Declarative Networking

Timothy Roscoe
Joint work with Boon Thau Loo, Tyson Condie,
Joseph M. Hellerstein, Petros Maniatis, and Ion Stoica
Intel Research at Berkeley and U.C. Berkeley

1st June 2005

Despite nearly 40 years of research into routing on packet-oriented networks, routing protocols are still highly complex (with much of this complexity pushed into "policy") and difficult to configure and manage. Much of this complexity in the Internet is a reflection of complex business relationships which map poorly onto the actual protocol design, but it is remarkable that during this time practically *no* new abstractions have emerged to simplify or modularize the business of network routing – in stark contrast to other fields of systems design, such as operating systems (kernels, processes, etc.), distributed systems (RPC, objects, state machines, etc.), and databases (the relational model).

In the Declarative Networking group in Berkeley, we are investigating the connection between network routing and distributed query processing. Our insight is to view updating routing tables in a network as the ongoing process of maintaining a distributed data structure with particular properties – properties which characterize the network in question. The properties of this distributed data structure can be expressed declaratively as a query over distributed network state, which is executed continuously (in the sense of "continuous query processing").

We have tried this approach, with some success, in two different settings: IP network routing (a paper in this year's SIGCOMM) and overlay networks.

Overlay networks represent an interesting, and attractive, initial subproblem in this space. We use the term "overlay network" broadly. Overlay networks are used today in a variety of distributed systems ranging from file-sharing and storage systems to communication infrastructures. Overlays of various kinds have recently received considerable attention in the networked systems research community, partly due to the availability of the PlanetLab platform. However, a broader historical perspective is that overlay functionality has implicitly long been a significant component of wide-area distributed systems.

Despite this, designing, building and adapting these overlays to an intended application and the target environment is a difficult and time consuming process.

To ease the development and the deployment of such overlay networks, our group has built a system which uses a variant of the Datalog declarative logic language to express overlay networks in a highly compact and reusable form. The language can specify a Narada-style mesh network in 13 rules, and the Chord structured overlay in only 35 rules – this includes code to perform churn handling and "stabilization". The language is directly parsed and compiled into a per-node graph of dataflow elements which construct and maintain the overlay networks. The system is implemented in C++ and executes with small overhead and memory footprint.

Unlike some other proposals for overlay toolkits, we do not aim for performance results on a par with handcrafted overlay implementations and hand-tuned overlay parameters. Instead, we aim to show that declarative overlay descriptions can be implemented with *acceptable* performance, and that there are benefits to the declarative specification that go beyond the raw performance of a single overlay design. We believe that this is useful for rapidly prototyping new ideas, and eventually for deploying production systems as well. Experimental results show that the system can implement overlay networks with reasonable efficiency compared with existing implementations, which typically require two orders of magnitude more code.

In this respect, our argument for declarative specification of overlay properties, and in fact of network routing behaviour in general, is analogous to the argument for SQL and relational database management systems some 35 years ago: while remaining a fertile area for research, network routing and topology construction are a sufficiently well-understood technical problem that it's high time we abstracted this functionality into a domain-specific language for all but very small or very specialized forms of network. There is value in developers and operators paying a small price in runtime efficiency and/or performance in exchange for drastically reduced time spent in development, deployment, management, and evolution of routing logic.

The initial goals of our implementation are also akin to those of the early relational database systems: to explore the feasibility of the declarative approach in practice at a coarse grain, without trying to capture all possible optimizations in the first generation of the system.