Decimate: a portable and fault-tolerant scheduler extension efficiently handling a large number of dependent jobs

Samuel KORTAS^a, Habib TOYE^b, Ibrahim HOTEIT^{b,c}

(samuel.kortas, habib.toyemahamadoukele, ibrahim.hoteit)@kaust.edu.sa

^aKaust Supercomputing Laboratory, ^bDivision of Computer, Electrical and Mathematical Sciences & Engineering, ^cDivision of Physical Science & Engineering



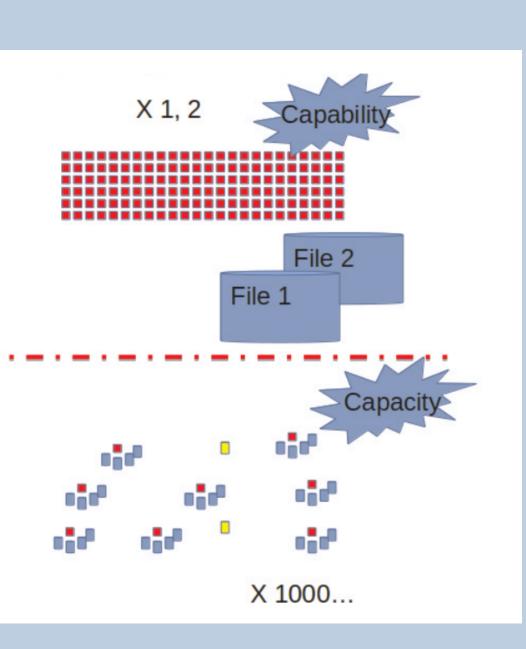
Better handling of capacity & capability jobs on Shaheen XC-40

Some of our scientists use KAUST Cray XC-40 Shaheen to explore parameters sweeping involving thousands of jobs saving thousands of temporary files. Most are non HPC experts but they need a result in a guaranteed time. These complex workflows producing a large number of small files cause challenging problems in terms of scheduling and file system stress.

Resources are configured/tuned for capability jobs, limited in number but running on a high number of nodes, and producing a small number of big files.

All the opposite of what our users' jobs need!

Limitation may come from scheduler not configured for 1000s of jobs, filesystem not tuned to support 1000s jobs each writing small files at scale hardware and numerical



A SLURM-extension available as a python module

Decimate extends the scheduling and executing environment allowing the user to:

• Submit arbitrary number of jobs regardless of limitation in the scheduling policy, manage jobs as a single workflow easing their submission, monitoring, deletion or reconfiguration against a large set of combinations of parameters.

• Benefit from a centralized log file.

• Check for correctness of the results of a job via a user-configurable shell or Python script and make a decision either to stop the whole workflow, to resubmit the failing components as is, or to modify it dynamically.

Once installed in a SLURM-scheduled environment as a regular python module via the command:

pip install decimate [--user] or conda install -c hpc4all decimate

Decimate provides additional commands and parameters:

dbatch	submits the job (SLURM sbatch command with additional parameters handling fault-tolerance, and hidding limitation)	dstat	gives a detailed status of the jobs currently running, pending, waiting to be submitted or finished weither completed or returned with failure.
dlog	access to centralized log file for the whole workflow	dkill	terminates all job of the whole workflow

failures that have to be taken into account.

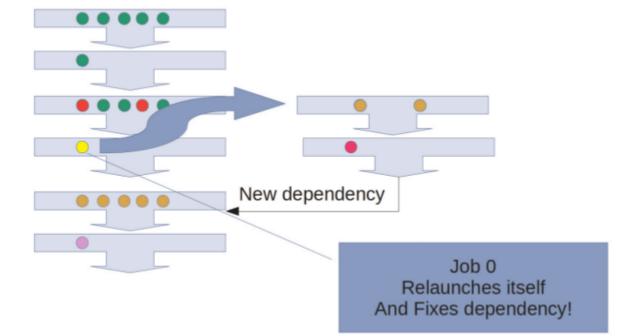
Our strategy:

- Pack 'many jobs' together to make them appear as big ones
- Reduce stress on filesystem by using Ramdisk and messages.
- Package these features as a SLURM scheduler extension easy to use and to install, distributed as an Open Source software.

Automated restart in case of failure

In case of failure of one part of the workflow, Decimate automatically detects the failure, signals it to the user and launches the misbehaving part after having fixed the job dependency. By default if the same failure happens three consecutive times, *Decimate* cancels the whole workfow.

dbatch --job-name step1 --array 1-5 --check



--check-file check.sh --max-retry 3 step1.sh

dbatch --job-name step2 --depends step1 step2.sh

Example of checking script check.sh echo \$job_step -- \$attempt -- \$task_id echo is_job_completed=\$is_job_completed

grep 'job DONE' \$output_file

Expected Results SUCCESS: job went fine FAILED: job needs to retart HALT: the whole workflow failed

As the checking script is evaluated by the first job of the next step, it has knowledge of the status of the whole workflow. In case of failure, one can easily decide at that point either to resubmit the failed parts of the last step, either to stop the whole workflow or even to modify dynamically the environment before retempting a step.

Decimate also provides corresponding Python methods for each of these shell commands.

Three extended execution modes

Sustained and automated feeding of the queue: Decimate guarantees that at the most a certain number of jobs appears in the queue at the same time (satisfying limitation in place). The remaining jobs are automatically submitted at the completion of the previous ones:

200 jobs 4 active only

dbatch --max-jobs=4 --array=1-200 my_job.sh

executes 200 jobs handled as arrays with at most 4 jobs appearing in the scheduling queue.

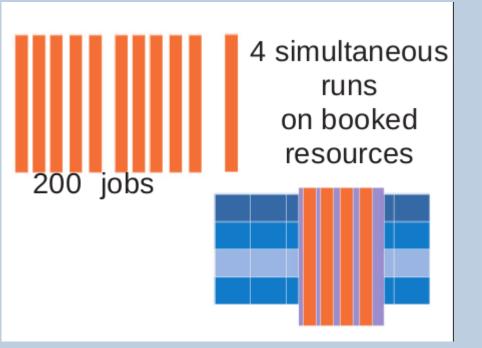
Execution on a pool of nodes: *Decimate* can transparently schedule all the jobs inside a given set of nodes booked for a longer duration period: 4 simultaneous runs on booked resources:

dbatch --parallel-runs=4 --array=1-200 my_job.sh

Books 4x3 nodes for 30 minutes and executes 4 jobs of 3 nodes simultaneously on these resources.

Parametric jobs support

Decimate computes every job possible, group them in blocks of similar footprint (nodes x time), submit them as separate arrays



Possible	#SBATCH
Parameters	#SBATCH
Combinations	job to be tested with
	different sets of

and manage them in a fault tolerant environment

dbatch --param--file=my_params.txt my_job.sh

History and current status

Decimate is the result of a continuous ongoing effort started in 2007 at Electricité de France to provide a portable framework capable of running hundreds of scenarios on a user workstation and transparently scaling to thousands of possibilities explored on the company's Supercomputer.

As the main developper moved to KAUST, the need for a robust and fault-tolerant environment to run 1000 simultaneous forcasts lead to its complete rewritting and packaging as a scheduler extension available as a python module and distributed under Open Source License.

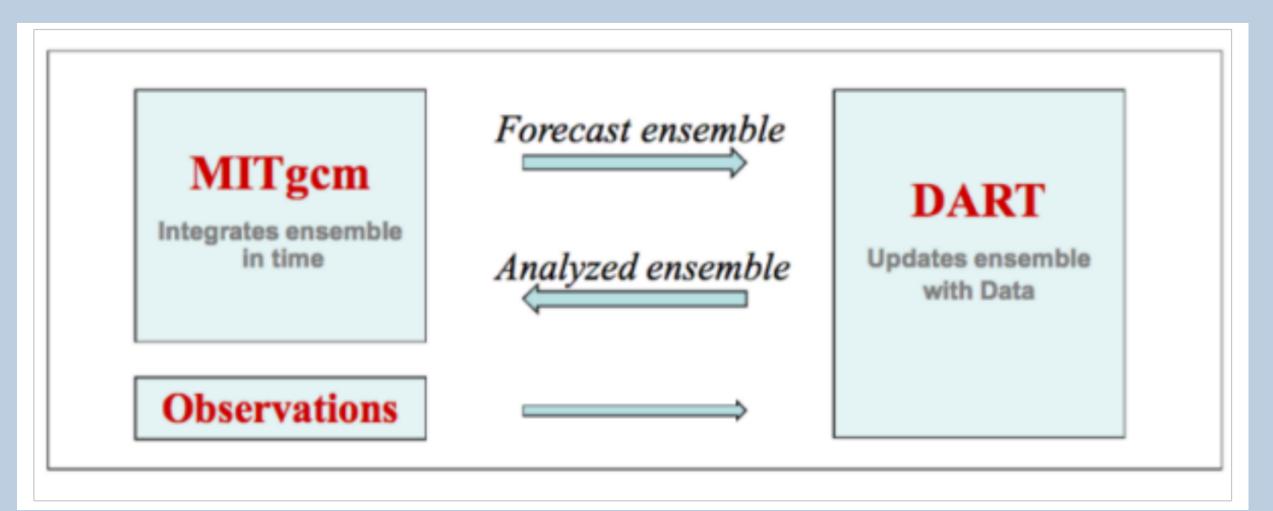
Development and maintenance of *Decimate* are currently supported by KAUST Supercomputing Core Laboratory

Perspectives

- Running an Ocean Ensemble data Assimilation with even more members.
- Comparison with other tools... To our knowledge, none of them has a shell

An Ocean Ensemble data Assimilation challenging use case

DART-MITgcm assimilation system combines MITgcm forecast ensemble simulations with observational data to compute the best possible estimate of the state of the ocean through DART Ensemble Kalman filters. It also quantifies uncertainties in the final solution (often described by the covariance matrix of the state estimate), derived from the knowledge of the prior uncertainties and the observational data errors covariance.



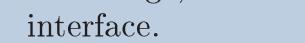
A workflow of 130,000 jobs, running 1000 MITgcm members on 3 Haswell 32-core nodes followed by one DART filter running on one node.

Results

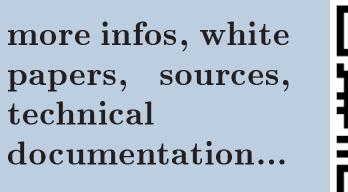
arams.txt

Decimate allowed to alleviate the burden of constantly having to monitor the ongoing workflow and manually acting in case of numerical or hardware failure. A rough estimation was reporting that 20% of restitution time was effectively consumed into these manual operations.

100%[IHF0] launch-0:0:no active job in the queue, changing all WAITING in ABORTED??? [MSG] step 2-filter-0:1.1 SUCCESS: 100% FAILURE: 0% -> [] [MSG] step 2-mitgcm-0:1-999 MIXED SUCCESS: 100% FAILURE: 0% -> [] [MSG] step 2-mitgcm-0:1-10:1.1 SUCCESS: 100% FAILURE: 0% -> [] [MSG] step 3-filter-0:1.1 SUCCESS: 100% FAILURE: 0% -> [] [MSG] step 3-filter-0:1.1 SUCCESS: 100% FAILURE: 0% -> [] [MSG] step 3-filter-0:1.1 SUCCESS: 100% FAILURE: 0% -> [] [MSG] step 4-mitgcm-0:1-999 SUCCESS: 100% FAILURE: 0% -> [] [MSG] step 4-mitgcm-0:1-999 MIXED [MSG] step 4-mitgcm-0:1-10:1 SUCCESS: 100% FAILURE: 0% -> [] [MSG] step 4-mitgcm-0:1-299 MIXED [MSG] step 4-mitgcm-0:1-399 SUCCESS: 100% FAILURE: 0% -> [] [MSG] step 4-mitgcm-0:1-399 SUCCESS: 100% FAILURE: 0% -> [] [MSG] step 4-mitgcm-0:1-10 SUCCESS: SUCCESS: 100% FAILURE: 0% -> [] [MSG] step 5-filter-0:1-1 SUCCESS: SUCCESS: 100% FAILURE: 0% -> []	<pre># Welcome to dart_mitgcm v 0.4.1! # # (using DECIMATE Framework 0.9) # # # # # # # # # # # # # # # # # # #</pre>	
<pre>cmmbers c</pre>	<pre># members python /project/k1029/Sam/dart_mitgcm/0.4.1/sles11.3_gnu5.1.0/dart_mitgcm/dart_mitgcm.py -sa</pre>	
<pre>running on cd13 (shaked) prince /united / signated rigingca/0.4.1/sigs1.3.gut5.1.0/drt_signated rigingca/0.4.r/signated / signated / s</pre>	running on cdl3 (shaheen) python /project/k1029/Sam/dart_mitgcm/0.4.1/sles11.3_gnu5.1.0/dart_mitgcm/dart_mitgcm.py -sa	
<pre>python /project/kl029/5aw/dart_mitgcx/6.4.1/stall.j_nu/5.1.0/dart_mitgcx/</pre>	<pre>python /project/k1029/Sam/dart_mitgcm/0.4.1/sles11.3_gnu5.1.0/dart_mitgcm/dart_mitgcm.py -sa</pre>	
<pre>>.true.s in ensemble innager_mil.</pre>	INF0 1 !!!! WARNING WARNING mismatched tag single restart file in values:	
b.true.in restart_file_toil_mil. invoi *.true.in restart_file_toil_mil. invoi *.true.in restart_file_toil_mil. invoi *.true.in restart_file_toil_mil. invoi *.true.in restart_file_toil_mil. invoi *.true.in restart_file_toil_mil. invoi *.true.in restart_file_toil_mil. invoi *.true.in restart_file_toil_mil. invoi *.true.in restart_file_toil_mil. invoi *.true.in restart_file_toil_mil. invoi *.true.in restart_file_toil_mil. invoi *.true.in restart_file_toil_mil. invoite *.true.in restart_in restart_file_toil_mil. invoite *.true.in restart_in r		
<pre>[IHF0] iff WARNING WARNING mismatched tag restart_in_file_name values set in model, state used in file second step of mitgcm was searched tag to in perfect. ics: eff second step of mitgcm was restarted once after 1% of failure second in the queue, changing all WAITING in ADORTED777 (Information active found [12.aitgo=3,11.a) (Information active found [12.aitgo=3,11.a) (Information active found in the queue, changing all WAITING in ADORTED777 (Information active found in the queue, changing all WAITING in ADORTED777 (Information active found [12.aitgo=3,11.a) (Information active found in the queue, changing all WAITING in ADORTED777 (Information active found in the queue, changing all WAITING in ADORTED777 (Information active found in the queue, changing all WAITING in ADORTED777 (Information active found in the queue, changing all WAITING in ADORTED777 (Information active found in the queue, changing all WAITING in ADORTED777 (Information active found in the queue, changing all WAITING in ADORTED777 (Information active found in the queue, changing all was found in the queue, changing all was found in the queue, found in the queue, changing all was found in the queue of the found in the queu</pre>	>.true.< in restart_file_tool_nml.	
Second step of mitgem was restarted once after 1% of failure information of the first first on masspaced value		
<pre>setup on manual intervention bet is setup on the setup of the setup o</pre>		
[INF0] input, mil. template did not change signer last time, no need to [120:4107] [INF0] input, mil. template did not change signer last time, no need to [120:4107] [INF0] input, mil. template did not change signer last time, no need to [120:4107] [INF0] input, mil. template did not change signer last time, no need to [120:4107] [INF0] input, mil. template did not change signer last time, no need to [120:4107] [INF0] input, mil. template did not change signer last time, no need to [120:4107] [INF0] input, mil. template did not change signer last time, no need to [120:4107] [INF0] input, mil. template did not change signer last time, no need to [120:4107] [INF0] input, mil. template did not change signer last time, no need to [120:4107] [INF0] input, mil. template did not change signer last time, no need to [120:4107] [INF0] input, mil. template did not change signer last time, no need to [120:4107] [INF0] input, mil. template did not change signer last time, no need to [120:4107] [INF0] input, mil. template did not change signer last time, no need to [120:4107] [INF0] input, mil. template did not change signer last time, no need to [120:4107] [INF0] input, mil. template did not change signer last time, no need to [120:4107] [INF0] inter did not change signer last time, no need to [120:4107] [INF0] inter did not change signer last time, no need tin [120:4107] <td< td=""><td>>smoother 1cs< in smoother n</td><td></td></td<>	>smoother 1cs< in smoother n	
100% INFO] Laurch-010:no active job in the queue, changing all WAITING in ADMITED77 W66] step 2-filter-01-1 SUCCESS SUCCESS: 100% FAILURE: 0% -> [[] W66] step 2-atigen-01-1999 MUXED SUCCESS: 100% FAILURE: 0% -> [] W56] step 3-filter-01-1 SUCCESS SUCCESS: 100% FAILURE: 0% -> [] W56] step 3-filter-01-1 SUCCESS SUCCESS: 100% FAILURE: 0% -> [] W56] step 3-filter-01-1 SUCCESS SUCCESS: 100% FAILURE: 0% -> [] W56] step 4-filter-01-1 SUCCESS SUCCESS: 100% FAILURE: 0% -> [] W56] step 5-filter-01-1 SUCCESS SUCCESS: 100% FAILURE: 0% -> [] W56] step 5-filter-01-1 SUCCESS SUCCESS: 100% FAILURE: 0% -> [] W56] step 5-filter-01-1 SUCCESS SUCCESS: 100% FAILURE: 0% -> [] W56] step 5-filter-01-1 SUCCESS SUCCESS: 100% FAILURE: 0% -> [] W56] step 5-filter-01-1 SUCCESS SUCCESS: 100% FAILURE: 0% -> [] W56] step 5-sitigen-12-999 MUXED SUCCESS SUCCESS: 100% FAILURE: 0% -> [] W56] step 6-filter-01-1 SUCCESS SUCCESS: 100% FAILURE: 0% -> [] W56] step 6-sitigen-12-999 MUXED SUCCESS SUCESS: 100% FAILURE: 0% -> [] W56] step 6-sitigen-12-999 MUXED SUC		
1 step 2-filter-0:1-1 SUCCESS SUCCESS: 100% FAILURE: 0% -> [] 1 step 2-sitige=0:1:999 HUXED SUCCESS SUCCESS: 100% FAILURE: 0% -> [] 1 step 2-sitige=0:1:1 SUCCESS SUCCESS: 100% FAILURE: 0% -> [] 1 step 3-sitige=0:1:1 SUCCESS SUCCESS: 100% FAILURE: 0% -> [] 1 step 3-sitige=0:1:209 HUXED SUCCESS SUCCESS: 100% FAILURE: 0% -> [] 1 step 4-sitige=0:1:209 HUXED SUCCESS SUCCESS: 100% FAILURE: 0% -> [] 1 step 4-sitige=0:1:20 SUCCESS SUCCESS: 100% FAILURE: 0% -> [] 1 step 4-sitige=0:1:24,235-236,238,241,243-245,257:1999] HUXED SUCCESS SUCCESS: 100% FAILURE: 0% -> [] 1 step 4-sitige=0:1:24,235-236,238,241,243-245,575:1999] HUXED SUCCESS SUCCESS: 100% FAILURE: 0% -> [] 1 step 5-sitige=1:34,350,352 SUCCESS SUCCESS: 100% FAILURE: 0% -> [] 1 step 6-sitige=0:1:99 HUXED SUCCESS SUCCESS: 100% FAILURE: 0% -> [] 1 step 6-sitige=0:1:99 HUXED SUCCESS SUCCESS: 100% FAILURE: 0% -> [] With no manual intervention 5 steps made in 3h30 on a crowded machine Ole scientific results were published runni: Success success success success success in 0% FAILURE: 0% -> []	[INFO] launch-010:2 unconsistent steps_were found: [12-mitgcm-3,11-m] 100%[INFO] launch-010:no active job in the gueue, changing all WAITING in ABORTED???	
$\frac{1956}{156} \frac{1}{1569} \frac{1}{1167} \frac{1}{1028} \frac{1}{10$	[MSG] step 2-filter-0:1-1 SUCCESS SUCCESS: 100% FAILURE: 0% -> []	
istep 3-mitgca-0:1-999 SUCCESS 100% FAILURE: 0% -> [] M56 istep 4-mitgca-0:1-399 WICESS SUCCESS: 100% FAILURE: 0% -> [] M56 istep 4-mitgca-0:1-399 WICESS SUCCESS: 100% FAILURE: 0% -> [] M56 istep 4-mitgca-0:1-399 WIXED SUCCESS: 100% FAILURE: 0% -> [] M56 istep 5-mitgca-0:1-399 WIXED SUCCESS: 100% FAILURE: 0% -> [] M56 istep 5-mitgca-0:1-399 MIXED SUCCESS: 100% FAILURE: 0% -> [] M56 istep 5-mitgca-0:1-399 MIXED SUCCESS: 100% FAILURE: 0% -> [] M56 istep 6-mitgca-0:1-399 MIXED SUCCESS: 100% FAILURE: 0% -> [] M56 istep 6-mitgca-0:1-399 MIXED SUCCESS: 100% FAILURE: 0% -> [] M56 istep 6-mitgca-0:1-399 MIXED SUCCESS: 100% FAILURE: 0% -> [] M56 istep 6-mitgca-0:1-399 MIXED SUCCESS: 100% FAILURE: 0% -> []	[MSG] step 2-mitgcm-1:614,617-620,622,624-625 SUCCESS SUCCESS: 100% FAILURE: 0% →> []	
$\frac{1566}{1569} = \frac{1}{1560} = \frac{1}{1526} = $		
is tep 4-mitigen-1:124,235-236,230,241,243-245 Success success: 10% FATLURE: 0% -> [] MSG is tep 5-filter-0:1-1 Success Success: 10% FATLURE: 0% -> [] MSG is tep 5-mitigen-1:349 MIXED Success: 10% FATLURE: 0% -> [] MSG is tep 5-mitigen-1:349 MIXED Success: 10% FATLURE: 0% -> [] MSG is tep 6-mitigen-0:1-3 Success Success: 10% FATLURE: 0% -> [] MSG is tep 6-mitigen-0:1-3 Success Success: 10% FATLURE: 0% -> [] MSG is tep 6-mitigen-0:1-3 Success Success: 10% FATLURE: 0% -> [] MSG is tep 6-mitigen-0:1-4 Success Success: 9% FATLURE: 0% -> [] MSG is tep 6-mitigen-0:1-4 Success Success: 9% FATLURE: 0% -> [] MSG is tep 6-mitigen-0:1-3 Success Success: 9% FATLURE: 0% -> [] MSG is tep 6-mitigen-0:1-3 Success Success: 10% FATLURE: </td <td></td> <td></td>		
Inset 1 step 5-sitgen-0:1-999 MixeD SUCCESS 99% FAILURE: 0% - [061,675] Inset 6-sitgen-0:1-399 SUCCESS SUCCESS 100% FAILURE: 0% - [1] Inset 6-sitgen-0:1-399 SUCCESS SUCCESS 100% FAILURE: 0% - [1] Inset 6-sitgen-0:1-399 MixeD SUCCESS 100% FAILURE: 0% - [1] Inset 6-sitgen-0:1-399 MixeD SUCCESS 100% FAILURE: 0% - [1] Inset 6-sitgen-0:1-399 MixeD SUCCESS 100% FAILURE: 0% - [1] With no manual intervention 5 steps made in 3h30 on a crowded machine - [1] - [1] Ole scientific results were published runni: - [1] - [1] - [1] - [1]	[MSG] step 4-mitgcm-1:124,235-236,238,241,243-245 SUCCESS SUCCESS: 100% FAILURE: 0% -> []	
Istep 6-nitigen-0:1-999 SUCCESS SUCCESS: 100% FAILURE: 0% -> [] MIXED SUCCESS SUCCESS: 100% FAILURE: 0% -> [] With no manual intervention 5 steps made in 3h30 on a crowded machine Ole scientific results were published runni:	[MSG] step 5-mitgcm-0:1-999 MIXED SUCCESS: 99% FAILURE: 0% -> [861,875]	
bis d step 6-sitgen-0:1-999 MIXED SUCCESS 99% FAILURE: 0% -> [345, 350, 352] With no manual intervention 5 steps made in 3h30 on a crowded machine Ole scientific results were published runni:		
With no manual intervention 5 steps made in 3h30 on a crowded machine	MSG step 6-mitgcm-0:1-999 MIXED SUCCESS: 99% FAILURE: 0% -> [345,350,352]	
5 steps made in 3h30 on a crowded machine	The I see a site a site and a site of the second se	
5 steps made in 3h30 on a crowded machine	With no manual intervention	
ole scientific results were published runni		
÷	5 steps made in 3h30 on a crowded machine	
÷		
÷		
÷		
÷	ale scientific results were nublished runn	ir
similation model on 1000 members. [1]	sie beienning repuins were published runn.	11
similation model on 1000 members. [1]	• • 1 • 1 1 1 1000 1 [1]	
	similation model on 1000 members 111	

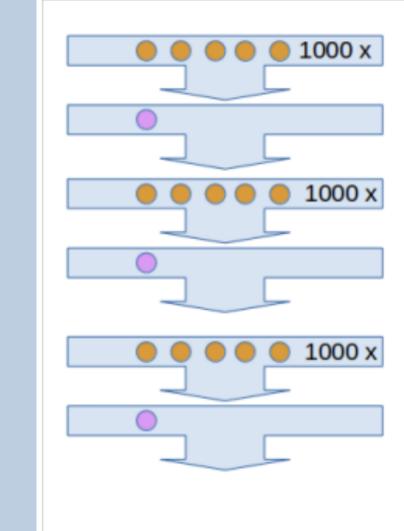


- Support of other scheduler: PBS.
- Support of no-scheduler mode, running on a regular workstation.



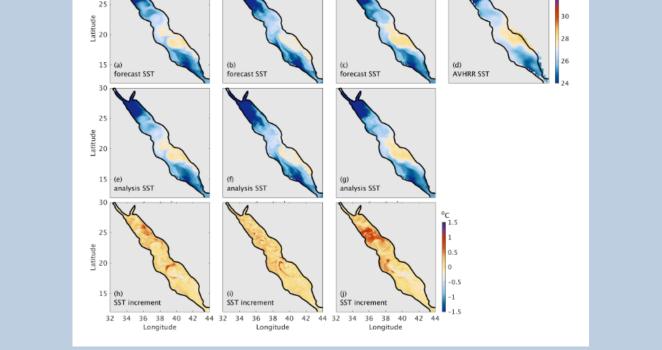


http://decimate.hpc4all.org



First set of mitgcm (1000 x 3-node runs) barrier Apply the filter (dart) (1 x 16-node run) barrier second set of mitgcm (1000 x 3-node runs) barrier Apply the filter (dart) (1 x 16-node run)

- barrier
- (1000 x 3-node runs) set of mitgcm
- barrier Apply the filter (dart) (1 x 16-node run)



Decimate is now extensively used at KAUST in Computational Chemistry, Uncertainty Quantification, or to explore any sets of parameters.

References

[1] A Fault-Tolerant HPC Scheduler Extension for Large and Operational Ensemble Data Assimilation: Application to the Red Sea, Toye H., Kortas S., Zhan P., Hoteit I., Journal of Computational Science 27 (2018) 46-56

[2] http://decimate.readthedocs.org

Acknowledgements

Va

the

Research exposed in this poster was supported by King Abdullah University of Science and Technology (KAUST), and made use of the resources of the KAUST Supercomputing Core Laboratory.