The joys of beekeeping

Leo Gordon

EMBL-EBI

wellcome trust sanger institute

Software pipeline (Q)

- ***What makes a software pipeline different from a script/program?*** (assuming top-down approach)

Software pipeline (A)

- Usually a computation that doesn't need interaction
- Availability of multiple computers (cluster/farm/grid) to perform the computation
- Availability of convenient tools/languages to automate running of this computation

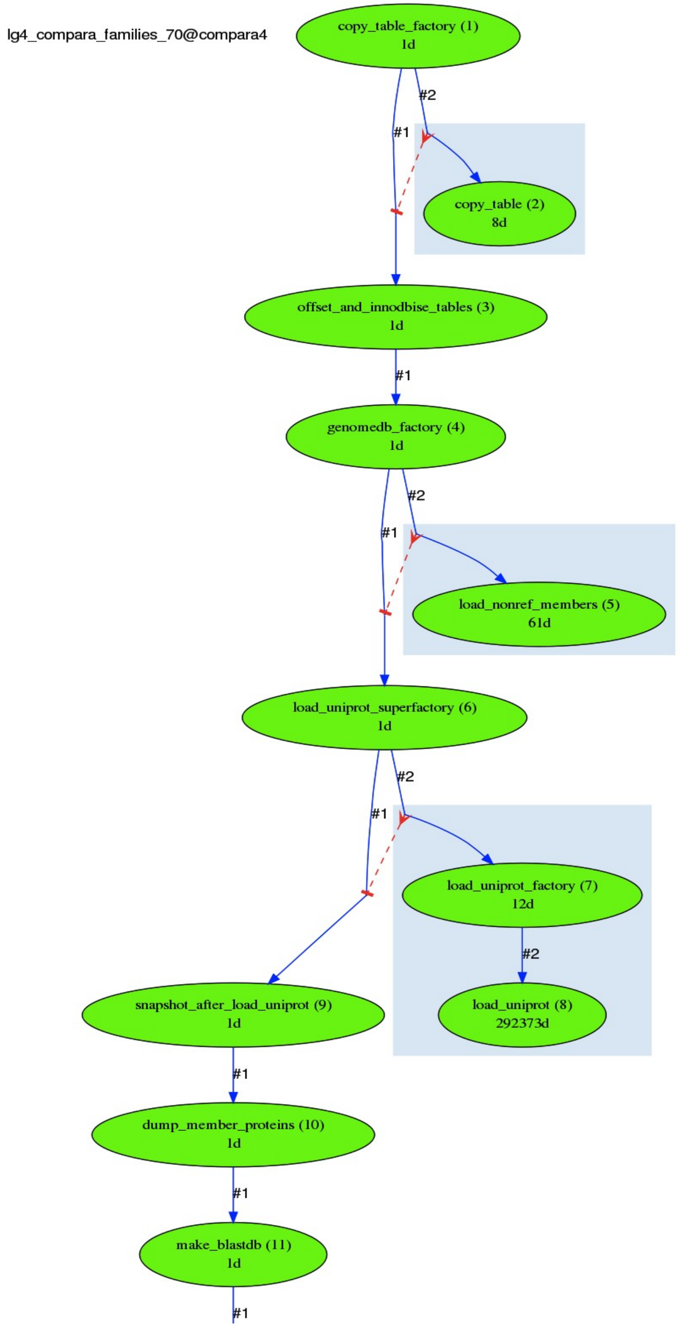
Software pipeline (requirements)

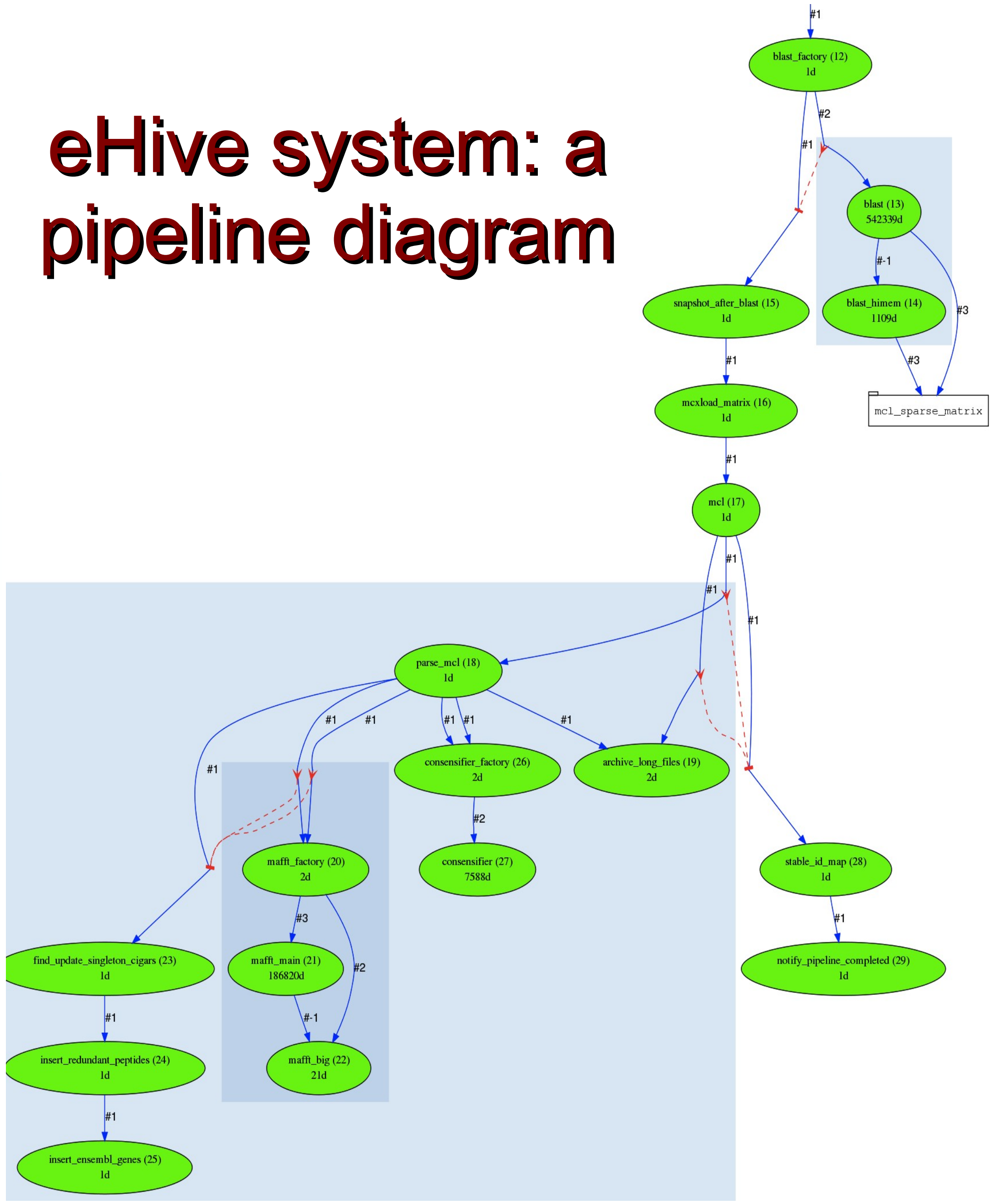
- Some code may take too long to compute sequentially
  - Transform independent loop iterations into individual “jobs”
    - rotate time into space
- Some code may crash eventually
  - Use checkpointing, restarting only individual jobs that crashed
    - rather than restarting the whole computation
  - Checkpointing in a non-sequential process means complex states
- Some code may be resource-greedy
  - Jobs can be given different (estimated) resources - memory, execution time, disk/temp storage space
  - Feedback collected on the actual usage of resources

eHive system: analyses, jobs, rules

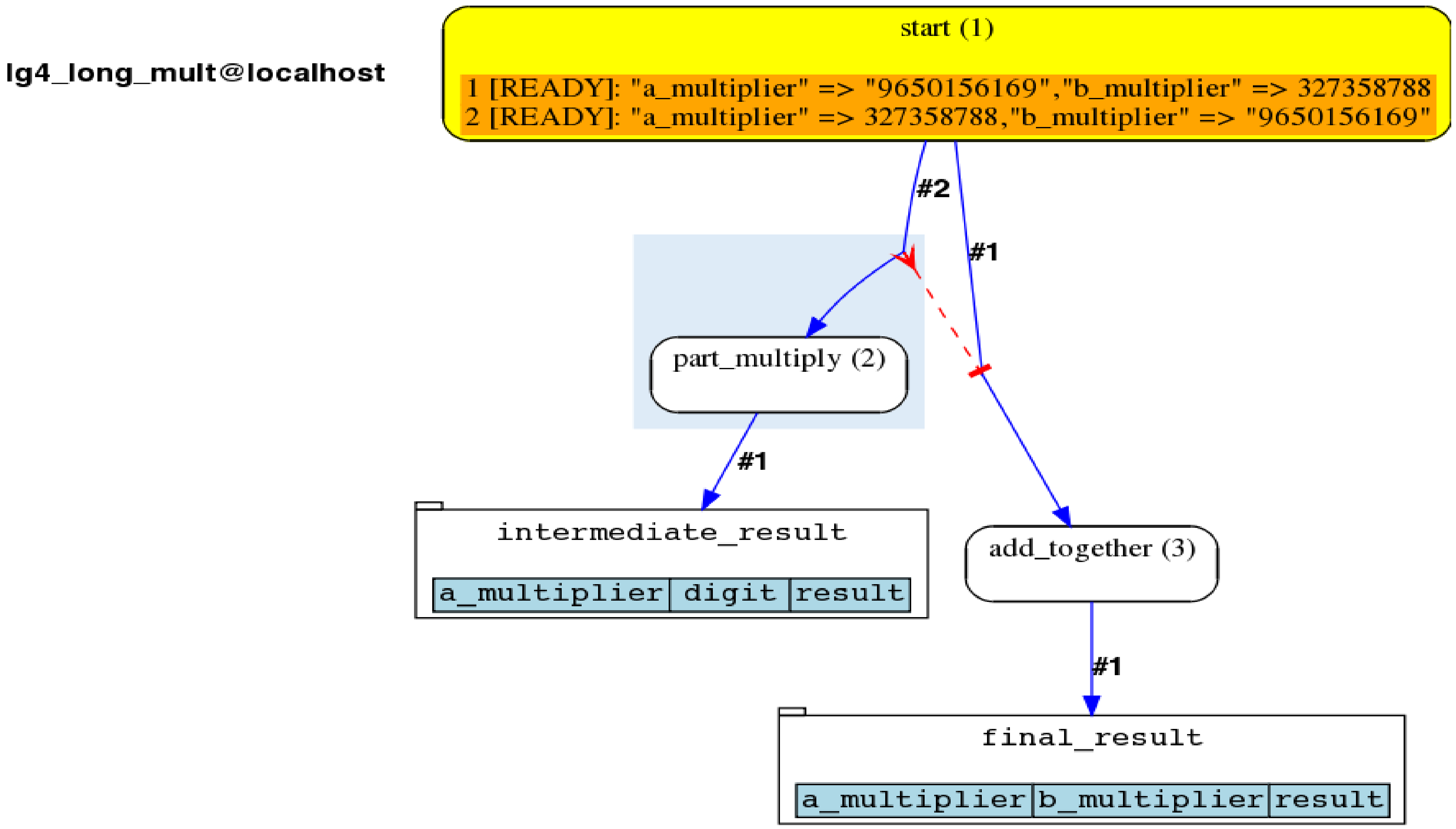
- Database-centric
- Individual pipeline instances created from config files and become “pipeline databases” with eHive schema
  - possibly extended with pipeline-specific tables
- Pipeline database contains a flow diagram with “*analyses*” as its nodes and “*rules*” as its edges.
- *Analyses* are abstract classes, *jobs* are specific instances, units of computation.
- *Jobs* can create other jobs (*dataflow rules*), block other jobs (*semaphores*) or whole analyses (*control rules*)

eHive system: a pipeline diagram

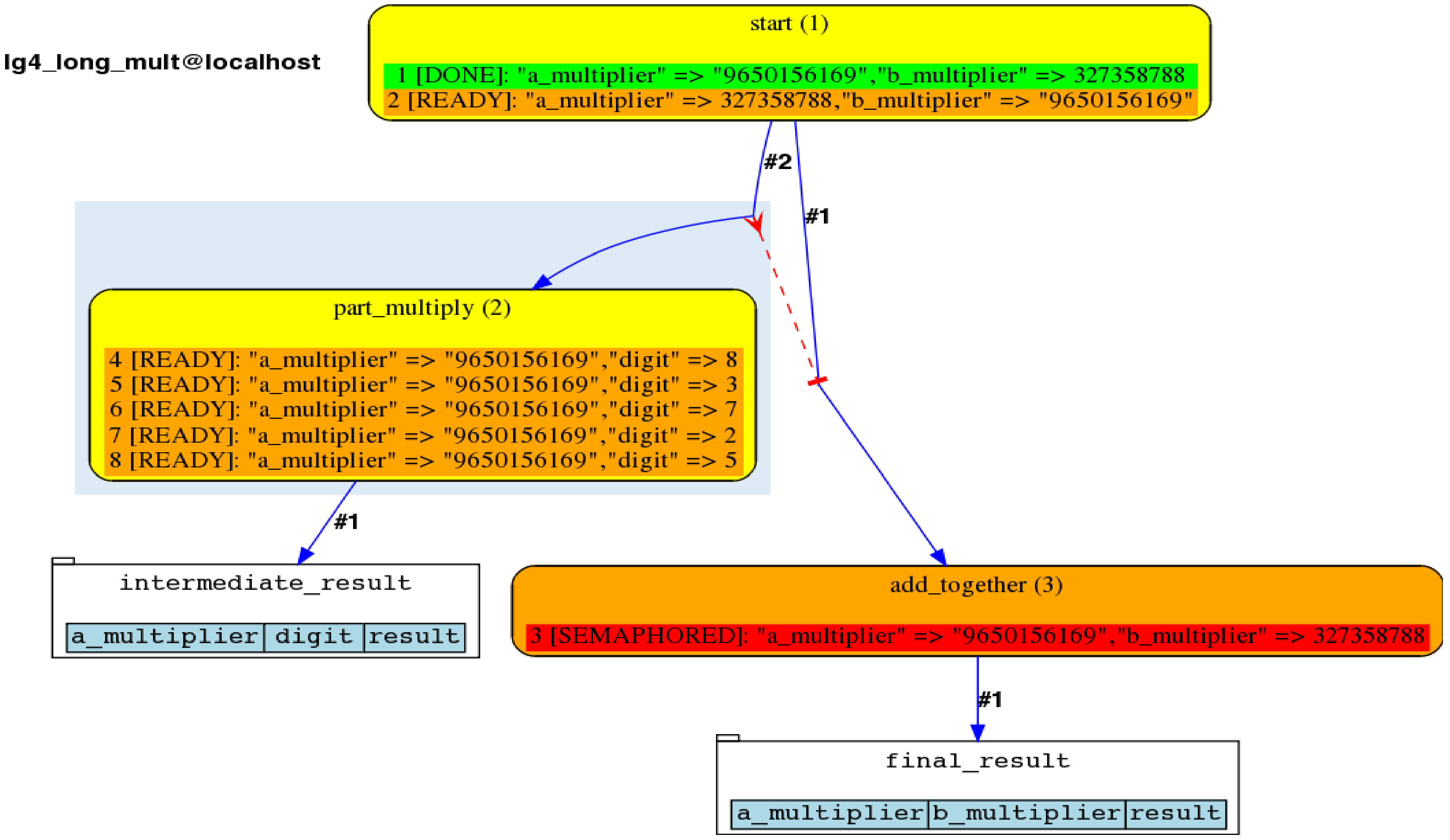

lg4_compara_families_70@compara4
copy_table_factory (1) 1d
#1
#2
copy_table (2) 8d
offset_and_innodbise_tables (3) 1d
#1
genomedb_factory (4) 1d
#1
#2
load_nonref_members (5) 61d
load_uniprot_superfactory (6) 1d
#1
#2
load_uniprot_factory (7) 12d
#2
load_uniprot (8) 292373d
snapshot_after_load_uniprot (9) 1d
#1
dump_member_proteins (10) 1d
#1
make_blastdb (11) 1d
#1



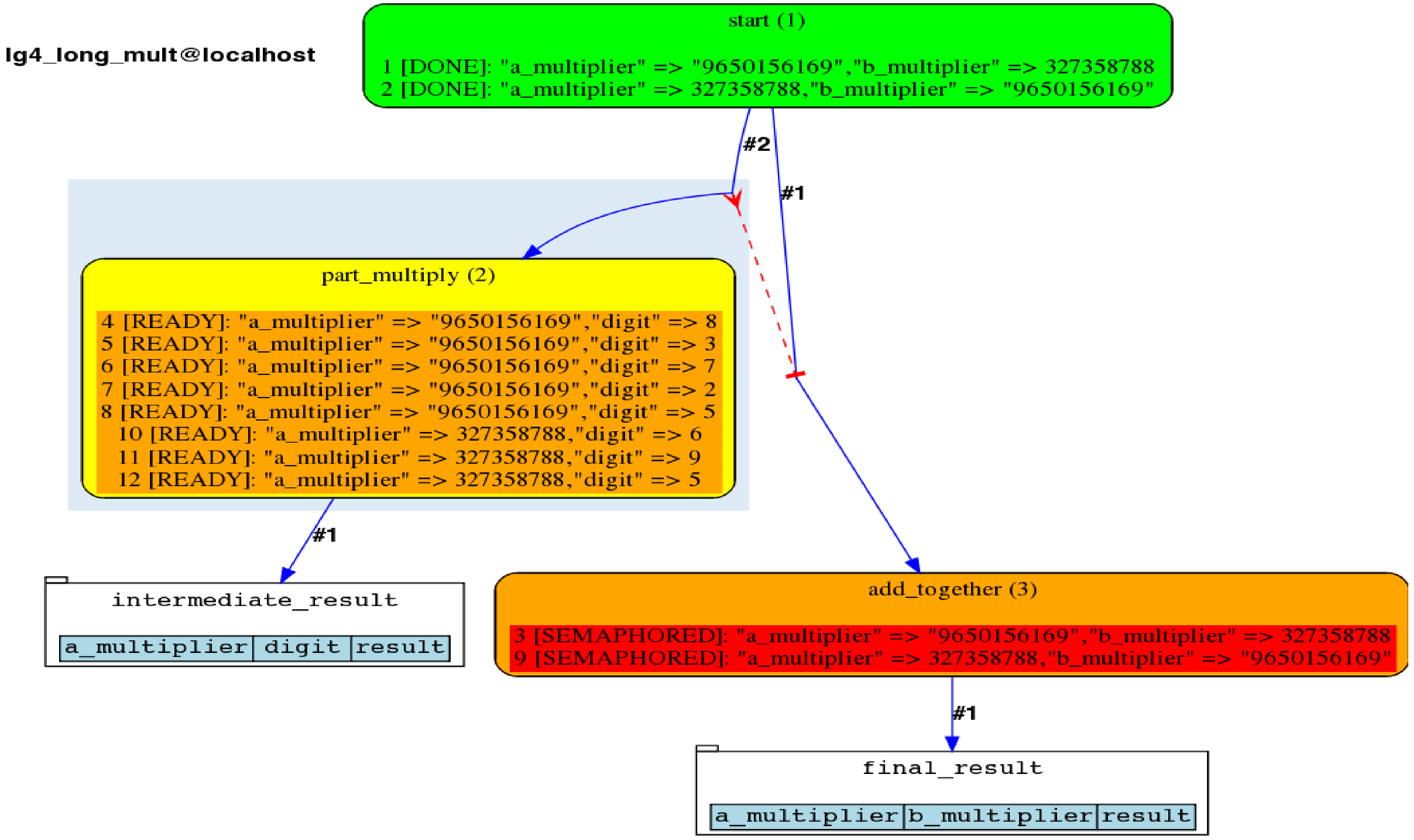
LongMultiplication example pipeline (1)



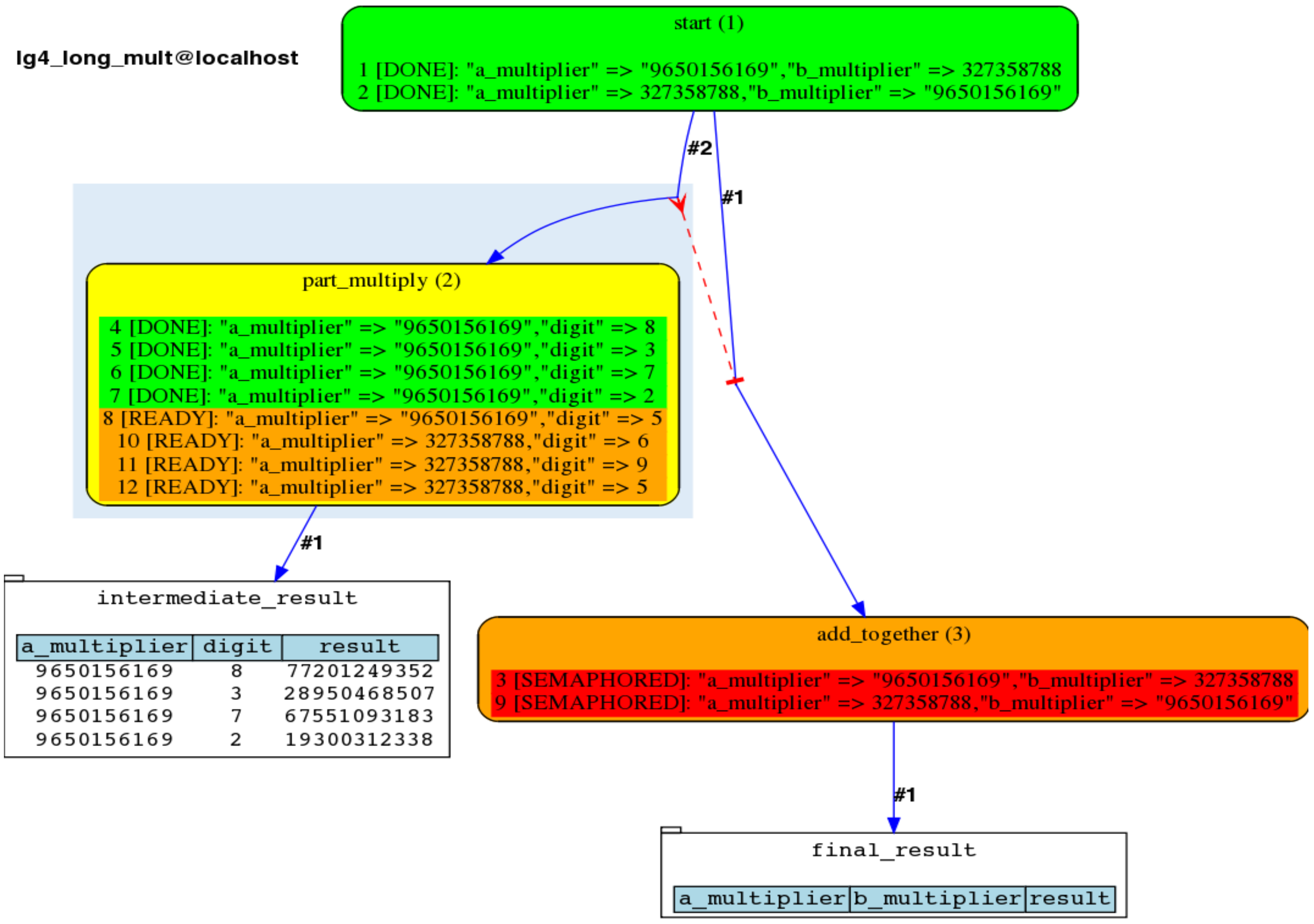
LongMultiplication example pipeline (2)


lg4_long_mult@localhost
start (1)
1 [DONE]: "a_multiplier" => "9650156169","b_multiplier" => 327358788
2 [READY]: "a_multiplier" => 327358788,"b_multiplier" => "9650156169"
#2
#1
part_multiply (2)
4 [READY]: "a_multiplier" => "9650156169","digit" => 8
5 [READY]: "a_multiplier" => "9650156169","digit" => 3
6 [READY]: "a_multiplier" => "9650156169","digit" => 7
7 [READY]: "a_multiplier" => "9650156169","digit" => 2
8 [READY]: "a_multiplier" => "9650156169","digit" => 5
#1
intermediate_result
a_multiplier | digit | result
add_together (3)
3 [SEMAPHORED]: "a_multiplier" => "9650156169","b_multiplier" => 327358788
#1
final_result
a_multiplier | b_multiplier | result

LongMultiplication example pipeline (3)


lg4_long_mult@localhost
start (1)
1 [DONE]: "a_multiplier" => "9650156169","b_multiplier" => 327358788
2 [DONE]: "a_multiplier" => 327358788,"b_multiplier" => "9650156169"
#2
#1
part_multiply (2)
4 [READY]: "a_multiplier" => "9650156169","digit" => 8
5 [READY]: "a_multiplier" => "9650156169","digit" => 3
6 [READY]: "a_multiplier" => "9650156169","digit" => 7
7 [READY]: "a_multiplier" => "9650156169","digit" => 2
8 [READY]: "a_multiplier" => "9650156169","digit" => 5
10 [READY]: "a_multiplier" => 327358788,"digit" => 6
11 [READY]: "a_multiplier" => 327358788,"digit" => 9
12 [READY]: "a_multiplier" => 327358788,"digit" => 5
#1
intermediate_result
a_multiplier
digit
result
add_together (3)
3 [SEMAPHORED]: "a_multiplier" => "9650156169","b_multiplier" => 327358788
9 [SEMAPHORED]: "a_multiplier" => 327358788,"b_multiplier" => "9650156169"
#1
final_result
a_multiplier
b_multiplier
result

LongMultiplication example pipeline (4)



lg4_long_mult@localhost
start (1)
1 [DONE]: "a_multiplier" => "9650156169","b_multiplier" => 327358788
2 [DONE]: "a_multiplier" => 327358788,"b_multiplier" => "9650156169"
#2
#1
part_multiply (2)
4 [DONE]: "a_multiplier" => "9650156169","digit" => 8
5 [DONE]: "a_multiplier" => "9650156169","digit" => 3
6 [DONE]: "a_multiplier" => "9650156169","digit" => 7
7 [DONE]: "a_multiplier" => "9650156169","digit" => 2
8 [READY]: "a_multiplier" => "9650156169","digit" => 5
10 [READY]: "a_multiplier" => 327358788,"digit" => 6
11 [READY]: "a_multiplier" => 327358788,"digit" => 9
12 [READY]: "a_multiplier" => 327358788,"digit" => 5
#1
add_together (3)
3 [SEMAPHORED]: "a_multiplier" => "9650156169","b_multiplier" => 327358788
9 [SEMAPHORED]: "a_multiplier" => 327358788,"b_multiplier" => "9650156169"
#1

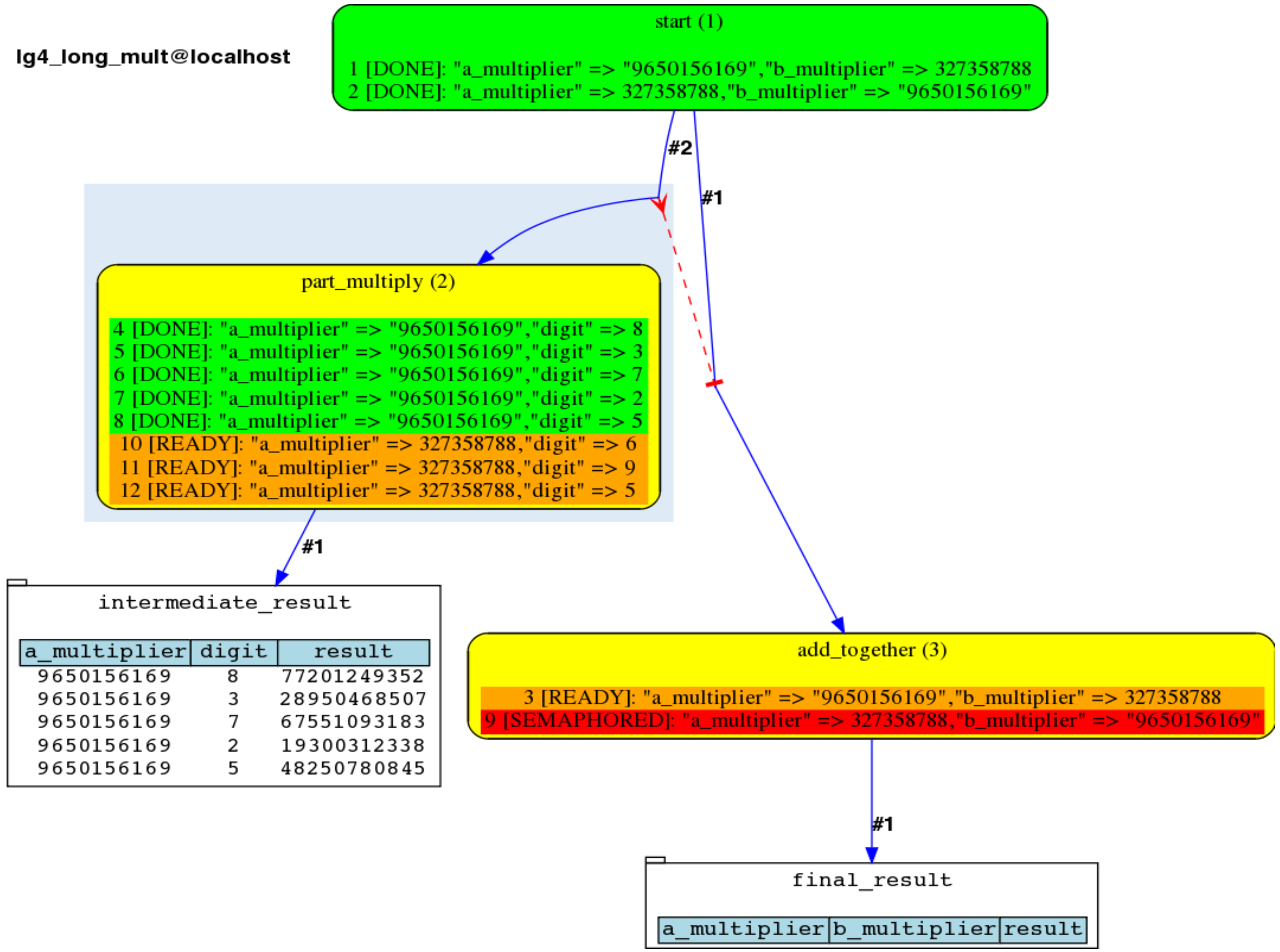
intermediate_result

a_multiplier	digit	result
9650156169	8	77201249352
9650156169	3	28950468507
9650156169	7	67551093183
9650156169	2	19300312338

final_result

a_multiplier	b_multiplier	result

LongMultiplication example pipeline (5)



start (1)

```
1 [DONE]: "a_multiplier" => "9650156169","b_multiplier" => 327358788
2 [DONE]: "a_multiplier" => 327358788,"b_multiplier" => "9650156169"
```

part_multiply (2)

```
4 [DONE]: "a_multiplier" => "9650156169","digit" => 8
5 [DONE]: "a_multiplier" => "9650156169","digit" => 3
6 [DONE]: "a_multiplier" => "9650156169","digit" => 7
7 [DONE]: "a_multiplier" => "9650156169","digit" => 2
8 [DONE]: "a_multiplier" => "9650156169","digit" => 5
10 [READY]: "a_multiplier" => 327358788,"digit" => 6
11 [READY]: "a_multiplier" => 327358788,"digit" => 9
12 [READY]: "a_multiplier" => 327358788,"digit" => 5
```

add_together (3)

```
3 [READY]: "a_multiplier" => "9650156169","b_multiplier" => 327358788
9 [SEMAPHORED]: "a_multiplier" => 327358788,"b_multiplier" => "9650156169"
```

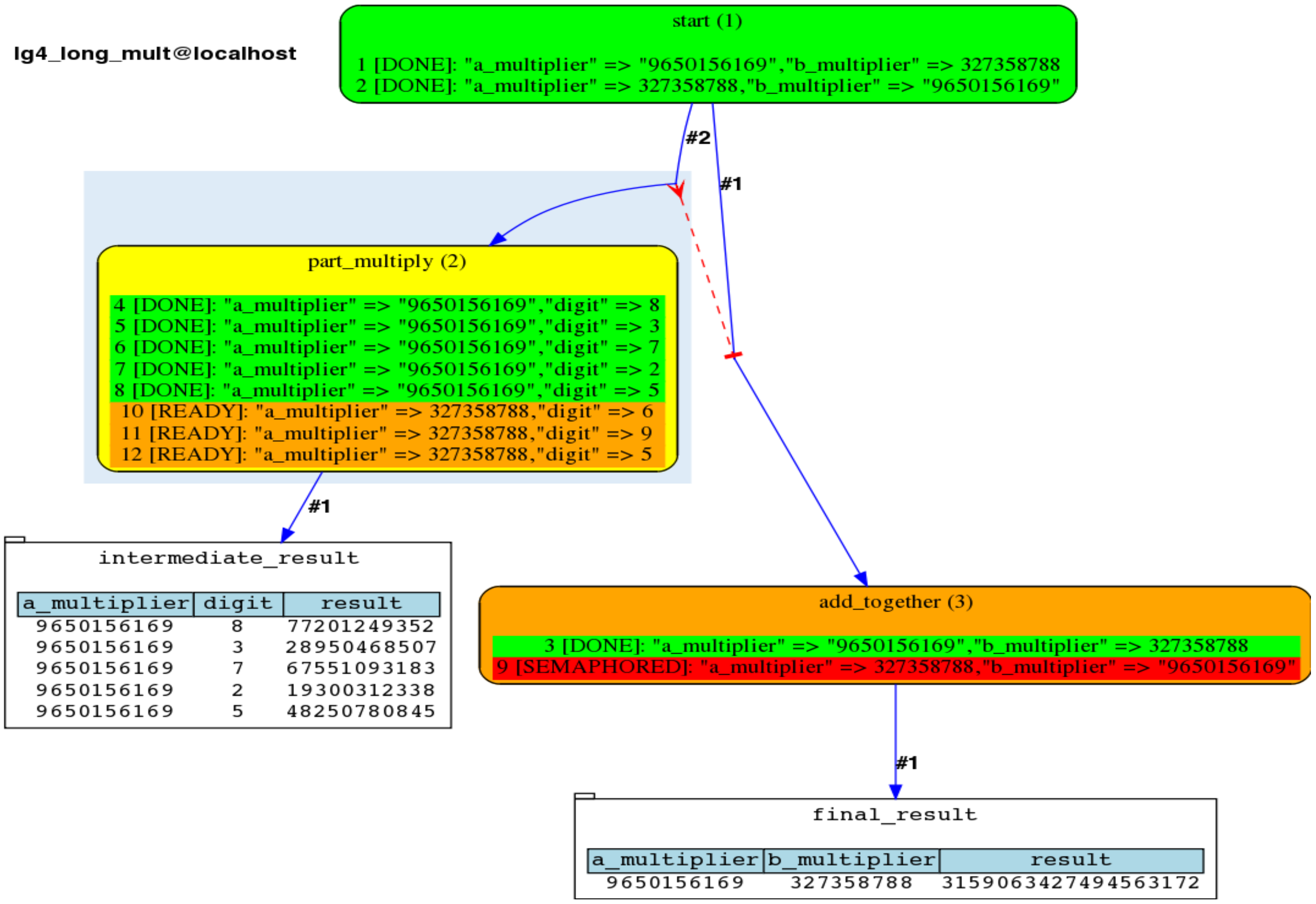
intermediate_result

a_multiplier	digit	result
9650156169	8	77201249352
9650156169	3	28950468507
9650156169	7	67551093183
9650156169	2	19300312338
9650156169	5	48250780845

final_result

a_multiplier	b_multiplier	result

LongMultiplication example pipeline (6)



start (1)

1 [DONE]: "a_multiplier" => "9650156169","b_multiplier" => 327358788
2 [DONE]: "a_multiplier" => 327358788,"b_multiplier" => "9650156169"

part_multiply (2)

4 [DONE]: "a_multiplier" => "9650156169","digit" => 8
5 [DONE]: "a_multiplier" => "9650156169","digit" => 3
6 [DONE]: "a_multiplier" => "9650156169","digit" => 7
7 [DONE]: "a_multiplier" => "9650156169","digit" => 2
8 [DONE]: "a_multiplier" => "9650156169","digit" => 5
10 [READY]: "a_multiplier" => 327358788,"digit" => 6
11 [READY]: "a_multiplier" => 327358788,"digit" => 9
12 [READY]: "a_multiplier" => 327358788,"digit" => 5

intermediate_result

a_multiplier	digit	result
9650156169	8	77201249352
9650156169	3	28950468507
9650156169	7	67551093183
9650156169	2	19300312338
9650156169	5	48250780845

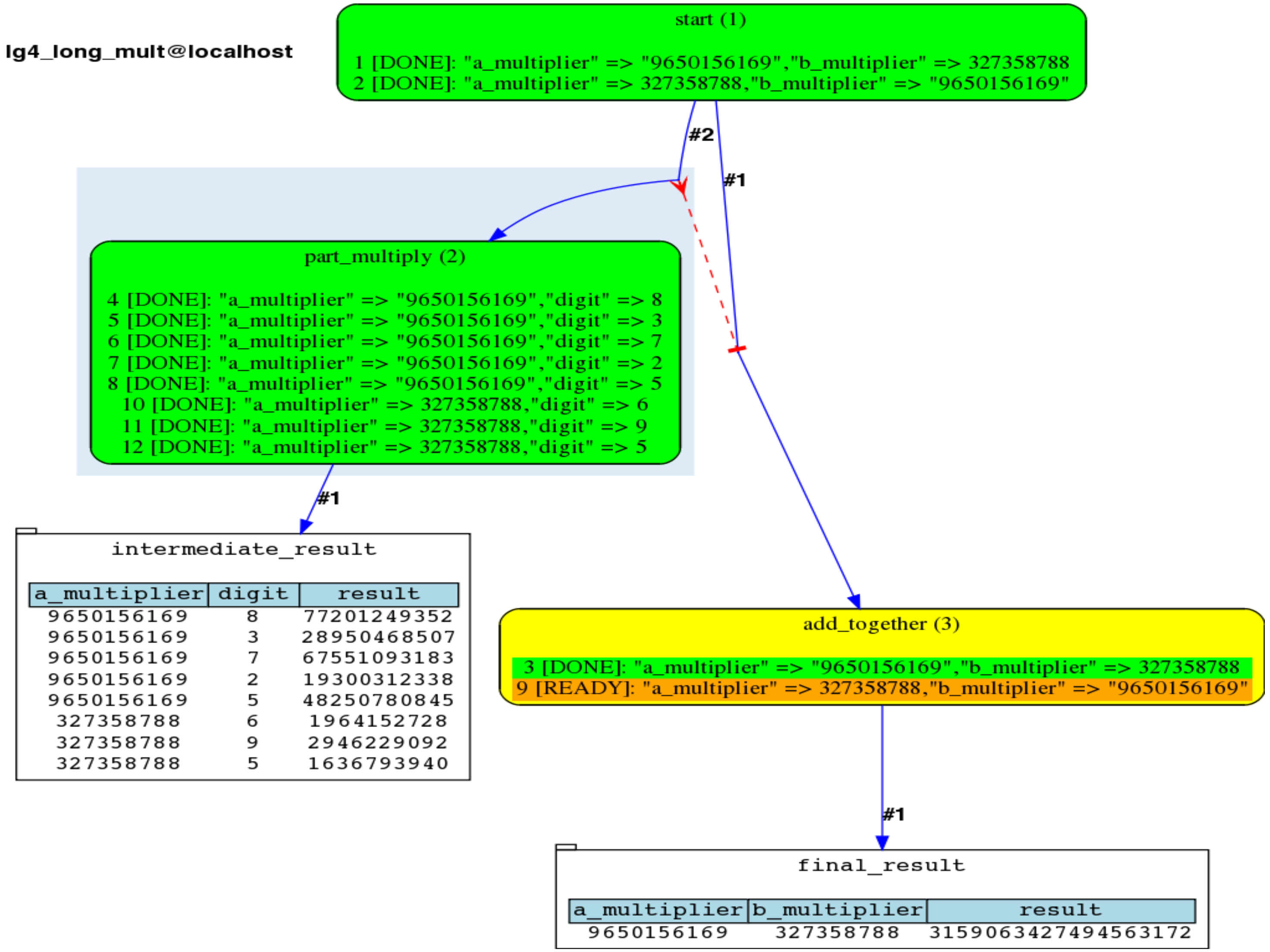
add_together (3)

3 [DONE]: "a_multiplier" => "9650156169","b_multiplier" => 327358788
9 [SEMAPHORED]: "a_multiplier" => 327358788,"b_multiplier" => "9650156169"

final_result

a_multiplier	b_multiplier	result
9650156169	327358788	3159063427494563172

LongMultiplication example pipeline (7)



start (1)

1 [DONE]: "a_multiplier" => "9650156169","b_multiplier" => 327358788
2 [DONE]: "a_multiplier" => 327358788,"b_multiplier" => "9650156169"

part_multiply (2)

4 [DONE]: "a_multiplier" => "9650156169","digit" => 8
5 [DONE]: "a_multiplier" => "9650156169","digit" => 3
6 [DONE]: "a_multiplier" => "9650156169","digit" => 7
7 [DONE]: "a_multiplier" => "9650156169","digit" => 2
8 [DONE]: "a_multiplier" => "9650156169","digit" => 5
10 [DONE]: "a_multiplier" => 327358788,"digit" => 6
11 [DONE]: "a_multiplier" => 327358788,"digit" => 9
12 [DONE]: "a_multiplier" => 327358788,"digit" => 5

intermediate_result

a_multiplier	digit	result
9650156169	8	77201249352
9650156169	3	28950468507
9650156169	7	67551093183
9650156169	2	19300312338
9650156169	5	48250780845
327358788	6	1964152728
327358788	9	2946229092
327358788	5	1636793940

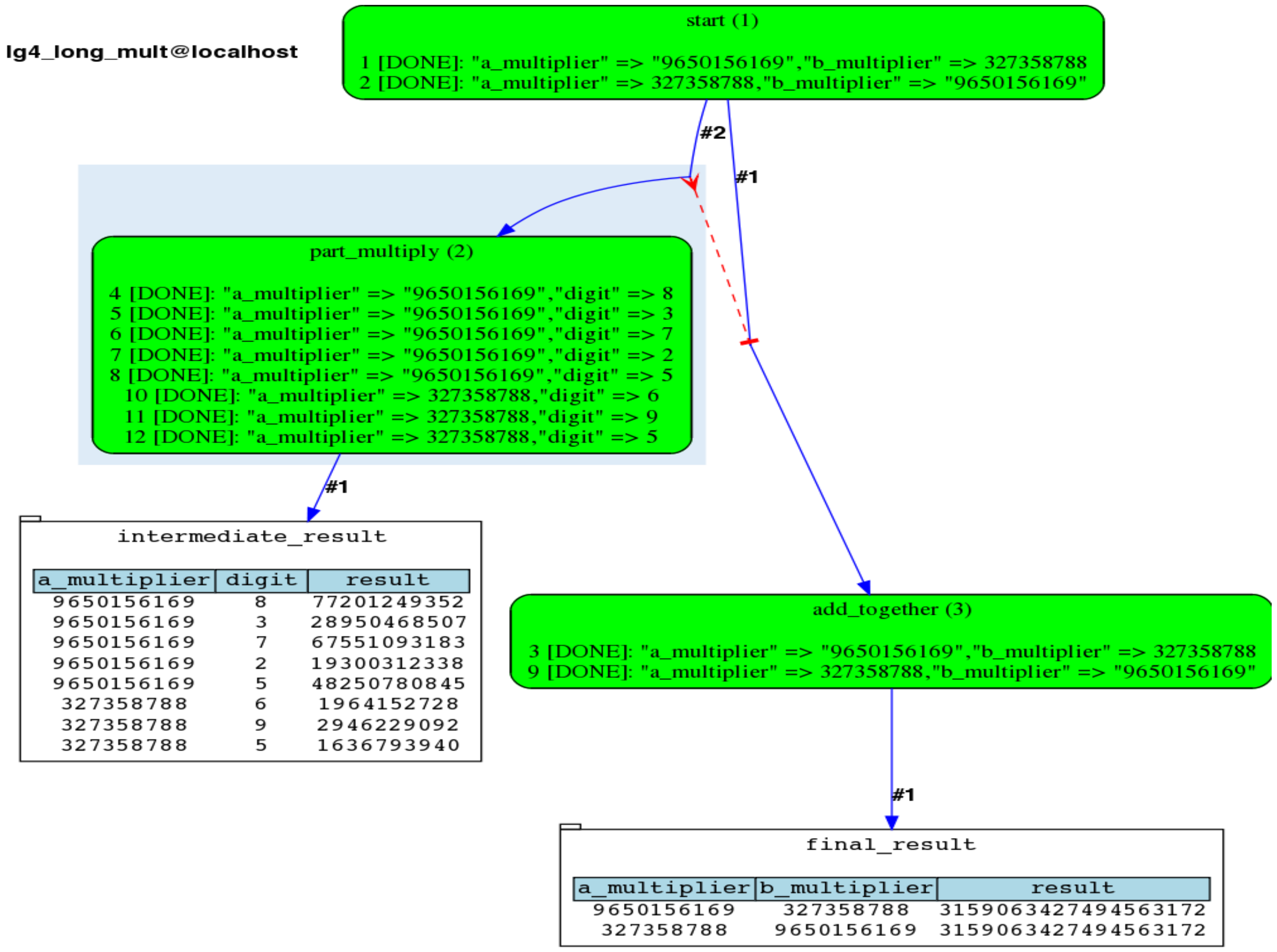
add_together (3)

3 [DONE]: "a_multiplier" => "9650156169","b_multiplier" => 327358788
9 [READY]: "a_multiplier" => 327358788,"b_multiplier" => "9650156169"

final_result

a_multiplier	b_multiplier	result
9650156169	327358788	3159063427494563172

LongMultiplication example pipeline (8)



```
start (1)

1 [DONE]: "a_multiplier" => "9650156169","b_multiplier" => 327358788
2 [DONE]: "a_multiplier" => 327358788,"b_multiplier" => "9650156169"
```

```
part_multiply (2)

4 [DONE]: "a_multiplier" => "9650156169","digit" => 8
5 [DONE]: "a_multiplier" => "9650156169","digit" => 3
6 [DONE]: "a_multiplier" => "9650156169","digit" => 7
7 [DONE]: "a_multiplier" => "9650156169","digit" => 2
8 [DONE]: "a_multiplier" => "9650156169","digit" => 5
10 [DONE]: "a_multiplier" => 327358788,"digit" => 6
11 [DONE]: "a_multiplier" => 327358788,"digit" => 9
12 [DONE]: "a_multiplier" => 327358788,"digit" => 5
```

```
add_together (3)

3 [DONE]: "a_multiplier" => "9650156169","b_multiplier" => 327358788
9 [DONE]: "a_multiplier" => 327358788,"b_multiplier" => "9650156169"
```

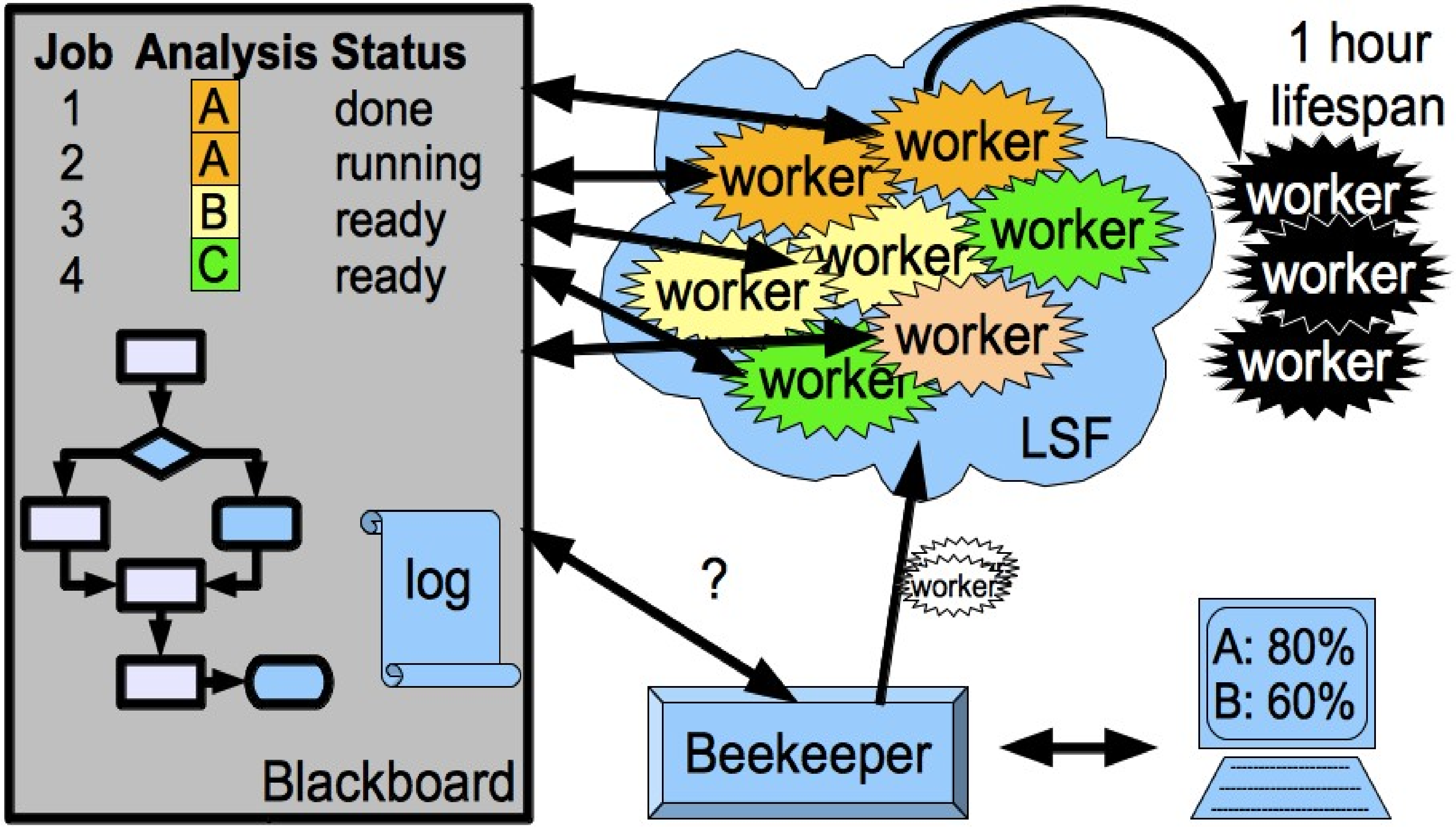
intermediate_result

a_multiplier	digit	result
9650156169	8	77201249352
9650156169	3	28950468507
9650156169	7	67551093183
9650156169	2	19300312338
9650156169	5	48250780845
327358788	6	1964152728
327358788	9	2946229092
327358788	5	1636793940

final_result

a_multiplier	b_multiplier	result
9650156169	327358788	3159063427494563172
327358788	9650156169	3159063427494563172

Workers, beekeeper, [re]specialization



Multi-meadow scheduling

- The same beekeeper can now submit both farm Workers and local Workers (busy farms, lots of light “linking code” between farm-needing processes)
- This preference (as well as desired resources) can be set for each Analysis
- Helps to save compute time, esp. combined with re-specialization of Workers

Using Git for development

- Development is done in Git (locally)

- Regularly pushed to the internal Git server visible to Sanger network users

- Less frequent (once per release?) exports to the externally-visible CVS server

- Fancy to try?

```
git clone git.internal:/repos/git/ensembl/compara/ensembl-hive.git
```

- eHive mailing list:

  ehive-users@ebi.ac.uk

Acknowledgements

Current and previous members of Compara team

All users of eHive system for testing, feedback and ideas

Paul Flicek, Steve Searle and the entire Ensembl Team

Funding

wellcome trust

EMBL

National Human Genome Research Institute

BBSRC bioscience for the future

European Commission Framework Programme 7

SEVENTH FRAMEWORK PROGRAMME

Quantomics

From Sequence to Consequence : Tools for the Exploitation of Livestock Genomes

GEN2PHEN

BLUEPRINT epigenome