YARN-3630: Notes for adaptive heartbeat policy

May 26, 2015, first draft

What can be used as an indicator of scheduler's load

Because all of the scheduler events are dispatched over a blocking queue, if the scheduler is overload, it must result in event congestion in the queue. To some extent, the event congestion can be seen as a necessary and sufficient condition of the schedulers's overload and then can be treated as a nice indicator of the scheduler's load.

How to determine the heartbeat interval for AM

We should first note that an AM heartbeat is not processed by the scheduler (the scheduler here means the scheduler's main thread), so generally it does not affect the scheduler's performance. But in some cases, for example there're many containers to release in a heartbeat, it will call the synchronized method and lead to the scheduler's performance decline, especially in a large and busy cluster where this happen frequently.

For a multi-tenant and highly contested cluster, users may in a disordered state in stimulating heartbeat, which should be normalized.

To address these, we may take several steps,

- set `nextHeartbeatInterval` in the `AllocateResponse` which guide AM stimulate heartbeat after a certain time gap;
- examine whether AM follows the guide, and take some measures if not;
- set new `nextHeartbeatInterval` by certain `heartbeatPolicy`;

A simple suggestion for the `heartbeatPolicy` is

$$\begin{aligned} \texttt{nextHeartbeatInterval} = \texttt{standardNextHeartbeatInterval} \\ + \texttt{nextHeartbeatDelay} * \texttt{rescaleFactor} \end{aligned} \tag{1}$$

Where `nextHeartbeatDelay` is determined by the load of the scheduler, and `rescaleFactor` is determined by the app's properties such as priority, weight and starvation level, etc. The simplest adaptive model is the linear model,

$$\texttt{nextHeartbeatDelay} = \texttt{numOfWaitingSchedulerEvents} * \texttt{RATIO} \tag{2}$$

where `RATIO` is a *parameter* which can determine the elasticity of the adaptive policy. A large value of `RATIO` is hard, and can lighten the load quickly, but all relative apps and nodes would be greatly affected.
According to different app's priorities, scheduler can set a `rescaleFactor` for the `nextHeartbeatDelay`. For example, it can rescale `nextHeartbeatDelay` by a coefficient of 0.5 or 2.0 depends if the app is on starvation and/or other factors. Scheduler should make statistics of all apps, and refers the statistical data, for example, it returns 1.0 if the app is at mean level. Another *parameter* `MAXSCALE` could be introduced to define the range the rescaling: [1/`MAXSCALE`, `MAXSCALE`]. Here `MAXSCALE` behaves like the "Gini coefficient" which represents the fairness or unfairness of the rescaling among apps.

How to determine the heartbeat interval for NM

Similar policy like above could also be applied for NM. But we should note that heartbeat from NM and the relative actions are the main reasons that overload scheduler, so it's better to firstly consider to slow down the NM heartbeats. We can achieve this by setting a relatively higher value of `RATIO` for NM heartbeats. An intelligent tuning of `RATIO` according to the root causes of scheduler's overload is attractive, but we need means to identify the causes of the overload.