

Performance Analysis of Publish/Subscribe Systems

H. Abbes^{1,2} J.-C. Dubacq²

¹Research Unit UTIC
École supérieure des Sciences et Technologies de Tunis

²Laboratory LIPN – UMR 7030
CNRS – Université Paris 13

EUMEDGRID Workshop on Grid Computing, 2007



Outline

- 1 Resource discovery in grids
 - Motivation
 - Exploiting your neighbourhood
- 2 Performance Analysis
 - Framework
 - Measurements

Outline

- 1 Resource discovery in grids
 - Motivation
 - Exploiting your neighbourhood
- 2 Performance Analysis
 - Framework
 - Measurements



Grids

Why putting grids around us?

Computational resources are more and more demanding.

- Computational grids
- Data grids, peer-to-peer networks
- Mobile computing, pervasive computing
- Inter-node communications, Instant grids!

What does not scale ?

- Fault tolerance (nodes, network)
- Data circulation (firewalls, rate, quantity)
- Intermittent node management (desktop grids)

Flexible grids need to be aware of new resources in a distributed way.



Grids

Why putting grids around us?

Computational resources are more and more demanding.

- Computational grids
- Data grids, peer-to-peer networks
- Mobile computing, pervasive computing
- Inter-node communications, Instant grids!

What does not scale ?

- Fault tolerance (nodes, network)
- Data circulation (firewalls, rate, quantity)
- Intermittent node management (desktop grids)

Flexible grids need to be aware of new resources in a distributed way.



Grids

Why putting grids around us?

Computational resources are more and more demanding.

- Computational grids
- Data grids, peer-to-peer networks
- Mobile computing, pervasive computing
- Inter-node communications, Instant grids!

What does not scale ?

- Fault tolerance (nodes, network)
- Data circulation (firewalls, rate, quantity)
- Intermittent node management (desktop grids)

Flexible grids need to be aware of new resources in a distributed way.



Grids

Why putting grids around us?

Computational resources are more and more demanding.

- Computational grids
- Data grids, peer-to-peer networks
- Mobile computing, pervasive computing
- Inter-node communications, Instant grids!

What does not scale ?

- Fault tolerance (nodes, network)
- Data circulation (firewalls, rate, quantity)
- Intermittent node management (desktop grids)

Flexible grids need to be aware of new resources in a distributed way.



Grids

Why putting grids around us?

Computational resources are more and more demanding.

- Computational grids
- Data grids, peer-to-peer networks
- Mobile computing, pervasive computing
- Inter-node communications, Instant grids!

What does not scale ?

- Fault tolerance (nodes, network)
- Data circulation (firewalls, rate, quantity)
- Intermittent node management (desktop grids)

Flexible grids need to be aware of new resources in a distributed way.



Fallacies of Distributed Computing

Joy/Lyon/Deutsch/Gosling

- 1 The network is reliable.
- 2 Latency is zero.
- 3 Bandwidth is infinite.
- 4 The network is secure.
- 5 Topology doesn't change.
- 6 There is one administrator.
- 7 Transport cost is zero.
- 8 The network is homogeneous.

(copied from Wikipedia)



Outline

- 1 Resource discovery in grids
 - Motivation
 - Exploiting your neighbourhood
- 2 Performance Analysis
 - Framework
 - Measurements



Living in a pervasive computing world.

Make yourself known!

Communication between random nodes is often limited. We need to build an overlay network.

- Main requirement: announce of existence (broadcast one-to-all);
- Good performance (low latency, congestion management, data rate);
- Topology of the overlay network: full graph, star graph (tree?), ring;
- Already existing: Ethernet level, LAN, VPN... not desktop grid;
- Security in the grid;
- Some initiative already in process: Host Identification Protocol, ZeroConf.



Living in a pervasive computing world.

Make yourself known!

Communication between random nodes is often limited. We need to build an overlay network.

- Main requirement: announce of existence (broadcast one-to-all);
- Good performance (low latency, congestion management, data rate);
- Topology of the overlay network: full graph, star graph (tree?), ring;
- Already existing: Ethernet level, LAN, VPN... not desktop grid;
- Security in the grid;
- Some initiative already in process: Host Identification Protocol, ZeroConf.



Living in a pervasive computing world.

Make yourself known!

Communication between random nodes is often limited. We need to build an overlay network.

- Main requirement: announce of existence (broadcast one-to-all);
- Good performance (low latency, congestion management, data rate);
- Topology of the overlay network: full graph, star graph (tree?), ring;
- Already existing: Ethernet level, LAN, VPN... not desktop grid;
- Security in the grid;
- Some initiative already in process: Host Identification Protocol, ZeroConf.



Living in a pervasive computing world.

Make yourself known!

Communication between random nodes is often limited. We need to build an overlay network.

- Main requirement: announce of existence (broadcast one-to-all);
- Good performance (low latency, congestion management, data rate);
- Topology of the overlay network: full graph, star graph (tree?), ring;
- Already existing: Ethernet level, LAN, VPN... not desktop grid;
- Security in the grid;
- Some initiative already in process: Host Identification Protocol, ZeroConf.



Living in a pervasive computing world.

Make yourself known!

Communication between random nodes is often limited. We need to build an overlay network.

- Main requirement: announce of existence (broadcast one-to-all);
- Good performance (low latency, congestion management, data rate);
- Topology of the overlay network: full graph, star graph (tree?), ring;
- Already existing: Ethernet level, LAN, VPN... not desktop grid;
- Security in the grid;
- Some initiative already in process: Host Identification Protocol, ZeroConf.



Living in a pervasive computing world.

Make yourself known!

Communication between random nodes is often limited. We need to build an overlay network.

- Main requirement: announce of existence (broadcast one-to-all);
- Good performance (low latency, congestion management, data rate);
- Topology of the overlay network: full graph, star graph (tree?), ring;
- Already existing: Ethernet level, LAN, VPN... not desktop grid;
- Security in the grid;
- Some initiative already in process: Host Identification Protocol, ZeroConf.



Living in a pervasive computing world.

Make yourself known!

Communication between random nodes is often limited. We need to build an overlay network.

- Main requirement: announce of existence (broadcast one-to-all);
- Good performance (low latency, congestion management, data rate);
- Topology of the overlay network: full graph, star graph (tree?), ring;
- Already existing: Ethernet level, LAN, VPN... not desktop grid;
- Security in the grid;
- Some initiative already in process: Host Identification Protocol, ZeroConf.



Uniqueness of designation

Be unique!

Resources use unique identifiers necessary for host-to-host communications.

- Second requirement: verification of uniqueness;
- Well-known distributed allocation problem;
- Requires many-to-one (does not scale) or other means (retry in case of conflict);
- No perfect solution (asynchronous) → retries.
- Unique identifier required also for security conscious protocols;
- Good solutions based on host identity for uniqueness.



Network topology

Don't get lost!

Routing through the network is also one of the expected functionality of a grid building infrastructure

- Complete graph: most simple;
- Star graph: used by the difficult to reach;
- Cloud: general internet routing;
- Ring: very efficient for all-to-all communications (not for general purpose communications)

Implementations:

- Kernel level → virtual devices, unmodified applications;
- User-space level → virtual nodes, node-specific process translation (e.g. web services, virtual addresses);
- Application level → does not scale.



Network topology

Don't get lost!

Routing through the network is also one of the expected functionality of a grid building infrastructure

- Complete graph: most simple;
- Star graph: used by the difficult to reach;
- Cloud: general internet routing;
- Ring: very efficient for all-to-all communications (not for general purpose communications)

Implementations:

- Kernel level → virtual devices, unmodified applications;
- User-space level → virtual nodes, node-specific process translation (e.g. web services, virtual addresses);
- Application level → does not scale.



Network topology

Don't get lost!

Routing through the network is also one of the expected functionality of a grid building infrastructure

- Complete graph: most simple;
- Star graph: used by the difficult to reach;
- Cloud: general internet routing;
- Ring: very efficient for all-to-all communications (not for general purpose communications)

Implementations:

- Kernel level → virtual devices, unmodified applications;
- User-space level → virtual nodes, node-specific process translation (e.g. web services, virtual addresses);
- Application level → does not scale.



Network topology

Don't get lost!

Routing through the network is also one of the expected functionality of a grid building infrastructure

- Complete graph: most simple;
- Star graph: used by the difficult to reach;
- Cloud: general internet routing;
- Ring: very efficient for all-to-all communications (not for general purpose communications)

Implementations:

- Kernel level → virtual devices, unmodified applications;
- User-space level → virtual nodes, node-specific process translation (e.g. web services, virtual addresses);
- Application level → does not scale.



Network topology

Don't get lost!

Routing through the network is also one of the expected functionality of a grid building infrastructure

- Complete graph: most simple;
- Star graph: used by the difficult to reach;
- Cloud: general internet routing;
- Ring: very efficient for all-to-all communications (not for general purpose communications)

Implementations:

- Kernel level → virtual devices, unmodified applications;
- User-space level → virtual nodes, node-specific process translation (e.g. web services, virtual addresses);
- Application level → does not scale.



Network topology

Don't get lost!

Routing through the network is also one of the expected functionality of a grid building infrastructure

- Complete graph: most simple;
- Star graph: used by the difficult to reach;
- Cloud: general internet routing;
- Ring: very efficient for all-to-all communications (not for general purpose communications)

Implementations:

- Kernel level → virtual devices, unmodified applications;
- User-space level → virtual nodes, node-specific process translation (e.g. web services, virtual addresses);
- Application level → does not scale.



Outline

- 1 Resource discovery in grids
 - Motivation
 - Exploiting your neighbourhood
- 2 Performance Analysis
 - Framework
 - Measurements



Zeroconf

Bonjour® and Avahi

- Zeroconf is a set of techniques sanctioned by various IETF RFCs:
 - Sets up local IP address (IPV4LL/RFC 3927, IPV6/RFC 2462)
 - Two independent parts: mDNS/DNS-SD.
 - mDNS conveys DNS over multicast (224.0.0.251 port 5353 UDP)
 - DNS-SD: encapsulated in DNS request/replies, provides answers to service discovery.
-  Bonjour
 - Developed by Apple® (initially called Rendezvous)
 - Windows+Java API
-  AVAHI
 - Free (LGPL) alternative to Bonjour.
 - Included in most Linux® distributions.
- Similar works: UPnP/DPWS (Microsoft® effort)

Zeroconf

Bonjour® and Avahi

- Zeroconf is a set of techniques sanctioned by various IETF RFCs:
 - Sets up local IP address (IPV4LL/RFC 3927, IPV6/RFC 2462)
 - Two independent parts: mDNS/DNS-SD.
 - mDNS conveys DNS over multicast (224.0.0.251 port 5353 UDP)
 - DNS-SD: encapsulated in DNS request/replies, provides answers to service discovery.



- Developed by Apple® (initially called Rendezvous)
- Windows+Java API





- Free (LGPL) alternative to Bonjour.
- Included in most Linux® distributions.

- Similar works: UPnP/DPWS (Microsoft® effort)



Zeroconf

Bonjour® and Avahi

- Zeroconf is a set of techniques sanctioned by various IETF RFCs:
 - Sets up local IP address (IPV4LL/RFC 3927, IPV6/RFC 2462)
 - Two independent parts: mDNS/DNS-SD.
 - mDNS conveys DNS over multicast (224.0.0.251 port 5353 UDP)
 - DNS-SD: encapsulated in DNS request/replies, provides answers to service discovery.
-  Bonjour
 - Developed by Apple® (initially called Rendezvous)
 - Windows+Java API
-  AVAHI
 - Free (LGPL) alternative to Bonjour.
 - Included in most Linux® distributions.
- Similar works: UPnP/DPWS (Microsoft® effort)



Zeroconf

Bonjour® and Avahi

- Zeroconf is a set of techniques sanctioned by various IETF RFCs:
 - Sets up local IP address (IPV4LL/RFC 3927, IPV6/RFC 2462)
 - Two independent parts: mDNS/DNS-SD.
 - mDNS conveys DNS over multicast (224.0.0.251 port 5353 UDP)
 - DNS-SD: encapsulated in DNS request/replies, provides answers to service discovery.

-  Bonjour

- Developed by Apple® (initially called Rendezvous)
- Windows+Java API

-  **AVAHI**



- Free (LGPL) alternative to Bonjour.
- Included in most Linux® distributions.

- Similar works: UPnP/DPWS (Microsoft® effort)



Zeroconf

Bonjour® and Avahi

- Zeroconf is a set of techniques sanctioned by various IETF RFCs:
 - Sets up local IP address (IPV4LL/RFC 3927, IPV6/RFC 2462)
 - Two independent parts: mDNS/DNS-SD.
 - mDNS conveys DNS over multicast (224.0.0.251 port 5353 UDP)
 - DNS-SD: encapsulated in DNS request/replies, provides answers to service discovery.
-  Bonjour
 - Developed by Apple® (initially called Rendezvous)
 - Windows+Java API
-  **AVAHI**
 - Free (LGPL) alternative to Bonjour.
 - Included in most Linux® distributions.
- Similar works: UPnP/DPWS (Microsoft® effort)



Pastry

« La cerise sur le gâteau »

Pastry (<http://freepastry.rice.edu/>)

- Framework for peer-to-peer applications
- Builds the overlay network and abstracts fault-tolerance
- Provides routing and load balancing
- Free-Pastry is a free implementation
- Works over the Internet
- Can also transfer data efficiently



Pastry

« La cerise sur le gâteau »

Pastry (<http://freepastry.rice.edu/>)

- Framework for peer-to-peer applications
- Builds the overlay network and abstracts fault-tolerance
- Provides routing and load balancing
- Free-Pastry is a free implementation
- Works over the Internet
- Can also transfer data efficiently



Pastry

« La cerise sur le gâteau »

Pastry (<http://freepastry.rice.edu/>)

- Framework for peer-to-peer applications
- Builds the overlay network and abstracts fault-tolerance
- Provides routing and load balancing
- Free-Pastry is a free implementation
- Works over the Internet
- Can also transfer data efficiently





Grid'5000

French experimental Grid platform

Grid 5000:

- has been mentioned already
- is a research effort developing a large scale nation wide infrastructure for Grid research

What we did:

- Compilation of a specific kernel/distribution with support for Pastry, Avahi and Bonjour.
- Reservation of 308 nodes on Orsay site. Runs of all measurements with complete logging of timings.





Grid'5000

French experimental Grid platform

Grid 5000:

- has been mentioned already
- is a research effort developing a large scale nation wide infrastructure for Grid research

What we did:

- Compilation of a specific kernel/distribution with support for Pastry, Avahi and Bonjour.
- Reservation of 308 nodes on Orsay site. Runs of all measurements with complete logging of timings.



Outline

- 1 Resource discovery in grids
 - Motivation
 - Exploiting your neighbourhood
- 2 Performance Analysis
 - Framework
 - **Measurements**



Sequential registration

- Registering one service per node;
- Large-enough delay $\delta > \mu_R$ between the registrations;
- Registration time on the node between start of registration/end of registration (network latency for multicasting/acknowledgement);



Sequential registration

- Registering one service per node;
- Large-enough delay $\delta > \mu_R$ between the registrations;
- Registration time on the node between start of registration/end of registration (network latency for multicasting/acknowledgement);



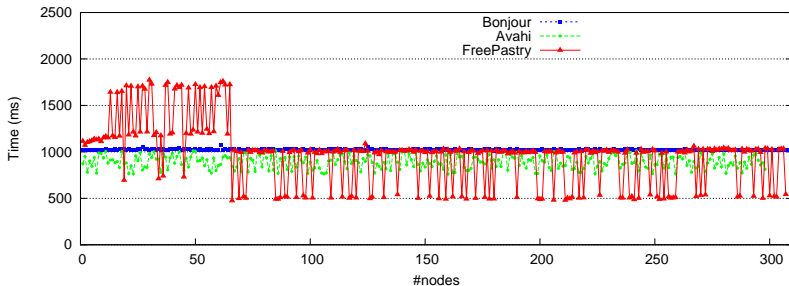
Sequential registration

- Registering one service per node;
- Large-enough delay $\delta > \mu_R$ between the registrations;
- Registration time on the node between start of registration/end of registration (network latency for multicasting/acknowledgement);



Sequential registration

- Registering one service per node;
- Large-enough delay $\delta > \mu_R$ between the registrations;
- Registration time on the node between start of registration/end of registration (network latency for multicasting/acknowledgement);



Simultaneous registration

- Registering one service per node;
- Very small delay $\delta \approx 0$ between the registrations;
- Registration time on the node between start of registration/end of registration;
- Free-Pastry performance collapse (right scale $140\times$ larger than left scale)

Simultaneous registration

- Registering one service per node;
- Very small delay $\delta \approx 0$ between the registrations;
- Registration time on the node between start of registration/end of registration;
- Free-Pastry performance collapse (right scale $140\times$ larger than left scale)



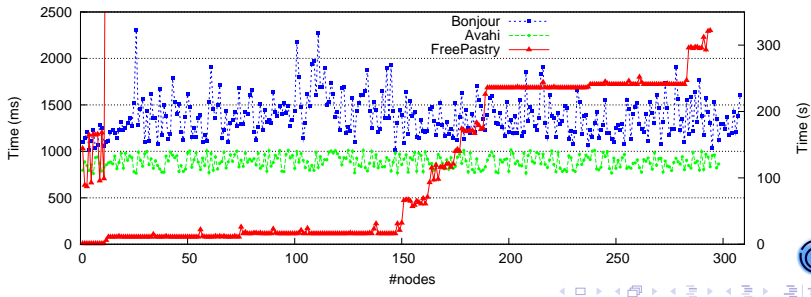
Simultaneous registration

- Registering one service per node;
- Very small delay $\delta \approx 0$ between the registrations;
- Registration time on the node between start of registration/end of registration;
- Free-Pastry performance collapse (right scale $140\times$ larger than left scale)



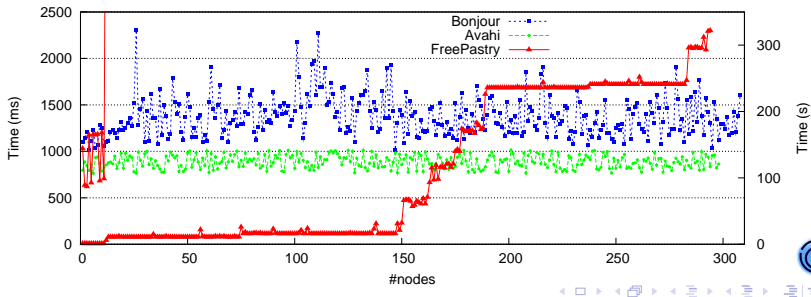
Simultaneous registration

- Registering one service per node;
- Very small delay $\delta \approx 0$ between the registrations;
- Registration time on the node between start of registration/end of registration;
- Free-Pastry performance collapse (right scale $140\times$ larger than left scale)



Simultaneous registration

- Registering one service per node;
- Very small delay $\delta \approx 0$ between the registrations;
- Registration time on the node between start of registration/end of registration;
- Free-Pastry performance collapse (right scale $140\times$ larger than left scale)



Browsing time

- Registering one service per node;
- Measurement of time between service registration and discovery per first-node and browsing process;

Bonjour No loss of information; discovery time less than 1s.

Avahi Heavy loss of registered services in simultaneous registration (60%) and time goes to 220 s; sequential registration much better (less than 2 s).

FreePastry About 3% losses in all registration modes.



Browsing time

- Registering one service per node;
- Measurement of time between service registration and discovery per first-node and browsing process;

Bonjour No loss of information; discovery time less than 1s.

Avahi Heavy loss of registered services in simultaneous registration (60%) and time goes to 220 s; sequential registration much better (less than 2 s).

FreePastry About 3% losses in all registration modes.



Browsing time

- Registering one service per node;
- Measurement of time between service registration and discovery per first-node and browsing process;

Bonjour No loss of information; discovery time less than 1s.

Avahi Heavy loss of registered services in simultaneous registration (60%) and time goes to 220 s; sequential registration much better (less than 2 s).

FreePastry About 3% losses in all registration modes.



Browsing time

- Registering one service per node;
- Measurement of time between service registration and discovery per first-node and browsing process;

Bonjour No loss of information; discovery time less than 1s.

Avahi Heavy loss of registered services in simultaneous registration (60%) and time goes to 220 s; sequential registration much better (less than 2 s).

FreePastry About 3% losses in all registration modes.



Browsing time

- Registering one service per node;
- Measurement of time between service registration and discovery per first-node and browsing process;

Bonjour No loss of information; discovery time less than 1s.

Avahi Heavy loss of registered services in simultaneous registration (60%) and time goes to 220 s; sequential registration much better (less than 2 s).

FreePastry About 3% losses in all registration modes.



Summary

- No service is better **for all stats** than the other.
- **No source code** for Bonjour.
- **Heavy loss of messages** for Avahi.
- **Performance collapse** for Freepastry with a great number of registrations.
- Perspectives
 - Blinking registrations, error bars.
 - Overlay networks (openHIP) to cross the link-local barrier.
 - Investigate UPnP and other discovery protocols.



Summary

- No service is better **for all stats** than the other.
- **No source code** for Bonjour.
- **Heavy loss of messages** for Avahi.
- **Performance collapse** for Freepastry with a great number of registrations.
- Perspectives
 - Blinking registrations, error bars.
 - Overlay networks (openHIP) to cross the link-local barrier.
 - Investigate UPnP and other discovery protocols.



Summary

- No service is better **for all stats** than the other.
- **No source code** for Bonjour.
- **Heavy loss of messages** for Avahi.
- **Performance collapse** for Freepastry with a great number of registrations.
- Perspectives
 - Blinking registrations, error bars.
 - Overlay networks (openHIP) to cross the link-local barrier.
 - Investigate UPnP and other discovery protocols.



Summary

- No service is better **for all stats** than the other.
 - **No source code** for Bonjour.
 - **Heavy loss of messages** for Avahi.
 - **Performance collapse** for Freepastry with a great number of registrations.
-
- Perspectives
 - Blinking registrations, error bars.
 - Overlay networks (openHIP) to cross the link-local barrier.
 - Investigate UPnP and other discovery protocols.



Summary

- No service is better **for all stats** than the other.
 - **No source code** for Bonjour.
 - **Heavy loss of messages** for Avahi.
 - **Performance collapse** for Freepastry with a great number of registrations.
-
- Perspectives
 - Blinking registrations, error bars.
 - Overlay networks (openHIP) to cross the link-local barrier.
 - Investigate UPnP and other discovery protocols.



Summary

- No service is better **for all stats** than the other.
- **No source code** for Bonjour.
- **Heavy loss of messages** for Avahi.
- **Performance collapse** for Freepastry with a great number of registrations.

- Perspectives
 - Blinking registrations, error bars.
 - Overlay networks (openHIP) to cross the link-local barrier.
 - Investigate UPnP and other discovery protocols.



Summary

- No service is better **for all stats** than the other.
- **No source code** for Bonjour.
- **Heavy loss of messages** for Avahi.
- **Performance collapse** for Freepastry with a great number of registrations.

- Perspectives
 - Blinking registrations, error bars.
 - Overlay networks (openHIP) to cross the link-local barrier.
 - Investigate UPnP and other discovery protocols.



For Further Reading I



Steinberg and Cheshire.

Zero Configuration Networking: The Definitive Guide.

O'Reilly Media, Inc., first edition, December 2005.



Rezmerita, Morlier, Néri and Cappello.

Private virtual cluster: Infrastructure and protocol for instant grids.

In Euro-Par 2006, Parallel Processing, volume 4128 of LNCS, Springer.



Abbes, Cérin, Dubacq and Jemni.

Performance Analysis of Publish/Subscribe Systems.

<https://hal.ccsd.cnrs.fr/docs/00/15/93/88/PDF/acdj.pdf>.

Merci

Thank you

شكراً

