

This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government.

# 30 Years of HPC from Gigafllops to Exafllops: No science left behind or thinning the herd?

SAND2018-4611C

Steve Plimpton  
Sandia National Laboratories

2018 Salishan Conference on High-Speed Computing



Sandia National Laboratories is a multi-mission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security



Administration is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract number DE-AC05-04OR21400.

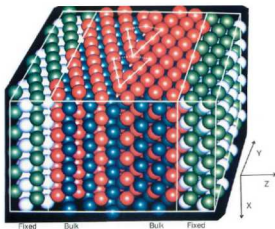
# Impact of advancing HPC on particle simulations

- Many methods/models are  $\sim O(N)$  cost in particle count
- Also scale as  $\sim O(N/P)$  in parallel, for large enough  $N/P$
- 1000x machine  $\Rightarrow$  1000x more particles or time or combo

# Impact of advancing HPC on particle simulations

- Many methods/models are  $\sim O(N)$  cost in particle count
- Also scale as  $\sim O(N/P)$  in parallel, for large enough  $N/P$
- 1000x machine  $\Rightarrow$  1000x more particles or time or combo

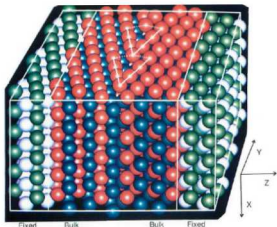
**30 yrs ago:**  
my thesis  
1000 atoms  
50K steps



# Impact of advancing HPC on particle simulations

- Many methods/models are  $\sim O(N)$  cost in particle count
- Also scale as  $\sim O(N/P)$  in parallel, for large enough  $N/P$
- 1000x machine  $\Rightarrow$  1000x more particles or time or combo

**30 yrs ago:**  
my thesis  
1000 atoms  
50K steps

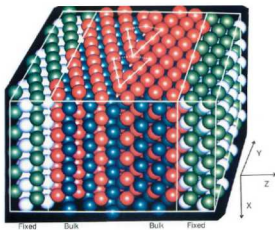


**Today:**  
V Bulatov,  
et al (LLNL)  
2.1B atoms  
460M steps

# Impact of advancing HPC on particle simulations

- Many methods/models are  $\sim O(N)$  cost in particle count
- Also scale as  $\sim O(N/P)$  in parallel, for large enough  $N/P$
- 1000x machine  $\Rightarrow$  1000x more particles or time or combo

**30 yrs ago:**  
my thesis  
1000 atoms  
50K steps



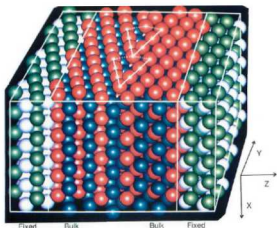
**Today:**  
V Bulatov,  
et al (LLNL)  
2.1B atoms  
460M steps

- Linpack: 1 BG/Q core / 1 Cray YMP proc = **41x** !!

# Impact of advancing HPC on particle simulations

- Many methods/models are  $\sim O(N)$  cost in particle count
- Also scale as  $\sim O(N/P)$  in parallel, for large enough  $N/P$
- 1000x machine  $\Rightarrow$  1000x more particles or time or combo

**30 yrs ago:**  
my thesis  
1000 atoms  
50K steps



**Today:**  
V Bulatov,  
et al (LLNL)  
2.1B atoms  
460M steps

- Linpack: 1 BG/Q core / 1 Cray YMP proc = **41x** !!
- Cray YMP proc  $\Rightarrow$  third of BG/Q Sequoia  $\Rightarrow$  **21M** faster
- MD atom-steps/s  $\Rightarrow$  **8.5M** faster

# Newspapers and ukuleles

Ben Bagdikian (1920-2016): Pulitzer prize journalist, professor



# Newspapers and ukuleles

Ben Bagdikian (1920-2016): Pulitzer prize journalist, professor



“Trying to be a first-rate reporter on the average American newspaper is like trying to play Bach’s *St. Matthew’s Passion* on a ukulele.”

## HPC and ukuleles

“Trying to do first-rate science on the average petascale (soon exascale) machine is like trying to play Bach’s *St. Matthew’s Passion* on a million ukuleles.”



What I'm **NOT** trying to say

# What I'm **NOT** trying to say

- HPC is **underfunded** (like newspapers)
  - this is one of best times in my memory for scientific HPC
  - lots of buzz about exascale, machine learning, beyond Moore

# What I'm **NOT** trying to say

- HPC is **underfunded** (like newspapers)
  - this is one of best times in my memory for scientific HPC
  - lots of buzz about exascale, machine learning, beyond Moore
- Chip designers or machine architects are creating underpowered or **deficient hardware** (ukuleles)
  - current & future hardware is incredibly sophisticated
  - in the right hands, ukuleles can be amazing instruments
  - google **Jake Shimabukuro**

# What I'm **NOT** trying to say

- HPC is **underfunded** (like newspapers)
  - this is one of best times in my memory for scientific HPC
  - lots of buzz about exascale, machine learning, beyond Moore
- Chip designers or machine architects are creating underpowered or **deficient hardware** (ukuleles)
  - current & future hardware is incredibly sophisticated
  - in the right hands, ukuleles can be amazing instruments
  - google **Jake Shimabukuro**
- No one is doing **first-rate science** on petascale machines
  - Gordon Bell prizes at SC conference
  - INCITE allocations by DOE

So what am I trying to say?

# So what am I trying to say?

- We're making machines that are **increasingly complex**
  - harder to program
  - harder to achieve high performance on
  - optimized for a narrower portion of computational science

# So what am I trying to say?

- We're making machines that are **increasingly complex**
  - harder to program
  - harder to achieve high performance on
  - optimized for a narrower portion of computational science
- Software developers (apps, libs, system) spend a **lot of time & energy** figuring out how to optimize for, and use these machines effectively
  - we suffer from **Ukulele Syndrome**

# So what am I trying to say?

- We're making machines that are **increasingly complex**
  - harder to program
  - harder to achieve high performance on
  - optimized for a narrower portion of computational science
- Software developers (apps, libs, system) spend a **lot of time & energy** figuring out how to optimize for, and use these machines effectively
  - we suffer from **Ukulele Syndrome**
- These two realities have a cost:
  - may be optimal for flops, but is **sub-optimal for science**
  - lots of time/\$\$ not spent on science
  - leaving many apps and scientists on the sidelines
  - there **ain't that many** Jake Shimabukuros

Do you suffer from Ukulele Syndrome?

# Different than Ukulele Acquisition Syndrome (UAS)

- Google “ukulele syndrome”, will find real thing called UAS



- “What is **UAS**? It's that insatiable desire to acquire the next ukulele for your need, to find that perfect ukulele for that favorite song, sound, or moment.”

# Different than Ukulele Acquisition Syndrome (UAS)

- Google “ukulele syndrome”, will find real thing called UAS



- “What is **UAS**? It’s that insatiable desire to acquire the next ukulele for your need, to find that perfect ukulele for that favorite song, sound, or moment.”
- People **addicted** to buying more and more hardware
- Many of you and your orgs have serious cases of HAS

# Different than Ukulele Acquisition Syndrome (UAS)

- Google “ukulele syndrome”, will find real thing called UAS



- “What is **UAS**? It’s that insatiable desire to acquire the next ukulele for your need, to find that perfect ukulele for that favorite song, sound, or moment.”
- People **addicted** to buying more and more hardware
- Many of you and your orgs have serious cases of HAS
- Sorry, there is **no known cure** for UAS or HAS

# Five symptoms of Ukulele Syndrome

When you get a new, faster machine ...

# Five symptoms of Ukulele Syndrome

When you get a new, faster machine ...

- Spend **time and \$\$**
  - re-writing your kernels or code for Nth time
  - writing papers for Mth time on how code adapted/performs

# Five symptoms of Ukulele Syndrome

When you get a new, faster machine ...

- Spend **time and \$\$**
  - re-writing your kernels or code for Nth time
  - writing papers for Mth time on how code adapted/performs
- Use a **brute force algorithm** instead of more clever one
  - b/c it runs faster or more scalably, or at higher flop rate

# Five symptoms of Ukulele Syndrome

When you get a new, faster machine ...

- Spend **time and \$\$**
  - re-writing your kernels or code for Nth time
  - writing papers for Mth time on how code adapted/performs
- Use a **brute force algorithm** instead of more clever one
  - b/c it runs faster or more scalably, or at higher flop rate
- See your **code performance decrease**:  
relative to PUP = peak ukulele performance

# Five symptoms of Ukulele Syndrome

When you get a new, faster machine ...

- Spend **time and \$\$**
  - re-writing your kernels or code for Nth time
  - writing papers for Mth time on how code adapted/performs
- Use a **brute force algorithm** instead of more clever one
  - b/c it runs faster or more scalably, or at higher flop rate
- See your **code performance decrease**:  
relative to PUP = peak ukulele performance
- Run your code on **smaller fraction** of new machine:
  - b/c it doesn't scale to more nodes
  - b/c running bigger problems for same short time isn't useful

# Five symptoms of Ukulele Syndrome

When you get a new, faster machine ...

- Spend **time and \$\$**
  - re-writing your kernels or code for Nth time
  - writing papers for Mth time on how code adapted/performs
- Use a **brute force algorithm** instead of more clever one
  - b/c it runs faster or more scalably, or at higher flop rate
- See your **code performance decrease**:  
relative to PUP = peak ukulele performance
- Run your code on **smaller fraction** of new machine:
  - b/c it doesn't scale to more nodes
  - b/c running bigger problems for same short time isn't useful
- Write **allocation proposals** for computer time that:
  - jump thru hoops to show code can use 50%+ of the machine
  - run lots of small jobs (together) to boost machine usage

# What is the impact on science, mission or otherwise?

- **Opportunity cost** for continually adapting apps to new machines
  - doesn't advance the models
  - nor the numerical methods & algorithms
  - nor the science (at least directly)

# What is the impact on science, mission or otherwise?

- **Opportunity cost** for continually adapting apps to new machines
  - doesn't advance the models
  - nor the numerical methods & algorithms
  - nor the science (at least directly)
- Deploying computers that are **hard to use and code for**:
  - shrinks the number of apps that can use it
  - we do less multiphysics, less multiscale
  - shrinks the pool of people that can program them
  - **casual HPC users** are not Jake Shimabukuro, but they are often **better scientists**
  - if make machines hard for them to code & use, we lose their science contributions

# Why you might be singing the Ukulele Blues

**Balance ratios** on past, present, future HPC platforms  
Thanks to Si Hammond (Sandia) for this data!

- **Local balance** = flops to pay for on-node word (8 bytes)
- **Remote balance** = flops to pay for off-node word

# Why you might be singing the Ukulele Blues

**Balance ratios** on past, present, future HPC platforms  
Thanks to Si Hammond (Sandia) for this data!

- **Local balance** = flops to pay for on-node word (8 bytes)
- **Remote balance** = flops to pay for off-node word

Year	Machine	Linpack	Flops/ local	Flops/ remote
1988	Cray YMP	2.1 Gflops	0.52	0.52
1997	ASCI Red (SNL)	1.6 Tflops	8.3	20
2011	RoadRunner (LANL)	1.0 Pflops	6.7	170

## Current blues

- **Local balance** = flops to pay for on-node word (8 bytes)
- **Remote balance** = flops to pay for off-node word

Year	Machine	Linpack	Flops/ local	Flops/ remote
1988	Cray YMP	2.1 Gflops	0.52	0.52
1997	ASCI Red (SNL)	1.6 Tflops	8.3	20
2011	RoadRunner (LANL)	1.0 Pflops	6.7	170
2012	Sequoia (LLNL)	17 Pflops	32	160
2013	Titan (ORNL)	18 Pflops	29	490

# Asian blues

- **Local balance** = flops to pay for on-node word (8 bytes)
- **Remote balance** = flops to pay for off-node word

Year	Machine	Linpack	Flops/ local	Flops/ remote
1988	Cray YMP	2.1 Gflops	0.52	0.52
1997	ASCI Red (SNL)	1.6 Tflops	8.3	20
2011	RoadRunner (LANL)	1.0 Pflops	6.7	170
2012	Sequoia (LLNL)	17 Pflops	32	160
2013	Titan (ORNL)	18 Pflops	29	490
2011	K-Computer (Japan)	11 Pflops	15	95
2013	Tianhe-2 (China)	34 Pflops	22	2100
2016	Sunway TaihuLight (China)	93 Pflops	130	1500

# Exascale blues

Year	Machine	Linpack	Flops/ local	Flops/ remote
1988	Cray YMP	2.1 Gflops	0.52	0.52
1997	ASCI Red (SNL)	1.6 Tflops	8.3	20
2011	RoadRunner (LANL)	1.0 Pflops	6.7	170
2012	Sequoia (LLNL)	17 Pflops	32	160
2013	Titan (ORNL)	18 Pflops	29	490
2011	K-Computer (Japan)	11 Pflops	15	95
2013	Tianhe-2 (China)	34 Pflops	22	2100
2016	Sunway TaihuLight (China)	93 Pflops	130	1500
~2021	TBD Door #1	1.0 Eflops	80	3200
~2021	TBD Door #2	1.0 Eflops	300	10000

## Commentary on the blues

- **Congratulations!** billion X speed-up in 30 years!

## Commentary on the blues

- **Congratulations!** billion X speed-up in 30 years!
- Cray YMP main memory was like cache memory today
  - small and fast

# Commentary on the blues

- **Congratulations!** billion X speed-up in 30 years!
- Cray YMP main memory was like cache memory today
  - small and fast
- Growing imbalance ratios mean:
  - fewer codes achieve high single-node performance
  - fewer codes achieve good scalability

## Commentary on the blues

- **Congratulations!** billion X speed-up in 30 years!
- Cray YMP main memory was like cache memory today
  - small and fast
- Growing imbalance ratios mean:
  - fewer codes achieve high single-node performance
  - fewer codes achieve good scalability
- Bottom line: we're **selecting** for certain kinds of apps

# Growing imbalance is good news for particle apps

- Particle apps:
  - lots of flops per memory access (expensive models)
  - particle/particle interactions are local (comm is local)
  - zillions of particles  $\Rightarrow$  lots of threads

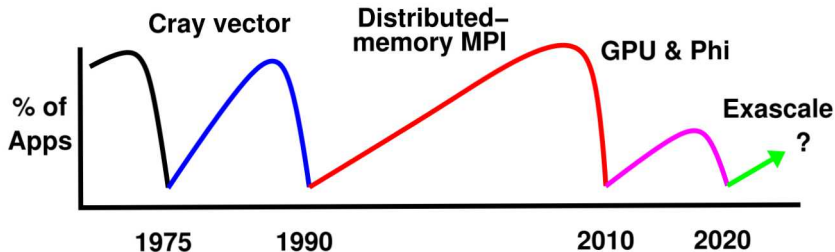
# Growing imbalance is good news for particle apps

- Particle apps:
  - lots of flops per memory access (expensive models)
  - particle/particle interactions are local (comm is local)
  - zillions of particles  $\Rightarrow$  lots of threads
- So I shouldn't be complaining ...  
we're **thinning the herd** of apps, less competition for cycles

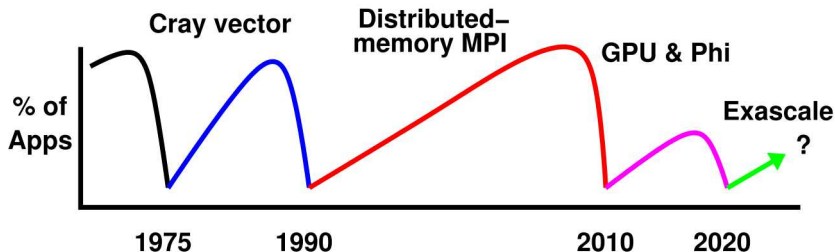
# Growing imbalance is good news for particle apps

- Particle apps:
  - lots of flops per memory access (expensive models)
  - particle/particle interactions are local (comm is local)
  - zillions of particles  $\Rightarrow$  lots of threads
- So I shouldn't be complaining ...  
we're **thinning the herd** of apps, less competition for cycles
- But ...
  - particles don't represent **broad swath** of computational science, or majority of apps that need HPC
  - physics often isn't short-range
  - hard to reach long timescales with explicit timestepping

# Qualitative history of apps on largest HPC platforms



# Qualitative history of apps on largest HPC platforms



- Curves from each era actually **overlap** somewhat
- Peaks would be **much lower** if only counted apps that ran efficiently on entire large machines
- Folks with **quantitative** data on this?
  - supercomputer centers (DOE NERSC, Swiss NSC, etc)
  - big labs with wide variety of apps

# Another quote to explain the politics of HPC



Clark Kerr  
president of UC Berkeley  
later of UC system



“I find the three major administrative problems  
on a campus are ...

# Another quote to explain the politics of HPC



Clark Kerr  
president of UC Berkeley  
later of UC system



“I find the three major administrative problems  
on a campus are ...

**sex** for the students,  
**athletics** for the alumni, and  
**parking** for the faculty.”

Supercomputers also have 3 constituencies (at least)

# Supercomputers also have 3 constituencies (at least)



“I find the three major administrative problems when building a new supercomputer are ...

# Supercomputers also have 3 constituencies (at least)



“I find the three major administrative problems when building a new supercomputer are ...

**flops** for the funders,  
**branding** for the vendors, and  
**access** (parking) for the scientists.”

# Supercomputers also have 3 constituencies (at least)



“I find the three major administrative problems when building a new supercomputer are ...

**flops** for the funders,  
**branding** for the vendors, and  
**access** (parking) for the scientists.”

- Funders = students: flops are sexy, grab the headlines
- Vendors = alumni: our technology is in the fastest machines
- Scientists = faculty:
  - access is more than time on the machine
  - also ease-of-use, ability to do many kinds of science

# Supercomputers also have 3 constituencies (at least)



“I find the three major administrative problems when building a new supercomputer are ...

**flops** for the funders,  
**branding** for the vendors, and  
**access** (parking) for the scientists.”

- Funders = students: flops are sexy, grab the headlines
- Vendors = alumni: our technology is in the fastest machines
- Scientists = faculty:
  - access is more than time on the machine
  - also ease-of-use, ability to do many kinds of science
- Can you make all 3 happy?

# Supercomputers also have 3 constituencies (at least)



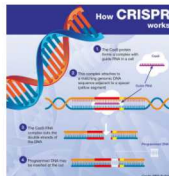
“I find the three major administrative problems when building a new supercomputer are ...

**flops** for the funders,  
**branding** for the vendors, and  
**access** (parking) for the scientists.”

- Funders = students: flops are sexy, grab the headlines
- Vendors = alumni: our technology is in the fastest machines
- Scientists = faculty:
  - access is more than time on the machine
  - also ease-of-use, ability to do many kinds of science
- Can you make all 3 happy?
- Maybe at a rich university, but not with a supercomputer
- B/c the 3 constituencies have competing interests

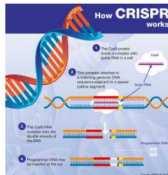
# Cell biology

- PCR (1983) = polymerase chain reaction, DNA replication
- Microarray chips (1995) = parallel gene expression (millions)
- DNA sequencing (2001) = \$10K/Mb  $\Rightarrow$ , few \$0.01/Mb
- CRISPR (2012) = genome editing in living cells



# Cell biology

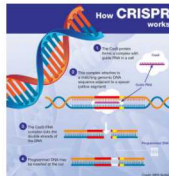
- PCR (1983) = polymerase chain reaction, DNA replication
- Microarray chips (1995) = parallel gene expression (millions)
- DNA sequencing (2001) = \$10K/Mb  $\Rightarrow$ , few \$0.01/Mb
- CRISPR (2012) = genome editing in living cells



- All these technologies rapidly became **ubiquitous**
- Any lab, any grad student can use them
- Don't need add-on experts to write an NIH proposal

# Cell biology

- PCR (1983) = polymerase chain reaction, DNA replication
- Microarray chips (1995) = parallel gene expression (millions)
- DNA sequencing (2001) = \$10K/Mb  $\Rightarrow$ , few \$0.01/Mb
- CRISPR (2012) = genome editing in living cells



- All these technologies rapidly became **ubiquitous**
- Any lab, any grad student can use them
- Don't need add-on experts to write an NIH proposal
- Could we aspire to that **ease-of-use** for HPC machines?

# User facilities with billion \$ instruments

- Hubble telescope (NASA/ESA), SNS (ORNL), Z-machine (Sandia)



# User facilities with billion \$ instruments

- Hubble telescope (NASA/ESA), SNS (ORNL), Z-machine (Sandia)



- Hubble: 1.3M observations, SNS: 20K users, Z: 3160 shots

# User facilities with billion \$ instruments

- Hubble telescope (NASA/ESA), SNS (ORNL), Z-machine (Sandia)



- Hubble: 1.3M observations, SNS: 20K users, Z: 3160 shots
- All solicit user proposals (Hubble from amateurs!)
- Facilities **shield users** from nearly all **complexity**

# User facilities with billion \$ instruments

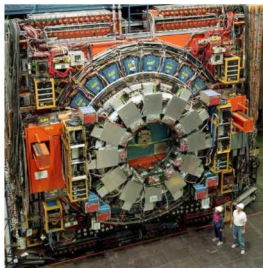
- Hubble telescope (NASA/ESA), SNS (ORNL), Z-machine (Sandia)



- Hubble: 1.3M observations, SNS: 20K users, Z: 3160 shots
- All solicit user proposals (Hubble from amateurs!)
- Facilities **shield users** from nearly all **complexity**
- What if 20x new HPC machine just gave all users 20x more?

# High-energy particle physics

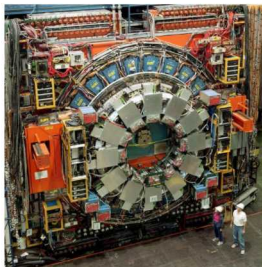
- CERN, FermiLab, etc



- Every new accelerator requires **one-of-a-kind new detectors** to be useful
- Performs handful of (high-impact, highly complex) science experiments in a narrow sub-field of physics

# High-energy particle physics

- CERN, FermiLab, etc



- Every new accelerator requires **one-of-a-kind new detectors** to be useful
- Performs handful of (high-impact, highly complex) science experiments in a narrow sub-field of physics
- **Is HPC more like** cell bio, user facilities, or HE physics?

# No silver bullet solutions

- Let's not build hard-to-use machines that require **heroic ukulele players** to model a handful of good science problems
- Rather, let's create easy-to-use machines & software to enable huge numbers of **mere-mortal computational scientists** in many fields to do amazing science

# No silver bullet solutions

- Let's not build hard-to-use machines that require **heroic ukulele players** to model a handful of good science problems
- Rather, let's create easy-to-use machines & software to enable huge numbers of **mere-mortal computational scientists** in many fields to do amazing science
- Hope you view my remarks as **inducements** to:
  - insulate users from growing complexity of HPC machines
  - make life easier for the apps and the science

# No silver bullet solutions

- Let's not build hard-to-use machines that require **heroic ukulele players** to model a handful of good science problems
- Rather, let's create easy-to-use machines & software to enable huge numbers of **mere-mortal computational scientists** in many fields to do amazing science
- Hope you view my remarks as **inducements** to:
  - insulate users from growing complexity of HPC machines
  - make life easier for the apps and the science
- **Metric**: A tool is most powerful, when users don't need experts to use it, and can do things with it the tool creators didn't imagine

# No silver bullet solutions

- Let's not build hard-to-use machines that require **heroic ukulele players** to model a handful of good science problems
- Rather, let's create easy-to-use machines & software to enable huge numbers of **mere-mortal computational scientists** in many fields to do amazing science
- Hope you view my remarks as **inducements** to:
  - insulate users from growing complexity of HPC machines
  - make life easier for the apps and the science
- **Metric**: A tool is most powerful, when users don't need experts to use it, and can do things with it the tool creators didn't imagine

David Parnas: **"Complexity is not a goal. I don't want to be remembered as an engineer of complex systems."**