Computational Challenges in Relativistic Cosmology

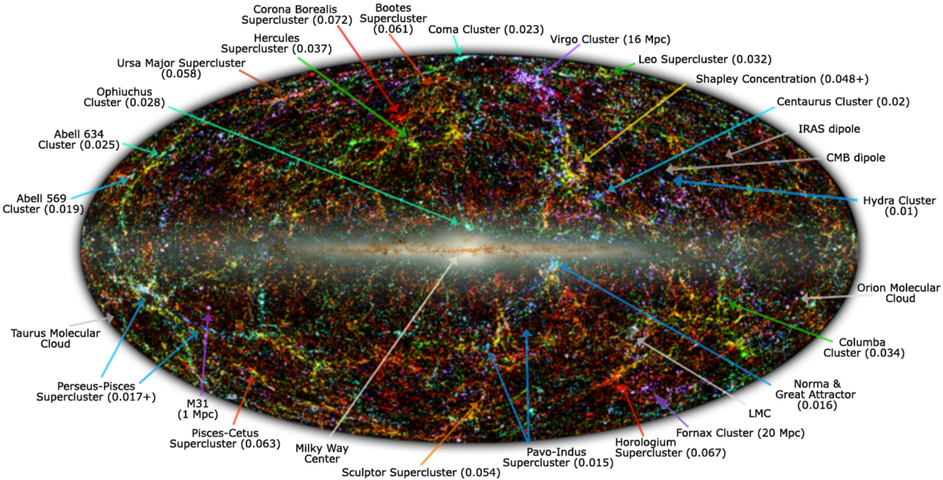
Vincent Reverdy

Department of Astronomy
University of Illinois at Urbana-Champaign (UIUC)

July 4th, 2017

Classical Numerical Cosmology

The local Universe



High performance cosmological simulations

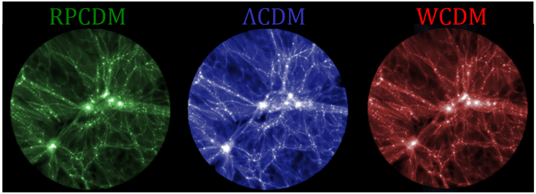
Vlasov-Poisson equations

$$\frac{\mathrm{d}f}{\mathrm{d}t} = \frac{\partial f}{\partial t} + \frac{\vec{p}}{a^2} \cdot \frac{\partial f}{\partial \vec{x}} - \vec{\nabla}\Phi \cdot \frac{\partial f}{\partial \vec{p}} = 0$$

$$\nabla^2 \Phi = 4\pi G a^2 \bar{\rho} \delta \qquad \delta = \frac{\rho - \bar{\rho}}{\bar{\rho}}$$

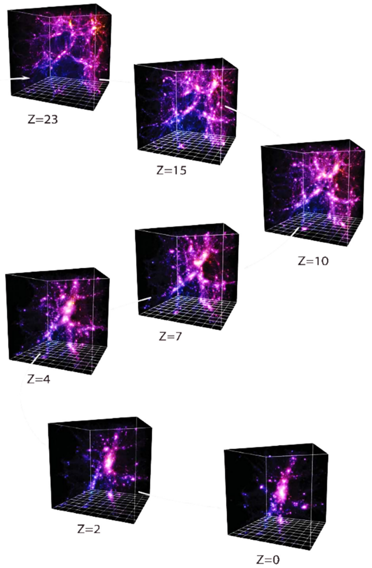
2 regimes

- linear regime $\delta \ll 1$
- non-linear regime $\delta > 1$



Numerical simulation

- **Need of HPC to explore non-linearities**



Distances and perturbations

Homogeneous FLRW metric

$$ds^2 = -c^2 dt^2 + a(t)^2(dx^2 + dy^2 + dz^2)$$

Perturbed FLRW metric

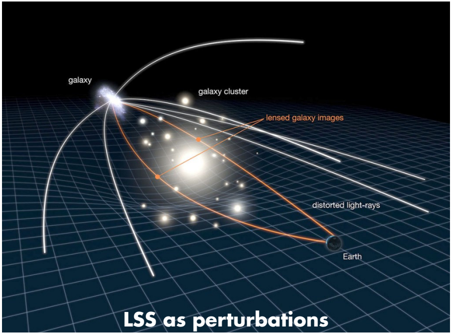
$$ds^2 = -c^2 \left(1 + 2\frac{\Phi}{c^2}\right) dt^2 + a(t)^2 \left(1 - 2\frac{\Phi}{c^2}\right)(dx^2 + dy^2 + dz^2)$$

Affine connections

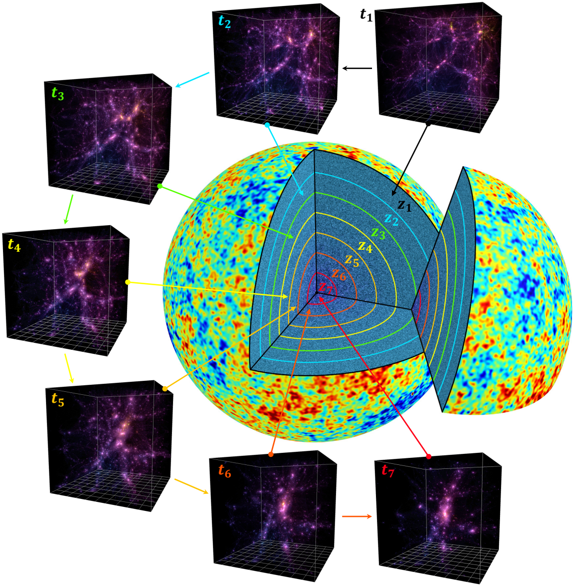
$$\Gamma^{\alpha}_{\beta\gamma} = \frac{1}{2} g^{\alpha\delta} \left(\frac{\partial g_{\delta\beta}}{\partial x^{\alpha}} + \frac{\partial g_{\delta\gamma}}{\partial x^{\beta}} - \frac{\partial g_{\beta\gamma}}{\partial x^{\delta}}\right)$$

Geodesics equation

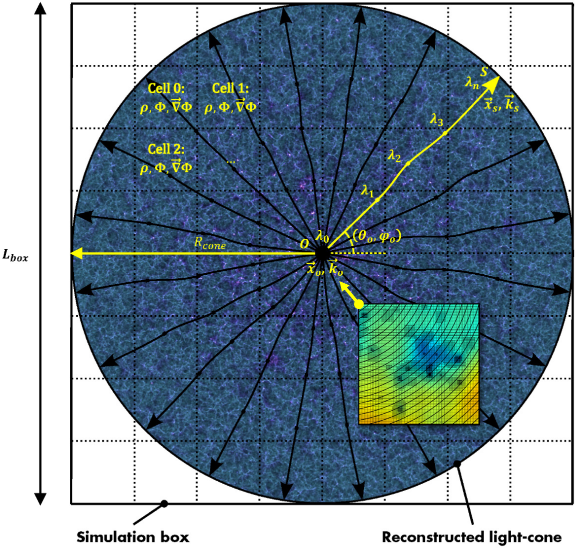
$$\frac{d^2 x^{\alpha}}{d\lambda^2} = -\Gamma^{\alpha}_{\beta\gamma} \frac{dx^{\beta}}{d\lambda} \frac{dx^{\gamma}}{d\lambda}$$



Cone construction



Integration of geodesics



Generic approach

- **Direct integration of geodesics**

Integration of geodesics

$$\frac{d^2\eta}{d\lambda^2} \approx -\frac{2a'}{a}\frac{d\eta}{d\lambda}\frac{d\eta}{d\lambda} - \frac{2}{c^2}\frac{d\Phi}{d\lambda}\frac{d\eta}{d\lambda} + 2\frac{\partial\Phi}{\partial\eta}\left(\frac{d\eta}{d\lambda}\right)^2$$

$$\frac{d^2x}{d\lambda^2} \approx -\frac{2a'}{a}\frac{d\eta}{d\lambda}\frac{dx}{d\lambda} + \frac{2}{c^2}\frac{d\Phi}{d\lambda}\frac{dx}{d\lambda} - 2\frac{\partial\Phi}{\partial x}\left(\frac{d\eta}{d\lambda}\right)^2$$

$$\frac{d^2y}{d\lambda^2} \approx -\frac{2a'}{a}\frac{d\eta}{d\lambda}\frac{dy}{d\lambda} + \frac{2}{c^2}\frac{d\Phi}{d\lambda}\frac{dy}{d\lambda} - 2\frac{\partial\Phi}{\partial y}\left(\frac{d\eta}{d\lambda}\right)^2$$

$$\frac{d^2z}{d\lambda^2} \approx -\frac{2a'}{a}\frac{d\eta}{d\lambda}\frac{dz}{d\lambda} + \frac{2}{c^2}\frac{d\Phi}{d\lambda}\frac{dz}{d\lambda} - 2\frac{\partial\Phi}{\partial z}\left(\frac{d\eta}{d\lambda}\right)^2$$

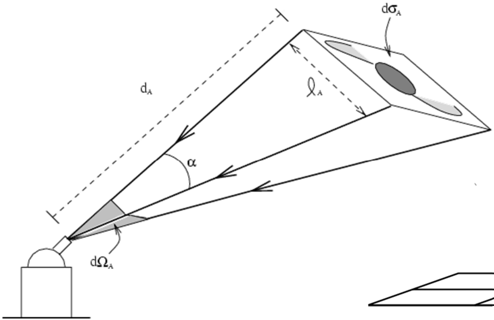
DEUS-FUR + MAGRATHEA

- Full-sky light cones from FUR
- Billions of AMR cells/observer
- 1 million photon/observer
- Minimum number of approx

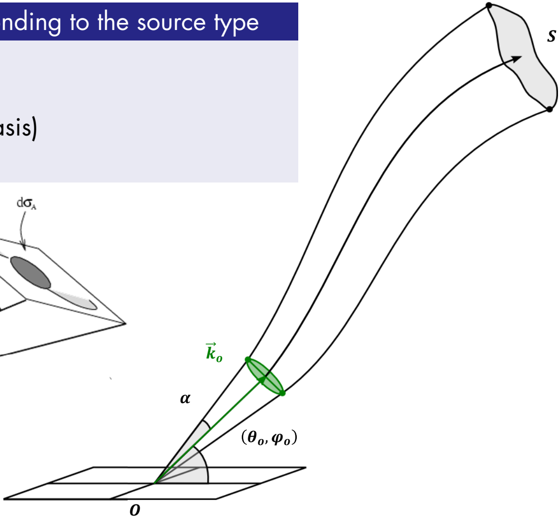
Perturbed trajectories and stop condition

Different possibillities corresponding to the source type

- Affine parameter
- Time
- Perpendicular plane (Sachs basis)
- **Redshift**

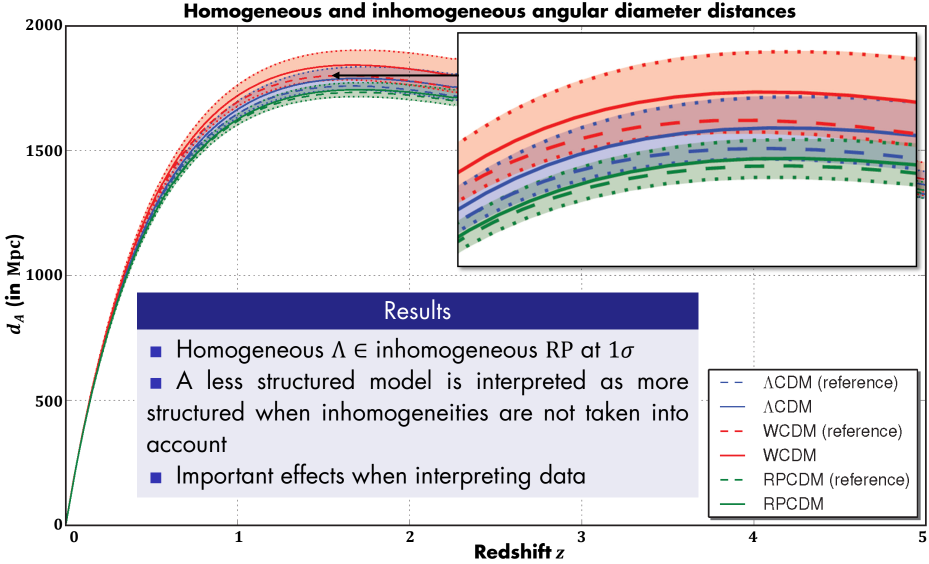


Unperturbed angular diameter distance



Perturbed angular diameter distance

Results



Results

- Homogeneous $\Lambda \in$ inhomogeneous RP at 1σ
- A less structured model is interpreted as more structured when inhomogeneities are not taken into account
- Important effects when interpreting data

Defining Numerical Relativistic Cosmology

Where do we want to go? Defining the ideal simulation

Let's imagine we have a yottascale (10^{24}) supercomputer: 1 billion times more powerful than today with 1 billion times more memory than today

Where do we want to go? Defining the ideal simulation

Let's imagine we have a yottascale (10^{24}) supercomputer: 1 billion times more powerful than today with 1 billion times more memory than today

What?

- Full GR: no background whatsoever
- Cosmological structure formation

Where do we want to go? Defining the ideal simulation

Let's imagine we have a yottascale (10^{24}) supercomputer: 1 billion times more powerful than today with 1 billion times more memory than today

What?

- Full GR: no background whatsoever
- Cosmological structure formation

What size range?

- L_{min}: small galaxies?
- L_{max}: the Observable Universe?

Where do we want to go? Defining the ideal simulation

Let's imagine we have a yottascale (10^{24}) supercomputer: 1 billion times more powerful than today with 1 billion times more memory than today

What?

- Full GR: no background whatsoever
- Cosmological structure formation

What size range?

- L_{min}: small galaxies?
- L_{max}: the Observable Universe?

What mass range?

- M_{min}: mass of small galaxies?
- M_{max}: mass of the Observable Universe?

Where do we want to go? Defining the ideal simulation

Let's imagine we have a yottascale (10^{24}) supercomputer: 1 billion times more powerful than today with 1 billion times more memory than today

What?

- Full GR: no background whatsoever
- Cosmological structure formation

What size range?

- L_{min}: small galaxies?
- L_{max}: the Observable Universe?

What mass range?

- M_{min}: mass of small galaxies?
- M_{max}: mass of the Observable Universe?

What time range?

- $t_{\text{beginning}}$: redshift of the CMB?
- t_{end}: current epoch?

Where do we want to go? Defining the ideal simulation

Let's imagine we have a yottascale (10^{24}) supercomputer: 1 billion times more powerful than today with 1 billion times more memory than today

What?

- Full GR: no background whatsoever
- Cosmological structure formation

What size range?

- L_{min}: small galaxies?
- L_{max}: the Observable Universe?

What mass range?

- M_{min}: mass of small galaxies?
- M_{max}: mass of the Observable Universe?

What time range?

- $t_{\text{beginning}}$: redshift of the CMB?
- t_{end}: current epoch?

What physics?

- Pure gravity
- Hydrodynamics?
- Baryonic physics?
- Singularities and black holes?
- Neutrinos?
- Magnetic fields?
- Pre-CMB physics?
- etc. . .

Why?

Why?

- Compute the order of magnitude of the backreaction?

Why?

- Compute the order of magnitude of the backreaction?
- Answer the backreaction conjecture?

Why?

- Compute the order of magnitude of the backreaction?
- Answer the backreaction conjecture?
- Have more realistic simulations?

Why?

- Compute the order of magnitude of the backreaction?
- Answer the backreaction conjecture?
- Have more realistic simulations?
- Have better virtual catalogs to prepare observational surveys?

Why?

- Compute the order of magnitude of the backreaction?
- Answer the backreaction conjecture?
- Have more realistic simulations?
- Have better virtual catalogs to prepare observational surveys?
- Better accuracy and precision in cosmological predictions?

Why?

- Compute the order of magnitude of the backreaction?
- Answer the backreaction conjecture?
- Have more realistic simulations?
- Have better virtual catalogs to prepare observational surveys?
- Better accuracy and precision in cosmological predictions?
- Numerical demonstration of classical cosmology?

Why?

- Compute the order of magnitude of the backreaction?
- Answer the backreaction conjecture?
- Have more realistic simulations?
- Have better virtual catalogs to prepare observational surveys?
- Better accuracy and precision in cosmological predictions?
- Numerical demonstration of classical cosmology?
- Understand the emergence of physics phenomena?

Why?

- Compute the order of magnitude of the backreaction?
- Answer the backreaction conjecture?
- Have more realistic simulations?
- Have better virtual catalogs to prepare observational surveys?
- Better accuracy and precision in cosmological predictions?
- Numerical demonstration of classical cosmology?
- Understand the emergence of physics phenomena?
- Anything else?

Why more power?

Why more power?

Traditional answers: the philosophy of more

Why more power?

Traditional answers: the philosophy of more

- More stuff: bigger, longer, larger

Why more power?

Traditional answers: the philosophy of more

- More stuff: bigger, longer, larger
- More resolution: in space, in time, in mass

Why more power?

Traditional answers: the philosophy of more

- More stuff: bigger, longer, larger
- More resolution: in space, in time, in mass
- More simulations: exploration of parameter space, statistical accuracy

Why more power?

Traditional answers: the philosophy of more

- More stuff: bigger, longer, larger
- More resolution: in space, in time, in mass
- More simulations: exploration of parameter space, statistical accuracy
- More physics: multiphysics code

Why more power?

Traditional answers: the philosophy of more

- More stuff: bigger, longer, larger
- More resolution: in space, in time, in mass
- More simulations: exploration of parameter space, statistical accuracy
- More physics: multiphysics code

Will this really answer our questions?

Why more power?

Traditional answers: the philosophy of more

- More stuff: bigger, longer, larger
- More resolution: in space, in time, in mass
- More simulations: exploration of parameter space, statistical accuracy
- More physics: multiphysics code

Will this really answer our questions?

- More stuff $\Rightarrow$ better evaluation of global quantities

Why more power?

Traditional answers: the philosophy of more

- More stuff: bigger, longer, larger
- More resolution: in space, in time, in mass
- More simulations: exploration of parameter space, statistical accuracy
- More physics: multiphysics code

Will this really answer our questions?

- More stuff $\Rightarrow$ better evaluation of global quantities
- More resolution $\Rightarrow$ less numerical errors

Why more power?

Traditional answers: the philosophy of more

- More stuff: bigger, longer, larger
- More resolution: in space, in time, in mass
- More simulations: exploration of parameter space, statistical accuracy
- More physics: multiphysics code

Will this really answer our questions?

- More stuff $\Rightarrow$ better evaluation of global quantities
- More resolution $\Rightarrow$ less numerical errors
- More simulations $\Rightarrow$ better accuracy on statistical quantities

Why more power?

Traditional answers: the philosophy of more

- More stuff: bigger, longer, larger
- More resolution: in space, in time, in mass
- More simulations: exploration of parameter space, statistical accuracy
- More physics: multiphysics code

Will this really answer our questions?

- More stuff $\Rightarrow$ better evaluation of global quantities
- More resolution $\Rightarrow$ less numerical errors
- More simulations $\Rightarrow$ better accuracy on statistical quantities
- More physics $\Rightarrow$ more realistic (but less understanding)

Why more power?

Traditional answers: the philosophy of more

- More stuff: bigger, longer, larger
- More resolution: in space, in time, in mass
- More simulations: exploration of parameter space, statistical accuracy
- More physics: multiphysics code

Will this really answer our questions?

- More stuff $\Rightarrow$ better evaluation of global quantities
- More resolution $\Rightarrow$ less numerical errors
- More simulations $\Rightarrow$ better accuracy on statistical quantities
- More physics $\Rightarrow$ more realistic (but less understanding)

Does not really bring any new knowledge...

On realistic simulations

On realistic simulations

On multiphysics simulations

- Increase the number of degrees of freedom
- Increased degeneracy
- Add dimensions in a parameter space with local extrema

On realistic simulations

On multiphysics simulations

- Increase the number of degrees of freedom
- Increased degeneracy
- Add dimensions in a parameter space with local extrema

Reality vs correctness

- Mimicking reality does not provide any guarantee of correctness
- Proving correctness is difficult, and even more with multiphysics simulations
- Multiphysics increase confusion instead of explainability

On realistic simulations

On multiphysics simulations

- Increase the number of degrees of freedom
- Increased degeneracy
- Add dimensions in a parameter space with local extrema

Reality vs correctness

- Mimicking reality does not provide any guarantee of correctness
- Proving correctness is difficult, and even more with multiphysics simulations
- Multiphysics increase confusion instead of explainability

The role of simulations

- The goal of simulations is not to produce realistic results
- Producing realistic results is an optimization problem
- Neural networks can do it far better and way faster than most physical models

On realistic simulations

On multiphysics simulations

- Increase the number of degrees of freedom
- Increased degeneracy
- Add dimensions in a parameter space with local extrema

Reality vs correctness

- Mimicking reality does not provide any guarantee of correctness
- Proving correctness is difficult, and even more with multiphysics simulations
- Multiphysics increase confusion instead of explainability

The role of simulations

- The goal of simulations is not to produce realistic results
- Producing realistic results is an optimization problem
- Neural networks can do it far better and way faster than most physical models

(Personal) conclusion

Simulations are not about the result, they are about the process

Less is more

Less is more

The philosophy of more

- More stuff
- More resolution
- More simulations
- More physics

Less is more

The philosophy of more

- More stuff
- More resolution
- More simulations
- More physics

The philosophy of less

Less is more

The philosophy of more

- More stuff
- More resolution
- More simulations
- More physics

The philosophy of less

- "Less" physics: start from more fundamental equations

Less is more

The philosophy of more

- More stuff
- More resolution
- More simulations
- More physics

The philosophy of less

- "Less" physics: start from more fundamental equations
- Less approximations

Less is more

The philosophy of more

- More stuff
- More resolution
- More simulations
- More physics

The philosophy of less

- "Less" physics: start from more fundamental equations
- Less approximations
- Lower the number of degrees of freedom

Less is more

The philosophy of more

- More stuff
- More resolution
- More simulations
- More physics

The philosophy of less

- "Less" physics: start from more fundamental equations
- Less approximations
- Lower the number of degrees of freedom
- Improved correctness

Less is more

The philosophy of more

- More stuff
- More resolution
- More simulations
- More physics

The philosophy of less

- "Less" physics: start from more fundamental equations
- Less approximations
- Lower the number of degrees of freedom
- Improved correctness
- More generic

Less is more

The philosophy of more

- More stuff
- More resolution
- More simulations
- More physics

The philosophy of less

- “Less” physics: start from more fundamental equations
- Less approximations
- Lower the number of degrees of freedom
- Improved correctness
- More generic
- Understanding emergence

Current Limitations in Numerical Relativistic Cosmology

What do we have? What do we need?

What do we have? What do we need?

What we do have

What do we have? What do we need?

What we do have

- The fundamental physics is known: $G_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$

What do we have? What do we need?

What we do have

- The fundamental physics is known: $G_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$
- The numerical methods are known

What do we have? What do we need?

What we do have

- The fundamental physics is known: $G_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$
- The numerical methods are known
- The algorithms are known

What do we have? What do we need?

What we do have

- The fundamental physics is known: $G_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$
- The numerical methods are known
- The algorithms are known
- Parallelization techniques are known

What do we have? What do we need?

What we do have

- The fundamental physics is known: $G_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$
- The numerical methods are known
- The algorithms are known
- Parallelization techniques are known
- Implementation is a technical detail

What do we have? What do we need?

What we do have

- The fundamental physics is known: $G_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$
- The numerical methods are known
- The algorithms are known
- Parallelization techniques are known
- Implementation is a technical detail

What we do need

$\Rightarrow$ Therefore it is a problem of computational power

Computational power

Computational power

Present and future

- We already have petascale supercomputers (10^{15} FLOPS)
- Exascale supercomputers are coming (10^{18} FLOPS)

Computational power

Present and future

- We already have petascale supercomputers (10^{15} FLOPS)
- Exascale supercomputers are coming (10^{18} FLOPS)

A lot of room at the bottom

- A lot of time in current simulation codes is spend doing nothing
- Lot of opportunity for optimizations at the bottom of computing
- Computer science, computer architecture, compiler, programming languages

Computational power

Present and future

- We already have petascale supercomputers (10^{15} FLOPS)
- Exascale supercomputers are coming (10^{18} FLOPS)

A lot of room at the bottom

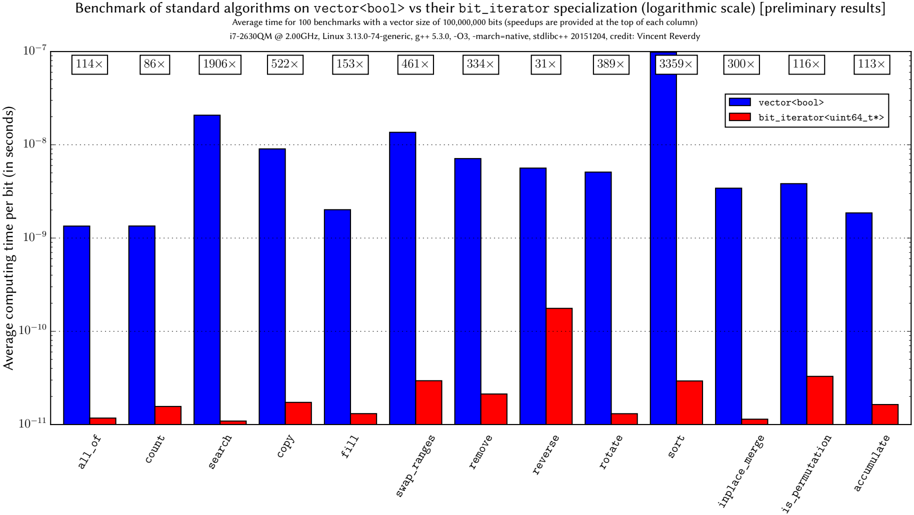
- A lot of time in current simulation codes is spend doing nothing
- Lot of opportunity for optimizations at the bottom of computing
- Computer science, computer architecture, compiler, programming languages

Applications			
High level libraries			
Wrappers and bindings	Python	R	Java
Optimized libraries	Interpreters (Python, R...)		Virtual machines (JVM)
Compiled, native, low level languages (C, C++...)			
Compilers, mostly written in C and C++ (GCC, LLVM...)			
Machine layer, assembly instructions			

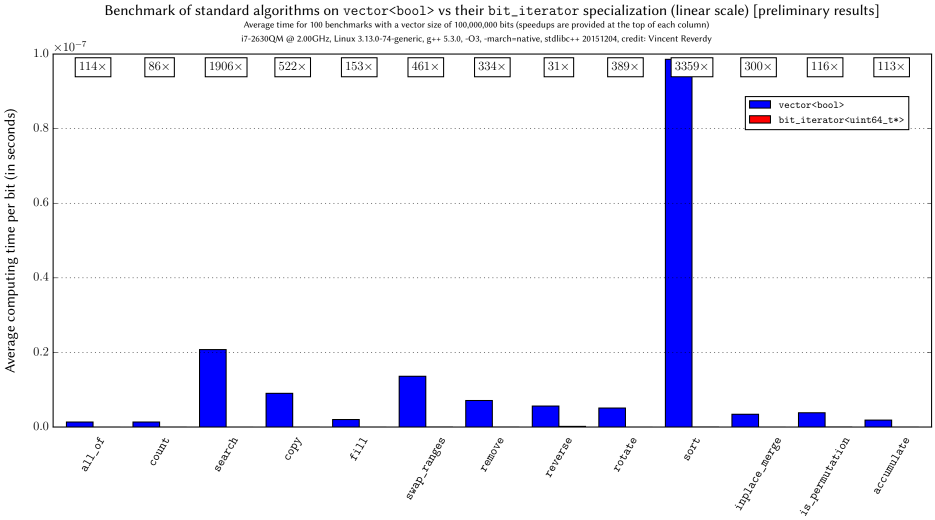
Propagation of optimizations

Softwares are built as stacks. Low-level optimizations can be propagated back to the higher levels while ensuring maximum performances and genericity.

Example of optimization: bit level parallelism



Bit level parallelism (available in a next revision of C++)



Challenges in Numerical Relativistic Cosmology

Upcoming challenge: data structures

Upcoming challenge: data structures

Data transfer timings (Credits: Google Research)

Operation	Approx. time	Remark
L1 cache reference	0.5 ns	
One cycle on a 3 GHz processor	1 ns	
Branch mispredict	5 ns	
L2 cache reference	7 ns	$14\times$ L1 cache
Mutex lock or unlock	25 ns	
Main memory reference	100 ns	$200\times$ L1 cache
Send 1 KB over a 1 Gbps network	10 μs	
Read 1 MB sequentially from main memory	250 μs	
Round trip within the same datacenter	500 μs	
Read 1 MB sequentially from a SSD	1 ms	$4\times$ memory
Disk seek	10 ms	$20\times$ datacenter RT
Read 1 MB sequentially from disk	20 ms	$80\times$ memory
Send packet California→Netherlands→California	150 ms	

Upcoming challenge: data structures

Data transfer timings (Credits: Google Research)		
Operation	**Approx. time**	**Remark**
L1 cache reference	0.5 ns	
One cycle on a 3 GHz processor	1 ns	
Branch mispredict	5 ns	
L2 cache reference	7 ns	14× L1 cache
Mutex lock or unlock	25 ns	
Main memory reference	100 ns	200× L1 cache
Send 1 KB over a 1 Gbps network	10 μs	
Read 1 MB sequentially from main memory	250 μs	
Round trip within the same datacenter	500 μs	
Read 1 MB sequentially from a SSD	1 ms	4× memory
Disk seek	10 ms	20× datacenter RT
Read 1 MB sequentially from disk	20 ms	80× memory
Send packet California→Netherlands→California	150 ms	

Consequences

- Most of the time, pure computing time is not the problem anymore
- Most of the time, data transfer is the problem:
  [disk] → [memory] → [cache]
  [cache] ← [node memory] ↔ [node memory] → [cache]
- Once everything is in cache, computations are fast

⇒ Trees and graphs for AMR

Challenges in computational sciences

Challenges in computational sciences

A history of challenges

Challenges in computational sciences

A history of challenges

1. Formalisms

Challenges in computational sciences

A history of challenges

1. Formalisms
2. Computational power

Challenges in computational sciences

A history of challenges

1. Formalisms
2. Computational power
3. Algorithms

Challenges in computational sciences

A history of challenges

1. Formalisms
2. Computational power
3. Algorithms
4. Parallelism [present]

Challenges in computational sciences

A history of challenges

1. Formalisms
2. Computational power
3. Algorithms
4. Parallelism [present]
5. Data structures [upcoming]

Challenges in computational sciences

A history of challenges

1. Formalisms
2. Computational power
3. Algorithms
4. Parallelism [present]
5. Data structures [upcoming]
6. Code complexity [future]

Challenges in computational sciences

A history of challenges

1. Formalisms
2. Computational power
3. Algorithms
4. Parallelism [present]
5. Data structures [upcoming]
6. Code complexity [future]

A stack of challenges

Challenges in computational sciences

A history of challenges

1. Formalisms
2. Computational power
3. Algorithms
4. Parallelism [present]
5. Data structures [upcoming]
6. Code complexity [future]

A stack of challenges

1. Type theory and category theory [theoretical computer science]

Challenges in computational sciences

A history of challenges

1. Formalisms
2. Computational power
3. Algorithms
4. Parallelism [present]
5. Data structures [upcoming]
6. Code complexity [future]

A stack of challenges

1. Type theory and category theory [theoretical computer science]
2. Programming languages and compilers [computer science]

Challenges in computational sciences

A history of challenges

1. Formalisms
2. Computational power
3. Algorithms
4. Parallelism [present]
5. Data structures [upcoming]
6. Code complexity [future]

A stack of challenges

1. Type theory and category theory [theoretical computer science]
2. Programming languages and compilers [computer science]
3. Data organization [computer science, computer engineering]

Challenges in computational sciences

A history of challenges

1. Formalisms
2. Computational power
3. Algorithms
4. Parallelism [present]
5. Data structures [upcoming]
6. Code complexity [future]

A stack of challenges

1. Type theory and category theory [theoretical computer science]
2. Programming languages and compilers [computer science]
3. Data organization [computer science, computer engineering]
4. Parallelization [computer science, computer engineering]

Challenges in computational sciences

A history of challenges

1. Formalisms
2. Computational power
3. Algorithms
4. Parallelism [present]
5. Data structures [upcoming]
6. Code complexity [future]

A stack of challenges

1. Type theory and category theory [theoretical computer science]
2. Programming languages and compilers [computer science]
3. Data organization [computer science, computer engineering]
4. Parallelization [computer science, computer engineering]
5. Numerical methods [applied mathematics]

Challenges in computational sciences

A history of challenges

1. Formalisms
2. Computational power
3. Algorithms
4. Parallelism [present]
5. Data structures [upcoming]
6. Code complexity [future]

A stack of challenges

1. Type theory and category theory [theoretical computer science]
2. Programming languages and compilers [computer science]
3. Data organization [computer science, computer engineering]
4. Parallelization [computer science, computer engineering]
5. Numerical methods [applied mathematics]
6. Solvers [applied mathematics]

Challenges in computational sciences

A history of challenges

1. Formalisms
2. Computational power
3. Algorithms
4. Parallelism [present]
5. Data structures [upcoming]
6. Code complexity [future]

A stack of challenges

1. Type theory and category theory [theoretical computer science]
2. Programming languages and compilers [computer science]
3. Data organization [computer science, computer engineering]
4. Parallelization [computer science, computer engineering]
5. Numerical methods [applied mathematics]
6. Solvers [applied mathematics]
7. Physics equations [physics]

Code complexity

Code complexity

The physicist's view

Implementation and algorithms are technical details.

Code complexity

The physicist's view

Implementation and algorithms are technical details.

The computer scientist's view

The physics is a technical detail.

Code complexity

The physicist's view

Implementation and algorithms are technical details.

The computer scientist's view

The physics is a technical detail.

Current approaches

Code complexity

The physicist's view

Implementation and algorithms are technical details.

The computer scientist's view

The physics is a technical detail.

Current approaches

- We tend to do work that should be done by computers

Code complexity

The physicist's view

Implementation and algorithms are technical details.

The computer scientist's view

The physics is a technical detail.

Current approaches

- We tend to do work that should be done by computers
- We bend our physics to make it fit instead of bending languages and compilers

Code complexity

The physicist's view

Implementation and algorithms are technical details.

The computer scientist's view

The physics is a technical detail.

Current approaches

- We tend to do work that should be done by computers
- We bend our physics to make it fit instead of bending languages and compilers
- Compilers can derive equations and do mathematical optimization

Code complexity

The physicist's view

Implementation and algorithms are technical details.

The computer scientist's view

The physics is a technical detail.

Current approaches

- We tend to do work that should be done by computers
- We bend our physics to make it fit instead of bending languages and compilers
- Compilers can derive equations and do mathematical optimization

What is a programming language?

$$S \xrightarrow{f} T$$

- S: a set of equations to solve
- T: a number (sequence of 0 and 1) to serve as input of a Turing machine
- f: the morphism computed by the compiler

Code complexity

The physicist's view

Implementation and algorithms are technical details.

The computer scientist's view

The physics is a technical detail.

Current approaches

- We tend to do work that should be done by computers
- We bend our physics to make it fit instead of bending languages and compilers
- Compilers can derive equations and do mathematical optimization

What is a programming language?

$$S \xrightarrow{f} T$$

- S: a set of equations to solve
- T: a number (sequence of 0 and 1) to serve as input of a Turing machine
- f: the morphism computed by the compiler
- Traditional approach: modifying $S \Rightarrow$ but f can be modified too

A type theory/category theory problem

A type theory/category theory problem

Code complexity

A type theory/category theory problem

Code complexity

- Parallelism, numerical methods, data structures and physics are all mixed together

A type theory/category theory problem

Code complexity

- Parallelism, numerical methods, data structures and physics are all mixed together
- Combinatorial explosions of complexity

A type theory/category theory problem

Code complexity

- Parallelism, numerical methods, data structures and physics are all mixed together
- Combinatorial explosions of complexity
- Everything but a technical detail

A type theory/category theory problem

Code complexity

- Parallelism, numerical methods, data structures and physics are all mixed together
- Combinatorial explosions of complexity
- Everything but a technical detail
- Boils down to a type theory/category theory problem

A type theory/category theory problem

Code complexity

- Parallelism, numerical methods, data structures and physics are all mixed together
- Combinatorial explosions of complexity
- Everything but a technical detail
- Boils down to a type theory/category theory problem
- Finding the right abstractions is mostly language independent

Conclusion

Conclusion

The ideal simulation code

Conclusion

The ideal simulation code

- Full GR on cosmological scales

Conclusion

The ideal simulation code

- Full GR on cosmological scales
- No metric background

Conclusion

The ideal simulation code

- Full GR on cosmological scales
- No metric background
- Should solve $G_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$ for a given distribution of mass. . . and that's it

Conclusion

The ideal simulation code

- Full GR on cosmological scales
- No metric background
- Should solve $G_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$ for a given distribution of mass... and that's it

Why?

Conclusion

The ideal simulation code

- Full GR on cosmological scales
- No metric background
- Should solve $G_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$ for a given distribution of mass... and that's it

Why?

- To understand the emergence of cosmology from numerical relativity

Conclusion

The ideal simulation code

- Full GR on cosmological scales
- No metric background
- Should solve $G_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$ for a given distribution of mass. . . and that's it

Why?

- To understand the emergence of cosmology from numerical relativity
- Regardless of backreaction effects

Conclusion

The ideal simulation code

- Full GR on cosmological scales
- No metric background
- Should solve $G_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$ for a given distribution of mass. . . and that's it

Why?

- To understand the emergence of cosmology from numerical relativity
- Regardless of backreaction effects

Challenges

Conclusion

The ideal simulation code

- Full GR on cosmological scales
- No metric background
- Should solve $G_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$ for a given distribution of mass... and that's it

Why?

- To understand the emergence of cosmology from numerical relativity
- Regardless of backreaction effects

Challenges

- Computational power is a no-problem

Conclusion

The ideal simulation code

- Full GR on cosmological scales
- No metric background
- Should solve $G_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$ for a given distribution of mass... and that's it

Why?

- To understand the emergence of cosmology from numerical relativity
- Regardless of backreaction effects

Challenges

- Computational power is a no-problem
- Data movement is a rising bottleneck

Conclusion

The ideal simulation code

- Full GR on cosmological scales
- No metric background
- Should solve $G_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$ for a given distribution of mass... and that's it

Why?

- To understand the emergence of cosmology from numerical relativity
- Regardless of backreaction effects

Challenges

- Computational power is a no-problem
- Data movement is a rising bottleneck
- Code complexity will come after

Conclusion

Conclusion

Final remarks

Conclusion

Final remarks

- The main problem of computing is moving from parallelism to data structures

Conclusion

Final remarks

- The main problem of computing is moving from parallelism to data structures
- There is a lot of room at the bottom of computing: low-level optimizations

Conclusion

Final remarks

- The main problem of computing is moving from parallelism to data structures
- There is a lot of room at the bottom of computing: low-level optimizations
- Code complexity boils down to a type theory problem: computer scientists needed

Conclusion

Final remarks

- The main problem of computing is moving from parallelism to data structures
- There is a lot of room at the bottom of computing: low-level optimizations
- Code complexity boils down to a type theory problem: computer scientists needed
- For a well-designed code, physics should almost be a technical detail

Conclusion

Final remarks

- The main problem of computing is moving from parallelism to data structures
- There is a lot of room at the bottom of computing: low-level optimizations
- Code complexity boils down to a type theory problem: computer scientists needed
- For a well-designed code, physics should almost be a technical detail

On simplicity

Conclusion

Final remarks

- The main problem of computing is moving from parallelism to data structures
- There is a lot of room at the bottom of computing: low-level optimizations
- Code complexity boils down to a type theory problem: computer scientists needed
- For a well-designed code, physics should almost be a technical detail

On simplicity

- Doing Full GR cosmological simulations is aiming for less

Conclusion

Final remarks

- The main problem of computing is moving from parallelism to data structures
- There is a lot of room at the bottom of computing: low-level optimizations
- Code complexity boils down to a type theory problem: computer scientists needed
- For a well-designed code, physics should almost be a technical detail

On simplicity

- Doing Full GR cosmological simulations is aiming for less
- ... and less is more: more genericity, more correctness, more explainability
- ... to understand the emergence of cosmology from numerical relativity

Conclusion

Final remarks

- The main problem of computing is moving from parallelism to data structures
- There is a lot of room at the bottom of computing: low-level optimizations
- Code complexity boils down to a type theory problem: computer scientists needed
- For a well-designed code, physics should almost be a technical detail

On simplicity

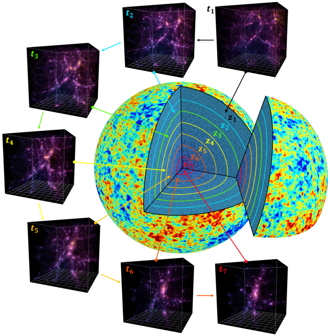
- Doing Full GR cosmological simulations is aiming for less
- . . . and less is more: more genericity, more correctness, more explainability
- . . . to understand the emergence of cosmology from numerical relativity

Conclusion

"Simplicity is the final achievement. After one has played a vast quantity of notes and more notes, it is simplicity that emerges as the crowning reward of art."
Fryderyk Chopin

Thank you for your attention

Any question?



For collaborations: vince.rev@gmail.com