The Bronze-Standard application

Principle, deployment and optimization on EGEE

Tristan Glatard

Johan Montagnat

Xavier Pennec

EGEE and SEE-GRID Summer School on Grid Application Support

Friday, June 29th

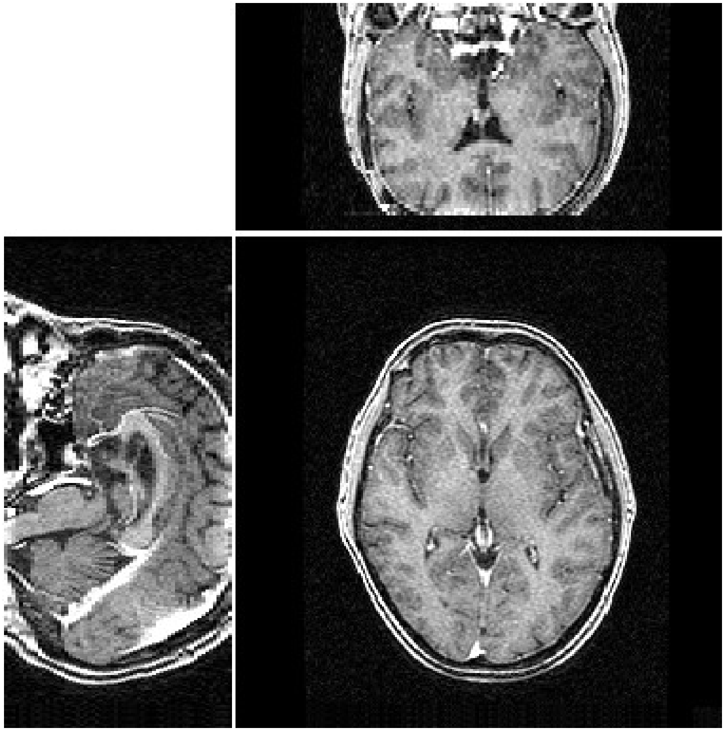
Outline

- **The Bronze-Standard application (10 min)**
  - The medical imaging context
  - Goals and methods
- **Grid deployment of the application (15 min)**
  - Grid challenges
  - Workflow deployment
- **Performance issues and optimization (15 min)**
  - Importance of the latency on EGEE
  - Timeout optimization

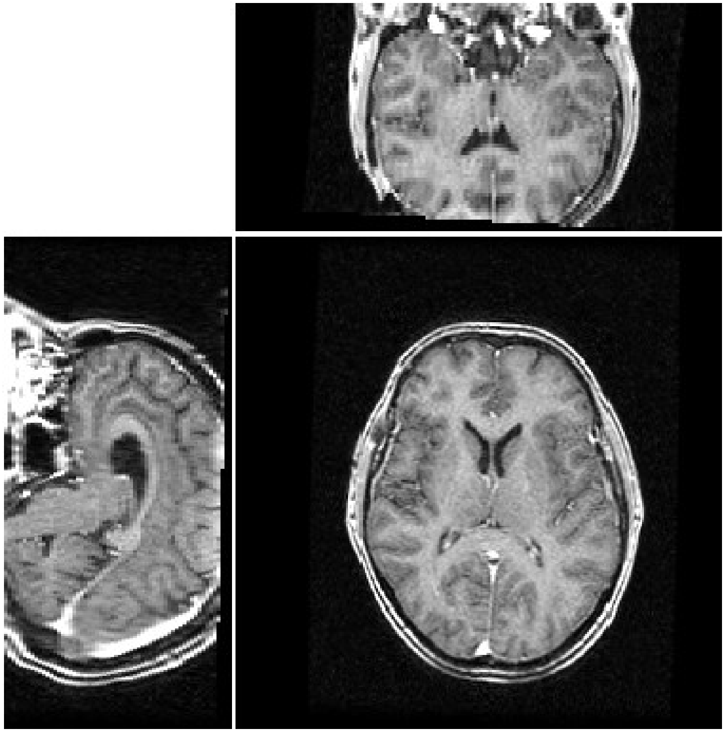
Medical image registration

- **Medical images registration**
  - Goal: fusing two images acquired in different frames
  - Input data : a target image and a floating image
  - Output data : a transformation and a result image

Before registration



After registration



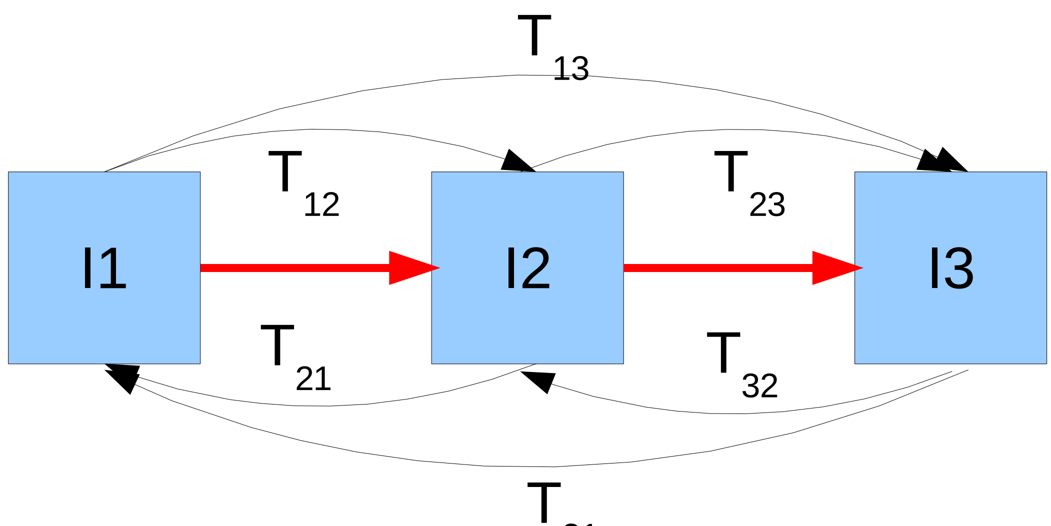
Performance evaluation of registration

- **Simulation of noisy data:**
  - Apply transformation (ground truth)
  - Measure how far the result is from the truth
- **Real data on controlled environment**
  - Imaging a physical phantom
  - Gold standard: measure the motion of the phantom
- **Database of in-vivo real images**
  - Representative of the clinical application
  - Span all sources of variability
  - No gold standard

The Bronze Standard idea

- **N images, m algorithms**
- **N.(N-1).m transformations measured**
- **N-1 transformations to estimate**

Redundancy



- **Exploit redundancy to compute**
  - Mean transformations $\overline{T}_{ij}$ (Bronze standard)
  - Variances on the transformations (Accuracy)

The Bronze Standard method

- **The $\bar{T}_{ij}$ transformations minimize:**

$$\sum_{i,j\in[1,n],k\in[1,m]} d\left(T_{i,j}^{k}, \bar{T}_{i,j}\right)^2$$

- **Norm on the rigid transformations:**

$$\mu^2(R(\theta,n),t) = \frac{\theta^2}{\sigma_{\mathrm{r}}^2} + \frac{\|t\|^2}{\sigma_{\mathrm{t}}^2}$$

- **Robust di**

$$d(T_1,T_2) = \min\left(\mu^2(T_1^{(-1)} \circ T_2)\ ,\ \chi^2\right)$$

The Bronze Standard method

- **The $\bar{T}_{ij}$ transformations minimize:**

$$\sum_{i,j\in[1,n],k\in[1,m]} d\left(T_{i,j}^{k}, \bar{T}_{i,j}\right)^{2}$$

- **Norm on the rigid transformations:**

$$\mu^2\left(R(\theta,n),t\right) = \frac{\theta^2}{\sigma_{\mathrm{r}}^2} + \frac{\|t\|^2}{\sigma_{\mathrm{t}}^2}$$

The transformation

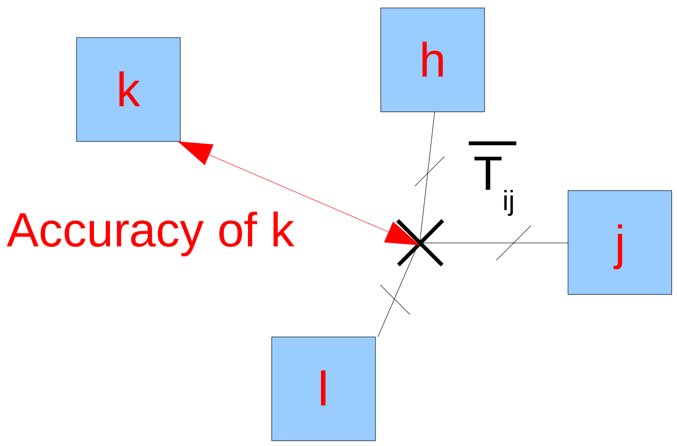
variances

- **Robust distance on the transformations:**

$$d(T_1,T_2) = \min\left(\mu^2(T_1^{(-1)}\circ T_2)\ ,\ \chi^2\right)$$

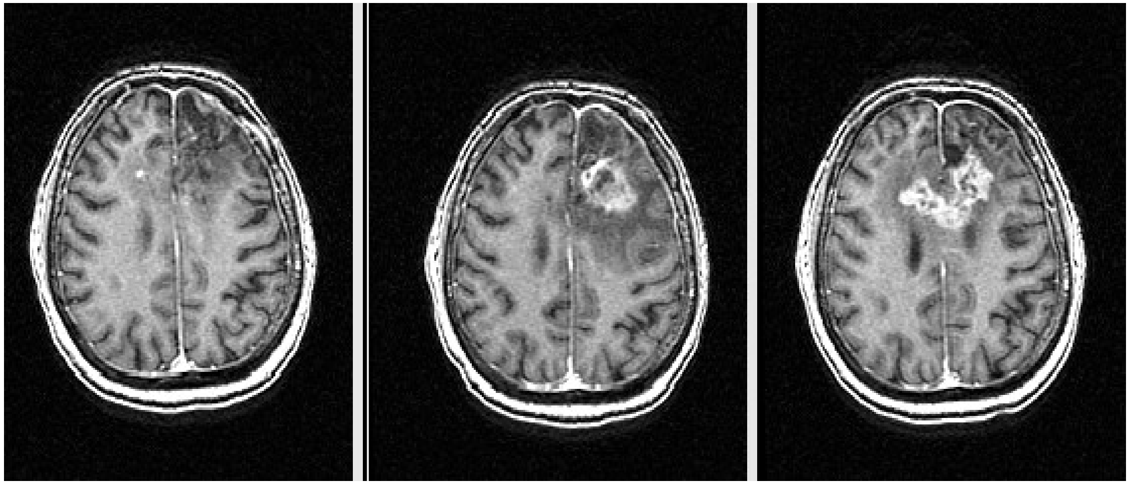
Assessing the algorithms

- **The accuracy of an algorithm k is given by:**
  - Computing the $\overline{T}_{ij}$ without k's results
  - Computing distances to $\overline{T}_{ij}$ for k's transformations



Clinical use-case

- **Follow-up of brain radiotherapy**
  - Requires several registrations
  - Precision of the tumor evolution estimation required
- **Image database**
  - 29 patients
  - 2 time points minimum
  - Gadolinium injected T1 MRIs
  - Example for one patient (3 time points):



t_1 t_2 t_3

Registration algorithms assessed

- **Rigid registration algorithms**
- **Feature-based (crest lines):**
  - CrestMatch
  - PFRegister (robust version)
- **Intensity-based:**
  - Baladin (bloc matching)
  - Yasmina (Powell optimization)
- **Initialized with CrestMatch's result:**
  - Ensures that all the algorithms converge to the same minimum
  - Measure of the accuracy

Accuracy results

- **Mean error on the transformations:**

$$\sigma_r = 0.130 \text{ deg} ; \sigma_\tau = 0.345 \text{ mm}$$

- **Error on the bronze standard:**

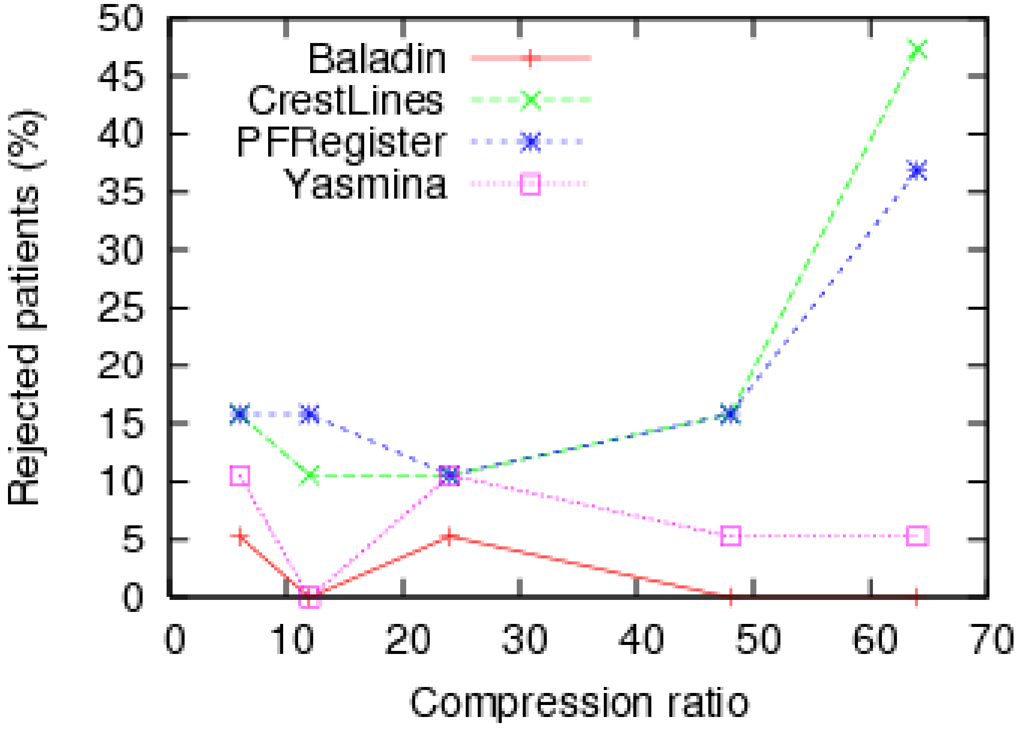
$$\sigma_r = 0.05 \text{ deg} ; \sigma_\tau = 0.148 \text{ mm}$$

- **Accuracy of the algorithms:**

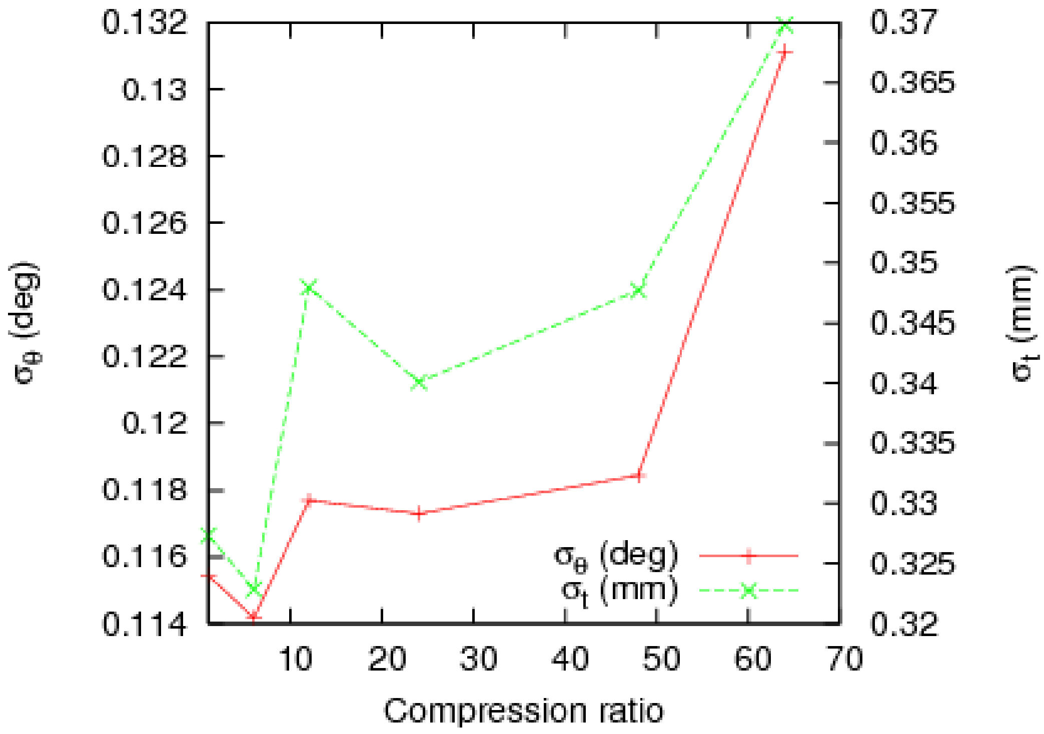
Algorithm	$\sigma_{\mathrm{r}}(deg)$	$\sigma_{\mathrm{t}}(mm)$
CrestMatch	0.150	0.424
PFRegister	0.180	0.416
Baladin	0.139	0.395
Yasmina	0.137	0.445

Impact of lossy compression

- **Robustness**

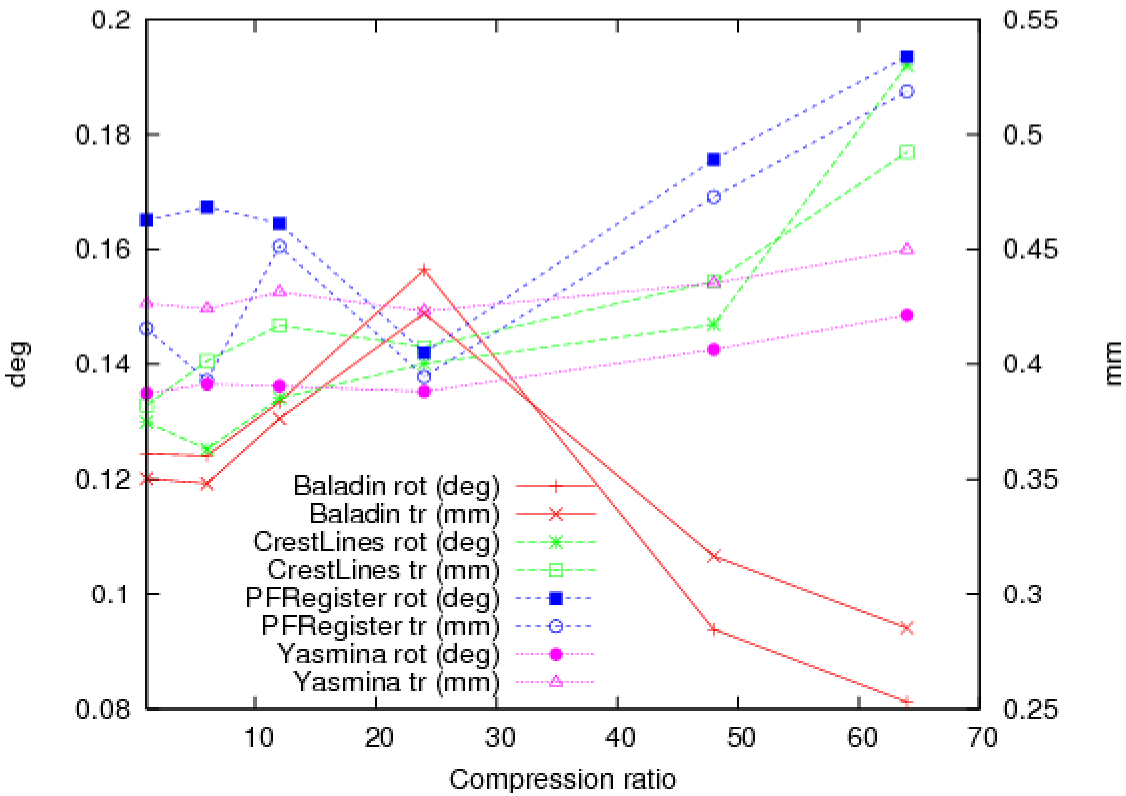


- **Repeatability**



Impact of lossy compression

- **Accuracy results:**



Outline

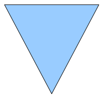
- **The Bronze-Standard application (10 min)**
  - The medical imaging context
  - Goals and methods
- **Grid deployment of the application (15 min)**
  - Grid challenges
  - Workflow deployment
- **Performance issues and optimization (15 min)**
  - Importance of the latency on EGEE
  - Timeout optimization

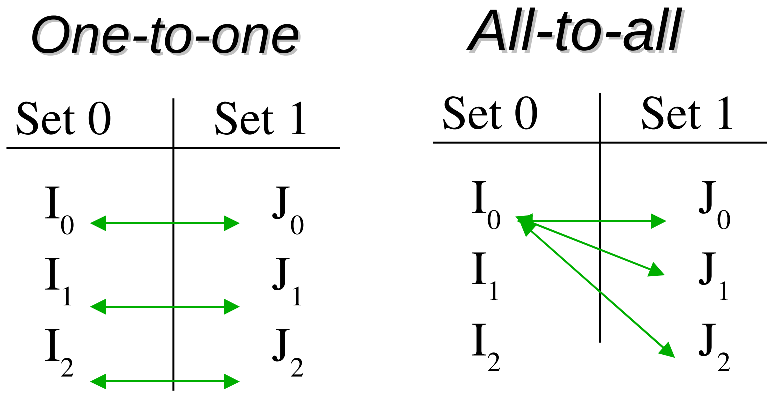
Grid challenges of the application

- **Constraints/needs:**
  1) Sharing algorithms from different institutes
  2) Sharing the data between the algorithms
  3) Computing power

- **Solutions:**
  1) Workflow of services
  2) Data storage on SE inside a VO
  3) Optimized grid execution

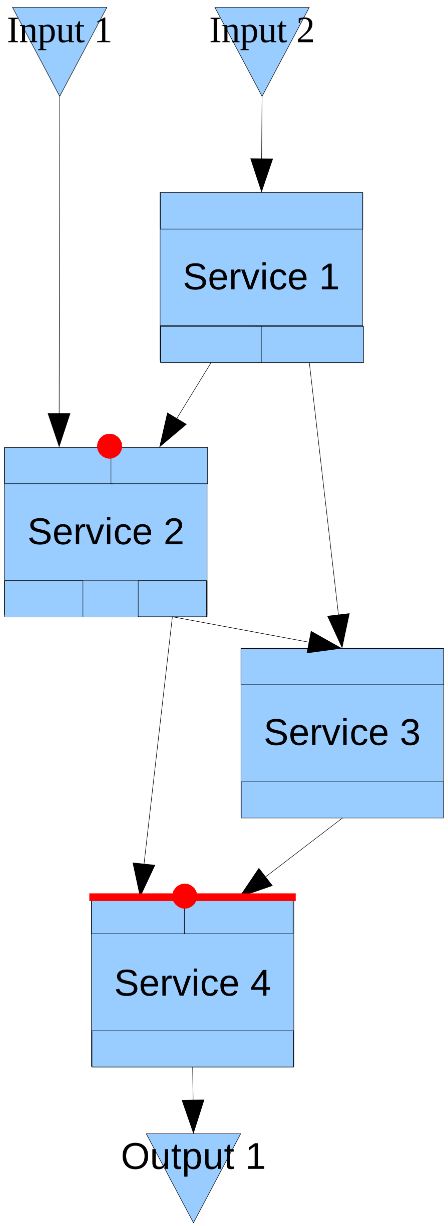
Service-based workflows

- **Graph description:**
  - Input/output of the application 
  - Data dependencies between services
  - Iteration strategy between services inputs

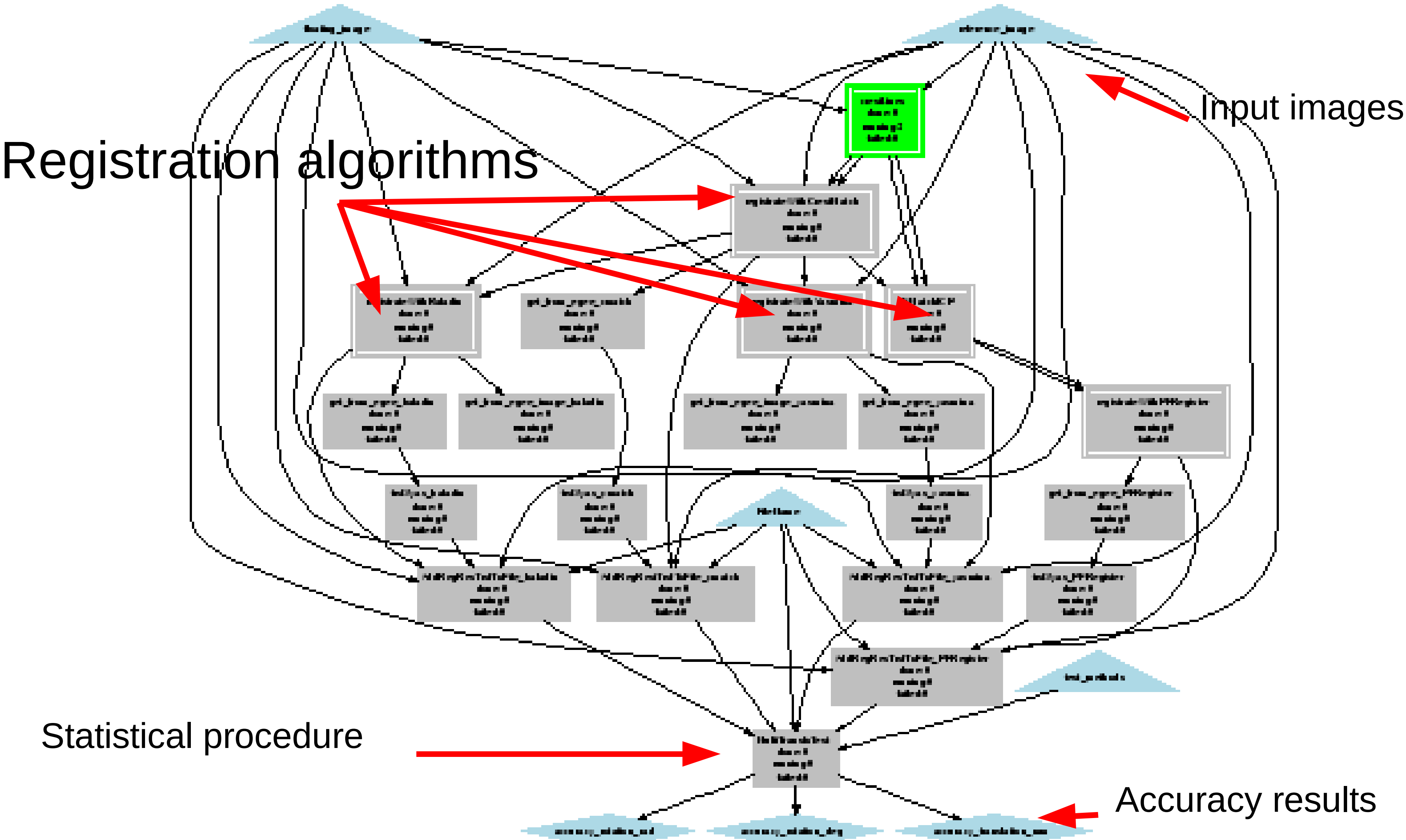


  - Data synchronization barriers

- **Instantiation on data *at execution time***



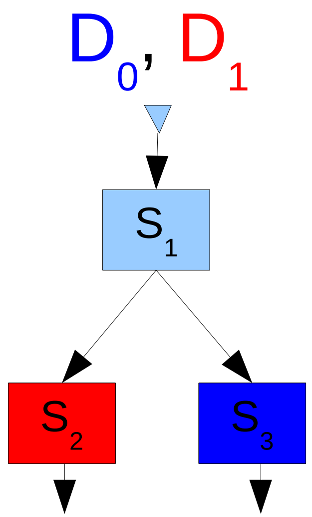
Workflow of the application



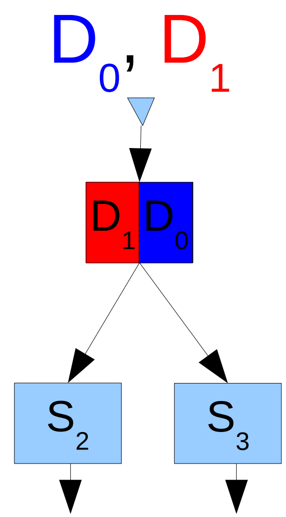
Parallelism in service workflows

- **3 kinds of parallelism can be exploited:**

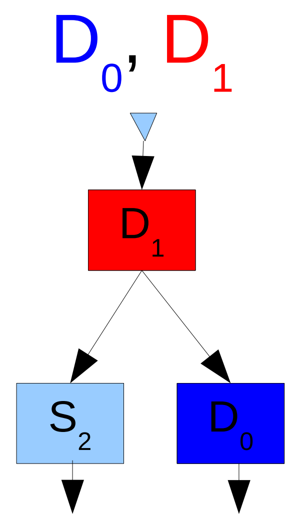
Workflow parallelism



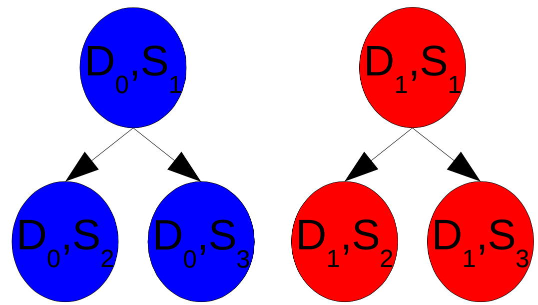
Data parallelism



Services parallelism (pipelining)

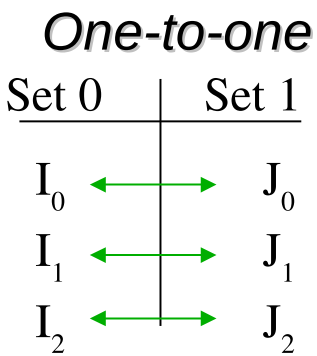


- **Data and service parallelism are intrinsic in task graphs:**



Iteration strategies in a parallel WF

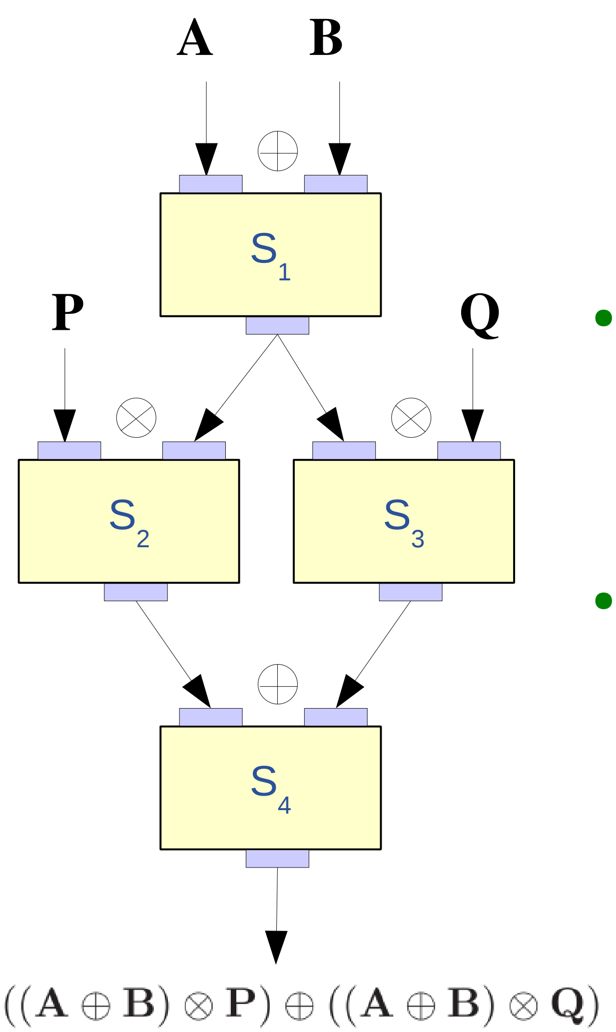
- **One-to-one operators assume ordered data set**



- **No problem if:**
  - Data parallelism is not present (order is preserved)
  - Service parallelism is not present

- **One to one operator in a data+service parallel execution:**
  - Keep track of the data graph
  - Two data segments are composed **iif** they are correlated
  - Correlation groups are defined by the user

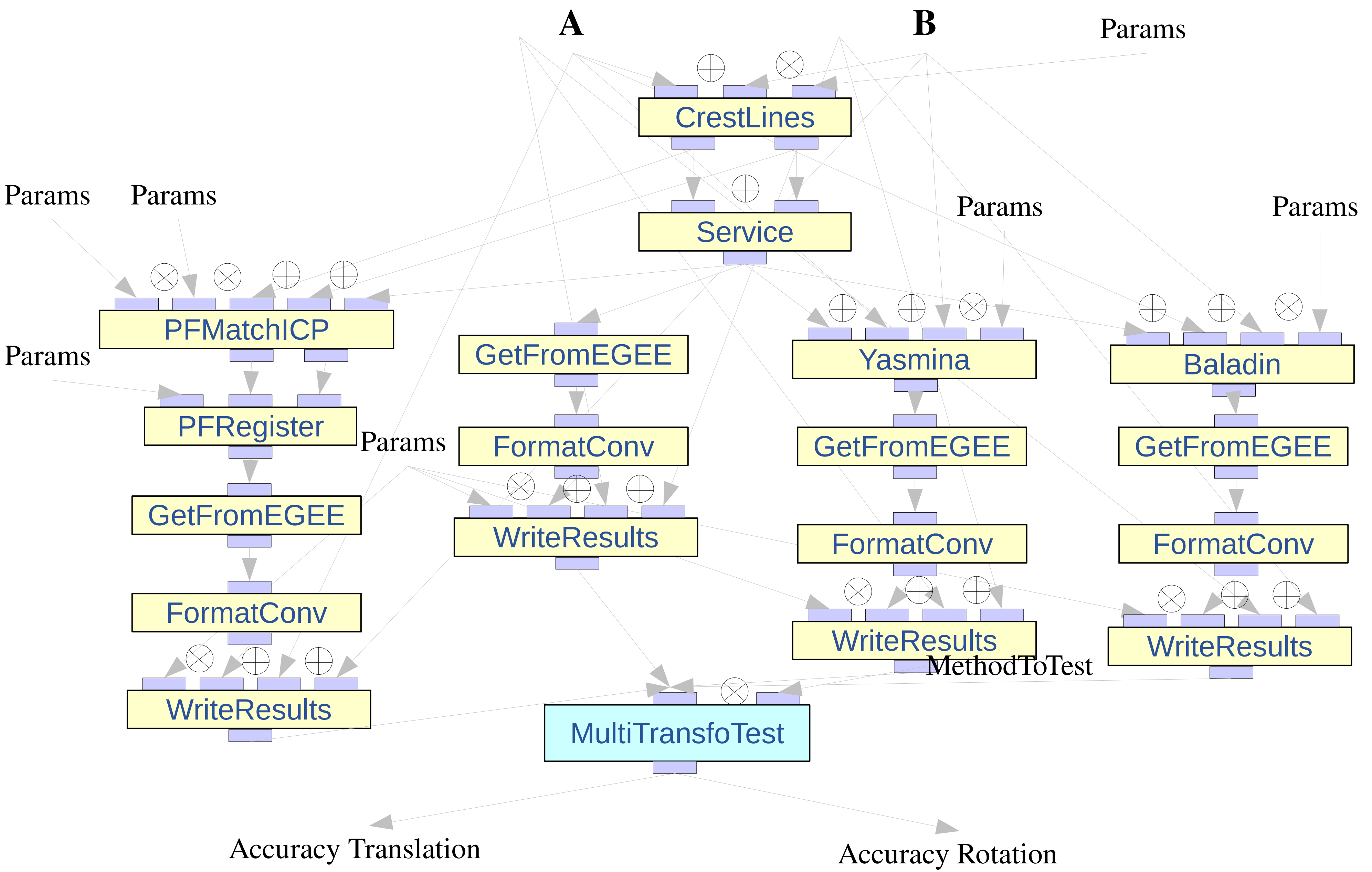
Explicit correlation through groups



- **The user defines correlation groups:**
  - $\mathbf{G} = \{(A_0, B_0), (A_1, B_1), \ldots\}$
  - No relation between A_i and P_k
- **Service S_1:**
  - $\oplus$ composition: A_i and B_j combined iff i=j
- **Service S_4:**
  - $(A_i \oplus B_i) \otimes P_k$ and $(A_j \oplus B_j) \otimes Q_m$ combined iff i=j
  - $((A_i \oplus B_i) \otimes P_k) \oplus ((A_i \oplus B_i) \otimes Q_m)$ for all k and m

(slide from J.Montagnat)

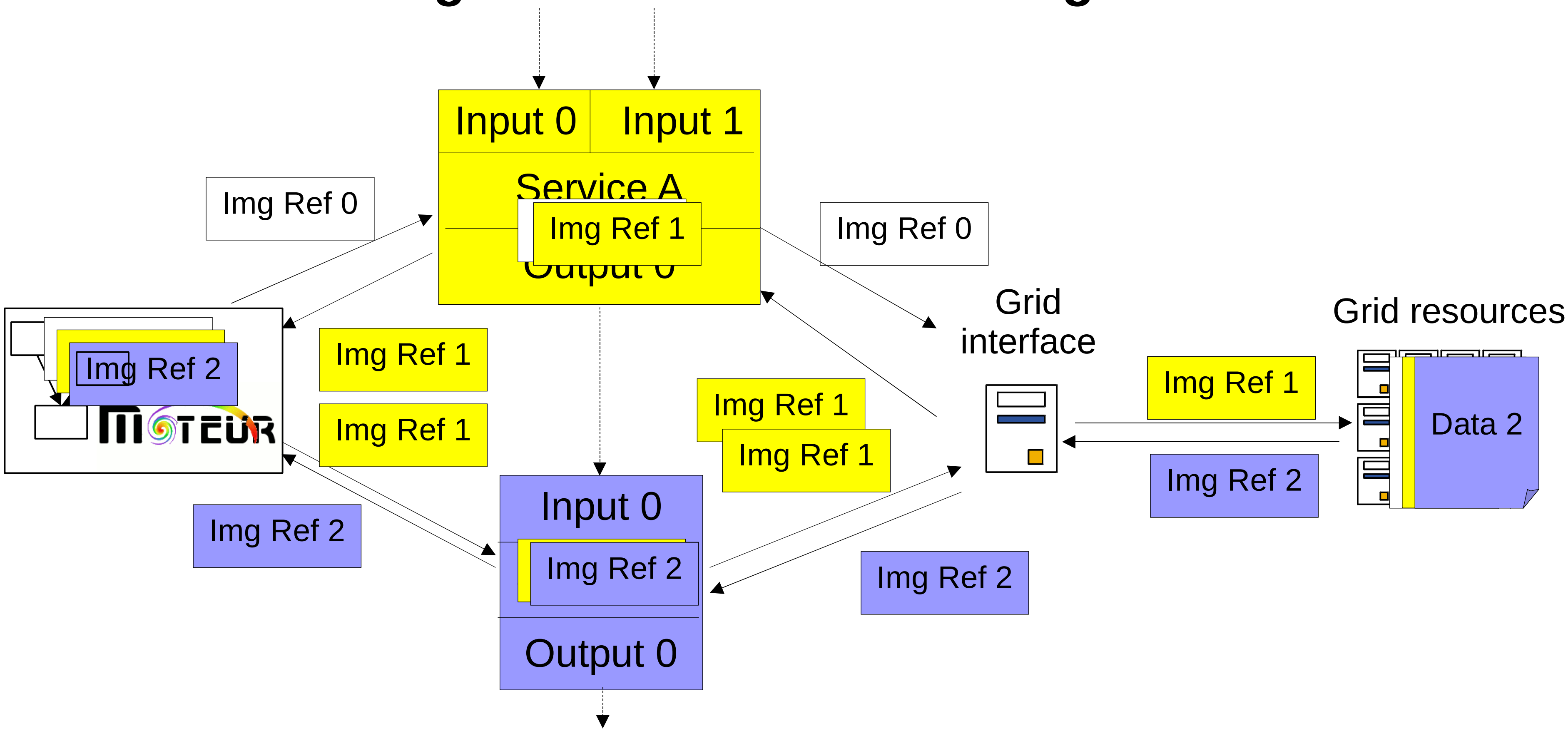
Bronze Standard application example



(slide from J.Montagnat)

Execution on EGEE

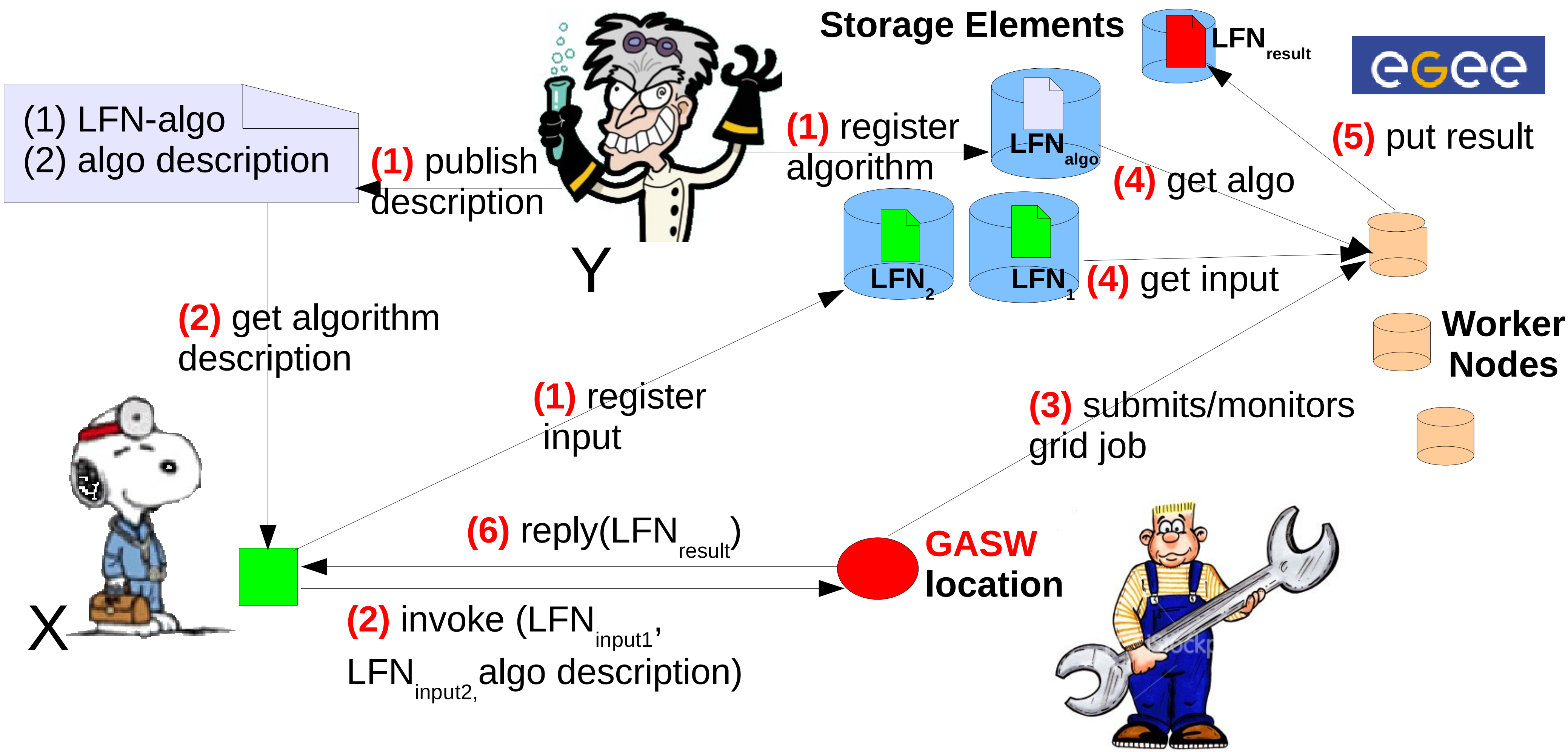
- **Development of MOTEUR, a parallel service workflow engine**
- **The workflow engine is isolated from the grid:**



- **Application codes need to be wrapped in Web-Services**

Generic Application Service Wrapper

- **The grid job handling can be decoupled from Y**



GASW algorithm descriptor

- **Executable access method:**
  - URL
  - Grid file
- **Input/Output**
  - Command-line options
  - Access methods (for files)
- **Sandbox files access method**

```
<description>
  <executable name="CrestLines.pl">
    <access type="URL">
        <path value="http://somewhere.eu/"/>
    </access>
    <value value="CrestLines.pl"/>

     <input name="image" option="-im1">
        <access type="LFN" />
    </input>
    <input name="scale" option="-s"/>
    <output name="crest_lines" option="-c2">
        <access type="LFN" />
    </output>

     <sandbox name="convert8bits">
        <access type="URL">
          <path value="http://elsewhere.dk/"/>
        </access>
        <value value="Convert8bits.pl"/>
    </sandbox>
  </executable>

</description>
```

GASW algorithm descriptor

- **Executable access method:**
  - URL
  - Grid file
- **Input/Output**
  - Command-line options
  - Access methods (for files)
- **Sandbox files access method**

```
<description>
  <executable name="CrestLines.pl">
    <access type="URL">
        <path value="http://somewhere.eu/"/>
    </access>
    <value value="CrestLines.pl"/>

     <input name="image" option="-im1">
        <access type="LFN" />
    </input>
    <input name="scale" option="-s"/>
    <output name="crest_lines" option="-c2">
        <access type="LFN" />
    </output>

     <sandbox name="convert8bits">
       <access type="URL">
         <path value="http://elsewhere.dk/"/>
       </access>
       <value value="Convert8bits.pl"/>
    </sandbox>
  </executable>

</description>
```

GASW algorithm descriptor

- **Executable access method:**
  - URL
  - Grid file
- **Input/Output**
  - Command-line options
  - Access methods (for files)
- **Sandbox files access method**

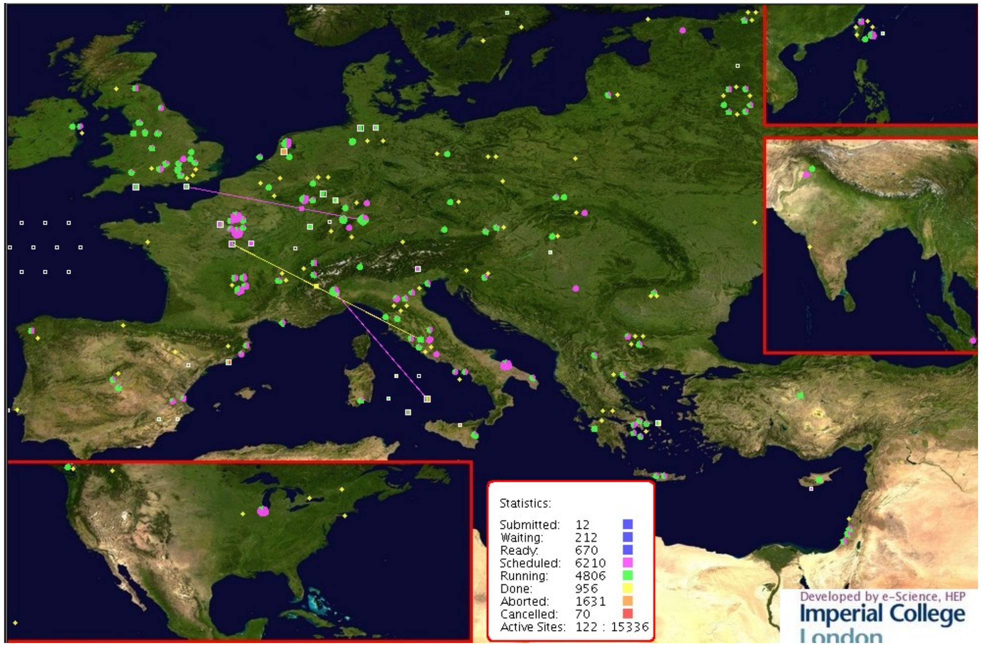
```
<description>
  <executable name="CrestLines.pl">
    <access type="URL">
          <path value="http://somewhere.eu/"/>
    </access>
    <value value="CrestLines.pl"/>

     <input name="image" option="-im1">
          <access type="LFN" />
    </input>
    <input name="scale" option="-s"/>
    <output name="crest_lines" option="-c2">
          <access type="LFN" />
    </output>

     <sandbox name="convert8bits">
         <access type="URL">
            <path value="http://elsewhere.dk/"/>
         </access>
         <value value="Convert8bits.pl"/>
    </sandbox>
  </executable>

</description>
```

Outline

- **The Bronze-Standard application (10 min)**
  - The medical imaging context
  - Goals and methods
- **Grid deployment of the application (15 min)**
  - Grid challenges
  - Workflow deployment
- **Performance issues and optimization (15 min)**
  - Importance of the latency on EGEE
  - Timeout optimization

Optimization on EGEE

- **Production grid:**
  - 50 countries
  - 237 clusters / 36500 CPU
  - 23 PB storage
  - 5000 users



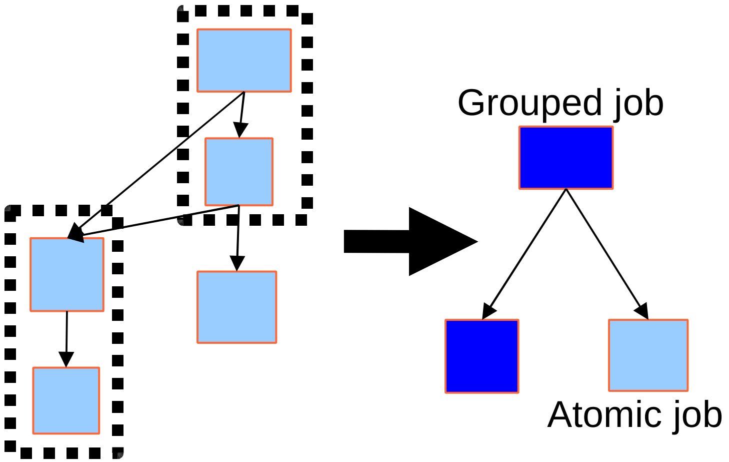
- **High throughput**

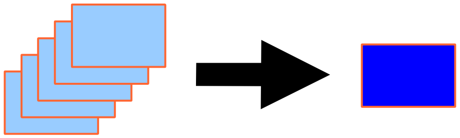
- **High latency**
  - Duration between submission and execution
  - ≈ 5 min +/- 5 min

- **Coming from**
  - Large-scale (network overheads, faults)
  - Multi-users (resources shared between users)

Latency reduction solutions

- **Reducing latency at the workflow level**
  - Grouping sequential jobs
    - \+ Lowers the size of the critical path
    - \- Increases job sizes
  - Grouping parallel jobs
    - \+ Lowers the impact of latency *variability*
    - \- Reduces parallelism





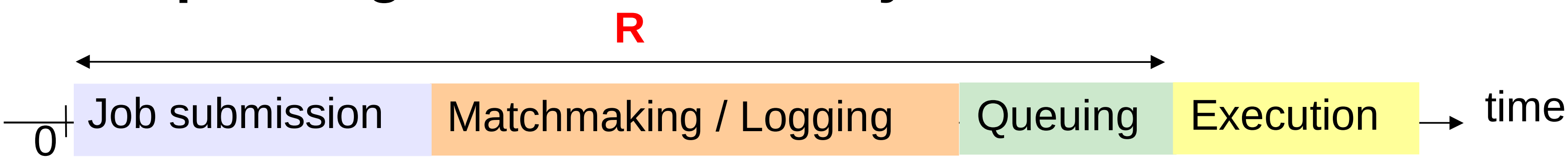
- **Reducing latency at the job level**
  - Redundant submissions
    - \+ Lowers the impact of variability
    - \- Scaling problem
  - Timeout and resubmission

Probabilistic approach

- **Objective: to minimize latency pay-off**
  - Time-out and resubmission
- **Model the job latency**
  - Compute expected execution time
- **Take into account the complexity of the system**
  - Difficult to provide deterministic modeling
  - Probabilistic modeling
- **Adapt to different system behaviors**
  - Highly reliable clusters
  - More error-prone grids

Grid latency modeling

- **Normal operating mode modeled by a random variable R**



  – Distribution of **R** supposed to be estimated (from off-line measures)

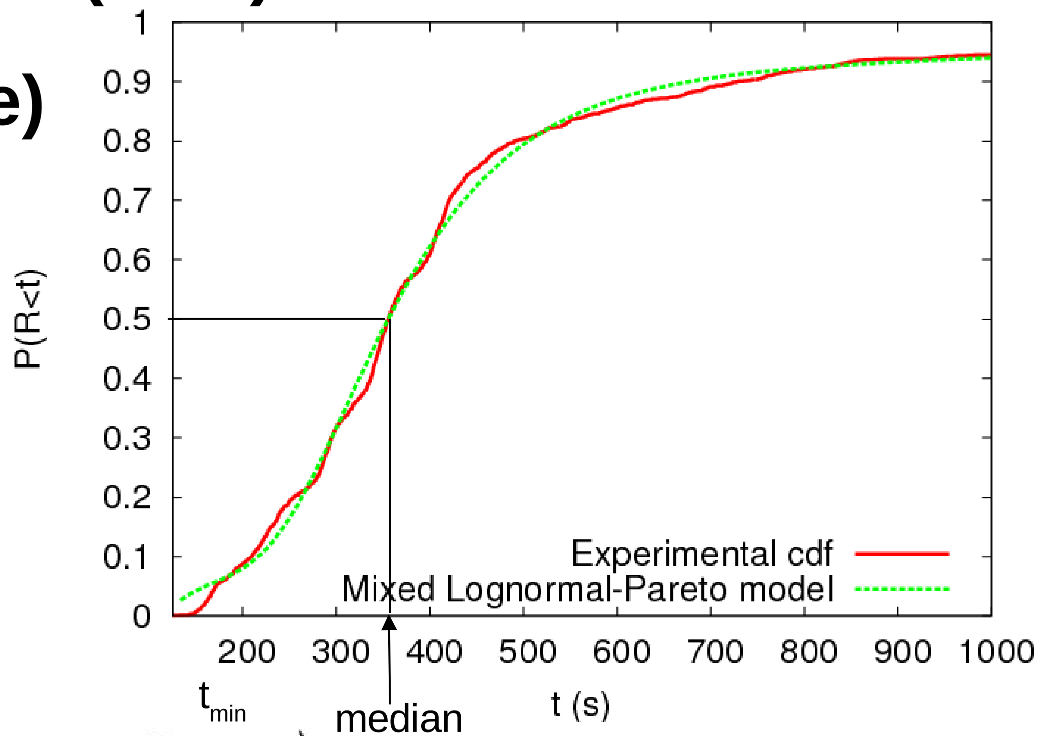
- **Faults modeled by an outlier ratio ρ**

- **Outliers may come from:**
  – Hardware failures and software bugs
  – Locally heavy load
  – Scheduling errors

- **Example on the EGEE production infrastructure:**
  – Measured outlier ratio ρ ≈ 2.5%
  – Mixed Log-normal/Pareto model for **R**

Model of the latency on EGEE

- **Cumulative density function (c.d.f) of R**



- **Data acquisition (red curve)**
  - Probe jobs (`/bin/hostname`)
  - Constant number in the system
  - Total: 2137 jobs
  - Outliers threshold: 10.000s

- **Model fitting (green curve)**

$$P^{model}(R<t) = \underbrace{(1-\alpha(t))\,\Phi\left(\frac{\ln(t-t_{\min})-\mu}{\sigma}\right)}_{\text{Body: log-normal}} + \underbrace{\alpha(t)\left(1-\left(\frac{a}{a+t}\right)^{\nu}\right)}_{\text{Tail: Pareto}}$$

  - $\alpha(t_{min})=0$; $\alpha(t_{max})=1$
  - Least-square minimization / K-S test for validation

Timeout optimization

- **Hypotheses**
  - Timeout => cancellation + resubmission
  - Neglect Cancel/Resubmit cost
  - Neglect Cancel/Resubmit overload => independent submissions

- **Execution time from i^th^ submission to completion**

Wall-clock time (r) — Latency in normal mode (R)

$$J_i = \begin{cases} r + R & \text{with probability } 1 - q \\ t_\infty + J_{i+1} & \text{with probability } q \end{cases}$$

Timeout value (t_∞) — Probability to timeout (q)

- **Probability to timeout**

Outlier ratio (ρ)

$$q = \rho + (1-\rho)P(r + R > t_\infty)$$

$$q = 1 - (1-\rho)F_R(t_\infty - r)$$

The execution time J

- **The c.d.f F_J is known for every nt_∞ :**

$$1-F_J(nt_\infty) = P(J > nt_\infty) = q^n$$

- **If $nt_\infty < t < (n+1)t_\infty$:**

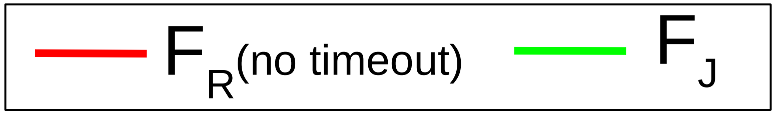
$$F_J(t) = \underbrace{1 - q^n}_{\text{Succeeded before } (n+1)^{th} \text{ submission}} + \overbrace{q^n}^{\text{Timed-out n times}} \overbrace{\underbrace{(1-\rho)}_{\text{Not an outlier}} F_R(t - nt_\infty)}^{(n+1)^{th} \text{ attempt succeeded}}$$

- **Expectation of J:**

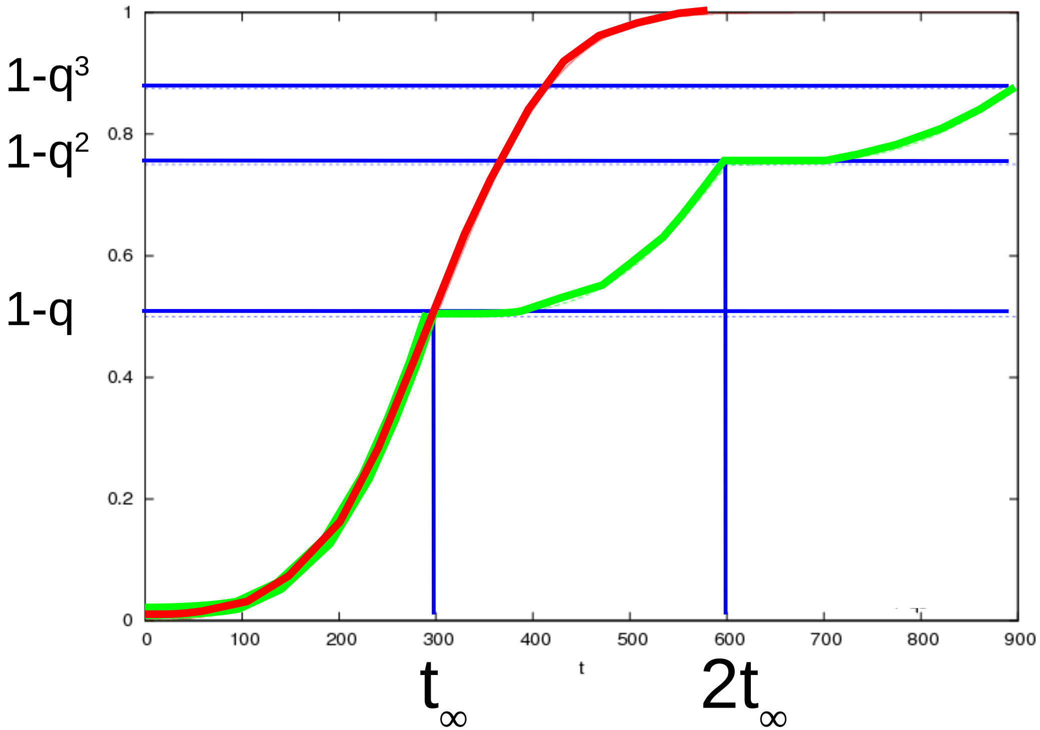
$$E_J(t_\infty) = \frac{1}{F_R(t_\infty)} \int_0^{t_\infty} u f_R(u) du + \frac{t_\infty}{(1-\rho) F_R(t_\infty)} - t_\infty$$

- **Best timeout value:** $\hat{t}_\infty = \underset{t_\infty}{\mathrm{argmin}}(E_J(t_\infty))$

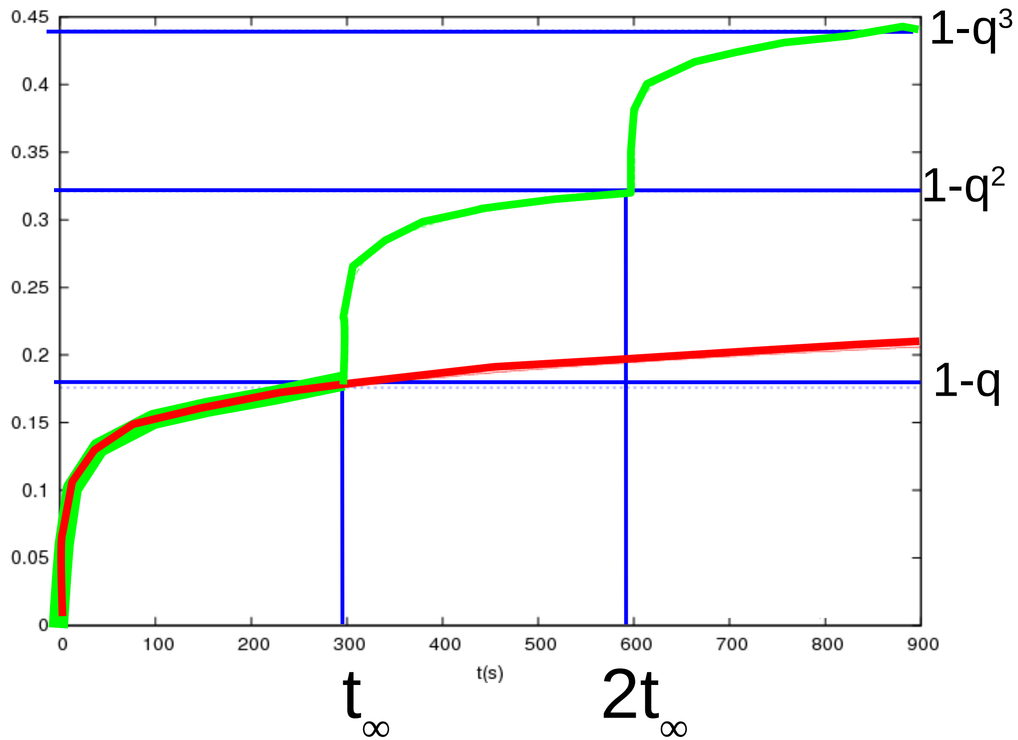
Illustration (without outliers)



- **Bad timeout choice ($F_R > F_J$)**



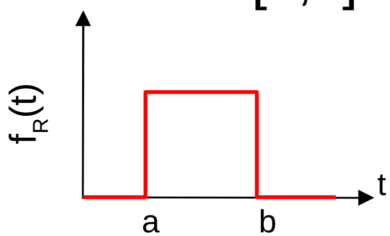
- **Good timeout choice($F_R < F_J$)**

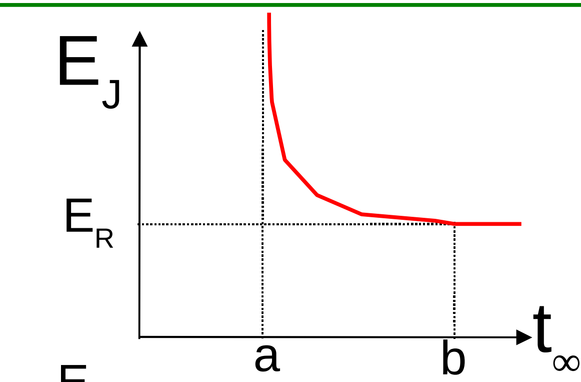


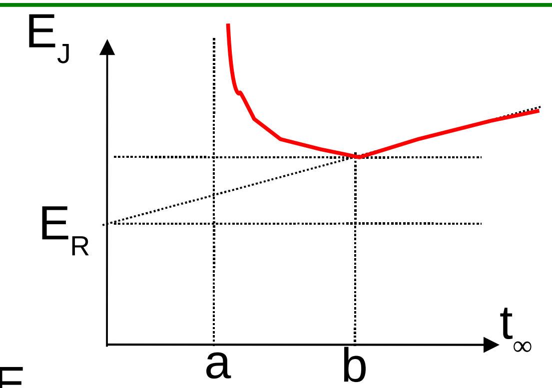
Light-tailed distributions

Latency distribution (p.d.f)	EJ without outlier	EJ with outliers

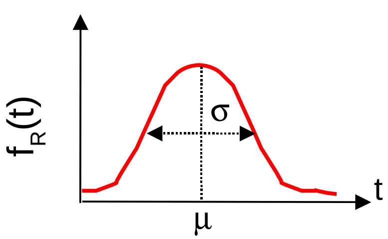
– Uniform [a,b]

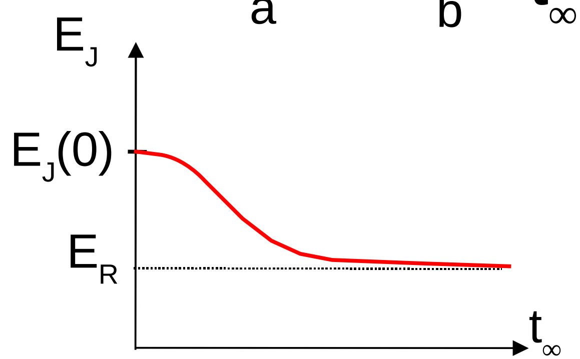


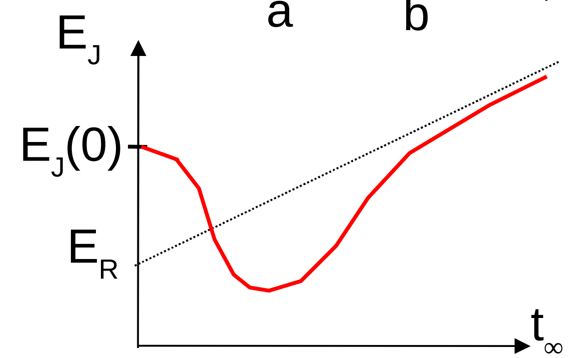




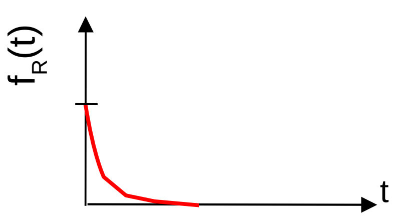
– Truncated Gaussian (μ,σ)

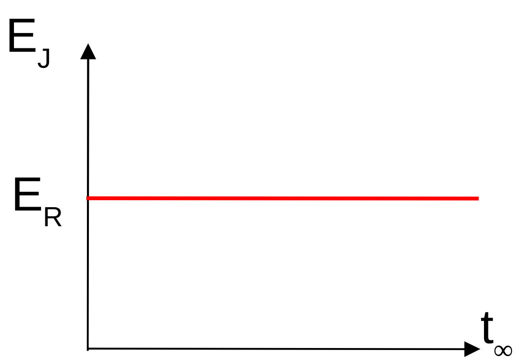


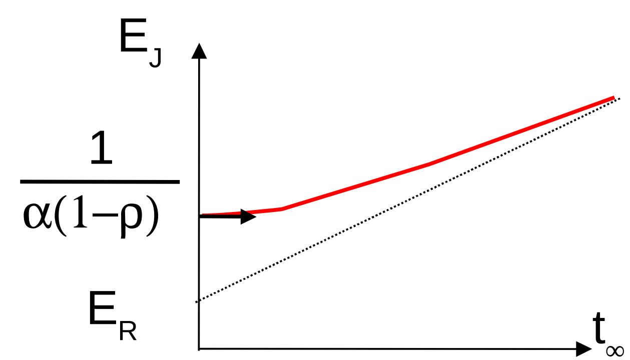




– Exponential (α)



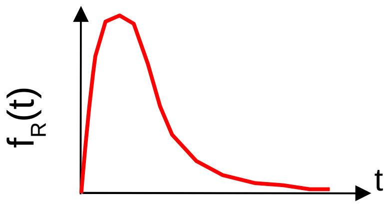


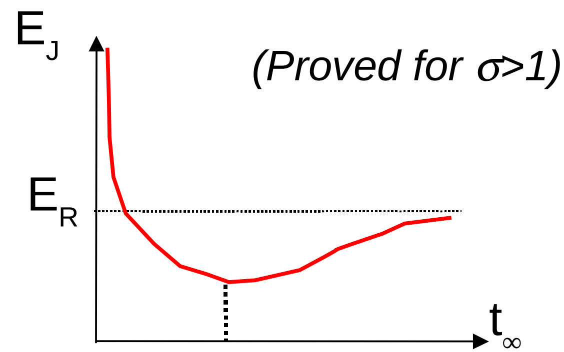


Heavy-tailed distributions

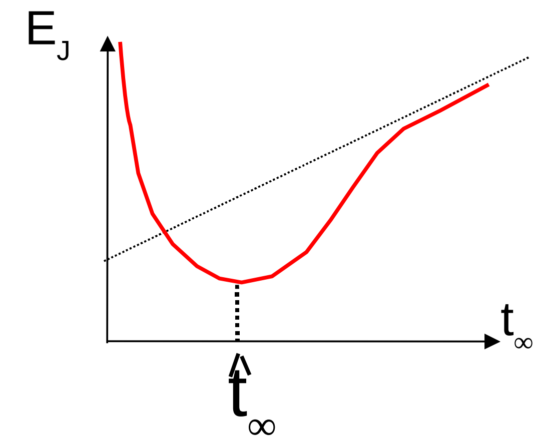
Latency distribution (p.d.f)	**EJ without outlier**	**EJ with outliers**

– Log-normal (μ,σ)

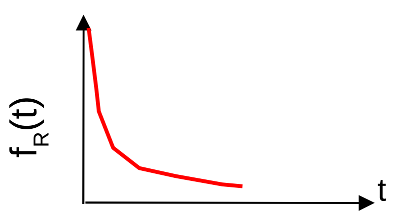


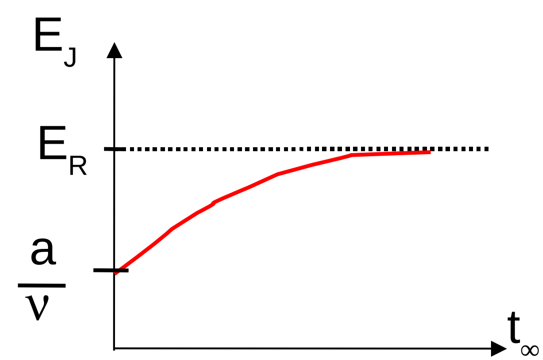


$e^{\mu}K(\sigma)$

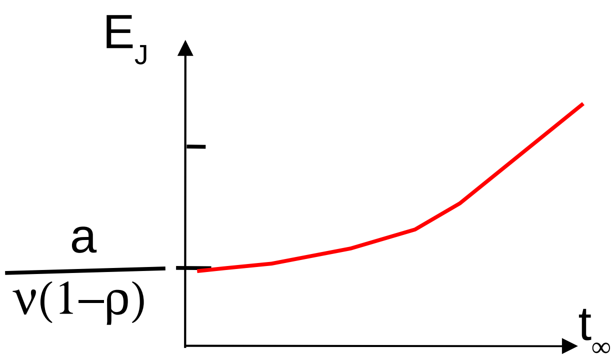


– Pareto (a,ν)





Speed-up $E_R/E_J > 1$



Results summary

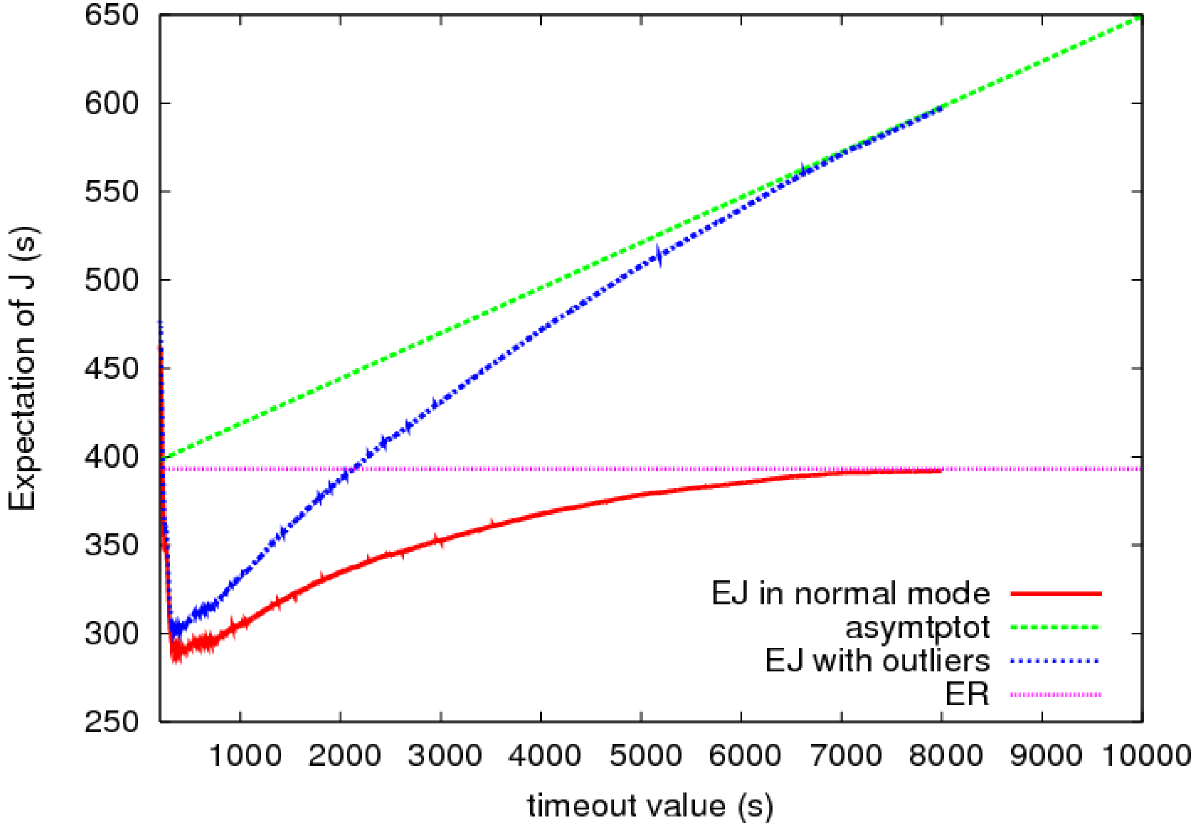
- **Optimal timeout values**

	Distribution of R	Without outliers ($\rho = 0$)	With outliers ($\rho > 0$)
Light-tailed	Uniform	$+\infty$ (or b)	b
Light-tailed	Trunc. Gaussian	$+\infty$	$0 < \hat{t}_\infty < +\infty$
	Exponential	any	0
Heavy-tailed	Log-normal (μ,σ)	$\hat{t}_\infty = e^\mu K(\sigma) < +\infty$	$0 < \hat{t}_\infty < +\infty$
Heavy-tailed	Pareto ($\nu > 1$)	0	0

- **Singular values:**
  - $\hat{t}_\infty = \infty$: do not set any timeout
  - $\hat{t}_\infty = 0$: probability to face a null latency is high

Experimental case

- **Optimization on the distribution measured on EGEE**



- **Without outliers (red curve)**
  - $\hat{t}_\infty = 360s$; $E_J(\hat{t}_\infty) = 289s$
  - Speed-up w.r.t E_R :1.36

With outliers (blue curve)

$\hat{t}_\infty = 358s$; $E_J(\hat{t}_\infty) = 300s$

- **Overestimating the timeout better than underestimating it**

References

- **On the medical imaging application:**
  - T. Glatard, X. Pennec, J. Montagnat. "Performance evaluation of grid-enabled registration algorithms using bronze-standards" in MICCAI'06, pages 152--160, oct 2006
- **On MOTEUR and its design:**
  - T. Glatard, J. Montagnat, D. Lingrand, X. Pennec. "Flexible and efficient workflow deployement of data-intensive applications on grids with MOTEUR" to appear in IJHPCA, 2007
- **On the optimization of the application on EGEE:**
  - T. Glatard, J. Montagnat, X. Pennec. "Optimizing jobs timeouts on clusters and production grids" in CCGrid'07, pages 100-107, may 2007

http://www.i3s.unice.fr/~glatard **glatard@i3s.unice.fr**