**Title**: [YARN-2022] Support in Preemption policy to save AM Containers

**Authors: Sunil Govind,**

**Last modified**: May 30 2014

* 1. **Problem statement**

As per the existing mechanism, ProportionalCapacityPreemption policy will preempt the containers as per below criteria:

1. Select Applications based on the order of submission time
2. From each applications, select container with below order:

Map --> Reducer --> Failed Map --> AM

AM Container will always be killed as per above criteria, when all containers of an Application is been iterated for performing preemption.

* 1. **Requirements**

In cases, where preemption has to be performed among multiple applications, it will be better to keep all AM containers as the last resources to preempt. With this approach, it is possible to trade an AM Container with another application’s worker container and thus save the AM container.

* AM Container has to be kept as least priority when preemption happens for multiple applications.
* AM containers alone can occupy the complete Queue resources under this type of behavior. This can be averted by preempting remaining AM containers after equilibrium is achieved with the total running AM containers in the Queue.
  1. **Proposal**

Solution for this proposal has mainly two parts,

* **Lower priority for AM containers in Preemption:** AM Containers can be skipped from the list of live containers from an application while considering for preemption. Later these AM containers can be preempted if required, to achieve required amount of preemption even after preempting all other type of containers.
* **Introduce a checkpoint to find the number of AM’s to be saved from a Queue**: Below configuration parameters will control the number of application running in a Queue.
* yarn.scheduler.capacity.<QueueName>.maximum-applications
* yarn.scheduler.capacity.<QueueName>.maximum-am-resource-percent
* yarn.scheduler.capacity.<QueueName>.user-limit-factor

These parameters have to be honored while deriving on a checkpoint to save AM containers. Once the number of containers limit is reached as per the derived checkpoint, other AM containers can be preempted without any lower priority constraint as discussed in this design earlier.

* 1. **Details**

YARN-2022 introduces a new mechanism to save maximum AM containers during preemption event.

1. Save AM Containers:

Below mentioned steps has to be taken care to achieve this behavior.

* New configuration named yarn.resourcemanager.monitor.capacity.preemption.skip\_am\_container will control this feature. If configured as true, all application master containers will be given least priority in the event of preemption.
* AM Container has to be marked in a better way to identify it from the other types of containers. This has to be done from **RMAppAttemptImpl# AMContainerAllocatedTransition**. During this container allocate event, Master Container can be found and a property can be set in the respective RMContainer to identify as MasterContainer.
* While preempting containers from an application, this marked RMContainer can be skipped for preemption. This container still has to be added to a pending AM Container list if needed to preempt based on forceful needs.

1. Check point to find the optimal number of AM Containers in a Queue:

To avoid race conditions such as AM containers alone running in a Queue, the optimum number of AM container which can run in a Queue has to be found.

* maximum-applications and maximum-am-resource-percent

As per the configurations such as yarn.scheduler.capacity.<QueueName>.maximum- applications and yarn.scheduler.capacity.<QueueName>.maximum-am-resource-percent, a maximum number of application that can run on a Queue can be defined.

Also if the same configuration is given at cluster level, then a total number of applications that can run in cluster also can be derived.

In effect a total number of applications to run in the cluster/queue can be derived as mentioned earlier, and a certain percentage of these Applications only have to be saved. This percentage limit can be configurable and if the limits are met w.r.t above total number of applications, remaining applications can be preempted.

yarn.resourcemanager.monitor.capacity.preemption.am\_container\_limit can be the name of this configuration.

* user-limit-factor and minimum-user-limit-percent

These two factors add a restriction on the application resources which can be allocated in user level. If the user quota is over, then scheduler will skip the application requests for the specific user.

As per preemption, it is not good to retain such application master containers also. These containers can be preempted.

So while trying to save an application master container, user quota for that application also will be checked. If application is over utilizing the resource capacity, the application master container also can be preempted.