Second-Order Challenges in Streaming Systems

Dr. John M. Chapin
Vanu, Inc.
1 Porter Square, Suite 18
Cambridge, MA 02140 USA
jchapin@vanu.com
http://www.vanu.com
Vanu, Inc. Overview

Founded 1998
MIT LCS spin-off

Supply signal processing software to radio manufacturers
“The software in software radio”

Focus on software engineering and portability

GSM base station

Monitoring system
Public safety terminal

Copyright © 2003, Vanu, Inc.
Some definitions

**Waveform**
An RF communications standard. Includes all layers.

**Software Defined Radio (SDR)**
An RF communications device where the waveform supported can be changed through a software load.

**X is portable to Y**
The cost of getting X running on Y is a small fraction of X’s original development cost.
Second-order challenges

Many good, useful ideas for streaming languages/systems have already been described in this workshop.

To support applications like those built at Vanu, Inc., there are additional challenges that must be addressed:

- Dynamic processing graphs
- Large-scale processing graphs
- Variable cost kernels
- Portable source representation
- Hybrid applications
Dynamic processing graphs

GSM waveform implementation
many options negotiated during call setup
for efficiency, have specialized modules for different options
after options known, instantiate specialized processing pipeline
and connect into existing graph
delete this pipeline during call teardown

Graph surgery must occur in 10s of milliseconds without
processing interruptions (elsewhere in graph) > ~1 msec.
Large-scale processing graphs

- Simple analog FM waveform: 5 kernels
- AMPS channel: 30 kernels
- Full AMPS basestation: > 1500 kernels

This was with coarse kernels
Want finer granularity to exploit compiler strip-mining, etc.

Recommend compiler algorithms designed for 500 kernels in tightly optimized unit, 100k kernels in full application.
Variable-cost kernels

Traditional DSP algorithms have constant cost/sample. Much better average-case performance possible with variable cost/sample.

- Convolutional decoding based on Dijkstra shortest-paths search
- CCK decoding using initial (+1, -1, X) discriminator

Recommend scheduling based on bounded variability rather than invariant costs.
Want common source representation for DSP, FPGA, complex SOC (e.g. Vertex II), etc.

- Complex, expensive signal processing software
- Different architectures used on different platforms

OK if output code is not optimal (but must be Pretty Good™)

Big US DOD SDR program pushing hard for this

“A major goal of the JTRS program is to use these [independently procured] waveforms across the family of JTRS radios regardless of manufacturer.”
Most real-world applications are hybrid
streaming code for signal processing
non-streaming code for control, networking, human interface, ...

Need portable, efficient constructs for
configuring/controlling/monitoring
moving high speed data streams back and forth
moving events and parameters back and forth

Must be able to debug as an integrated single application.
How RDL handles hybrid applications

**C++ code**

```cpp
class MyMonitor :
  public SigProcExt {
    void SigProcEvent();
  }

SPS.Register(
  "Monitor",
  new MyMonitor()
);
```
Summary

This talk identified 5 challenges in the full streaming-system problem that go beyond the obvious initial challenges.

Dynamic processing graphs
Large-scale processing graphs
Variable cost kernels
Portable source representation
Hybrid applications