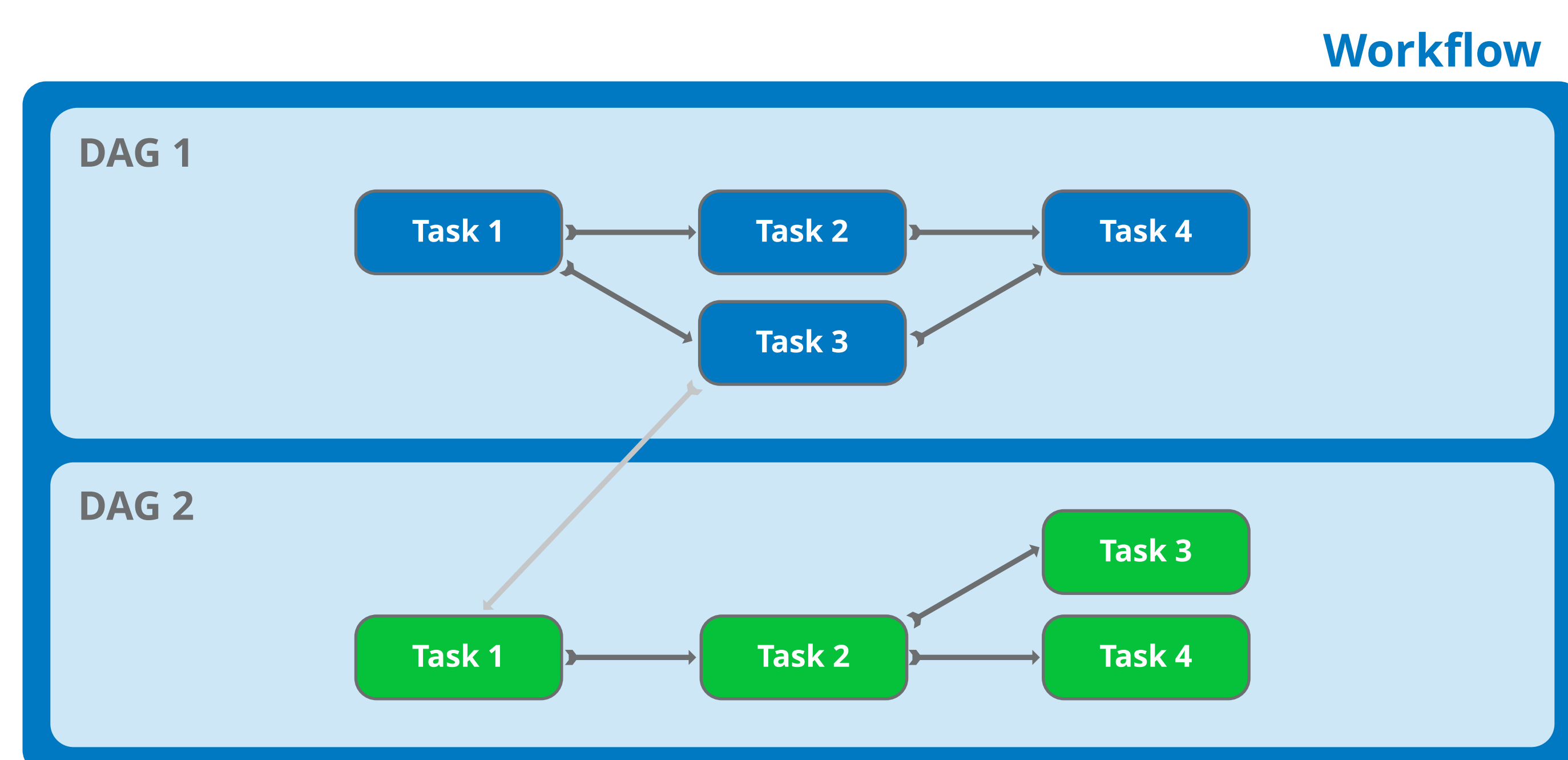


Lightflow - A lightweight, distributed workflow system

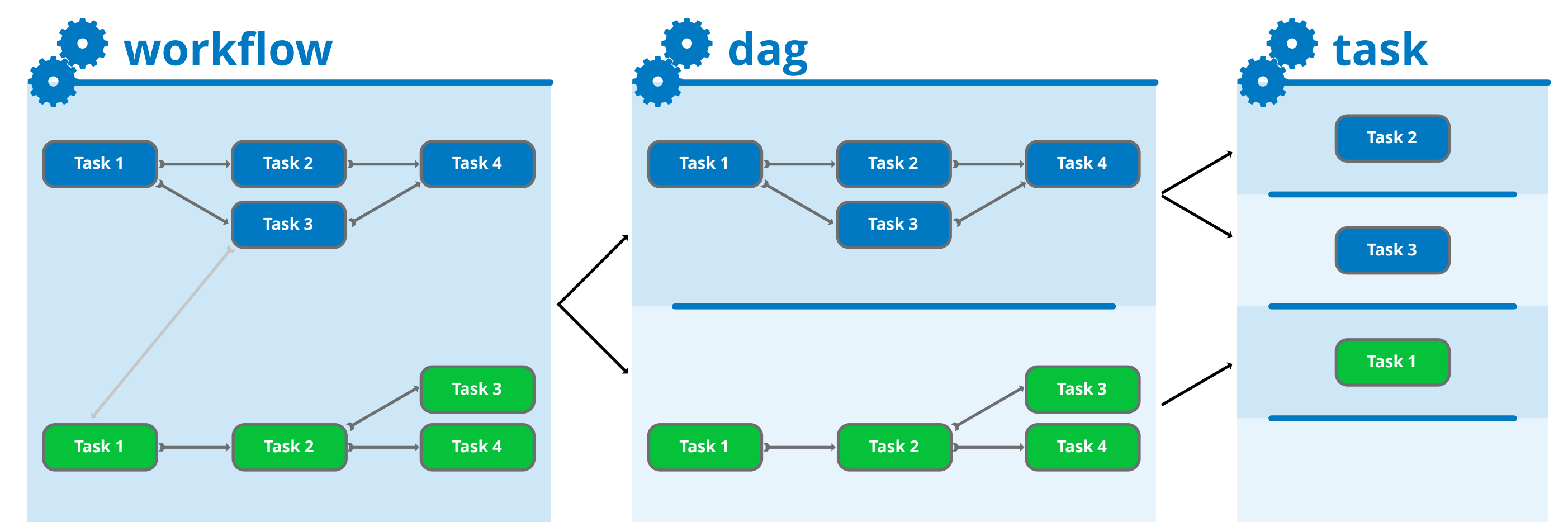
Andreas Moll, Stephen Mudie, Robbie Clarken, Paul Martin

After more than 10 years of operation, the beamlines at the Australian Synchrotron are well established and the demand for automation of research tasks is growing. In order to meet these demands we have developed a generic distributed workflow system.

Architecture



Lightflow models a workflow as a set of individual tasks arranged as a directed acyclic graph (DAG). This specification encodes the direction that data flows as well as dependencies between tasks. Each workflow consists of one or more DAGs. While the arrangement of tasks within a DAG cannot be changed at runtime, other DAGs can be triggered from within a task, therefore enabling a workflow to be adapted to varying inputs or changing conditions during runtime.






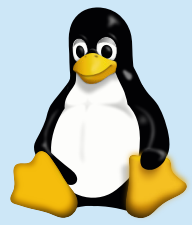
Lightflow employs a worker-based queuing system, in which workers consume individual tasks. This allows the processing of workflows to be distributed.

In order to avoid single points of failure, such as a central daemon as is often found in other workflow tools, the queuing system is also used to manage and monitor workflows and DAGs.

Tasks can receive data from upstream tasks and send data to downstream tasks. Any data that can be serialised can be shared between tasks. Typical examples for data flowing from task to task are file paths, pandas DataFrames or numpy arrays.

Lightflow provides a fully featured command line interface for starting, stopping and monitoring workflows and workers.

Implementation

 <p>Lightflow is written in Python 3 and supports Python 3.5+ It uses the Celery library for queuing tasks and the NetworkX module for managing the directed acyclic graphs.</p>	 <p>As redis is a common database found at many beamlines at the Australian Synchrotron, it is the default backend for Celery in Lightflow. However, any other Celery backend can be used as well.</p>
 <p>Lightflow uses MongoDB in order to store data that is persistent during a workflow run. Examples include the aggregation of values, calculation of running averages, or the storage of flags.</p>	 <p>Lightflow is being developed and tested on Linux, with Debian and RedHat being the main platforms.</p>

Examples

The phaseID pipeline at the SAXS/WAXS beamline identifies diffraction peak positions within SAXS profiles and infers the most likely space group. This pipeline enables researchers to rapidly determine phase diagrams for self-assembled lyotropic liquid crystal systems.

The data compression and management pipeline at the two crystallography beamlines (MX1, MX2) has recently been upgraded to use Lightflow in order to take advantage of a distributed system to compress multiple experiments at the same time.

Workflow definition

```
from lightflow.models import Dag
from lightflow.tasks import PythonTask

# the callback function for the tasks
def print_info(data, store, signal, context):
    print('Task {task_name} being run in DAG {dag_name} for workflow {workflow_name} ({workflow_id})'.format(**context.to_dict()))

# create the main DAG
d = Dag('main_dag')

# task that limits the branching to certain successor tasks
branch_task = PythonTask(name='branch_task', callback=print_info)

# first task, first lane
lane1_print_task = PythonTask(name='lane1_print_task', callback=print_info)

# first task, second lane
lane2_print_task = PythonTask(name='lane2_print_task', callback=print_info)

# first task, third lane
lane3_print_task = PythonTask(name='lane3_print_task', callback=print_info)

# joins all three lanes together and waits for the predecessor tasks to finish processing
join_task = PythonTask(name='t_join_me', callback=print_info)

# set up the graph of the DAG as illustrated above. Please note how a list of tasks
# defines tasks that are run in parallel (branched out).
d.define({branch_task: [lane1_print_task, lane2_print_task, lane3_print_task],
          lane1_print_task: join_task,
          lane2_print_task: join_task,
          lane3_print_task: join_task})
```

Download

The source code has been published as open source on GitHub and on PyPI.

```
$ pip install lightflow
```



<https://github.com/AustralianSynchrotron/Lightflow>

<https://australiansynchrotron.github.io/Lightflow>