Klever - Feature #8149

Think on proper progress evaluation when several jobs are solved at once or/and much computational resources are available

04/24/2017 01:06 PM - Evgeny Novikov

Status:	Closed	**Start date:**	09/20/2017
Priority:	Urgent	**Due date:**	10/26/2017
Assignee:	Ilja Zakharov	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	1.0		
Published in build:			

Description

At the moment a progress of solving verification tasks and corresponding remaining times are shown properly just when one job is solving and there are computational resources that are enough just for solving one verification task. This makes the progress and times almost useless when several jobs are solved at once or/and much computational resources are available.

Subtasks:

Feature # 8445: Visualize useful progress	**Closed**
Feature # 8446: Fix and improve progress reporting	**Closed**
Feature # 8491: Update the progress reporting implementation in Bridge	**Rejected**

Related issues:

Related to Klever - Bug #8442: Names of components have been changed and test...	**Rejected**	**09/19/2017**	
Has duplicate Klever - Feature #8444: Restore progress calculation	**Rejected**	**09/19/2017**	
Blocked by Klever - Feature #8536: Untie coverage report from any components ...	**Closed**	**10/31/2017**	**11/01/2017**

History

#1 - 04/24/2017 03:45 PM - Evgeny Novikov

BTW, the same problem is for several sub-jobs.

#2 - 06/22/2017 09:34 PM - Evgeny Novikov

- Assignee deleted (Ilja Zakharov)

- Priority changed from Urgent to High

There are too many very high priority issues. This one can be done after really important features will be supported.

#3 - 09/20/2017 11:45 AM - Evgeny Novikov

- Assignee set to Ilja Zakharov

- Priority changed from High to Urgent

- Target version set to 1.0

In addition to the requested improvements it will be necessary to fix progress calculation at all (that was mentioned in #8444).

#4 - 09/20/2017 11:57 AM - Evgeny Novikov

Ilja suggests to change the progress API between Bridge and Core so that there will be additional report fields intended just for progress calculation and there shouldn't any references to concrete Core component names (something similar to #8442).

#5 - 10/10/2017 03:40 PM - Ilja Zakharov

To restore functionality of progress evaluation I propose to do the following changes in Bridge first:
1) Implement a separate request to Bridge (not as a report) with the following data: {
"total tasks to be generated": 11111,
"tasks failed": 122,

"tasls solved": 500,
"average wall time to finish": 3766,
"wall time spend on solution": 2300
}
This request Core can do several times during the solution.
If Core has several sub-jobs it will not send any data at all or do it as there is the only sub-job.

2) Bridge should just store and visualize the data as is calculating the percentage as (failed + solved) * 100/total.

3) If Core provided data to Bridge and the job has PROCESSING status Bridge should send received data to scheduler during service/get_jobs_and_tasks request adding the following section: {
....
"jobs progress": {
"job_id": {data as is sent by Core}
}
}
Addiction of the section can be done only on update the data to economy traffic.

#6 - 10/11/2017 12:05 PM - Evgeny Novikov

Ilja Zakharov wrote:

> To restore functionality of progress evaluation I propose to do the following changes in Bridge first:
> 1) Implement a separate request to Bridge (not as a report) with the following data: {
> "total tasks to be generated": 11111,
> "tasks failed": 122,
> "tasls solved": 500,
> "average wall time to finish": 3766,
> "wall time spend on solution": 2300
> }

Minor improvements:
"tasks failed" -> "failed tasks"
"tasls solved" -> "solved tasks"
"average wall time to finish" -> "expected time for solving tasks"
"wall time spend on solution" -> "elapsed time for solving tasks"

> This request Core can do several times during the solution.

My suggestion is to send just changed values since, say, "total tasks to be generated" will not ever change.

> If Core has several sub-jobs it will not send any data at all or do it as there is the only sub-job.

So, Bridge should expect that there can be no progress reports for some verification jobs.

> 2) Bridge should just store and visualize the data as is calculating the percentage as (failed + solved) * 100/total.

I suggest the following formula: 100 * solved / (total - failed). This should be calculated just if failed < total. Otherwise if failed = total progress can be hidden because of tasks generation/solution finishes.

#7 - 10/11/2017 05:31 PM - Ilja Zakharov

> My suggestion is to send just changed values since, say, "total tasks to be generated" will not ever change.

I disagree with this point. Since Bridge does not need doing any complicated calculation, let's allow changing the numbers. However, I am not sure that we will change them, but anyway such artificial restrictions are really not necessary.

#8 - 10/12/2017 09:37 AM - Evgeny Novikov

Ilja Zakharov wrote:

> > My suggestion is to send just changed values since, say, "total tasks to be generated" will not ever change.
>
> I disagree with this point. Since Bridge does not need doing any complicated calculation, let's allow changing the numbers. However, I am not sure that we will change them, but anyway such artificial restrictions are really not necessary.

I didn't restrict any changes, although it isn't clear. I just suggested to send data incrementally, i.e. send just changed values and likely even after some configurable period of time. For instance, users can request to update a progress just each 30 seconds or each 5 minutes.

#9 - 10/16/2017 10:52 AM - Evgeny Novikov

Ilja Zakharov wrote:

> If Core has several sub-jobs it will not send any data at all or do it as there is the only sub-job.

I suggest a quite simple for implementation and useful for users approach to evaluate a progress for jobs with sub-jobs. Like with task let's evaluate the number of sub-jobs (total, solved, failed) and an average wall time spent on their solution. So, Bridge will show a progress of sub-jobs solution rather than tasks solution. Regarding to data there can be following additional fields:

```
{
   "sub-jobs to be solved": 50,
   "failed sub-jobs": 5,
   "solved sub-jobs": 25,
```

```
   "expected time for solving sub-jobs": 3766,
   "elapsed time for solving sub-jobs": 2300
}
```

Also, I suggest to name "total tasks to be generated" as "tasks to be generated".

#10 - 10/16/2017 11:37 AM - Ilja Zakharov

> Regarding to data there can be following additional fields

Ok, lets do it also. I would propose to send the data within the same request but make it possible to either attach either data about the tasks and sub-jobs, only about sub-jobs or only about tasks. Because corresponding information core will likely calculate indifferent places.

#11 - 10/30/2017 10:30 AM - Evgeny Novikov

We need a specification that will conclude this discussion and describe all new requests and their semantics in all details.

#12 - 10/31/2017 02:15 PM - Ilja Zakharov

Lets discuss it here: https://goo.gl/H2WFsw.

#13 - 11/01/2017 04:46 PM - Ilja Zakharov

Added issue #8536 as blocking since I am doing progress calculation on base of refactoring of Job.py which I cannot finish without separate total coverage request.

#14 - 11/07/2017 07:22 PM - Ilja Zakharov

- Status changed from New to Resolved

The updated progress implementation is available in branch 8149-new-progress. I will perform additional tests, but for simple examples all work nicely for both jobs with subjobs and without them.

#15 - 11/08/2017 02:35 PM - Ilja Zakharov

- Status changed from Resolved to Open

Decided to update the implementation.

#16 - 11/10/2017 11:21 AM - Ilja Zakharov

- Status changed from Open to Resolved

The final implementation is in branch "8149-new-progress". I see no any explicit bugs there at the moment.

#17 - 11/14/2017 01:38 PM - Evgeny Novikov

- Status changed from Resolved to Closed

I merged the branch to master in 459f75e7. At last we have quite proper progress evaluation and visualization that is extremely valuable for large production jobs.