] ATM Forum, "Private Network-Network Interface (PNNI) v. 1.1 Specifications," http://www.atmforum.com/standards/approved.html, Apr. 2002.
[10] I. Cidon, R. Rom, and Y. Shavitt, "Multirate Routing Combined with Resource Reservation," *IEEE INFOCOM'97*, Apr. 1997.
[11] G. Apostolopoulos *et al.*, "QoS Routing Mechanisms and OSPF Extensions," RFC 2676, Aug. 1999.
[12] D. Medhi, "QoS Routing with Path Caching: A Framework and Network Performance," *IEEE Commun. Mag.*, vol. 40, no. 12, Dec. 2002.
[13] S. Roy and J. J. Garcia-Luna-Aceves, "An Efficient Path Selection Algorithm for On-Demand Link-State Hop-by-Hop Routing," *Proc. IEEE ICCCN*, 2002.
[14] B. Fortz, J. Rexford, and M. Thorup, "Traffic Engineering with Traditional IP Routing Protocols," *IEEE Commun. Mag.*, vol. 40, no. 10, Oct. 2002.
[15] X. Xiao *et al.*, "Traffic Engineering with MPLS in the Internet," *IEEE Network*, vol. 14, no. 2, Mar./Apr. 2000.
[16] S. Fahmy and R. Jain, *Handbook of Communications Technologies: The Next Decade*, chapter "Resource ReSerVation Protocol (RSVP)," CRC Press, July 1999.
[17] D. Awduche *et al.*, "RSVP-TE: Extensions to RSVP for LSP Tunnels," RFC 3209, Dec. 2001.
[18] S. Blake *et al.*, "An Architecture for Differentiated Services," RFC 2475, Dec. 1998.
[19] K. Kompella and D. Awduche, "Notes on Path Computation in Constraint-Based Routing," Internet Draft, July 2000, http://www.awduche.com/papers/papers2000/draft-kompella-te-pathcomp-00.txt
[20] Z. Wang and J. Crowcroft, "Quality-of-Service Routing for Supporting Multimedia Applications," *IEEE JSAC*, vol. 14, no. 7, Sept. 1996, pp. 1228–34.
[21] R. Guerin and A. Orda, "QoS-Based Routing in Networks with Inaccurate Information: Theory and Algorithms," *IEEE/ACM Trans. Net.*, June 1999, pp. 350–64.
[22] S. Nelakuditi, Z. Zhang, and R. Tsang, "Adaptive Proportional Routing: A Localized QoS Routing Approach," *Proc. Conf. Comp. Commun. (IEEE INFOCOM'00)*, Mar. 2000.

[23] H. Mahon *et al.*, "Requirements for a Policy Management System," Internet Draft, Nov. 2000, http://www.watersprings.org/pub/id/draft-mahon-policy-mgmt-00.txt
[24] Cisco white paper, "Policy Based Routing," http://www.cisco.com/warp/public/cc/techno/protocol/tech/plicy wp.htm, Aug. 2002.
[25] K. Varadhan, R. Govindan, and D. Estrin, "Persistent Route Oscillations in Inter-Domain Routing," *Computer Networks*, vol. 32, no. 1, Jan. 2000, pp. 1–16.
[26] R. Mahajan, D. Wetherall, and T. Anderson, "Understanding BGP Misconfiguration," *Proc. SIGCOMM'02*, Aug. 2002, pp. 3–16.
[27] T. G. Griffin and G. Wilfong, "On the Correctness of IBGP Configuration," *Proc. SIGCOMM'02*, Aug. 2002, pp. 17–29.
[28] J. Strassner *et al.*, "Policy Core Information Model - Version 1 Specification," RFC 3060, Feb. 2001.
[29] Y. Snir *et al.*, "Policy QoS Information Model," Internet Draft, July 2001, http://www.potaroo.net/ietf/ids/draft-ietf-policy-qos-info-model-04. txt
[30] D. Durham *et al.*, "The COPS (Common Open Policy Service) Protocol," RFC 2748, Nov. 1999.
[31] Cisco Systems, *IOS Release 12.0: Network Protocols Configuration Guide*, chapter "Configuring IP Routing Protocol-Independent Features," Cisco Systems, 2000.
[32] M. Marsh, "Policy Routing Using Linux 1/e," Sams, 2001.
[33] Z. Wang and J. Crowcroft, "Quality-of-Service Routing for Supporting Multimedia Applications," London, GB, Oct. 1994.
[34] S. Chen, "Routing Support for Providing Guaranteed End-to-End Quality-of-Service," Ph.D. thesis, University of Illinois at Urbana-Cahmpaign, May 1999.
[35] Q. Ma and P. Steenkiste, "Quality of Service Routing with Performance Guarantees," *Int'l. IFIP Wksp. Quality of Service*, May 1997.
[36] P. Goyal and G. Hjalmtysson, "QoS Routing for Best-Effort Flows," *Proc. Int'l. Wksp. Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, (Basking Ridge, New Jersey), June 1999.
[37] K. G. Shin, C. C. Chou, and S. K. Kweon, "Distributed Route Selection for Establishing Real-Time Channels," *IEEE Trans. Parallel and Distributed Systems*, vol. 11, no. 3, Mar. 2000, pp. 318–35.
[38] A. Juttner *et al.*, "Lagrange Relaxation-Based Method for the QoS Routing Problem," *Proc. Conf. Comp. Commun., (IEEE INFOCOM'01)*, Apr. 2001.
[39] H. F. Salama, D. S. Reeves, and Y. Viniotis, "A Distributed Algorithm for Delay-Constrained Unicast Routing," *IEEE INFOCOM'97*, Apr. 1997.
[40] T. Korkmaz and M. Krunz, "Multi-Constrained Optimal Path Selection," *Proc. Conf. Comp. Commun., (IEEE INFOCOM'01)*, Apr. 2001.
[41] E. Altman *et al.*, "Competitive Routing in Networks with Polynomial Cost," *Proc. Conf. Comp. Commun., (IEEE INFOCOM'00)*, Mar. 2000.
[42] C. Casetti *et al.*, "A New Class of QoS strategies Based on Network Graph Reduction," *Proc. Conf. Comp. Commun., (IEEE INFOCOM'02)*, June 2002.
[43] G. Apostolopoulos *et al.*, "Improving QoS Routing Performance Under Innacurate Link State," *Proc. 16th Int'l. Teletraffic Congress, (ITC'16)*, Edinburgh, United Kingdom, June 1999.
[44] A. Shaikh, J. Rexford, and K. Shin, "Load-Sensitive Routing of Long-Lived IP Flows," *Proc. ACM SIGCOMM'99*, Sept. 1999.
[45] J. Wang and K. Nahrstedt, "Hop-by-Hop Routing Algorithms for Premium-Class Traffic in DiffServ Networks," *Proc. Conf. Comp. Commun. (IEEE INFOCOM'02)*, June 2002.
[46] F. Hao and E. Zegura, "On Scalable QoS Routing: Performance Evaluation of Topology Aggregation," *Proc. Conf. Comp. Commun. (IEEE INFOCOM'00)*, Mar. 2000.
[47] A. Orda and A. Sprintson, "A Scalable Approach to the Partition of QoS Requirements in Unicast and Multicast," *Proc. Conf. Comp. Commun. (IEEE INFOCOM'02)*, June 2002.
[48] F. Ergun, R. Sinha, and L. Zhang, "QoS Routing with Performance-Dependent Costs," *Proc. Conf. Comp. Commun.*, *(IEEE INFOCOM'00)*, Mar. 2000.
[49] J. Moy, "OSPF Version 2," RFC 2178, July 1997.
[50] L. Kou, G. Markowsky, and L. Berman, "A Fast Algorithm for Steiner Tree," *Acta Informatica*, 1981, pp. 141–45.
[51] R. Widyono, "The Design and Evaluations of Routing Algorithms for Real-Time Channels," TR-94-024, University of California at Berkeley, International Computer Science Institute, June 1994.
[52] V. P. Kompella, J. C. Pasquale, and G. C. Polyzos, "Multicast Routing for Multimedia Communication," *IEEE/ACM Trans. Net.*, June 1993.
[53] Q. Sun and H. Langendorfer, "Efficient Multicast Routing for Delay-Sensitive Applications," *2nd Int'l. Wksp. Protocols for Multimedia Systems (PROMS'95)*, Oct. 1995, pp. 452–58.
[54] V. P. Kompella, J. C. Pasquale, and G. C. Polyzos, "Two Distributed Algorithms for Multicasting Multimedia Information," *ICCCN'93*, San Diego, CA, June 1993, pp. 343–49.
[55] S. Chen, K. Nahrstedt, and Y. Shavitt, "A QoS-Aware Multicast Routing Protocol," *IEEE JSAC*, vol. 18, no. 12, Dec. 2000.
[56] G. N. Rouskas and I. Baldine, "Multicast Routing with End-to-End Delay and Delay-Variation Constraints," *IEEE JSAC*, Apr. 1997, pp. 346–56.
[57] G. Apostolopoulos *et al.*, "Quality of Service Based Routing: A Performance Perspective," *ACM Comp. Commun. Review*, vol. 28, no. 4, Sept.1998.
[58] C. S. Yang and M. Y. Luo, "Efficient Support for Content-Based Routing in Web Server Clusters," *2nd USENIX Symp. Internet Technologies and Systems*, (Boulder, Colorado, USA), Oct. 1999.
[59] G. Barish and K. Obraczka, "World Wide Web Caching: Trends and Techniques," *IEEE Commun. Mag.*, Internet Technology Series, vol. 38, no. 5, May 2000.
[60] G. Apostolopoulos *et al.*, "Design, Implementation and Performance of a Content-Based Switch," *Proc. Conf. Comp. Commun. (IEEE INFOCOM'00)*, Mar. 2000.
[61] Cisco white paper, "Cisco Content Routing Protocols," http://www.cisco.com/warp/public/cc/pd/cxsr/cxrt/tech/ccrp wp.htm, Mar. 2001.
[62] B. Subbiah and Z. A. Uzmi, "Content-Aware Networking in the Internet: Issues and Challenges," *Proc. ICC'01*, June 2001.
[63] D. G. Andersen *et al.*, "Resilient Overlay Networks," *Proc. ACM SOSP*, Oct. 2001.
[64] S. Savage *et al.*, "Detour: A Case for Informed Internet Routing and Transport," *IEEE Micro*, vol. 1, no. 19, Jan. 1999, http://www.cs.washington.edu/research/networking/detour/ and http://ramp.ucsd.edu/index.html, pp. 50–59.
[65] Y. Chu, S. Rao, and H. Zhang, "A Case for End-System Multicast," *Proc. ACM Sigmetrics*, June 2000.
[66] M. Kwon and S. Fahmy, "Topology-Aware Overlay Networks for Group Communication," *Proc. ACM NOSSDAV*, May 2002, http://www.cs.purdue.edu/homes/fahmy/, pp. 127–36.

BIOGRAPHIES

OSSAMA YOUNIS (oyounis@cs.purdue.edu) received the B.S. and M.S. degrees from the computer sciences department, faculty of engineering, Alexandria University, Egypt, in 1995 and 1999, respectively. Since 2000 he has been pursuing his Ph.D. degree at the computer sciences department, Purdue University. His research interests include Internet routing and tomography, sensor networks, and distributed systems.

SONIA FAHMY (fahmy@cs.purdue.edu) is an assistant professor at the computer sciences department at Purdue University. She obtained her Ph.D. degree from Ohio State University in September 1999. Her research interests are in the design and evaluation of network architectures and protocols. She is a member of the ACM, IEEE, Phi Kappa Phi, Sigma Xi, and Upsilon Pi Epsilon. Please see http://www.cs.purdue.edu/homes/fahmy/ for more information.