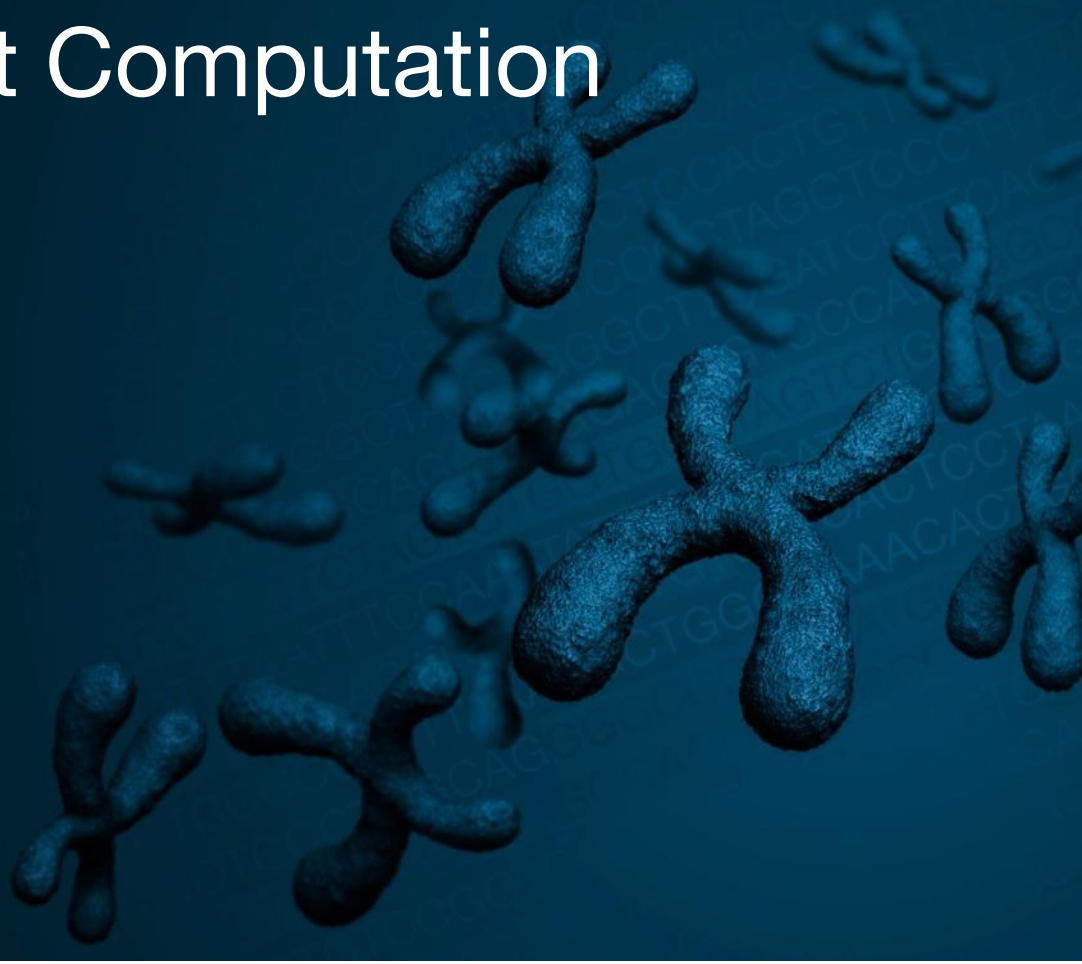
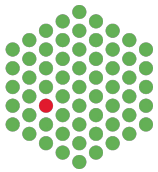
eHive: a System for Massive High-throughput Computation

Brandon Walts
Leo Gordon
Matthieu Muffato
Andrew Yates
Paul Flicek



EMBL-EBI



e!Ensembl

Motivation

- Problem space
  - Large data sets, growing faster than Moore's law
  - Many analyses are easy to run in parallel
  - Analysis tools tend to follow the UNIX philosophy - do one thing but do it well
  - Lots of data handling between analyses
  - Provenance and reproducibility are important
  - Regular repeats of analysis as part of a production cycle
- Infrastructure
  - Large compute farm, managed by a scheduler (LSF)
  - Data in a mix of RDBMS and flat files

Thus, eHive

- First release in 2004
- Currently controls 450 cpu-years per year of compute for Ensembl
- Adopted by several institutions outside of EMBL-EBI

What's with the name?

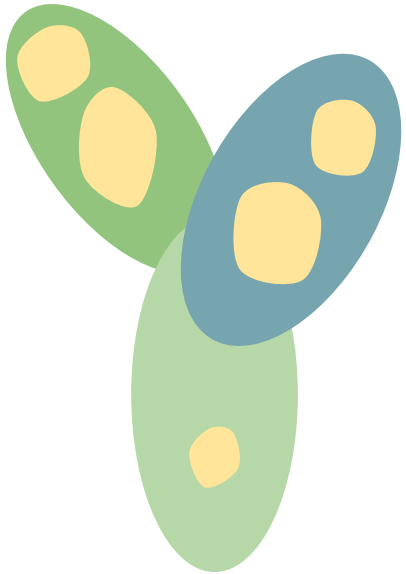
eHive's approach to computation is based on a swarm of autonomous agents - naturally leading to a beehive metaphor:

What's with the name?

- Independent agents ("workers") perform computation.

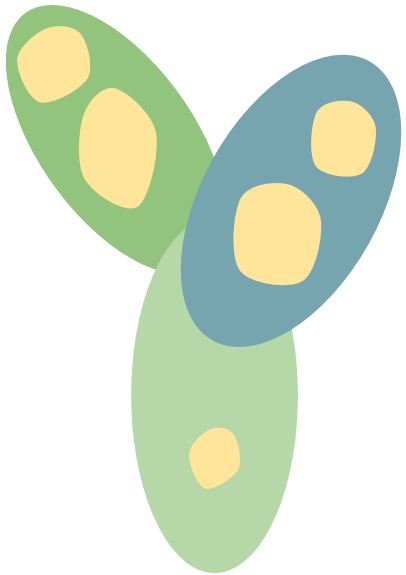
What's with the name?

- Agents have access to resources of different types ("meadows").



What's with the name?

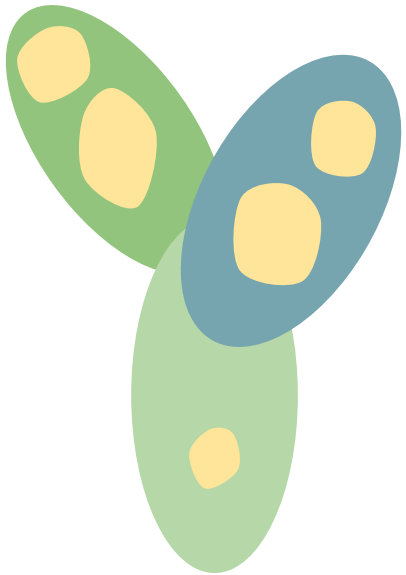
- There is an overseeing process ("beekeeper"), but it is lightweight -- concerned with managing worker population and identifying problems.



What's with the name?

- There is a central database ("blackboard") that workers update to coordinate their activity

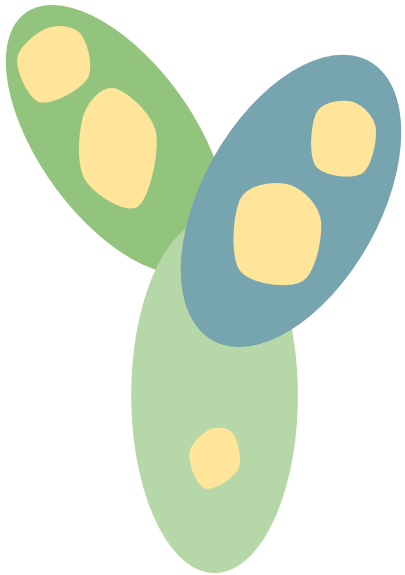
analysis	job	state
download	1	done
align	2	ready



eHive lifecycle

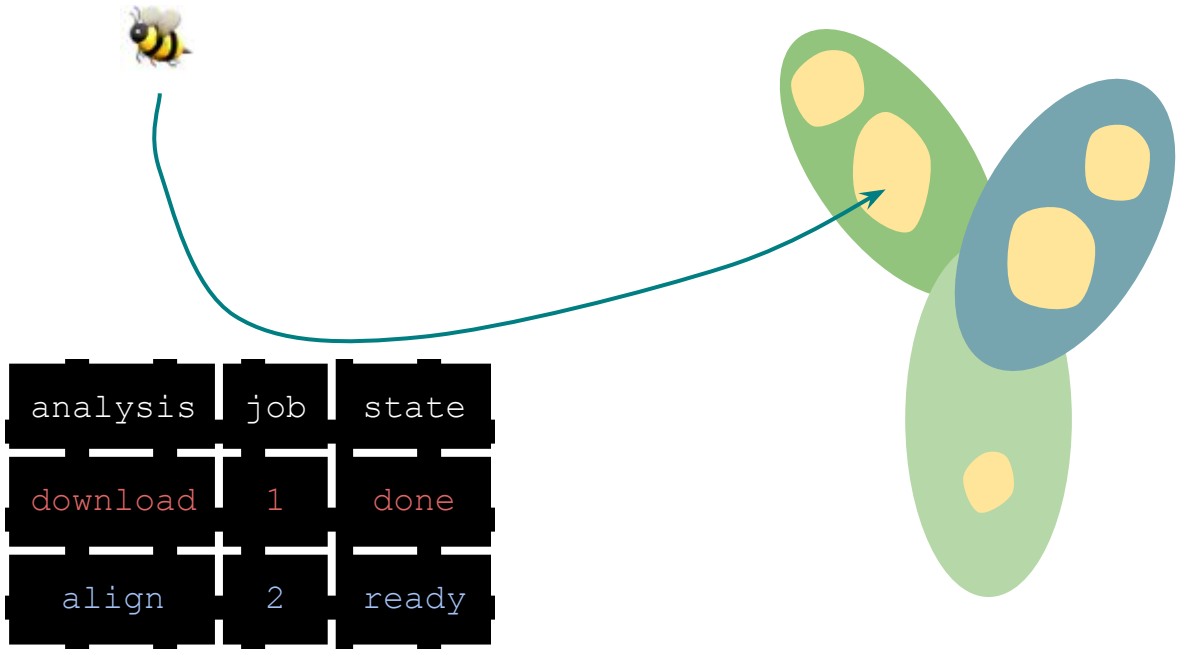
- The beekeeper checks the current job list and worker population, creating new workers if necessary

analysis	job	state
download	1	done
align	2	ready



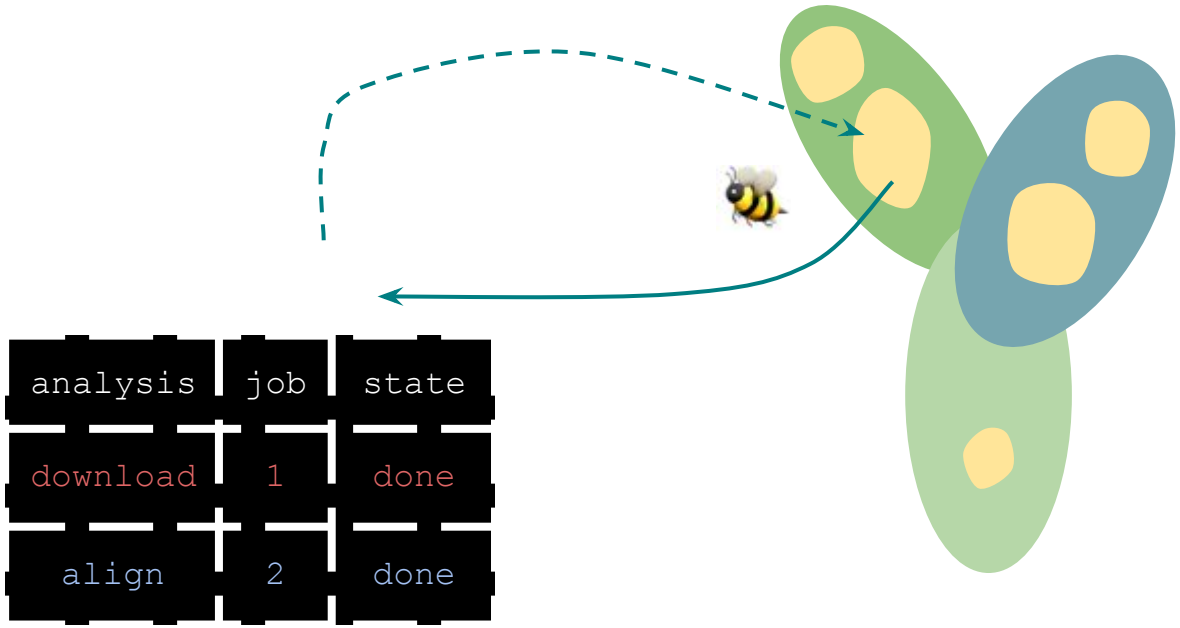
eHive lifecycle

- Worker checks for a job it is able to execute, claims it, specializes if necessary, and begins execution



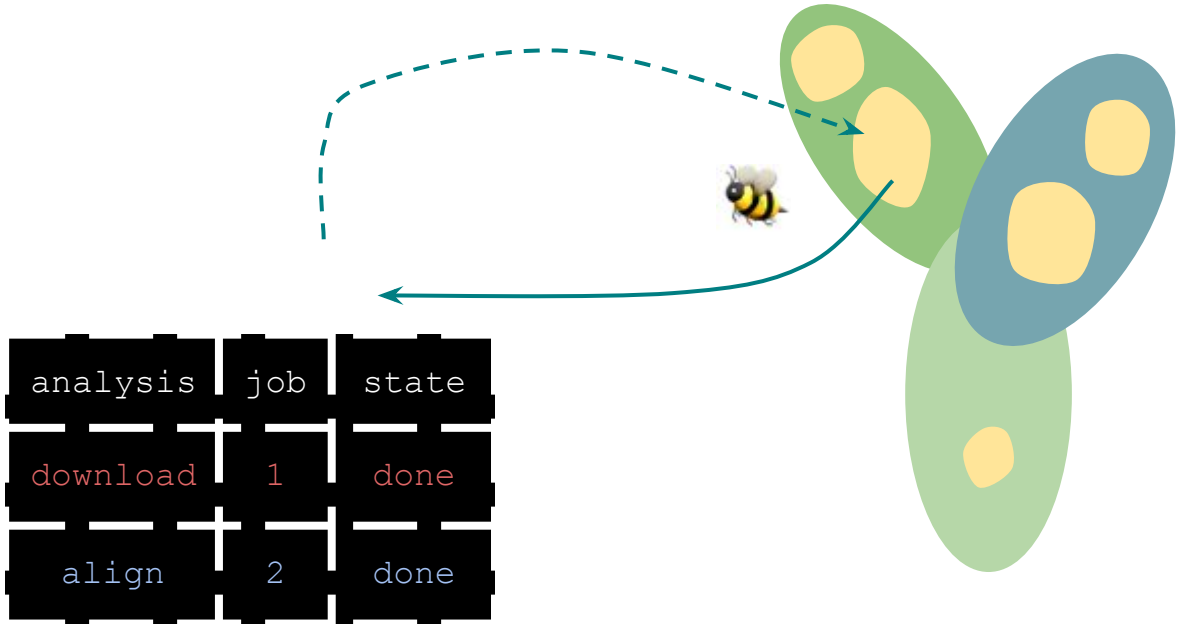
eHive lifecycle

- Worker completes running job. Updates the job list, then checks to see if there is more work to do.



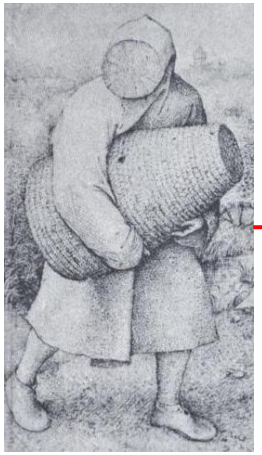
eHive lifecycle

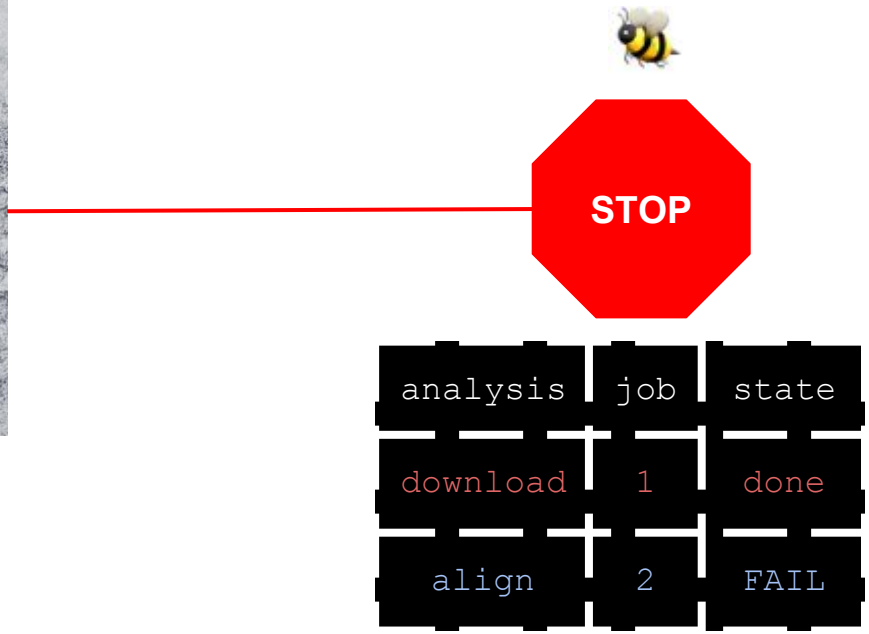
- For short jobs, workers can claim a batch to do in one cycle, reducing dispatch and startup overhead

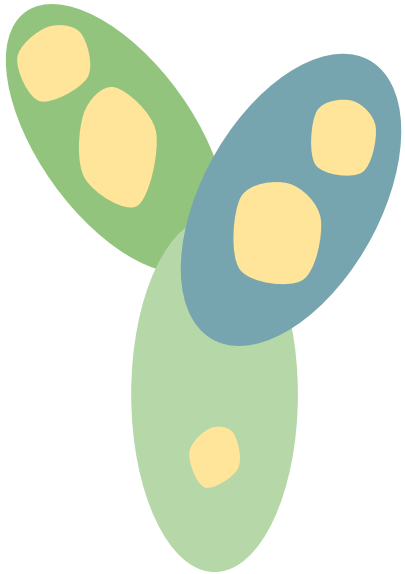


eHive lifecycle

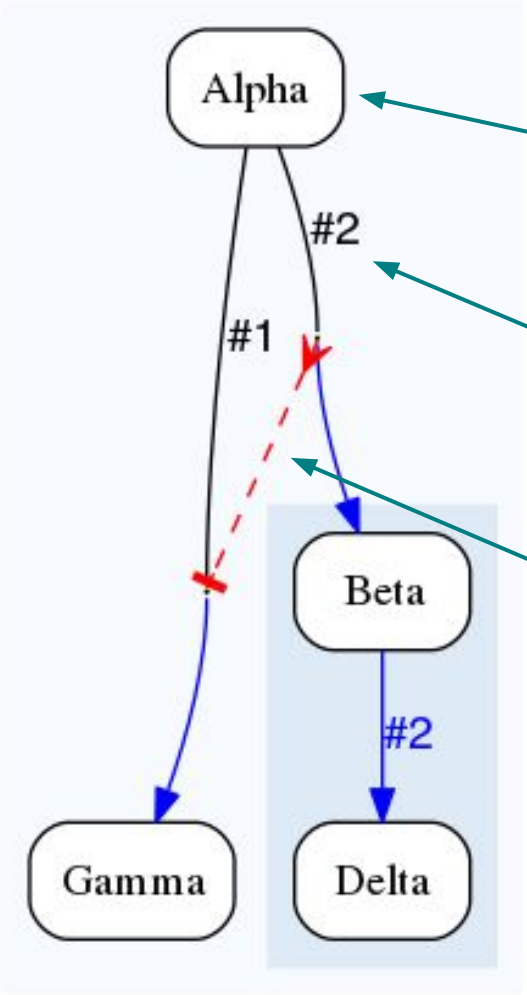
- If errors occur, the beekeeper notes this and can partially or completely stop workflow execution







Workflow structure



An eHive workflow is composed of

analyses

which are connected by

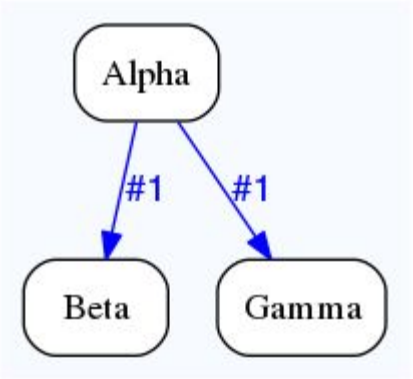
dataflows

and which can be controlled by

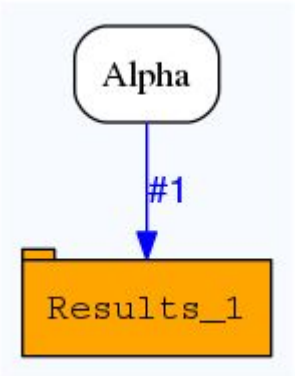
semaphores

Dataflows and events

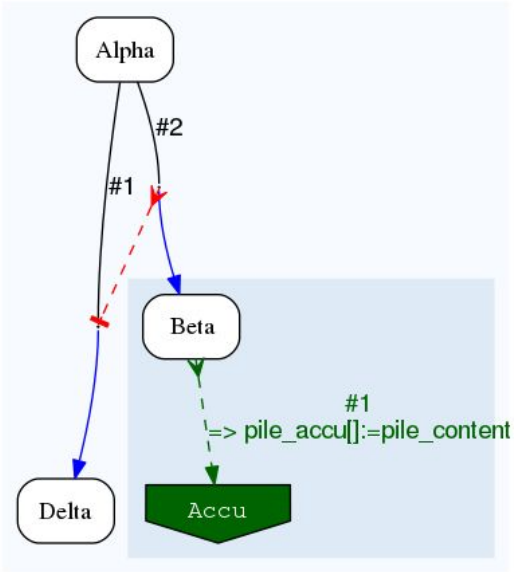
- When each job runs, it can generate zero or more dataflow events.
- Different events can be transmitted on different "branches."
- The consequences of these events are determined by how they are wired in the workflow



Create ("seed") new jobs



Store data in a table

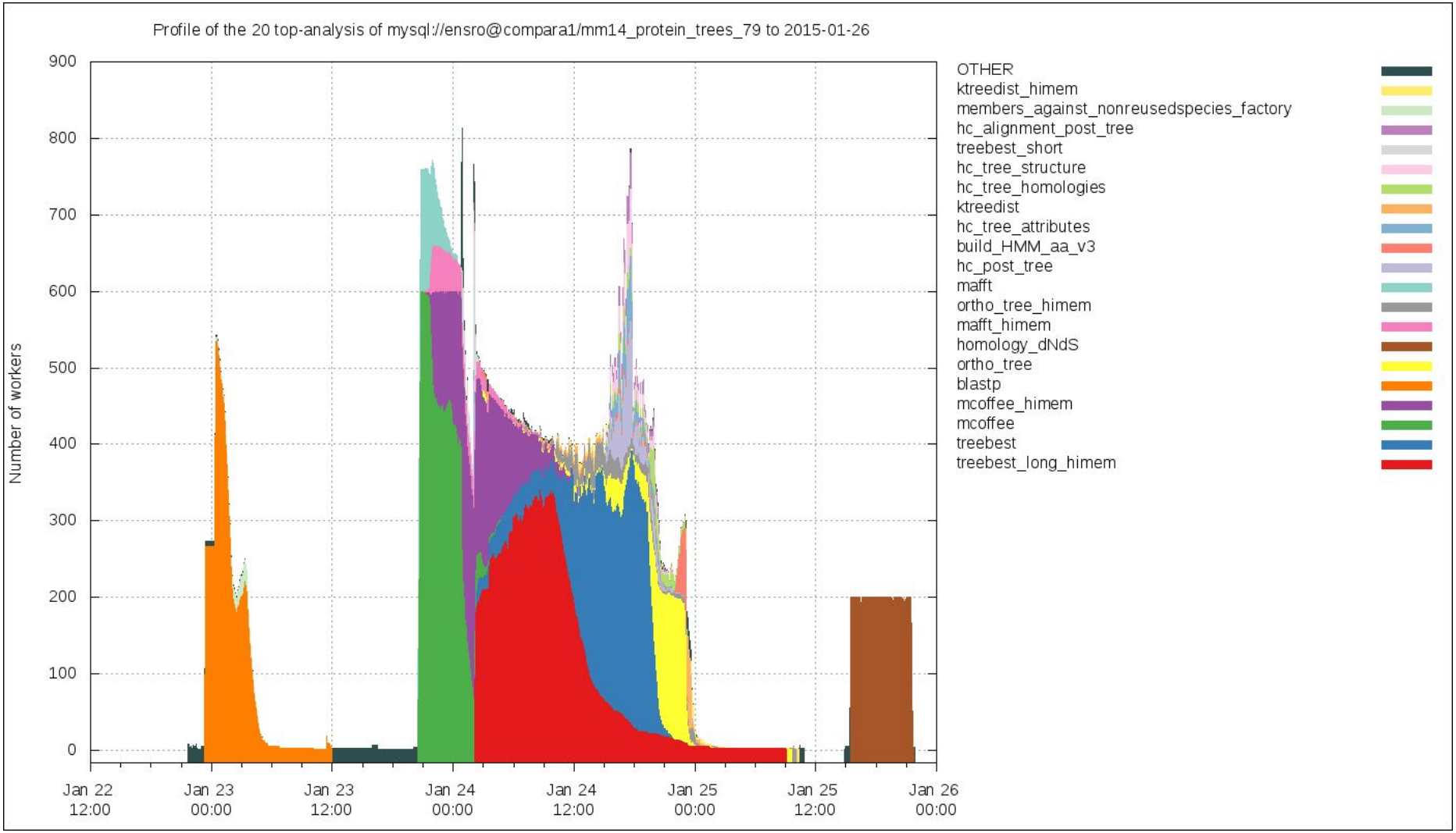


Store data in an "accu" data structure

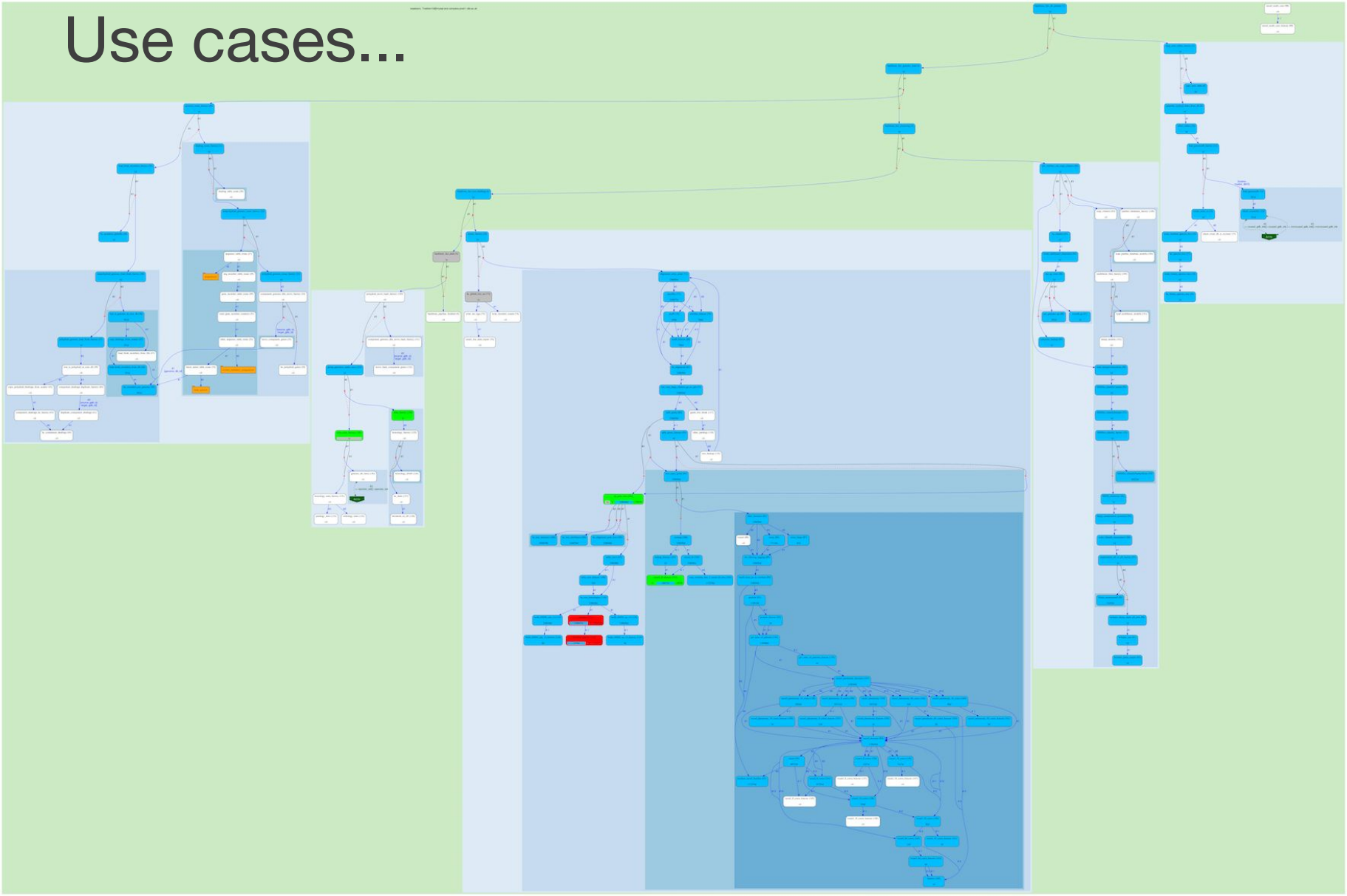
What's in the box?

- eHive code as a collection of Perl modules
- Scripts to instantiate and execute workflows
- Visualization and debugging utilities
  - Workflow structure
  - Resource usage
- Interfaces for different schedulers
  - LSF is officially supported
  - SGE, PBS Pro, and HTCondor reference implementations available
- guiHive - web based workflow management tool

Workflow analytics



Use cases...



Obtaining eHive and guiHive

https://ensembl-hive.readthedocs.io/

https://github.com/Ensembl/ensembl-hive
https://github.com/Ensembl/guiHive

https://hub.docker.com/r/ensemblorg/guihive/

Ensembl Acknowledgements

The Entire Ensembl Team

Bronwen L. Aken[1], Premanand Achuthan[1], Wasiu Akanni[1], M. Ridwan Amode[1], Friederike Bernsdorff[1], Jyothish Bhai[1], Konstantinos Billis[1], Denise Carvalho-Silva[1], Carla Cummins[1], Peter Clapham[2], Laurent Gil[1], Carlos García Girón[1], Leo Gordon[1], Thibaut Hourlier[1], Sarah E. Hunt[1], Sophie H. Janacek[1], Thomas Juettemann[1], Stephen Keenan[1], Matthew R. Laird[1], Ilias Lavidas[1], Thomas Maurel[1], William McLaren[1], Benjamin Moore[1], Daniel N. Murphy[1], Rishi Nag[1], Victoria Newman[1], Michael Nuhn[1], Chuang Kee Ong[1], Anne Parker[1], Mateus Patricio[1], Harpreet Singh Riat[1], Daniel Sheppard[1], Helen Sparrow[1], Kieron Taylor[1], Anja Thormann[1], Alessandro Vullo[1], Brandon Walts[1], Steven P. Wilder[1], Amonida Zadissa[1], Myrto Kostadima[1], Fergal J. Martin[1], Matthieu Muffato[1], Emily Perry[1], Magali Ruffier[1], Daniel M. Staines[1], Stephen J. Trevanion[1], Fiona Cunningham[1], Andrew Yates[1], Daniel R. Zerbino[1] and Paul Flicek[1,2,*]

[1]European Molecular Biology Laboratory, European Bioinformatics Institute, Wellcome Genome Campus, Hinxton, Cambridge CB10 1SD, UK and [2]Wellcome Trust Sanger Institute, Wellcome Genome Campus, Hinxton, Cambridge, CB10 1SA, UK

Funding

wellcome

EMBL

National Human Genome Research Institute

Open Targets

BBSRC bioscience for the future

TRANSFORMING GENETIC MEDICINE INITIATIVE

MedBioinformatics

Co-funded by the European Union