Drain decommissioning nodes through available resources

Once a node is in DECOMMISSIONING state, SchedulerNode.getAvailableResource() will return 0 so that no new container will be allocated any more on the node. So the node is basically being drained.

NodeDecomSchedulerEvent and NodeRecomSchedulerEvent are introduced to notify schedulers (CapacityScheduler, FairScheduler, FiFoScheduler etc) about the decommissioning or recommission events so that scheduler can update its cluster resources which could be important for scheduling.

Automatic and asynchronous tracking of decommissioning nodes status

DecommissioningNodeWatcher inside ResourceTrackingService tracks DECOMMISSIONING nodes status automatically and asynchronously after client/admin made the graceful decommission request. ResourceTrackerService handles node registration and frequent heart beat with latest contain status so it is a natural place to hook up automatic decommissioning status tracking logic. ResourceTrackerService now tracks DECOMMISSIONING nodes to decide when, after all running containers on the node have completed, will be transitioned into DECOMMISSIONED state (NodeManager will be told to shutdown).

Under MR application, a node, after completes all its containers, may still serve it map output data during the duration of the application for reducers. A fully graceful mechanism would keep such DECOMMISSIONING nodes until all involved applications complete. It could be however undesirable under long-running applications scenario where a bunch of "idle" nodes would stay around for long period of time. DecommissioningNodesWatcher balance such concern with a timeout policy --- a DECOMMISSIONING node will be DECOMMISSIONED no later than DECOMMISSIONING_TIMEOUT regardless of running containers or applications. If running containers finished earlier, it continues wait up to DECOMMISSIONING_TIMEOUT for the applications to finish. Following are the sub status of a decommissioning node:

- NONE --- Node is not in DECOMMISSIONING state.
- WAIT_CONTAINER --- wait for running containers to complete.
- WAIT_APP --- wait for running application to complete (after all containers complete)
- TIMEOUT --- Timeout waiting for either containers or applications to complete
- READY --- Nothing to wait, ready to be decommissioned

- DECOMMISSIONED --- The node has already been decommissioned

Status of all decommissioning node are logged periodically (every 20 seconds) in human friendly format to resource manager logs.

To be efficient, DecommissioningNodesWatcher skip tracking application containers on a particular node before the node is in DECOMMISSIONING state. It only tracks containers once the node is in DECOMMISSIONING state. DecommissioningNodesWatcher basically is no cost when no node is under DECOMMISSIONING. This sacrifices the possibility that an idle node once host containers of a still running application.

When decommissioning node TIMEOUT, it will be decommissioned regardless. Proper events will be send to ensure the node be deactivated and owning tasks will be rescheduled as necessary.

Per-Node decommission timeout support

Hosts could be subject to shutdown/terminate with short notice. For example, EC2 SPOT instance could be shutdown with 2-minute termination notice. In such cases, we like to graceful decommission such nodes with a small timeout that is before the anticipated termination time. It will avoid Hadoop re-actively deal with LOST instances where stop all containers could subject to long timeout delay.

Further, we support the flexibility to dynamically adjust the timeout of DECOMMISSIONING nodes. **HostsFileReader** now supports optional timeout value after each hostname (or ip). **NodesListManager** detect and update decommission timeout on individual RMNode as necessary. DecommissioningNodesWatcher evaluates timeout based on the dynamic decommission timeout on individual RMNode.

NodesListManager detect and handle include and exclude list changes

1. Recommission DECOMMISSIONED or DECOMMISSIONING nodes that are no longer excluded;
2. Gracefully decommission excluded nodes that are not already in DECOMMISSIONED nor DECOMMISSIONING state;
3. Take no action for excluded nodes that are already in DECOMMISSIONED or DECOMMISSIONING state.

NodesListManager also try to maintain DecomissionedNMsMetrics under variety of include and exclude list changes.