MIT2000-10
Software Technologies for Wireless Communication and Multimedia Networks

Stephen J. Garland, John V. Guttag and David Karger

1. Motivation

As communication penetrates into more aspects of daily life, the number of special-purpose devices, networks, and systems is increasing rapidly. At the same time, computation technologies are producing increasingly powerful general-purpose devices and systems. These two trends raise an interesting research problem, namely, how to design multiple-purpose communication devices and systems that efficiently adapt general capabilities to specific tasks. Adaptable devices and systems would

- reconfigure themselves to reduce the number of separate devices (e.g., radios, televisions, electronic pagers, cellular telephones, and GPS units) required to perform a variety of useful functions (e.g., listening to audio, watching video, receiving messages, engaging in two-way conversations, and finding one's location), some of which may not have been anticipated when the devices were manufactured;

- reconfigure themselves to continue functioning when other elements in a communication system change (e.g., to use CDMA rather than AMPS for cellular telephony, or to use DTV or PAL rather than NTSC for television) and to do so automatically by installing new software instead of requiring manual installation of new hardware;

- respond smoothly to changes in communication channels or user location, thereby continuing to provide service under severe conditions (e.g., multipath effects) or, alternatively, taking advantage of favorable conditions (e.g., the absence of noise) to conserve power or increase throughput; and
- provide more effective transport, as measured by latency, throughput, or accuracy, for different kinds of data (e.g., voice, video, and financial transactions) by incorporating application requirements into network protocols.

As part of the SpectrumWare project (http://nms.lcs.mit.edu/projects/spectrumware) and Oxygen effort (http://oxygen.lcs.mit.edu), we have been developing a flexible hardware and software environment that can adapt to changing communication and computation requirements. SpectrumWare performs real-time signal processing and other computations in portable application-level software. Because it runs on general-purpose processors under commercial operating systems, SpectrumWare benefits from ongoing developments in both hardware and software technology. Because it is software-based, SpectrumWare provides more flexibility than do traditional hardware-based systems for changing or upgrading functionality, for adjusting operating parameters dynamically, and for enabling applications to interact with the physical world. This proposal seeks to continue work on SpectrumWare by building, as part of the Oxygen effort, wireless access points for multimedia networks that scale from one to several buildings. To handle a wide variety of applications in a wide variety of environments, this proposal also seeks to continue development of efficient and flexible algorithms for software-based signal processing together with an application-driven, automated control framework for highly adaptive communication systems.

2. Research approach

The heart of SpectrumWare is an innovative runtime system that differs radically from other systems in at least three important respects, each of which provides additional adaptability.

1) The system uses a statistical model of computation to optimize signal-processing components for the current and/or average case, not for a hypothetical worst case, as do hardware-based systems.
2) The system uses application requirements, rather a fixed set of components operating at fixed sampling rates, to select what data to process and how to process it.
3) The system achieves better performance and provides added utility by using modularized components that fit application needs rather than current DSP fabrication techniques, analytical tools, or layered network models.

This proposal seeks to develop and exploit these aspects of SpectrumWare. More generally, it seeks to increase our understanding of many aspects of future systems and to help illuminate tradeoffs among functionality, computation, power, and communication. Furthermore, it seeks to increase the pace of innovation by making it possible to experiment with and deploy new systems almost entirely in software.

Statistical real-time

Conventional real-time systems (especially for signal processing) assume constant execution times and worst-case inputs. By contrast, SpectrumWare is a *statistical real-time system*, which assumes expected distributions of execution times and inputs. A statistical model is more consistent with the behavior of modern computing systems. For example, high-performance commodity microprocessors use multi-level memory systems, instruction pipelining, and speculative execution to provide superior average-case performance. On the other hand, current high-performance DSP systems forego taking advantage of such features because cache misses, pipeline stalls, and unsuccessful predictions can lead to significant degradations in worst-case performance. In SpectrumWare, the resources and algorithms for each part of a computation are tuned dynamically based upon application needs (e.g., for latency, throughput, or precision requirements), the quality of the incoming signal, and the current priorities of competing computations.

The proposed project will develop analytic techniques for statistical real-time systems. These techniques will enable systems to determine, for example, when to select and when to skip input data in order to meet processing deadlines and/or application requirements. Our experience building a software NTSC receiver shows that precise skipping of input data, which avoids frequent resynchronization, can significantly improve throughput without noticeably degrading picture quality. The proposed project will also develop general signal-processing feedback and control mechanisms. Our experience building a frequency hopping wireless network link shows that feedback to high-level applications about low-level channel conditions facilitates coordinated responses to changes in operating conditions.

Adaptable signal-processing algorithms

Current DSP components generally provide fixed functionality at fixed input/output rates. At times, however, one may want to vary functionality or data rates. For example, one may want a mobile device to transmit at a high rate when close to a stationary device (e.g., by modulating more bits per transmitted symbol to increase throughput), but to degrade gracefully to a less aggressive modulation to maintain good connectivity (e.g., as measured using the signal-to-noise ratio) when further away.

Components can trade off the accuracy of their results against their utilization of resources. For example, a correlation filter can provide efficient detection for signals in the presence of additive white Gaussian noise by using memory to accumulate—and computation to reorder—input samples, so as to process first those samples that contribute most to the detection decision. In the worst case, all samples must be processed and the computation time is the same; in the average case, or when occasional wrong decisions can be tolerated, fewer samples are needed and the computation time is much better.

In conjunction with lazy evaluation and out-of-band control channels, adaptable signal-processing (ASP) algorithms can be used to trade off the behavior of components within a signal-processing system to meet overall processing requirements. An essential requirement for a system that employs ASP is the ability of its components to operate with various amounts of computational resources, so that it can adjust the resources devoted to each component to achieve acceptable, efficient end-to-end performance. For example, a filtering algorithm can adjust output quality by varying the number of input samples it uses, dynamically altering its resource consumption as interference changes. A system experiencing better-than-expected interference conditions can relax filtering requirements and conserve computational resources or use them for other purposes.

Modularization

SpectrumWare provides flexibility in system design by moving the hardware/software boundary closer to the antenna. The proposed project seeks to exploit this flexibility by designing application-appropriate software modules, free from traditional constraints on when and how computations are performed. For example, our experience building software transmitters that pre-compute waveforms shows that modules can substitute preprocessing and inexpensive memory for expensive processing cycles.

One goal for modularization is to separate computation from control. For example, we seek to amortize the cost of expensive, but infrequent control computations over continuous computations performed on successive elements of received data. Modules that perform infrequent control computations help set parameters for modules that perform the continuous computation.

Another goal is to provide applications with greater control over the modules used by signal-processing systems in which they are embedded. Consider, for example, how a digital receiver might deal with errors when it processes 99.5% of the samples within 10 μsec, but takes 10 milliseconds in the worst case. If the receiver is part of a system receiving an audio stream, it may be best to use a module that buffers samples and relies on the fact that, on average, there is plenty of time to process all samples. On the other hand, if the receiver is embedded in a network interface, it may be best to use a module that stops processing after 10 μsec and randomly guesses whether the bit is zero or one. This module will introduce bit errors for 0.25% of the samples will be wrong. However, since there can also be channel-induced errors, higher level modules must be prepared to deal with errors, and this increase is not likely to have a material effect upon overall performance.

A related goal is to provide applications with the ability to download and install new signal processing modules. Such an ability would be very useful in building self-organizing scalable networks. For example, protocols could use beacons for triangulation to establish the basic topology of a network, and they could use a variety of “probes” during periodic channel characterizations to choose the most effective modulation techniques, downloading new modules when necessary to deal with different bandwidth requirements, noise environments, or network topologies.

Research agenda

Work on SpectrumWare will continue during the course of the multi-year Oxygen effort, with the goal of providing a prototype system of the kind described below in Section 3. This proposal requests support during the coming year for SpectrumWare activities directed towards that goal. Subsequent proposals may request additional support for work in subsequent years.

During the coming year, we plan to develop algorithms and software, of the kinds described above, which enable us to build adaptable networks of wired and wireless devices. In particular, we plan to develop:

1) An application program interface (API) that enables networked devices to trade off transmission rate against power consumption. The API will provide mechanisms for sensing signal-to-noise ratios and for adjusting transmission to reflect application requirements.
2) Protocols for implementing "radioactive networks" that enable network devices to respond to current operating conditions and application requirements by downloading and installing new software modules from a server-based library. Such software modules will enable the devices to provide new functionality (e.g., FM radio reception) or to change their transmission format.
3) Novel physical layer algorithms optimized to support digital communication over dynamically changing channels.

In subsequent years, we plan to provide additional features:

4) Mechanisms that enable mobile users to maintain network connectivity while moving from one location to another. These mechanisms will sense the condition of multiple

channels (e.g., providing local area 2.4Ghz connectivity or wider area 440Mhz connectivity) and provide for a smooth vertical handoff from one channel to another.

5) Multiband RF front ends, together signal processing software, that enable devices to function both as part of wireless local or wider area networks and also as receivers for FM radio or television broadcasts. Appropriate signal-processing software (e.g., for the US and Japan versions of NTSC) would be downloaded as needed from server-based libraries.

3. Prototype system and applications

A multi-year goal of this project is to construct a prototype wireless network that enables users to maintain continuous and ubiquitous connectivity to multimedia applications as they move about in a building at MIT or across the campus. The network will provide both handheld and stationary access points by integrating commercial off-the-shelf hardware with custom hardware for multiple antennae as well as with user interface devices such as speakers, microphones, video cameras, and displays. Network devices will contain custom-built I/O interfaces for transferring sampled wireless, video, and sound data to and from the processor. The wireless interface will be switchable over a range of RF front-ends and provide support for continuous data transfers, minimizing the effects of jitter and system latencies. Several RF front ends will enable trial applications involving software radios/TVs as well as highly configurable data networks.

The physical layer of the network will use SpectrumWare to adjust such factors as power consumption, modulation techniques, bandwidth, and error correction. Handheld devices will communicate with each other and with the stationary devices using infrared for desktop connectivity, high radio frequencies for in-building connectivity, and lower radio frequencies for campus-wide connectivity. Providing smooth vertical hand-offs between different modes of communication presents one set of challenges. Handling high data rates (e.g., 20 Mbps) for local connectivity presents another.

We expect that the experience gained by using actual wireless channels will raise new issues and drive further research.