In-transit analytics on distributed Clouds: applications and architecture

Omer F. Rana
School of Computer Science & Informatics
Cardiff University, UK
ranaof@cardiff.ac.uk

Structure

• Characterising “Big Data” applications
  – Application focus
  – What are these applications
• Common themes across applications
• Supporting in-transit analytics
  – An architecture for undertaking this
  – A Model for supporting in-transit analytics
  – Executing the model
Distributed, Dynamic, Data Intensive Applications

- Applications where data workload is equivalent to or greater than computational workload

- Properties of “dynamic” data applications
  - Real time: generated, re-distributed or partitioned data
  - On-demand: varying availability of data
  - Adaptive: change in processing or storage granularity

- Computational activities triggered due to data creation
  - Computation must respond to unplanned changes in data volumes or content

- To scale – various trade-offs necessary:
  - Move computation or data?
  - Frequency of access impacts various modes of analysis
  - Increasing interest in “in-memory” processing (high disk I/O overhead)
  - Exploitation of the full memory hierarchy (RAM, SSD, Disk) + other buffering strategies

Considered a variety of applications that conform to the above characteristics

Types of applications

- Variety of applications in multimedia streaming
  - e.g YouTube for multimedia streaming, xoIP (x=TV, Voice, etc)
  - Sensor networks (emergency response, security, environment, etc)
  - Processing requirements vary – over different timeframes

- Not just true for physical sciences
  - increasingly social scientists also face similar challenges (e.g. tension indicators in communities)

- Increasing availability of data over the Web and from government departments
  - Data from Facebook, Twitter, Flickr (text, audio, video, etc)
    - People as sensors
  - Data from government agencies – Police API, Demographic data (ONS), etc
Application Comparison

- Across four axes:
  (i) **Execution units** (sequential or parallel – varying granularity)
  (ii) **Communication** (messages, files, streaming, pub/sub, data reduction or shared data)
  (iii) **Coordination mechanism** (data/control flow, SPMD, master-worker, events, tuple space models) and
  (iv) **Execution environment** (captures the deployment environment: dynamic process/task creation, workflow execution, Web services, messaging (MPI), co-scheduling, data streaming, asynchronous data I/O, etc)

- Investigate characteristics in each of the above axes that would enable “adaptive behaviour”

---

<table>
<thead>
<tr>
<th>Application</th>
<th>Area</th>
<th>Lead Person/Site</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metagenomics + NGS</td>
<td>Biosciences</td>
<td>Amsterdam Medical Centre, Netherlands</td>
</tr>
<tr>
<td>ATLAS experiment (WLCG)</td>
<td>Particle Physics</td>
<td>CERN &amp; Daresbury Lab + RAL, UK</td>
</tr>
<tr>
<td>Large Synoptic Sky Survey (LSST)</td>
<td>Astrophysics</td>
<td>University of Edinburgh – Institute of Astronomy, UK</td>
</tr>
<tr>
<td>Virtual Observatory</td>
<td>Astrophysics</td>
<td>University of Edinburgh – Institute of Astronomy, UK</td>
</tr>
<tr>
<td>Cosmic Microwave Background</td>
<td>Astrophysics</td>
<td>Lawrence Berkeley National Laboratory, USA</td>
</tr>
<tr>
<td>Marine (Sea Mammal) Sensors</td>
<td>Biosciences</td>
<td>University of St. Andrews, UK, Scottish Oceans Institute</td>
</tr>
<tr>
<td>Climate/Earth System Grid</td>
<td>Earth Science</td>
<td>National Center for Atmospheric Research, USA</td>
</tr>
</tbody>
</table>
## Applications

<table>
<thead>
<tr>
<th>Application</th>
<th>Scientific Area</th>
<th>Lead Person/Site</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interactive Exploration of Environmental Data</td>
<td>Earth Science</td>
<td>University of Reading &amp; RAL, UK</td>
</tr>
<tr>
<td>Power Grids</td>
<td>Energy Informatics</td>
<td>University of Southern California, USA</td>
</tr>
<tr>
<td>Fusion (International Thermonuclear Experimental Reactor)</td>
<td>Chemistry/Physics</td>
<td>Oak Ridge National Laboratory &amp; Rutgers University, USA</td>
</tr>
<tr>
<td>Industrial Incident Notification and Response</td>
<td>Emergency Response Management</td>
<td>THALES, The Netherlands</td>
</tr>
<tr>
<td>MODIS Data Processing</td>
<td>Earth Science</td>
<td>Lawrence Berkeley National Laboratory, USA</td>
</tr>
<tr>
<td>Floating Sensors</td>
<td>Earth Science</td>
<td>Lawrence Berkeley National Laboratory, USA</td>
</tr>
<tr>
<td>Distributed Network Intrusion Detection</td>
<td>Security + Data Analytics</td>
<td>University of Minnesota, USA</td>
</tr>
</tbody>
</table>

---

**Application 1:**

Data Streaming and Complex Event Processing

---

**Bañares, José Ángel, Rana, Omer, Tolosana-Calasanz, Rafael and Pham, Congduc.** "Revenue creation for rate adaptive stream management in multi-tenancy environments". Lecture Notes in Computer Science 8193, pp. 122-137. Springer.


**Rafael Tolosana-Calasanz, José Á. Bañares, Omer Rana, Congduc Pham, Erotokritos Xydas, Charalampos Marmaras, Panagiotis Papadopoulos and Liana Cipcigan.** "Enforcing QoS on OpenNebula-based Shared Clouds for Highly Dynamic, Large-Scale Sensing Data Streams", to be presented at DPMSS workshop (from Sensor Networks to Clouds), at 14th IEEE/ACM Int. Symp. On Cluster, Cloud and Grid Computing (CCGrid), Chicago, May 2014.
Case study scenario

- Skelton building annual average electricity use per day

- 10 passenger cars (24 kWh battery), 5 maintenance vans (55 kWh battery)
- 20% battery SoC available for V2G
- Recharging for 20 miles per day in car EV and 40 miles per day in van
- Solar scenarios 0 kWp, 30 kWp, 60 kWp, 90 kWp
Case study scenario

• Skelton building annual average electricity use per day

http://www.eandfes.co.uk/

Case study scenario

• Skelton building annual average electricity use per day

http://www.eandfes.co.uk/

• 0 kWp Solar installation

http://www.eandfes.co.uk/

Case study scenario

• Skelton building annual average electricity use per day

http://www.eandfes.co.uk/

• 30 kWp Solar installation
Case study scenario

- Skelton building annual average electricity use per day

- 60 kWp Solar installation
- Solar PV tracking

http://www.eandfes.co.uk/

---

Case study scenario

- Skelton building annual average electricity use per day

- 90 kWp Solar installation
- Solar PV tracking

http://www.eandfes.co.uk/
Data Collection for Brokerage

Application 2:
Analysing social media data

Conejero, Javier, Rana, Omer, Burnap, Peter, Morgan, Jeffrey, Carrion, Carmen and Caminero, Blanca, “Characterising the power consumption of Hadoop Clouds: A social media analysis case study”. CLOSER 2013: 3rd International Conference on Cloud Computing and Services Science, Aachen, Germany, 8-10 May 2013.


Social Media Analysis

- Significant quantities of data generated from social media (… but “ethical” usage important)
  - Twitter: “firehose” (100%), “gardenhose” (10%), “spritzer” (1%)
  - Facebook status updates
- Integrating this data with other sources
  - ONS (in the UK) + other curated data
  - Maps related: (various options: Open Street Maps, Google Maps, Yahoo! Placefinder etc)
- Raw data not significant
  - Looking for particular types of “events” of interest
- Common analysis types
  - Sentiment and Opinion analysis
  - Connectivity between content generators
- Collaborative On-line Social Media Observatory (COSMOS)
  - “Tension” indicators in terrestrial and on-line communities
  - Integrating data with other (conventional) indicators

http://www.cosmosproject.net/
Streaming real-time data from 16.92.4.26
COSMOS: Architecture

- COSMOS integrates a variety of different services: Gender analysis, sentiment analysis, Open Street maps
- Can be integrated with user supplied services

Hadoop Cloud Architecture

- Hadoop cluster makes use of a number of VMs
- VMs are coordinated through the KVM Hypervisor
- Data is periodically staged to VM instances within the cluster
- Alter the number of VMs per machine
Common Theme: Pipelines

- **Existence of “pipelines”**
  - Stream/In-Memory analysis

- **Pipelines integrate various (distributed) applications (across different infrastructure)**
  - Build pipelines from “macro” components (“legacy” codes)

- **Pipeline stages have different emphasis**
  - Pre-Collect and store all in-coming data
  - Reduce size of incoming data stream

- **Data-driven pipeline execution**
  - Based on data-streaming – “event extraction”
  - Inclusion of “sensing” into the pipeline

- **Multiple, co-existing, concurrent pipelines**
  - Superscalar pipelines

```
Abstractions: Data flow “process networks” (actors and firing rules); Coordination: Pub-Sub + Events, Tuple Space models; Implementations: Yahoo Pipes!, Storm (bolts and sprouts, stream groups and topology), Pachube/Xively Cloud (Rate limited); Functional approaches: SCALA,Streamflow and Xbaya; Databases: EVE, Degub, Calder; SummingBird (used with Storm and Scalding), Commercial: Amazon Kinesis; Samza, Cascading, S4, Spark Cluster/Streaming, Google DataFlow/Millwheel; In Memory: Druid, VoltDB, MemSQL, NuoDB
```

---

**Pipelined Execution Semantics**

1) **Best Effort**

```
Best Effort: Finished -> Ready -> Busy -> Ready
```

2) **Blocking**

```
Blocking: Finished -> Blocked -> Ready
```

3) **Buffered**

```
Buffered: Finished -> Busy -> Ready
```

4) **Superscalar**

```
Superscalar: Ready -> Finished -> Busy -> Ready
```

5) **Streaming**

```
Streaming: Busy -> Busy -> Busy
```

---

"Parallel Computing Patterns for Grid Workflows"
Autonomic Streaming Pipeline

- **Streaming pipeline**
  - No “blocking” semantics
  - Continuous data transmission as a stream
  - Data processing order: arrival order (implicit) or time stamp (explicit)
  - After processing – result elements form the stream

- **Autonomic streaming**
  - Data stream “reacts” to changes in (operating) environment and producer/consumer data generation/consumption rate mismatch
  - Network congestion ➔ alter transmission data rate
  - Alternative modes of analysis: in-transit, at-source, at-sink, etc

In-transit Analysis

- Data processing while data is in movement from source to destination
- Question: what to process where and when
- Use of “slack” in network to support partial processing
- Application types:
  - Streaming & Data Fusion requirement
Reference Nets + Renew + DVega

- **“Nets-within-nets”**
  - Systems net and an object net
  - Net can express creation of new net instances ("creational inscriptions") – enabling dynamic self-modification of structure
  - Interaction via "synchronous channels"
  - Channel can contain variables whose binding is based on unification
  - Timed reference nets:
    - Time stamp attached to tokens
    - Use of timed inscriptions on arcs (control time stamps and firing delays)

- **Renew**
  - Java-based interpreter of Reference nets (an executable formalism)
  - Use tuples and Java expression as the inscription language
  - Objects nets can be Java objects

- **DVega**
  - Workflow engine based on Reference nets
  - Utilises a Linda-tuple space model for interaction between tasks in a workflow (referred to as a "message space")
Control Strategy + Adaptation

- Reference net model executes alongside real system
- Model used to tune behaviour
- Rule-based Reasoner coupled with other machine learning strategies

Adapting Transfer Rates based on Network Congestion

Lamda: data generation rate; B: bandwidth; omega: hard disk transfer rate
Network congestion added: intervals 11-24; control interval: 10 secs.
Adding in-transit processing nodes

deltaU: change in processing rate (i.e. number of data items processed/time)

Superscalar Pipelines and Rate Adaptation

R. Tolosana-Calasanz, J. B. Bañares, O. Rana, C. Pham and O. Rana
“Enforcing QoS in Scientific Workflow Systems Enacted Over Cloud Infrastructures”
Journal of Computer and System Science (Elsevier), 2012
Isolating multiple concurrent pipelines

Multiple input streams with different QoS demands

Input buffer

PU with dynamic capacity

output buffer

Manage input to guarantee QoS

Manage local resources of PU to guarantee QoS

Modify mu, omega, select routes, add in-transit proc.

Token-bucket (traffic shaping)

• Used to control the rate of traffic entering a component
• Influenced by the availability of processing units and outgoing rate

![Policing](image1.png)

• A **policer** typically drops excess traffic.

![Shaping](image2.png)

• A **shaper** typically delays excess traffic using a buffer to hold data and shape the flow when the data rate of the source is higher than expected.
Token-bucket based Rate Control

Key parameters of interest:
- \( b \) – token bucket size (depth); \( R \) – token generation rate; \( C \) – maximum arrival rate (e.g., link capacity)
- \( R \): specifies how much data can be sent or forwarded per unit time on average
- \( b \): specifies, for each burst, how much data can be sent within a given time

System Architecture

- 3 key components / node: Token Bucket, Processing Unit & output streaming

Scenario I: Gold customer takes additional resources

NO ADDITION OF RESOURCES FOR GOLDEN

ADDITION OF RESOURCES FOR GOLD CUSTOMERS
Integration with OpenNebula

Traffic shaping achieved through the use of a Token Bucket manager. A token bucket for each data stream

Monitoring number of accumulating packets in an intermediate buffer – triggers creation of new VM instances

OpenNebula 4.4 (32 physical nodes), 32GB/node, 8 cores/node. 4VMs (8VMs) send packets to 20VMs (40VMs) with 5 processes/VM, at 400 packets/s

<table>
<thead>
<tr>
<th>Packet Size</th>
<th>30 YMs Receiving Packets</th>
<th>40 YMs Receiving Packets</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Jitter (Mean)</td>
<td>Jitter (Mean)</td>
</tr>
<tr>
<td>8KB</td>
<td>2.00</td>
<td>2.00</td>
</tr>
<tr>
<td>16KB</td>
<td>3.00</td>
<td>3.00</td>
</tr>
<tr>
<td>32KB</td>
<td>3.00</td>
<td>3.00</td>
</tr>
<tr>
<td>64KB</td>
<td>3.00</td>
<td>3.00</td>
</tr>
</tbody>
</table>

VM cross talk and dynamic VM creation

Analytics in Federated Clouds

- Capability not available at a single provider
  - Cost of access, lack of trust in data transfer
- Avoids vendor lock-in by a single provider
- Security & fault tolerance
  - Single failure or breach does not affect site
  - Many providers operate in multiple regions (availability zones)
- Last mile issue
  - Limited benefit of moving everything to a single site
  - Lesson learned in Content Distribution Networks (e.g. Akamai)
  - Significant recent interest in Software Defined Networks
Developing Multi-Layered Federated Clouds

Multiple data access and processing layers
Deciding what to do where – creation of a “decision function”
Different objectives: L3: power, range; L2: stream aggregation; L1: throughput
(use of “Software Define Networks” at L2)
No need to migrate “raw” data to Cloud systems

Overview of the CometCloud Space

• Virtual shared space abstraction
  – Based on application properties
  – Mapped onto a set of peer nodes

• The space is accessible by all system nodes.
  – Access is independent of the physical locations of data tuples or hosts

• Coordination/interaction through the shared spaces
  – Runtime management, push/pull scheduling and load-balancing

• Dynamically constructed transient spaces enable application to exploit context locality
On-Demand Federation using CometCloud

- Cross-layer federation management using user and provider policies
- Federation is coordinated using Comet spaces at two levels
  
  - Management space
    - Orchestrate resources in the federation
    - Interchange operational messages
  
  - Shared execution spaces
    - Created on demand by agents
    - Provision local resources and connect to public clouds or external HPC systems


Implementation

- Requirements for a site to join the federation:
  - Java support
  - Valid credentials (authorized SSH keys)
  - Configure some parameters (i.e. address, ports, number of workers)

- Resources

<table>
<thead>
<tr>
<th>Resources</th>
<th>Cardiff</th>
<th>Rutgers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Machines</td>
<td>12</td>
<td>32</td>
</tr>
<tr>
<td>Core per Machine</td>
<td>12</td>
<td>8</td>
</tr>
<tr>
<td>Memory</td>
<td>12 GB</td>
<td>6 GB</td>
</tr>
<tr>
<td>Network</td>
<td>1 GbE</td>
<td>Infiniband</td>
</tr>
</tbody>
</table>

- Indiana Site
  - Uses FutureGrid (OpenStack, Infiniband interconnect, 2 cores/machine with 4GB memory) – also supports Cloudmesh Teefaa and Rain
**EnergyPlus and Building Optimisation**

- Real time optimisation of building energy use
  - sensors provide readings within an interval of 15-30 minutes,
  - Optimisation run over this interval
- The efficiency of the optimisation process depends on the capacity of the computing infrastructure
  - deploying multiple EnergyPlus simulations
- Closed loop optimisation
  - Set control set points
  - Monitor/acquire sensor data + perform analysis with EnergyPlus
  - Update HVAC and actuators in physical infrastructure

---

**Federated Clouds in Building Optimisation**

---

EnergyPlus and Building Optimisation
**Federation constraints**

Two metrics:
- Time to complete
- Results quality

Trading quality of results vs. overall simulation time

- Each Master decides how to compute the received job:
  - (i) where to compute the tasks:
    - (a) Single CometCloud or (b) federated CometCloud;
  - (ii) how many combinations to run giving the deadline received from the user.

**Evaluation**

- In our experiments we use two different configurations
  - (a) single cloud context where all the tasks have to be processed locally
  - (b) federation cloud context where the sites have the option of outsourcing tasks to remote sites.

- We use as inputs for our calculation
  - (i) CPU time of remote site as the amount of time spent by each worker to computer the tasks and
  - (ii) storage time on remote site as the amount of time needed to store data remotely.

- All the costs have been calculated in £ derived from Amazon EC2 cost.
Experiment 1: Job completed

Table III: Input Parameters: Experiment 1

<table>
<thead>
<tr>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
<th>Deadline</th>
</tr>
</thead>
<tbody>
<tr>
<td>16,18,20,22,24</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>1 Hour</td>
</tr>
</tbody>
</table>

Table IV: Results: Experiment 1

<table>
<thead>
<tr>
<th></th>
<th>Single Cloud</th>
<th>Federated Cloud</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nodes</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>Cost</td>
<td>£0</td>
<td>£7.46</td>
</tr>
<tr>
<td>Tasks</td>
<td>36</td>
<td>38</td>
</tr>
<tr>
<td>Deadline</td>
<td>1 hour</td>
<td>1 hour</td>
</tr>
<tr>
<td>Tuples exchanged</td>
<td>-</td>
<td>15</td>
</tr>
<tr>
<td>CPU on remote site</td>
<td>-</td>
<td>5626.45 Sec</td>
</tr>
<tr>
<td>Storage on remote site</td>
<td>-</td>
<td>1877.10 Sec</td>
</tr>
<tr>
<td>Completed tasks</td>
<td>34/38</td>
<td>38/38 in 55min 40s</td>
</tr>
</tbody>
</table>

- the federation site has two options: (i) run tasks on the local infrastructure (single cloud case) or (ii) outsource some tasks to a remote site (federation cloud case).
- A corresponding deadline of 1 hour, only 34 out of 38 can be completed.
- In the federation in 55 minutes by outsourcing 15 to the remote site.
- The process of outsourcing has an associated cost of 7.46 £.

Experiment 2: Job uncompleted

Table V: Input Parameter: Experiment 2

<table>
<thead>
<tr>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
<th>Deadline</th>
</tr>
</thead>
<tbody>
<tr>
<td>16,17,18,19,20,21,22,23,24</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>1 Hour</td>
</tr>
</tbody>
</table>

Table VI: Results: Experiment 2

<table>
<thead>
<tr>
<th></th>
<th>Single Cloud</th>
<th>Federated Cloud</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nodes</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>Cost</td>
<td>£0</td>
<td>£7.90</td>
</tr>
<tr>
<td>Tasks</td>
<td>72</td>
<td>72</td>
</tr>
<tr>
<td>Deadline</td>
<td>1 hour</td>
<td>1 hour</td>
</tr>
<tr>
<td>Tuples exchanged</td>
<td>-</td>
<td>15</td>
</tr>
<tr>
<td>CPU on remote site</td>
<td>-</td>
<td>5637.27 Sec</td>
</tr>
<tr>
<td>Storage on remote site</td>
<td>-</td>
<td>1869.41 Sec</td>
</tr>
<tr>
<td>Completed tasks</td>
<td>37/72</td>
<td>58/72</td>
</tr>
</tbody>
</table>

- In the context of single cloud federation (3 workers) only 37 out of 72 tasks are completed within the deadline of 1 hour.
- Exchanging 15 tuples between the two federation sites, with increased cost for execution and storage.
• we extend the deadline associated to 1 hour and 30 minutes
• when using the federation to outsource a percentage of tasks we observe that the number of tasks completed increases to 62

Summary of results

Experiment 3: Job uncompleted--parameters ranges extended:

Table VII: Input Parameters: Experiment 3

<table>
<thead>
<tr>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
<th>Deadline</th>
</tr>
</thead>
<tbody>
<tr>
<td>14,15,16,17,18,19,20</td>
<td>0,1</td>
<td>0,1</td>
<td></td>
<td>1h 30 min</td>
</tr>
<tr>
<td>21,22,23,24,25,26,27</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table VIII: Results: Experiment 3

<table>
<thead>
<tr>
<th></th>
<th>Single Cloud</th>
<th>Federated Cloud</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nodes</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>Cost</td>
<td>0</td>
<td>£ 10.70</td>
</tr>
<tr>
<td>Tasks</td>
<td>112</td>
<td>72</td>
</tr>
<tr>
<td>Deadline</td>
<td>1 h 30 min</td>
<td>1 h 30 min</td>
</tr>
<tr>
<td>Tuples exchanged</td>
<td>-</td>
<td>22</td>
</tr>
<tr>
<td>CPU on remote site</td>
<td>-</td>
<td>7983.74 sec</td>
</tr>
<tr>
<td>Storage on remote site</td>
<td>-</td>
<td>2687.15 sec</td>
</tr>
<tr>
<td>Completed tasks</td>
<td>42/112</td>
<td>62/112</td>
</tr>
</tbody>
</table>
Conclusion …

- Emergence of data-driven + data intensive applications
- Use of Cloud/data centres and edge nodes collectively
- Pipeline-based enactment a common theme
  - Various characteristics – buffer management and data coordination
  - Model development that can be integrated into a workflow environment
- Automating application adaptation
  - … as infrastructure changes
  - … as application characteristics change

3DPAS theme -- people

- Shantenu Jha (Rutgers University, USA and University of Edinburgh, UK) – theme coordinator
- J. D. Blower (Reading University, UK)
- Neil Chue Hong (Software Sustainability Institute, University of Edinburgh, UK),
- Simon Dobson (University of St Andrews, UK)
- Daniel S. Katz (Computation Institute, University of Chicago, USA & NSF Cyberinfrastructure, USA),
- Andre Luckow (Louisiana State University, USA)
- Omer Rana (Cardiff University, UK)
- Yogesh Simmhan (University of Southern California, USA)
- Jon Weissman (University of Minnesota, USA)
Collaborators ...

- **COSMOS**: Jeffrey Morgan, Peter Burnap, William Housley, Matthew Williams, Adam Edwards, Nick Avis (Cardiff)
- Ioan Petri (Cardiff, CS)
- Yacine Rezgui, Haijiang Li, Tom Beach (Cardiff ENGIN)
- Manish Parashar, Javier Diaz-Montes, Mengsong Zou (Rutgers University, US)
- Rafael Tolosana, Ricardo Rodriegez and Jose Banares (University of Zaragoza, Spain)
- Congduc Pham (University of Pau, France)
- Andreas Hoheisel (Fraunhofer Institute, Germany)