openQA Project - action #25892

Scheduling parallel jobs

2017-10-10 13:37 - nadvornik

Status:	Resolved	**Start date:**	2017-10-10
Priority:	High	**Due date:**	
Assignee:	EDiGiacinto	**% Done:**	0%
Category:	Concrete Bugs	**Estimated time:**	0.00 hour
Target version:	Done		
Difficulty:			

Description

After upgrade to the new scheduler I have this problem:
I have this group of parallel jobs:
A
B PARALLEL_WITH A
C PARALLEL_WITH A,B
D PARALLEL_WITH A,B

The jobs use barriers to synchronize so they finish approximately at the same time.

On an openQA server with 4 workers I have this group scheduled multiple times:
A1, B1, C1, D1, A2, B2, C2, D2, A3, B3, C3, D3, ...

First group of jobs A1, B1, C1, D1 finishes normally but then it sometimes ends up with workers doing jobs A2, A3, A4, A5 in parallel and waiting forever for the rest of group.

It looks like some race condition in use of _prefer_parallel() function.

Related issues:

Related to openQA Project - action #40415: Concurrent jobs with dependencies ...	**Resolved**	**2018-08-29**

History

#1 - 2017-11-23 08:42 - coolo

- Target version set to Ready

tricky to test case I assume

#2 - 2018-01-31 14:57 - mkravec

We have issues with scheduler at QAM-CaaSP & QA-CaaSP tests.
QAM start 7 clusters (8 jobs per cluster) at the same time, sometimes in addition to QA (12+5+5+5+1+1) clusters.
We have 2*24 dedicated workers (maybe a bit overloaded) handling from 50 to 85 cluster jobs at the same time.

I talked to Ettore and he proposed to start cluster test only when there are enough free workers. I like this solution, this would allow to share CaaSP workers with SLE again.
This way we prevent situations where we have:

- 5 workers in 1st cluster
- 7 workers in 2nd cluster ...
- 6 workers in 3rd cluster but we actually need 8 workers to finish cluster test.

Current QAM issues:

- https://openqa.suse.de/tests/overview?distri=caasp&version=2.0&build=%3A6584%3Agrub2.1517325020&groupid=127
- https://openqa.suse.de/tests/overview?distri=caasp&version=2.0&build=%3A6038%3Arpm.1517327138&groupid=127
- https://openqa.suse.de/tests/overview?distri=caasp&version=2.0&build=%3A6393%3Atar.1517330032&groupid=127
- https://openqa.suse.de/tests/overview?distri=caasp&version=2.0&build=%3A4905%3Anfs-utils.1517323718&groupid=127

QA clusters (29 jobs) are more reliable atm. because they are usually started at different time then QAM (56 jobs)

#3 - 2018-02-01 09:22 - coolo

- Priority changed from Normal to High

#4 - 2018-02-13 15:06 - thehejik

- File slenkins_jobs_deadlock.png added

I have similar problem with slenkins (which is using mutexes and PARALLEL_WITH="sutX,sutY" in *-control job only) on my local openqa instance (fully updated today).

Problem is with a distribution of the workers over irrelevant jobs that cannot be finished without triggering their sibling jobs.

Please see attached screen - I'm using 5 worker processes and some of those workers are blocked in different jobs. It will lead to a deadlock because those partially started jobs will stuck and then killed after 2 hours.

#5 - 2018-02-23 09:44 - pcervinka

We face similar situation in HPC group:
https://openqa.suse.de/tests/overview?distri=sle&version=15&build=473.4&groupid=130

Although are jobs triggered, sometimes is information about relation lost and test wait for each other in deadlock.

#6 - 2018-02-26 08:52 - sebchlad

I was updated by Coolo regarding this problem (which also affects HPC after pcervinka's nice improvements to HPC testing) and I understand this is rather significant work to be done, so we shall not expect any quick solution.

I just wonder however about what Coolo said and what Martin nicely described: "I talked to Ettore and he proposed to start cluster test only when there are enough free workers."

Isn't that quick workaround which we could have before a proper solution?
I understand/guess this would impact time exception but still it might be OK.

#7 - 2018-02-26 08:59 - coolo

Sure, we'll review your patches then.

#8 - 2018-02-26 09:57 - sebchlad

yeah I was actually wondering about this... I would perhaps answer the same way :-)

#9 - 2018-02-26 12:44 - EDiGiacinto

sebchlad wrote:

> I was updated by Coolo regarding this problem (which also affects HPC after pcervinka's nice improvements to HPC testing) and I understand this is rather significant work to be done, so we shall not expect any quick solution.
>
> I just wonder however about what Coolo said and what Martin nicely described: "I talked to Ettore and he proposed to start cluster test only when there are enough free workers."
>
> Isn't that quick workaround which we could have before a proper solution?

That workaround is the only other solution i see rather from migrating the whole scheduling to AMQP or start talking about SAT solvers - which would be even more painful.

> I understand/guess this would impact time exception but still it might be OK.

What will take most of the time, is to be able to deliver this feature with the guarantee to not impact other jobs.

#10 - 2018-02-27 11:48 - oholecek

We (me and nadvornik) took a brief look into this and think that passing $allocating (from https://github.com/os-autoinst/openQA/blob/master/lib/OpenQA/Scheduler/Scheduler.pm#L242) to job_grab and then to _prefer_parallel as '$running' should avoid scheduling of more parallel job groups when there are not enough workers for all parallel groups. What do you think?

Btw. what would migrating the whole scheduling to AMQP solve?

#11 - 2018-02-27 12:05 - EDiGiacinto

- Status changed from New to In Progress

- Assignee set to EDiGiacinto

#12 - 2018-02-28 10:51 - EDiGiacinto

oholecek wrote:

> We (me and nadvornik) took a brief look into this and think that passing $allocating (from https://github.com/os-autoinst/openQA/blob/master/lib/OpenQA/Scheduler/Scheduler.pm#L242) to job_grab and then to _prefer_parallel as '$running' should avoid scheduling of more parallel job groups when there are not enough workers for all parallel groups. What do you think?

It might just work, but i would also add a further check at the end of the allocation round

> Btw. what would migrating the whole scheduling to AMQP solve?

e.g. avoid using websockets to send jobs, treating worker_classes like queues and more important trying to formalize the problem during the process

#13 - 2018-03-01 08:37 - EDiGiacinto

EDiGiacinto wrote:

> oholecek wrote:
>
> > We (me and nadvornik) took a brief look into this and think that passing $allocating (from https://github.com/os-autoinst/openQA/blob/master/lib/OpenQA/Scheduler/Scheduler.pm#L242) to job_grab and then to _prefer_parallel as '$running' should avoid scheduling of more parallel job groups when there are not enough workers for all parallel groups. What do you think?
>
> It might just work, but i would also add a further check at the end of the allocation round

small update, this wasn't the only thing needed, i'm currently testing now on staging the required changes.

To guarantee (in a best-effort fashion) the assignment (scheduled -> assigned) in parallel of cluster jobs, will be introduced a new variable.

> Btw. what would migrating the whole scheduling to AMQP solve?
>
> e.g. avoid using websockets to send jobs, treating worker_classes like queues and more important trying to formalize the problem during the process

#14 - 2018-03-01 10:25 - ldevulder

pcervinka wrote:

> We face similar situation in HPC group:

Same for HA tests sometimes...

#15 - 2018-03-01 12:59 - coolo

Why would you need a new variable? Why isn't PARALLEL_WITH good enough?

#16 - 2018-03-01 13:32 - EDiGiacinto

Because forcing to start all cluster jobs only in parallel *as default* asks for starvation and deadlocks - if we want to hit that road, just say so and this change will be tight to PARALLEL_WITH, without other variables.

But imo i see other problems coming later and won't be able to fix them by just adding or removing more filtering - so i would personally go step by step, and eventually this behavior should go to default once we approve and decide it is good enough, not the opposite way around.

on a side note:
Even considering the allocating job as running, since _prefer_parallel is 'cutting' and not prioritizing as the name may suggest, will make the all scheduled jobs more subject to starvation until we change paradigm (again) and stop relying on database queries for scheduling (or if we want to scramble the queries, once more, but you are my guest then), so i see no 'real' solution given the current limitations, but just stacking changes on top.

#17 - 2018-03-02 13:21 - EDiGiacinto

This is the proposed PR: https://github.com/os-autoinst/openQA/pull/1592
will remove the option by coolo's request

#18 - 2018-03-07 08:01 - szarate

- Target version changed from Ready to Current Sprint

#19 - 2018-03-07 09:00 - coolo

- Status changed from In Progress to Resolved

merged

#20 - 2018-03-09 10:01 - coolo

- Target version changed from Current Sprint to Done

#21 - 2018-08-30 07:32 - EDiGiacinto

- Related to action #40415: Concurrent jobs with dependencies don't work if they are on different machines. added

Files

slenkins_jobs_deadlock.png	73.3 KB	2018-02-13	thehejik