Cooperative Application-level Processing on Hosts and their Attached Network Processors

Ada Gavrilovska, Karsten Schwan, Austen Mcdonald, Hailemelekot Seifu, Ola Nordstrom
Center for Experimental Research in Computer Systems (CERCS)
Georgia Institute of Technology
Atlanta, Georgia, 30332
{ada, schwan, austen, seif, nalo}@cc.gatech.edu

Overlay networks, ranging from simple front-end/back-end distinctions made in large scale server systems, to ones that extend across multiple Inter- or intra-net nodes, are finding increasing use in large scale distributed and grid applications. Services required by applications executing on top of such overlays include 'traditional' services such as data routing, multicast or quality of service support, security, or new services involving data customization and transformation for XML-based or multimedia and remote graphics applications. Application- or middleware-level implementations of these services require participating hosts to execute both application-level data manipulations, as well as the protocols used for data receipt and forwarding. They also require the data being transported to repeatedly cross hosts' I/O infrastructures.

An emerging class of programmable network processors (NPs) is becoming an attractive vehicle for deploying new functionality into the network infrastructure, with shorter development time then custom-design ASICs and with levels of cost/performance exceeding that of purely server-based infrastructures. NP hardware is optimized to efficiently move large volumes of packets between their incoming and outgoing ports, and typically, there is an excess of cycles available on the packet's fast path through the NP. Such 'headroom' has been successfully used to implement network-centric services such as software routing, network monitoring, intrusion detections, service differentiation, etc.

Our research goal is to utilize such programmable NPs for the execution of application- and middleware-level services required in large scale distributed applications, in order to attain improvements in end-user application performance, to more efficiently utilize server capacity, and to offer new services at no additional performance overheads perceived by end-users. By mapping service functionality to the combined resources offered by hosts and their attached network processors (ANPs), we obtain integrated host/ANP platforms much better suited for efficient service execution.

In order to enable the host-ANP cooperation, we are developing a communication architecture termed SPLITS (Software architecture for Programmable, LIghtweighT Stream handling), which allows developers to dynamically deploy and configure service functionality onto ANPs so as to best use the combined ANP/host resources. Services are composed with stream handlers, lightweight processing units applied on application-level messages, which run on ANPs, in the host kernel, or at application-level. SPLITS and stream handlers are implemented for hosts that run standard Linux OS kernels and for ANPs that are based on Intel's IXP network processor. Performance gains are due to the network-near execution of stream handlers on the ANP, the load reduction on the host system CPU and memory infrastructure, and the flexibility with which stream handling can be mapped across host-ANP boundary.

Our approach provides support for executing a rich set of lightweight middleware- and application-level services on the ANPs, directly on the overlay network's data paths, using fast ANP hardware. For more complex services, ANP resources may not be sufficient. In such cases, portions of the overlay processing are performed on the host node to which an ANP is attached, thereby splitting the protocol stack across host-ANP boundary. Sample services realized with SPLITS include content-based mirroring or filtering, variable-content multicast, stream merging and differentiation, as well as processing of XML-structured data as needed at the 'edges' of network infrastructures, and graphics services that perform per-client customizations of graphical displays. Experimental evaluations of SPLITS with NP/host combinations are shown to run more efficiently and offer better performance to applications compared to those that use pure host-based implementations.