Acceleration Targets:  
A Study of Popular Benchmark Suites  
Lisa Wu and Martha A. Kim  
June 10, 2012
The story starts like this:

Princess Ruruna and her helper Cain have a problem: To face dark silicon head on, they want to find applications that have acceleration potential. But what can they do to tackle the problem?

Let’s see what Tico the fairy has to say...

---

**THIS IS A VERY ELEMENTARY QUESTION...**

**But what should we accelerate?**
Let’s start by looking at some popular benchmark suites.
Let’s start by looking at some popular benchmark suites

1

* Do the benchmarks exhibit any common functionality?
Let's start by looking at some popular benchmark suites

1.

* Do the benchmarks exhibit any common functionality?
* If so, is it at or above the function level?
I will profile SPEC2006 and see if I can answer this question...
I will profile SPEC2006 and see if I can answer this question...

If the hottest function runs lightening fast, how much faster would the suite be?
I will profile SPEC2006 and see if I can answer this question...

![Graph showing the relationship between unique accelerators and max speedup of suite for SPEC2006. The graph plots max speedup on a logarithmic scale against unique accelerators on a linear scale, with a curve that indicates a significant increase in speedup as accelerators increase.]
I will profile SPEC2006 and see if I can answer this question...
I will profile SPEC2006 and see if I can answer this question...
I will profile SPEC2006 and see if I can answer this question...

If the hottest function runs lightning fast, how much faster would the suite be?

To get a 10X speedup, we need to accelerate over 189 unique functions!
Hmm...

What if we accelerated a bigger target?
Hmm...

What if we accelerated a bigger target?
Hmm...

What if we accelerated a bigger target?
Hmm...

What if we accelerated a bigger target?
Good! It only takes 21!

Oh wait...we need to accelerate 21 different applications for a 12x speedup?!
It seems that SPEC2006 cannot be accelerated easily...

How about other benchmark suites?
What about benchmark suites that are not written in C?

It seems that SPEC2006 cannot be accelerated easily...

How about other benchmark suites?
2

* What about benchmark suites that are not written in C?

* What impact does the language or programming environment have on acceleration potential?

It seems that SPEC2006 cannot be accelerated easily...

How about other benchmark suites?
Each source language provides a slightly different set of potential acceleration targets.

<table>
<thead>
<tr>
<th>Benchmark Suite</th>
<th>Granularity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>fine</td>
</tr>
<tr>
<td>SPEC2006</td>
<td>function</td>
</tr>
<tr>
<td>SPECJVM</td>
<td>method</td>
</tr>
<tr>
<td>DACAPO</td>
<td>method</td>
</tr>
<tr>
<td>UNLADEN-SWALLOW</td>
<td>function</td>
</tr>
</tbody>
</table>
Java? Go for it!
Java? Go for it!

DACAPRO

Max Speedup of Suite

Unique Accelerators

- method
- class
- package

Sunday, July 28, 2013
org.apache.axis.transport.http.HTTPSender.readHeadersFromSocket()
org.apache.axis.transport.http.HTTPSender

DACAPRO

Max Speedup of Suite

Unique Accelerators

method
class
package

org.apache.axis.transport.http.HTTPSender
org.apache.axis.transport.http

DACAPO

Max Speedup of Suite

Unique Accelerators

method
class
package
Okay... it takes 78 methods, 59 classes, or 33 packages to get a 10X speedup on Dacapo suite.
Cain, would you help me with other Java benchmark suites?

Of course!

How about SpecJVM?
Cain, would you help me with other Java benchmark suites?

Of course!

How about SpecJVM?
Cain, would you help me with other Java benchmark suites? Of course!

How about SpecJVM? 23 methods
Cain, would you help me with other Java benchmark suites?

How about SpecJVM?

23 methods
18 classes

Of course!
Cain, would you help me with other Java benchmark suites?

Of course!

How about SpecJVM?

23 methods

18 classes

or 14 packages. SpecJVM is better than Dacapo!
What have we concluded from our study today?
What have we concluded from our study today?

* Unstructured C code can only be accelerated in swaths of highly application-specific codes.
What have we concluded from our study today?

Unstructured C code can only be accelerated in swaths of highly application-specific codes.

Eco-cores

and DySER had the same conclusion

C-cores

*Unstructured C code can only be accelerated in swaths of highly application-specific codes.*
What have we concluded from our study today?

- Unstructured C code can only be accelerated in swaths of highly application-specific codes.
- Java has potential to use classes as targets. Is accelerating fifty unique classes worth a 10X performance gain?

Eco-cores

and DySER had the same conclusion

C-cores
What have we concluded from our study today?

* Unstructured C code can only be accelerated in swaths of highly application-specific codes.

* Java has potential to use classes as targets. Is accelerating fifty unique classes worth a 10X performance gain?

* Filling dark silicon will require tens to hundreds of specialized accelerators.

and DySER had the same conclusion.
We have some open questions...

Looks like we have a lot more research to do!
What are the benchmarks we should use to evaluate potential accelerators?

We have some open questions...

Looks like we have a lot more research to do!
What are the benchmarks we should use to evaluate potential accelerators?

The infrastructure associated with accelerators are increasingly important!

Looks like we have a lot more research to do!
What are the benchmarks we should use to evaluate potential accelerators?

The infrastructure associated with accelerators are increasingly important!

What happens when we factor in actual costs?

Looks like we have a lot more research to do!
Good work today! Hope you learned something about accelerating targets!

Questions?
fin