openQA Project - action #88459

low-prio single-machine jobs can starve out high-prio multi-machine tests

2021-02-05 09:52 - okurz

Status:	Resolved	**Start date:**	2021-02-05
Priority:	Normal	**Due date:**	2021-03-05
Assignee:	mkittler	**% Done:**	0%
Category:	Concrete Bugs	**Estimated time:**	0.00 hour
Target version:	Ready		
Difficulty:			

Description

Observation

We automatically spawn openqa-investigate jobs which so far are always single-machine only and all with a high priority number that should mean that these jobs should be triggered only when other jobs corresponding to the same worker classes had been preferred. Seems like in many cases multi-machine tests can not be started as long as multiple openqa-investigate jobs are scheduled as they are preferred when they should not be preferred.

Acceptance criteria

- **AC1:** Scheduled multi-machine jobs are started before openqa-investigate jobs for corresponding worker classes

Suggestions

- Review the scheduling code which AFAIR should try to prefer multi-machine jobs by blocking relevant worker slots until enough free slots are free to execute the cluster

History

#1 - 2021-02-18 09:43 - mkittler

- Assignee set to mkittler

> Review the scheduling code which AFAIR should try to prefer multi-machine jobs by blocking relevant worker slots until enough free slots are free to execute the cluster

There's indeed such code. It should prioritize jobs part of relevant parallel clusters and at some point even "hold a worker" to avoid starvation.

Judging by the logs the code is triggered, e.g.:

```
[2021-02-18T10:32:57.0587 CET] [debug] [pid:1401] Holding worker 967 for job 5480608 to avoid starvation
```

However, when looking at https://openqa.suse.de/admin/workers/967 it turned out that the worker has actually just started working on the single job https://openqa.suse.de/tests/5480515 (instead of https://openqa.suse.de/tests/5480608 which was still scheduled). So despite the log message it doesn't really work. I've also seen a few similar cases (using grep 'Discarding job' /var/log/openqa_scheduler and grep 'Holding worker' /var/log/openqa_scheduler on OSD to find relevant log messages).

#2 - 2021-02-18 10:30 - mkittler

Judging by the full scheduler log the situation described in the previous comment is actually not a problem at all. The scheduler run where the held job was assigned looks like this:

```
[2021-02-18T10:27:52.0180 CET] [debug] [pid:1401] Need to schedule 3 parallel jobs for job 5480608 (with prior
ity 0)
[2021-02-18T10:27:52.0180 CET] [debug] [pid:1401] Holding worker 599 for job 5480608 to avoid starvation
[2021-02-18T10:27:52.0181 CET] [debug] [pid:1401] Need to schedule 3 parallel jobs for job 5481843 (with prior
ity 8)
[2021-02-18T10:27:52.0181 CET] [debug] [pid:1401] Discarding job 5481843 (with priority 8) due to incomplete p
arallel cluster
[2021-02-18T10:27:52.0181 CET] [debug] [pid:1401] Need to schedule 3 parallel jobs for job 5481845 (with prior
ity 10)
```

```
[2021-02-18T10:27:52.0181 CET] [debug] [pid:1401] Discarding job 5481843 (with priority 10) due to incomplete
parallel cluster
[2021-02-18T10:27:52.0181 CET] [debug] [pid:1401] Need to schedule 1 parallel jobs for job 5482359 (with prior
ity 30)
…
[2021-02-18T10:27:52.0182 CET] [debug] [pid:1401] Need to schedule 1 parallel jobs for job 5480515 (with prior
ity 50)
[2021-02-18T10:27:52.0182 CET] [debug] [pid:1401] Need to schedule 1 parallel jobs for job 5480516 (with prior
ity 50)
[2021-02-18T10:27:52.0182 CET] [debug] [pid:1401] Need to schedule 1 parallel jobs for job 5480517 (with prior
ity 50)
…
```

So we're just "holding" a different worker for that job. This should be ok as well because it doesn't matter which worker we're "holding" as long as we're holding some worker.

Judging by the rest of the log the held worker is never used for a different job within the same scheduler run. And due to their increased priority the parallel jobs are also tried to be scheduled at the very beginning of each scheduler run.

I still see one flaw: It looks like we would always only hold one worker per parallel parent. The code was likely written having simple parallel clusters with one parent and one child in mind. However, when a parent has multiple children we're still holding only one worker which is not sufficient. E.g. the parallel clusters which are currently starving on OSD consist of one parent and 2 children so we needed to hold at least 2 workers until a 3rd worker becomes available.

#3 - 2021-02-18 12:55 - mkittler

- Status changed from Workable to In Progress

I've created a unit test to check whether the flaw mentioned in my last comment is actually present: https://github.com/os-autoinst/openQA/pull/3741

It doesn't seem that way. If there are 3 parallel jobs (like the jobs I've seen on OSD) but only 2 matching workers, the 2 matching workers are actually "held".

Of course if there's only one matching worker the scheduler is only able to hold one worker as we can not reserve busy workers. I've also tested this by putting pop @mocked_free_workers; before the last OpenQA::Scheduler::Model::Jobs->singleton->schedule in my unit test. Then only one Holding worker … line is logged like on OSD. I would assume that it is simply that case which we see on OSD. So there's still no problem.

#4 - 2021-02-18 13:28 - mkittler

I've been checking more concrete examples on OSD. It looks like there actually are cases where 2 workers are held for a 3-job-cluster:

```
[2021-02-18T14:08:27.0654 CET] [debug] [pid:1401] Need to schedule 3 parallel jobs for job 5482683 (with prior
ity 0)
[2021-02-18T14:08:27.0654 CET] [debug] [pid:1401] Holding worker 927 for job 5482686 to avoid starvation
[2021-02-18T14:08:27.0654 CET] [debug] [pid:1401] Holding worker 1254 for job 5482683 to avoid starvation
```

Here 5482686 and 5482683 are part of the same cluster. In this example we see that it also works with mixed WORKER_CLASSes (tap and sap_sle15).

The first time this job reaches zero-prio where the the worker reservation kicks in:

```
[2021-02-18T13:10:48.0651 CET] [debug] [pid:1401] Need to schedule 3 parallel jobs for job 5482683 (with prior
ity 0)
[2021-02-18T13:10:48.0651 CET] [debug] [pid:1401] Holding worker 600 for job 5482683 to avoid starvation
```

It took a while to get there (compare the timestamps):

```
[2021-02-18T12:15:51.0257 CET] [debug] [pid:1401] Need to schedule 3 parallel jobs for job 5482683 (with prior
ity 50)
…
[2021-02-18T12:19:13.0863 CET] [debug] [pid:1401] Need to schedule 3 parallel jobs for job 5482683 (with prior
ity 50)
[2021-02-18T12:19:13.0863 CET] [debug] [pid:1401] Discarding job 5482683 (with priority 50) due to incomplete
parallel cluster
…
[2021-02-18T13:10:47.0569 CET] [debug] [pid:1401] Need to schedule 3 parallel jobs for job 5482683 (with prior
ity 1)
[2021-02-18T13:10:47.0569 CET] [debug] [pid:1401] Discarding job 5482683 (with priority 1) due to incomplete p
arallel cluster
```

We could make increasing the prio more aggressive, e.g. make it configurable.

#5 - 2021-02-18 15:47 - mkittler

PR for reducing the prio more aggressively (configurable via env variable): https://github.com/os-autoinst/openQA/pull/3742

> [16:23] fvogt: As already explained, it takes a while until the prio decreases. The worker is not really aggressively doing that and only puts other workers on hold if the prio reaches 0.
> [16:24] I'd argue that it should do that if there's no higher prio job

It would make sense to implement this as well.

#6 - 2021-02-19 04:07 - openqa_review

- Due date set to 2021-03-05

Setting due date based on mean cycle time of SUSE QE Tools

#7 - 2021-02-19 16:40 - mkittler

> I've now made the prio decreasing 10 times faster on o3. Feel free to adjust this yourself as needed (adjust env variable systemctl edit openqa-scheduler). But note that the change is not actually effective yet because https://github.com/os-autoinst/openQA/pull/3742 has not been deployed yet.

#8 - 2021-02-22 14:37 - mkittler

- Status changed from In Progress to Feedback

The change to make the prio decreasing 10 times faster should now be effective on o3.

#9 - 2021-03-03 13:41 - favogt

mkittler wrote:

> The change to make the prio decreasing 10 times faster should now be effective on o3.

Looking at the current state of the TW group, there are a few normal prio tests and the usual amount of low-prio investigation jobs queued, but but multimachine tests mostly finished already. So this indeed seems to work as it should!

#10 - 2021-03-03 13:43 - ggardet_arm

At least on aarch64, multimachine are now scheduled properly!

#11 - 2021-03-05 07:36 - okurz

- Status changed from Feedback to Resolved

Thanks all for the feedback. This should be enough to call this done.