AlGorille: Algorithms for the Grid

Jens Gustedt

INRIA Lorraine & LORIA
AlGorille

AlGorille: Algorithms for the Grid

Jens Gustedt

INRIA Lorraine & LORIA

AlGorithmes pour la Grille

AlGorille: Algorithms for the Grid

Jens Gustedt

INRIA Lorraine & LORIA
AlGorithmes pour la grille

AlGorille: Algorithms for the Grid

Jens Gustedt

INRIA Lorraine & LORIA
AlGorithmes pour la grille

AlGorille: Algorithms for the Grid

Jens Gustedt

INRIA Lorraine & LORIA
AlGorithmes pour la grille

AlGorille: Algorithms for the Grid

Jens Gustedt

INRIA Lorraine & LORIA
AlGorithmes pour la grille

Outline

Algorithmics

Within Algorithmics we investigate efficient algorithms and prove statements about their resource usage.

Algorithm: A concise description of a technique to solve a given problem.

Efficiency: Ensure that the resource usage of an algorithms is as low as possible.

Proof: May be a theoretical proof (*e.g* on the bound of the number of operations) or a reproducible experimental study of the behavior.

Algorithmics

Within Algorithmics we investigate efficient algorithms and prove statements about their resource usage.

Algorithm: A concise description of a technique to solve a given problem.

Efficiency: Ensure that the resource usage of an algorithms is as low as possible.

Proof: May be a theoretical proof (*e.g* on the bound of the number of operations) or a reproducible experimental study of the behavior.

Algorithmics

Within Algorithmics we investigate efficient algorithms and prove statements about their resource usage.

Algorithm: A concise description of a technique to solve a given problem.

Efficiency: Ensure that the resource usage of an algorithms is as low as possible.

Proof: May be a theoretical proof (*e.g* on the bound of the number of operations) or a reproducible experimental study of the behavior.

Algorithmics

Within Algorithmics we investigate efficient algorithms and prove statements about their resource usage.

Algorithm: A concise description of a technique to solve a given problem.

Efficiency: Ensure that the resource usage of an algorithms is as low as possible.

Proof: May be a theoretical proof (*e.g* on the bound of the number of operations) or a reproducible experimental study of the behavior.

The Grid

The notion ‘Grid’ was introduced in 1998 by Foster & Kesselmann:

Idea: Use resources on the net

- computing power
- bandwidth
- storage

transparently, like we use electricity:

customer provider

anonymous

The Grid

The notion ‘Grid’ was introduced in 1998 by Foster & Kesselmann:

Idea: Use resources on the net

- computing power
- bandwidth
- storage

transparently, like we use electricity:

customer provider

anonymous

The Grid

The notion ‘Grid’ was introduced in 1998 by Foster & Kesselmann:

Idea: Use resources on the net

- computing power
- bandwidth
- storage

transparently, like we use electricity:

The Grid

The notion ‘Grid’ was introduced in 1998 by Foster & Kesselmann:

Idea: Use resources on the net

- computing power
- bandwidth
- storage

transparently, like we use electricity:

The Grid

The notion ‘Grid’ was introduced in 1998 by Foster & Kesselmann:

Idea: Use resources on the net

- computing power
- bandwidth
- storage

transparently, like we use electricity:

customer provider

anonymous

The Grid

The notion ‘Grid’ was introduced in 1998 by Foster & Kesselmann:

Idea: Use resources on the net

- computing power
- bandwidth
- storage

transparently, like we use electricity:

customer provider

anonymous

The Grid

The notion ‘Grid’ was introduced in 1998 by Foster & Kesselmann:

Idea: Use resources on the net

- computing power
- bandwidth
- storage

transparently, like we use electricity:

customer provider

anonymous

The Grid

The notion ‘Grid’ was introduced in 1998 by Foster & Kesselmann:

Idea: Use resources on the net

- computing power
- bandwidth
- storage

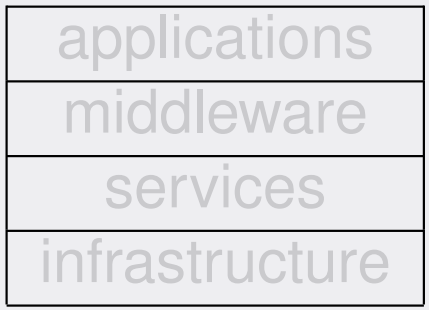
transparently, like we use electricity:

customer provider

anonymous

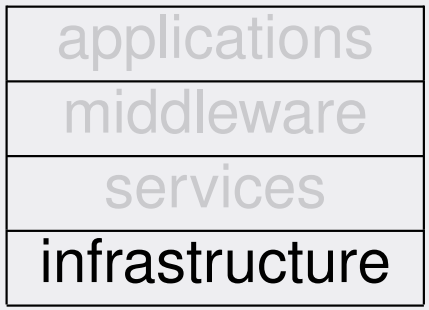
Our Contribution

Layer model of a grid architecture:



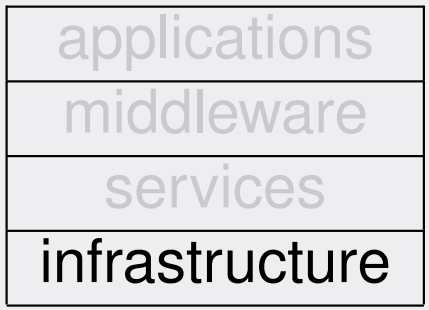
Our Contribution

Layer model of a grid architecture:



Our Contribution

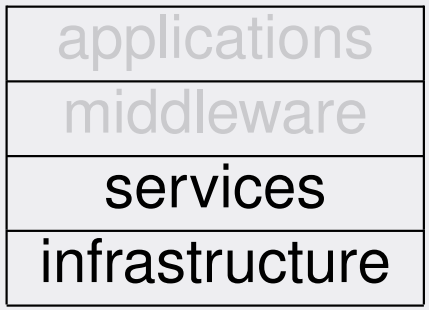
Layer model of a grid architecture:



soft and hardware

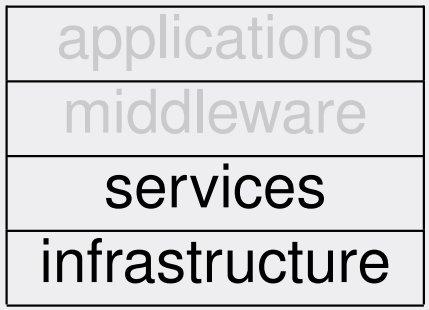
Our Contribution

Layer model of a grid architecture:



Our Contribution

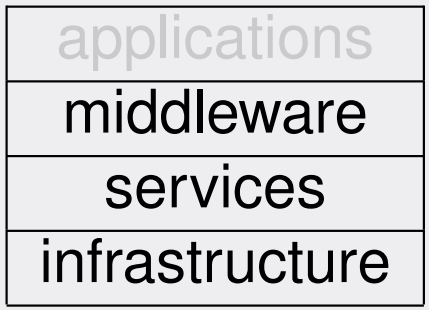
Layer model of a grid architecture:



scheduling, data management...

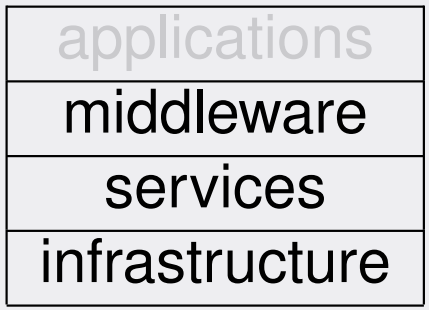
Our Contribution

Layer model of a grid architecture:



Our Contribution

Layer model of a grid architecture:



assemble service into platform API

Our Contribution

Layer model of a grid architecture:

applications
middleware
services
infrastructure

Our Contribution

Layer model of a grid architecture:

applications	⟸ here
middleware	
services	⟸ here
infrastructure	

Outline

Outline

Outline

Outline

Outline

Outline

Outline

The Team

Permanents:

Jens Gustedt	(DR INRIA)
Emmanuel Jeannot	(CR INRIA)
Martin Quinson	(MdC UHP/ESIAL)
Frédéric Suter	(MdC UHP)

External Collaborator:

Stephane Vialle	(prof Supélec, Metz Campus)

Temporary:

Pierre-Nicolas Clauss	Tchimou N'Takpé
Malek Cherier (SimGrid)	Olivier Dubuisson (Wrekavoc)
Xavier Delaruelle (Grid5000)	Léo Ghemtio (docking)
Luiz-Angelo Steffenel	(ATER Nancy 2)

Secretary:

Céline Simon	(UHP)

The Team

Permanents:

Jens Gustedt	(DR INRIA)
Emmanuel Jeannot	(CR INRIA)
Martin Quinson	(MdC UHP/ESIAL)
Frédéric Suter	(MdC UHP)

External Collaborator:

Stephane Vialle	(prof Supélec, Metz Campus)

Temporary:

Pierre-Nicolas Clauss	Tchimou N'Takpé
Malek Cherier (SimGrid)	Olivier Dubuisson (Wrekavoc)
Xavier Delaruelle (Grid5000)	Léo Ghemtio (docking)
Luiz-Angelo Steffenel	(ATER Nancy 2)

Secretary:

Céline Simon	(UHP)

The Team

Permanents:

Jens Gustedt	(DR INRIA)
Emmanuel Jeannot	(CR INRIA)
Martin Quinson	(MdC UHP/ESIAL)
Frédéric Suter	(MdC UHP)

External Collaborator:

Stephane Vialle	(prof Supélec, Metz Campus)

Temporary:

Pierre-Nicolas Clauss	Tchimou N'Takpé
Malek Cherier (SimGrid)	Olivier Dubuisson (Wrekavoc)
Xavier Delaruelle (Grid5000)	Léo Ghemtio (docking)
Luiz-Angelo Steffenel	(ATER Nancy 2)

Secretary:

Céline Simon	(UHP)

The Team

Permanents:

Jens Gustedt	(DR INRIA)
Emmanuel Jeannot	(CR INRIA)
Martin Quinson	(MdC UHP/ESIAL)
Frédéric Suter	(MdC UHP)

External Collaborator:

Stephane Vialle	(prof Supélec, Metz Campus)

Temporary:

Pierre-Nicolas Clauss	Tchimou N'Takpé
Malek Cherier (SimGrid)	Olivier Dubuisson (Wrekavoc)
Xavier Delaruelle (Grid5000)	Léo Ghemtio (docking)
Luiz-Angelo Steffenel	(ATER Nancy 2)

Secretary:

Céline Simon	(UHP)

Outline

Implication in the Scientific Community

- CoreGrid: EU
- GridExplorer, AGIR: national ACI
- Grid5000
- Alpage: ARA MDMSA
- RGE
- PRST Région Lorraine: MIS

Implication in the Scientific Community

- CoreGrid: EU
- GridExplorer, AGIR: national ACI
- Grid5000
- Alpage: ARA MDMSA
- RGE
- PRST Région Lorraine: MIS

Implication in the Scientific Community

- CoreGrid: EU
- GridExplorer, AGIR: national ACI
- Grid5000
- Alpage: ARA MDMSA
- RGE
- PRST Région Lorraine: MIS

Implication in the Scientific Community

- CoreGrid: EU
- GridExplorer, AGIR: national ACI
- Grid5000
- Alpage: ARA MDMSA
- RGE
- PRST Région Lorraine: MIS

Implication in the Scientific Community

- CoreGrid: EU
- GridExplorer, AGIR: national ACI
- Grid5000
- Alpage: ARA MDMSA
- RGE
- PRST Région Lorraine: MIS

Implication in the Scientific Community

- CoreGrid: EU
- GridExplorer, AGIR: national ACI
- Grid5000
- Alpage: ARA MDMSA
- RGE
- PRST Région Lorraine: MIS

Outline

Outline

The algorithmic challenge: overcome spatial and temporal distances

Grids are distributed environments:

- latency is proportional to the distance
- bandwidth is restricted

The algorithmic challenge: overcome spatial and temporal distances

Grids are distributed environments:

- latency is proportional to the distance (speed of light!)
- bandwidth is restricted

The algorithmic challenge: overcome spatial and temporal distances

Grids are distributed environments:

- latency is proportional to the distance (speed of light!)
- bandwidth is restricted

The algorithmic challenge: overcome spatial and temporal distances

Grids are distributed environments:

- latency is proportional to the distance (speed of light!)
- bandwidth is restricted

We have to organize

The algorithmic challenge: overcome spatial and temporal distances

Grids are distributed environments:

- latency is proportional to the distance (speed of light!)
- bandwidth is restricted

We have to organize

tasks

The algorithmic challenge: overcome spatial and temporal distances

Grids are distributed environments:

- latency is proportional to the distance (speed of light!)
- bandwidth is restricted

We have to organize

tasks

data

The algorithmic challenge: overcome spatial and temporal distances

Grids are distributed environments:

- latency is proportional to the distance (speed of light!)
- bandwidth is restricted

We have to organize

$$\left.\begin{array}{r}\text{tasks}\\ \\ \text{data}\end{array}\right\} \text{within} \left\{\right.$$

The algorithmic challenge: overcome spatial and temporal distances

Grids are distributed environments:

- latency is proportional to the distance (speed of light!)
- bandwidth is restricted

We have to organize

$$\left.\begin{array}{r}\text{tasks}\\ \\ \text{data}\end{array}\right\} \text{within} \left\{\begin{array}{l}\text{time}\\ \\ \end{array}\right.$$

The algorithmic challenge: overcome spatial and temporal distances

Grids are distributed environments:

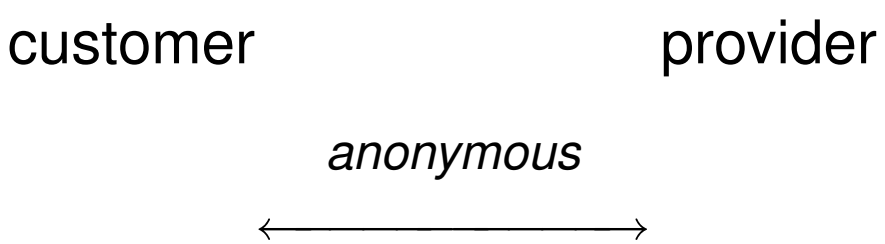
- latency is proportional to the distance (speed of light!)
- bandwidth is restricted

We have to organize

$$\left.\begin{array}{r}\text{tasks}\\ \\ \text{data}\end{array}\right\} \text{within} \left\{\begin{array}{l}\text{time}\\ \\ \text{space.}\end{array}\right.$$

Outline

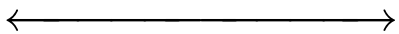
The conceptual challenge: reconcile differing objectives



The conceptual challenge: reconcile differing objectives

customer broker provider

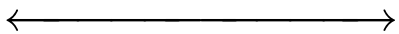
anonymous



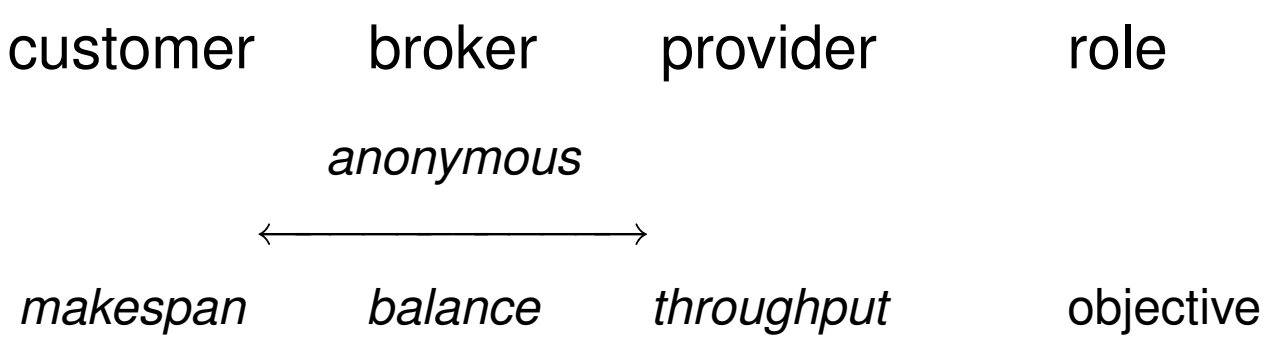
The conceptual challenge: reconcile differing objectives

customer broker provider role

anonymous



The conceptual challenge: reconcile differing objectives



Outline

The methodological challenge: performance evaluation

On top of all these difficulties: the problem of *performance evaluation*.

- Compare different algorithms in a reliable and reproducible way.
- Thereby demonstrate the progress our solutions provide.

We have to

- predict
- control
- evaluate

the performance in a setting that is as wide as possible.

The methodological challenge: performance evaluation

On top of all these difficulties: the problem of *performance evaluation*.

- Compare different algorithms in a reliable and reproducible way.
- Thereby demonstrate the progress our solutions provide.

We have to

- predict
- control
- evaluate

the performance in a setting that is as wide as possible.

The methodological challenge: performance evaluation

On top of all these difficulties: the problem of *performance evaluation*.

- Compare different algorithms in a reliable and reproducible way.
- Thereby demonstrate the progress our solutions provide.

We have to

- predict
- control
- evaluate

the performance in a setting that is as wide as possible.

The methodological challenge: performance evaluation

On top of all these difficulties: the problem of *performance evaluation*.

- Compare different algorithms in a reliable and reproducible way.
- Thereby demonstrate the progress our solutions provide.

We have to

- predict
- control
- evaluate

the performance in a setting that is as wide as possible.

The methodological challenge: performance evaluation

On top of all these difficulties: the problem of *performance evaluation*.

- Compare different algorithms in a reliable and reproducible way.
- Thereby demonstrate the progress our solutions provide.

We have to

- predict
- control
- evaluate

the performance in a setting that is as wide as possible.

The methodological challenge: performance evaluation

On top of all these difficulties: the problem of *performance evaluation*.

- Compare different algorithms in a reliable and reproducible way.
- Thereby demonstrate the progress our solutions provide.

We have to

- predict
- control
- evaluate

the performance in a setting that is as wide as possible.

The methodological challenge: performance evaluation

On top of all these difficulties: the problem of *performance evaluation*.

- Compare different algorithms in a reliable and reproducible way.
- Thereby demonstrate the progress our solutions provide.

We have to

- predict
- control
- evaluate

the performance in a setting that is as wide as possible.

The methodological challenge: performance evaluation

On top of all these difficulties: the problem of *performance evaluation*.

- Compare different algorithms in a reliable and reproducible way.
- Thereby demonstrate the progress our solutions provide.

We have to

- predict
- control
- evaluate

the performance in a setting that is as wide as possible.

The methodological challenge: performance evaluation

On top of all these difficulties: the problem of *performance evaluation*.

- Compare different algorithms in a reliable and reproducible way.
- Thereby demonstrate the progress our solutions provide.

We have to

- predict
- control
- evaluate

the performance in a setting that is as wide as possible.

Outline

Outline

Structuring applications for scalability

Models and algorithms for large scale computations

- Cellular Networks
- Matrix-based problems from Natural Science (Physics, biochemistry, ...)
- Graphs and other Dynamic Data Structures.

Structuring applications for scalability

Models and algorithms for large scale computations

- Cellular Networks
- Matrix-based problems from Natural Science (Physics, biochemistry, ...)
- Graphs and other Dynamic Data Structures.

Structuring applications for scalability

Models and algorithms for large scale computations

- Cellular Networks
- Matrix-based problems from Natural Science (Physics, biochemistry, ...)
- Graphs and other Dynamic Data Structures.

Outline

Transparent Resource Management

Efficient algorithms for

- Sequential and parallel task scheduling
- Data exchange
- Distribution and redistribution of data.

Transparent Resource Management

Efficient algorithms for

- Sequential and parallel task scheduling
- Data exchange
- Distribution and redistribution of data.

Transparent Resource Management

Efficient algorithms for

- Sequential and parallel task scheduling
- Data exchange
- Distribution and redistribution of data.

Outline

Experimental validation

- Simulation $\Longrightarrow$ SimGrid
- Emulation $\Longrightarrow$ Wrekavoc
- Large Scale Experiments $\Longrightarrow$ Grid5000
- Pilot Applications $\Longrightarrow$ parXXL

Experimental validation

- Simulation $\Longrightarrow$ SimGrid
- Emulation $\Longrightarrow$ Wrekavoc
- Large Scale Experiments $\Longrightarrow$ Grid5000
- Pilot Applications $\Longrightarrow$ parXXL

Experimental validation

- Simulation $\Longrightarrow$ SimGrid
- Emulation $\Longrightarrow$ Wrekavoc
- Large Scale Experiments $\Longrightarrow$ Grid5000
- Pilot Applications $\Longrightarrow$ parXXL

Experimental validation

- Simulation $\Longrightarrow$ SimGrid
- Emulation $\Longrightarrow$ Wrekavoc
- Large Scale Experiments $\Longrightarrow$ Grid5000
- Pilot Applications $\Longrightarrow$ parXXL

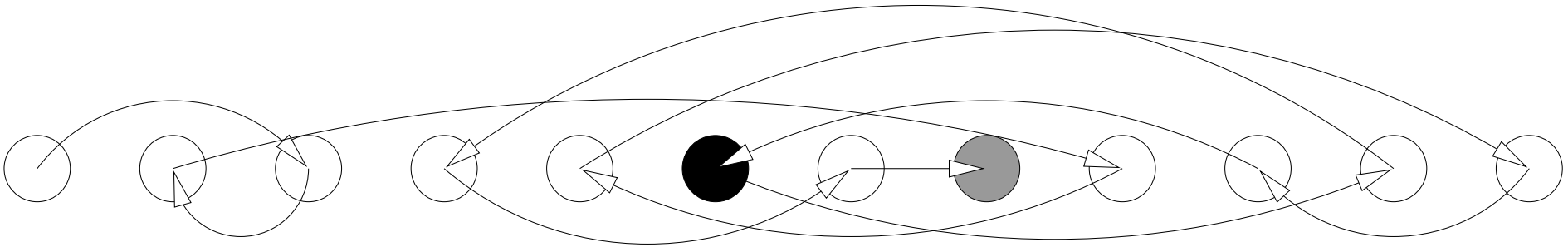
Context: Irregular Applications on a Large Scale

graphs : networks, web graph, internet, social networks
cellular networks, neural networks, cristals

Large problems, some millions or billions of nodes.

Complex neighborhood structure and local functionality

Context: Irregular Applications on a Large Scale



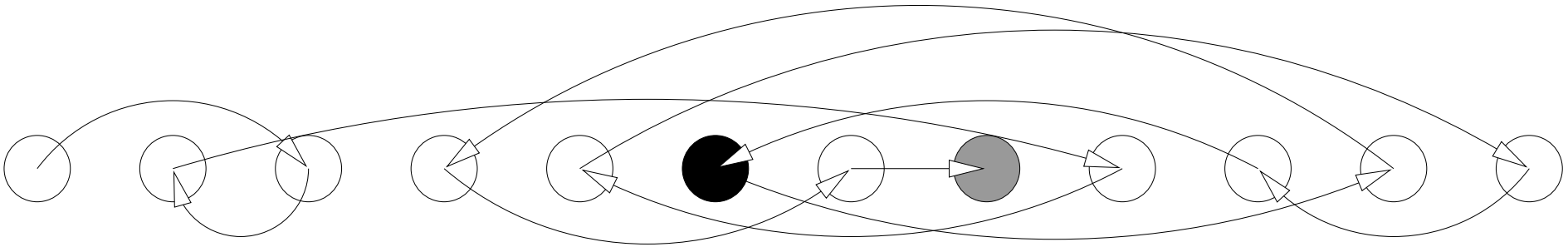
graphs networks web graph, internet, social networks

cellular networks neural networks, cristals

Large problems, some millions or billions of nodes.

Complex neighborhood structure and local functionality

Context: Irregular Applications on a Large Scale



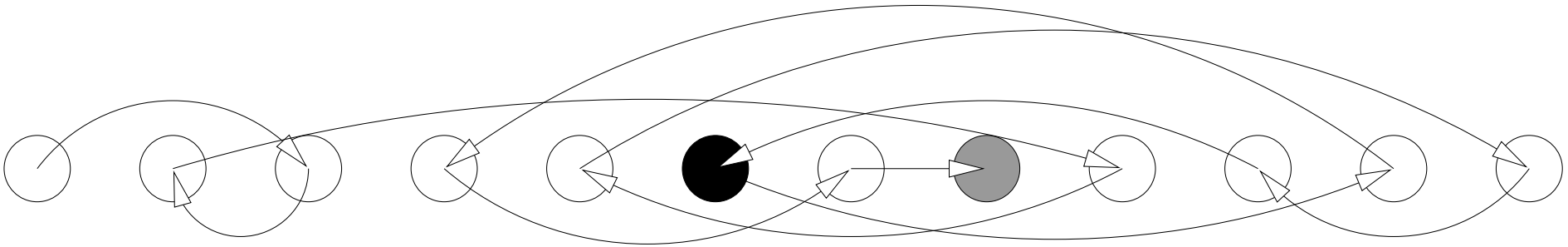
graphs networks web graph, internet, social networks

cellular networks neural networks, cristals

Large problems, some millions or billions of nodes.

Complex neighborhood structure and local functionality

Context: Irregular Applications on a Large Scale



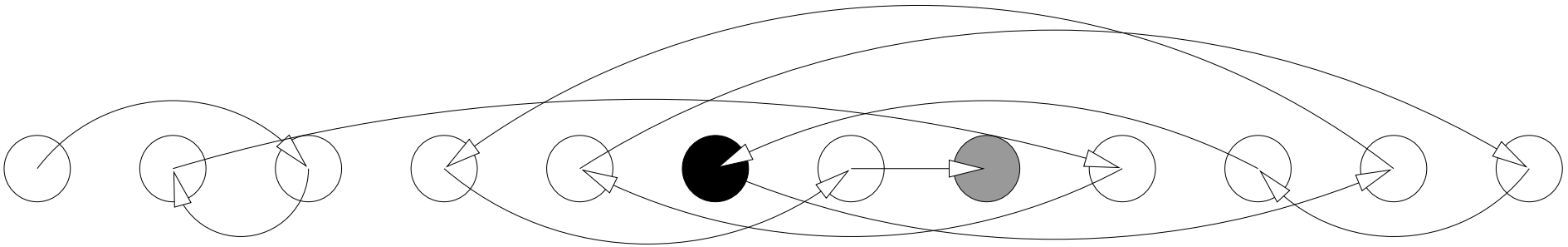
graphs networks web graph, internet, social networks

cellular networks neural networks, cristals

Large problems, some millions or billions of nodes.

Complex neighborhood structure and local functionality

Context: Irregular Applications on a Large Scale



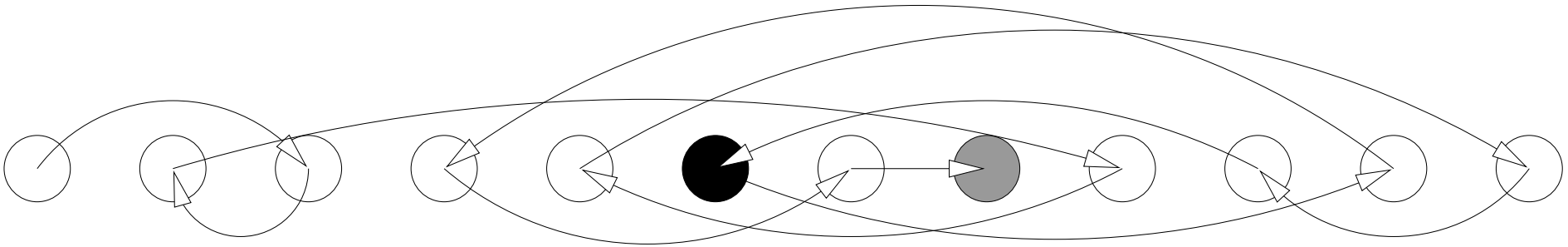
graphs networks web graph, internet, social networks

cellular networks neural networks, cristals

Large problems, some millions or billions of nodes.

Complex neighborhood structure and local functionality

Context: Irregular Applications on a Large Scale



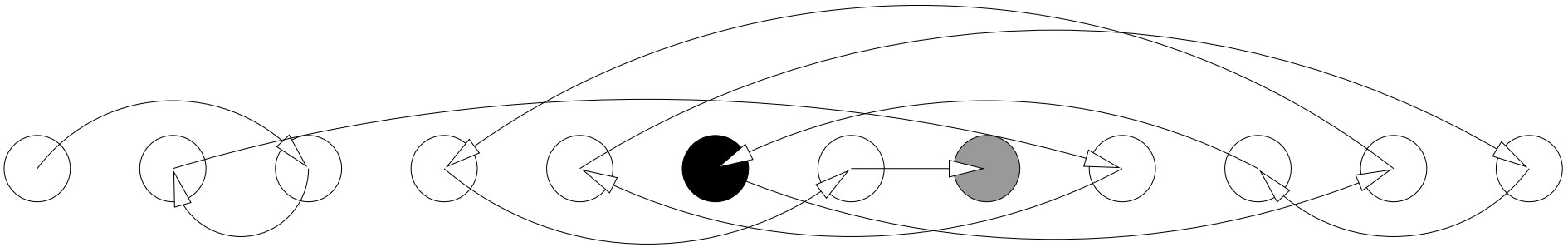
graphs networks web graph, internet, social networks

cellular networks neural networks, cristals

Large problems, some millions or billions of nodes.

Complex neighborhood structure and local functionality

Context: Irregular Applications on a Large Scale



graphs networks web graph, internet, social networks

cellular networks neural networks, cristals

Large problems, some millions or billions of nodes.

Complex neighborhood structure and local functionality

Outline

Software

parXXL Algorithmic toolbox and testbed for fine grained computation on coarse grained architectures.

AdOC Communication compression on the fly.

SimGrid/GRAS Grid-simulator (SimGrid) and toolbox (GRAS) for service development, simulation, evaluation and deployment.

Wrekavoc Emulation of heterogeneity.

Software

parXXL Algorithmic toolbox and testbed for fine grained computation on coarse grained architectures.

AdOC Communication compression on the fly.

SimGrid/GRAS Grid-simulator (SimGrid) and toolbox (GRAS) for service development, simulation, evaluation and deployment.

Wrekavoc Emulation of heterogeneity.

Software

parXXL Algorithmic toolbox and testbed for fine grained computation on coarse grained architectures.

AdOC Communication compression on the fly.

SimGrid/GRAS Grid-simulator (SimGrid) and toolbox (GRAS) for service development, simulation, evaluation and deployment.

Wrekavoc Emulation of heterogeneity.

Software

parXXL	Algorithmic toolbox and testbed for fine grained computation on coarse grained architectures.
AdOC	Communication compression on the fly.
SimGrid/GRAS	Grid-simulator (SimGrid) and toolbox (GRAS) for service development, simulation, evaluation and deployment.
Wrekavoc	Emulation of heterogeneity.

GRAS (Grid Reality And Simulation)

Why: On simulator, one develops a prototype, not an application

What: Grid Research & Development Framework

- Study on simulator, produce real-world programs seamlessly
- Two implementations of the same interface
- Running to Linux, Mac OSX, Solaris, AIX, IRIX (Windows?)
- Communication speed almost comparable to MPICH
- Allow heterogeneous communications

GRAS (Grid Reality And Simulation)

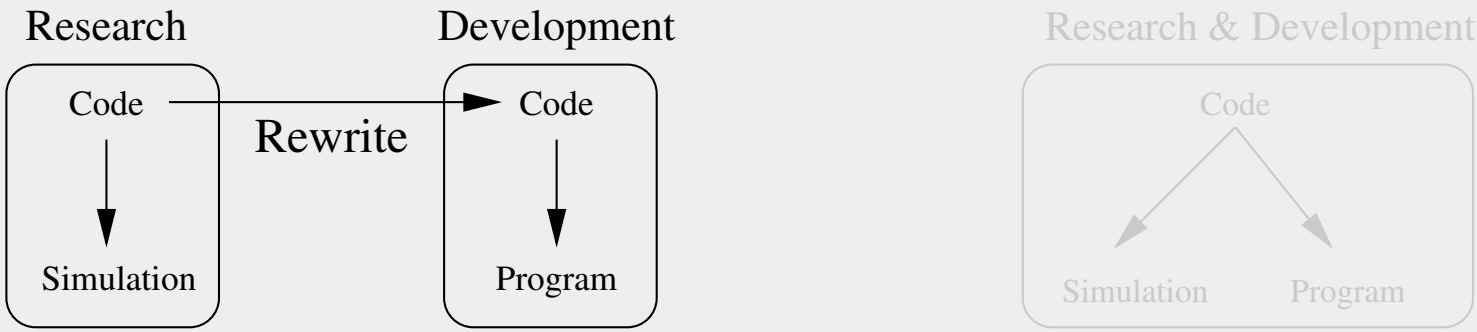
Why: On simulator, one develops a prototype, not an application

What: Grid Research & Development Framework

- Study on simulator, produce real-world programs seamlessly
- Two implementations of the same interface
- Running to Linux, Mac OSX, Solaris, AIX, IRIX (Windows?)
- Communication speed almost comparable to MPICH
- Allow heterogeneous communications

GRAS (Grid Reality And Simulation)

Why: On simulator, one develops a prototype, not an application

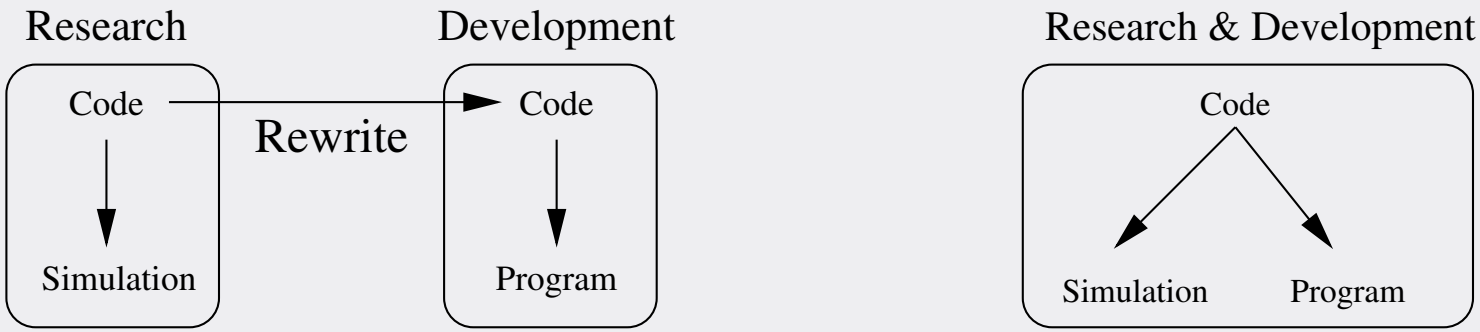


What: Grid Research & Development Framework

- Study on simulator, produce real-world programs seamlessly
- Two implementations of the same interface
- Running to Linux, Mac OSX, Solaris, AIX, IRIX (Windows?)
- Communication speed almost comparable to MPICH
- Allow heterogeneous communications

GRAS (Grid Reality And Simulation)

Why: On simulator, one develops a prototype, not an application

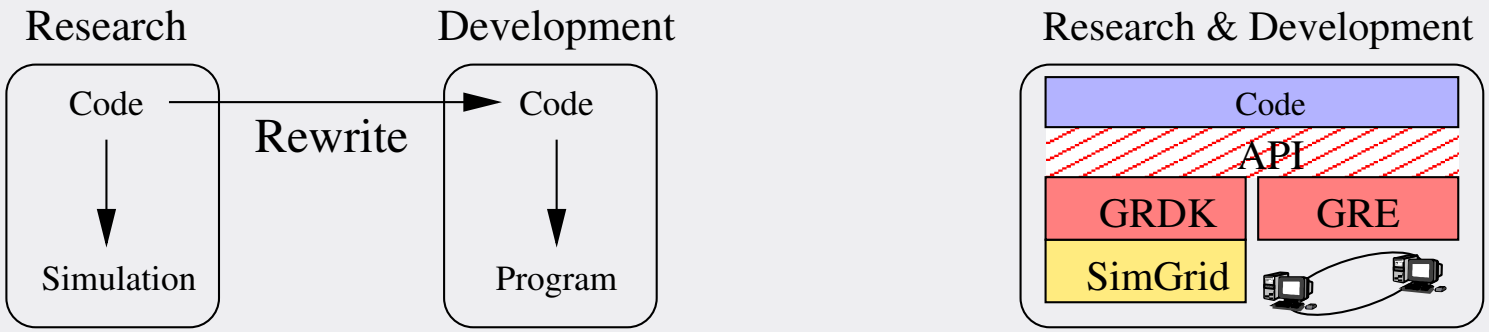


What: Grid Research & Development Framework

- Study on simulator, produce real-world programs seamlessly
- Two implementations of the same interface
- Running to Linux, Mac OSX, Solaris, AIX, IRIX (Windows?)
- Communication speed almost comparable to MPICH
- Allow heterogeneous communications

GRAS (Grid Reality And Simulation)

Why: On simulator, one develops a prototype, not an application



What: Grid Research & Development Framework

- Study on simulator, produce real-world programs seamlessly
- Two implementations of the same interface
- Running to Linux, Mac OSX, Solaris, AIX, IRIX (Windows?)
- Communication speed almost comparable to MPICH
- Allow heterogeneous communications

Outline

Conclusion

AlGorille

Algorithms for the Grid

Layer model of a grid architecture

Experi-mental valida-tion

applications
middleware
services
infrastructure

⟸ Structuring applications for scalability

⟸ Transparent resource management

A lot of Software

parXXL AdOc SimGrid Wrekavoc

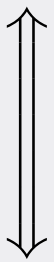
Conclusion

AlGorille

Algorithms for the Grid

Layer model of a grid architecture

Experi-mental valida-tion



applications
middleware
services
infrastructure

⟸ Structuring applications for scalability

⟸ Transparent resource management

A lot of Software

parXXL AdOc SimGrid Wrekavoc

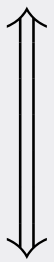
Conclusion

AlGorille

Algorithms for the Grid

Layer model of a grid architecture

Experi-mental valida-tion



applications
middleware
services
infrastructure

⟸ Structuring applications for scalability

⟸ Transparent resource management

A lot of Software

parXXL AdOc SimGrid Wrekavoc