The Imperative for High-Performance Audio Computing

LAC2009

John ffitch    Richard Dobson    Russell Bradford

University of Bath

LAC Parma, April 2009
Take note of what James Moorer said:

This is compute power far beyond what even the most starry-eyed fortune-teller could have imagined! It will change the very nature of what audio is, and what audio engineers do, since it changes what is possible at a fundamental level.
Introduction

Take note of what James Moorer said:

This is compute power far beyond what even the most starry-eyed fortune-teller could have imagined! It will change the very nature of what audio is, and what audio engineers do, since it changes what is possible at a fundamental level.
HiPAC

We continue to propose HiPAC — High-Performance Audio Computing — as an important new domain of study that explores the potential for new advanced processor architectures to transform the current landscape of audio synthesis, processing and music composition.

A key aspect of HiPAC from the point of view of the computer musician is that within a relatively short timescale, we can expect to see these technologies in consumer-grade hardware.
We continue to propose HiPAC — **High-Performance Audio Computing** — as an important new domain of study that explores the potential for new advanced processor architectures to transform the current landscape of audio synthesis, processing and music composition.

A key aspect of HiPAC from the point of view of the computer musician is that within a relatively short timescale, we can expect to see these technologies in consumer-grade hardware.
The Technology – History

We used to have “supercomputing” – Cray1, Connection Machine, MasPar. We can distinguish Single-Instruction-Multiple-Data (SIMD) machines and Multiple-Instruction-Multiple-Data (MIMD).

In the 1980s and 1990s one can find many experiments and suggestions. These were made obsolete by the brute power of faster machines.

Now the speed increases are ending, and we need to return to being clever.
We used to have “supercomputing” – Cray1, Connection Machine, MasPar. We can distinguish Single-Instruction-Multiple-Data (SIMD) machines and Multiple-Instruction-Multiple-Data (MIMD).

In the 1980s and 1990s one can find many experiments and suggestions. These were made obsolete by the brute power of faster machines.

Now the speed increases are ending, and we need to return to being clever.
The Technology – History

We used to have “supercomputing” – Cray1, Connection Machine, MasPar. We can distinguish Single-Instruction-Multiple-Data (SIMD) machines and Multiple-Instruction-Multiple-Data (MIMD).

In the 1980s and 1990s one can find many experiments and suggestions. These were made obsolete by the brute power of faster machines.

Now the speed increases are ending, and we need to return to being clever.
Modern SIMD fine-grained parallel architectures are associated today firstly with the vector extensions to standard CPUs.

And with the graphics accelerator cards now essential to all consumer workstations (especially where high performance in games is required).
Modern SIMD fine-grained parallel architectures are associated today firstly with the vector extensions to standard CPUs.

And with the graphics accelerator cards now essential to all consumer workstations (especially where high performance in games is required).
As it gets harder to increase the speed of a processor the manufacturers are providing multiple CPU cores on a chip. Normally 2 or 4 cores, 8 promised soon and .....
Intel Polaris 80-core
As it gets harder to increase the speed of a processor the manufacturers are providing multiple CPU cores on a chip. Normally 2 or 4 cores, 8 promised soon and the Intel Polaris with 80, with claim performance up to 1 Teraflop, but you need to wait about 10 years.

Also SIMD-style accelerator systems designed to work in conjunction with a host computer. We have experience of Clearspeed.
As it gets harder to increase the speed of a processor the manufacturers are providing multiple CPU cores on a chip. Normally 2 or 4 cores, 8 promised soon and the Intel Polaris with 80, with claim performance up to 1 Teraflop, but you need to wait about 10 years.

Also SIMD-style accelerator systems designed to work in conjunction with a host computer. We have experience of Clearspeed.
From Supercomputers to the Desktop

As it gets harder to increase the speed of a processor the manufacturers are providing multiple CPU cores on a chip. Normally 2 or 4 cores, 8 promised soon and the Intel Polaris with 80, with claim performance up to 1 Teraflop, but you need to wait about 10 years.

Also SIMD-style accelerator systems designed to work in conjunction with a host computer. We have experience of Clearspeed.
ClearSpeed™ CLEARSPEED ADVANCE™ X620
As it gets harder to increase the speed of a processor the manufacturers are providing multiple CPU cores on a chip. Normally 2 or 4 cores, 8 promised soon and the Intel Polaris with 80, with claim performance up to 1 Teraflop, but you need to wait about 10 years.

Also SIMD-style accelerator systems designed to work in conjunction with a host computer. We have experience of Clearspeed.

And the so-called GPGPU.....
As it gets harder to increase the speed of a processor the manufacturers are providing multiple CPU cores on a chip. Normally 2 or 4 cores, 8 promised soon and the Intel Polaris with 80, with claim performance up to 1 Teraflop, but you need to wait about 10 years.

Also SIMD-style accelerator systems designed to work in conjunction with a host computer. We have experience of Clearspeed.

And the so-called GPGPU.....
Earlier Attempts

In Audio:
- Transputer Array in Durham: 170 processors
- Midas in York: dataflow through SGI net

In Computer Science:
- Bath Parallel LISP Machine
- Timewarp
- ...etc
Earlier Attempts

In Audio:
- Transputer Array in Durham: 170 processors
- Midas in York: dataflow through SGI net

In Computer Science:
- Bath Parallel LISP Machine
- Timewarp
- ...etc
Earlier Attempts

In Audio:
- Transputer Array in Durham: 170 processors
- Midas in York: dataflow through SGI net

In Computer Science:
- Bath Parallel LISP Machine
- Timewarp
- ...etc
Earlier Attempts

In Audio:
- Transputer Array in Durham: 170 processors
- Midas in York: dataflow through SGI net

In Computer Science:
- Bath Parallel LISP Machine
- Timewarp
- ...etc
We have been working with Clearspeed plc to use their accelerator in interesting ways. They have a version of C extended with poly and mono variables. (Similar to MasPar system)

and there are others.

But do not forget Amdahl’s law.....
Software for Vector processing

We have been working with Clearspeed plc to use their accelerator in interesting ways. They have a version of C extended with \textit{poly} and \textit{mono} variables. (Similar to MasPar system)

and there are others.

But do not forget Amdahl’s law.....
Software for Vector processing

We have been working with Clearspeed plc to use their accelerator in interesting ways. They have a version of C extended with poly and mono variables. (Similar to MasPar system)

and there are others.

But do not forget Amdahl’s law.....
Software for Vector processing

We have been working with Clearspeed plc to use their accelerator in interesting ways. They have a version of C extended with poly and mono variables. (Similar to MasPar system)

and there are others.

But do not forget Amdahl’s law.....
Amdahl’s Law

The relationship between sequential and parallel computation is summarised in Amdahl’s law, which is stated as:

\[
\frac{1}{S + \frac{P}{N}}
\]

where \( S \) is the fraction of serial computation, \( P = 1 - S \) is the amount of parallelisable computation and \( N \) is number of processors.
The primary defining characteristics of a HiPAC dsp process:

- use of highly parallel fine-grained architectures (e.g. following the SIMD model), though we do not exclude more “conventional” multi-core computation
- real-time performance or better
- implies low latency
- ideal and “no-compromise” forms of algorithms
- new processes, and hence new effects and sounds, not simply “more of the same” - whether more reverbs or more voices.
A HiPAC case study - the Parallel Execution of Csound

We have proposed the The Sliding Phase Vocoder (SPV) as a canonical example of a HiPAC process. But now I present here a new (and different) application:

MultiCore Csound

The challenge is to make sensible use of a multicore processor, to provide more processing in real-time.
A HiPAC case study - the Parallel Execution of Csound

We have proposed the The Sliding Phase Vocoder (SPV) as a canonical example of a HiPAC process. But now I present here a new (and different) application:

**MultiCore Csound**

The challenge is to make sensible use of a multicore processor, to provide more processing in real-time.
A HiPAC case study - the Parallel Execution of Csound

We have proposed the The Sliding Phase Vocoder (SPV) as a canonical example of a HiPAC process. But now I present here a new (and different) application: 

**MultiCore Csound**

The challenge is to make sensible use of a multicore processor, to provide more processing in real-time.
This is work in progress, and not in the written paper. A weak form may appear at ICMC2009. We can compute different instances of instruments in parallel, as long as they do not use the same resource and as long as it is worth it.
A HiPAC case study - the Parallel Execution of Csound

This is work in progress, and not in the written paper. A weak form may appear at ICMC2009. We can compute different instances of instruments in parallel, *as long as they do not use the same resource* and *as long as it is worth it.*
This is work in progress, and not in the written paper. A weak form may appear at ICMC2009.

We can compute different instances of instruments in parallel, as long as they do not use the same resource and as long as it is worth it.
A HiPAC case study - the Parallel Execution of Csound

Is it Worthwhile?

We are creating a database of instruction counts for each opcode, parameterised by initialisation, instructions/sample and instructions/control.

There is no point in parallel execution if the overhead of threads is comparable with the cost.
A HiPAC case study - the Parallel Execution of Csound

<table>
<thead>
<tr>
<th>Opcode</th>
<th>init</th>
<th>Audio</th>
<th>Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>table.a</td>
<td>93</td>
<td>23.063</td>
<td>43.998</td>
</tr>
<tr>
<td>table.k</td>
<td>93</td>
<td>0</td>
<td>45</td>
</tr>
<tr>
<td>butterlp</td>
<td>9</td>
<td>29.005</td>
<td>5.478</td>
</tr>
<tr>
<td>butterhi</td>
<td>19</td>
<td>30.000</td>
<td>35</td>
</tr>
<tr>
<td>butterbp</td>
<td>20</td>
<td>30</td>
<td>71</td>
</tr>
<tr>
<td>oscil.ka</td>
<td>69</td>
<td>17</td>
<td>46</td>
</tr>
<tr>
<td>bilbar</td>
<td>371.5</td>
<td>1856.028</td>
<td>86</td>
</tr>
<tr>
<td>ags</td>
<td>497</td>
<td>917.921</td>
<td>79475.155</td>
</tr>
</tbody>
</table>
A HiPAC case study - the Parallel Execution of Csound

In what ways can brute distribution be wrong?

- Clashing use of global variable
- Adding into the output bus
- Other global structures
A HiPAC case study - the Parallel Execution of Csound

In what ways can brute distribution be wrong?

- Clashing use of global variable
- Adding into the output bus
- Other global structures
A HiPAC case study - the Parallel Execution of Csound

In what ways can brute distribution be wrong?

- Clashing use of global variable
- Adding into the output bus
- Other global structures
In what ways can brute distribution be wrong?

- Clashing use of global variable
- Adding into the output bus
- Other global structures
Compiler technology allows us to identify the points of contention.

Use of spin locks for output bus and others

Each k-cycle looks at a DAG of dependency, and schedules opcodes to maintain semantics.
Compiler technology allows us to identify the points of contention.

Use of spin locks for output bus and others

Each k-cycle looks at a DAG of dependency, and schedules opcodes to maintain semantics.
Compiler technology allows us to identify the points of contention.

Use of spin locks for output bus and others

Each k-cycle looks at a DAG of dependency, and schedules opcodes to maintain semantics.
A HiPAC case study - the Parallel Execution of Csound

Consider the fragment of Csound:

```
instr 1
    a1 oscil p4, p5, 1
    out a1
endin
instr 2
    gk oscil p4, p5, 1
endin
instr 3
    a1 oscil gk, p5, 1
    out a1
endin
```
A HiPAC case study - the Parallel Execution of Csound
A HiPAC case study - the Parallel Execution of Csound

Current state is that it runs, not totally integrated. The cost database not used yet.
A HiPAC case study - the Parallel Execution of Csound

Some performance figures running on a dual-core machine:

Best so far is **Electric Priest** that goes from 53s to 34s

**Xanadu** is less good at only 10% gain.

**Trapped in Convert** only shows 16% gain.

But they are gains!
A HiPAC case study - the Parallel Execution of Csound

Some performance figures running on a dual-core machine:

Best so far is **Electric Priest** that goes from 53s to 34s.

**Xanadu** is less good at only 10% gain.

**Trapped in Convert** only shows 16% gain.

But they are gains!
Some performance figures running on a dual-core machine:
Best so far is **Electric Priest** that goes from 53s to 34s
**Xanadu** is less good at only 10% gain.
**Trapped in Convert** only shows 16% gain.
But they are gains!
A HiPAC case study - the Parallel Execution of Csound

Some performance figures running on a dual-core machine:
Best so far is Electric Priest that goes from 53s to 34s
Xanadu is less good at only 10% gain.
Trapped in Convert only shows 16% gain.

But they are gains!
A HiPAC case study - the Parallel Execution of Csound

Some performance figures running on a dual-core machine:

Best so far is **Electric Priest** that goes from 53s to 34s

**Xanadu** is less good at only 10% gain.

**Trapped in Convert** only shows 16% gain.

But they are gains!
Parallel processing is coming apace, and if we do not embrace it in all its forms we will again be the poor relations. This hardware will soon be cheap enough. The “techies” must grasp it.

Thanks to Codemist Ltd and Clearspeed plc. This work is unsupported by public agencies.

The parallel Csound is the work of Chris Wilson for his BSc in Computer Science under supervision of John ffitch.
Conclusions

Parallel processing is coming apace, and if we do not embrace it in all its forms we will again be the poor relations. This hardware will soon be cheap enough. The “techies” must grasp it.

Thanks to Codemist Ltd and Clearspeed plc. This work is unsupported by public agencies.

The parallel Csound is the work of Chris Wilson for his BSc in Computer Science under supervision of John ffitch