Manager Node

Manager nodes are responsible for the Aspire UI, crawl control, the allocation of work to worker nodes and monitoring nodes which are down and taking corrective actions (such as designating a new main manager node and correcting the status of items that were being processed during failure).

A manager node is responsible for the seeds actively being crawled. Where there is more than one manager, a seed and all the documents it contains, are the sole responsibility of one manager. Seeds that are controlled by the same throttle policy are assigned to same manager node, so these seeds can be throttled together. Allocation of seeds to managers is performed at the time a crawl starts. Where there is more than one manager in a system, responsibility for seeds is distributed across the managers as evenly as possible, within the constraints imposed by the throttle policies.

Configuration

The manager node can be configured setting environment variables or JVM properties or using the settings json that is uploaded to the NoSQL database.

Main Manager

One of the manager nodes is designated as the main manager. This manager is responsible for:

- Noticing that a Manager Node has stopped and beginning the process to reassign the seeds allocated to that manager to others.
- Noticing that a Manager Node has started and beginning the process to assign seeds from other managers to it (without breaking the throttling policies).
- Noticing that a worker node has stopped and “resetting” the status of any jobs that were executing on that worker so they can be picked up and processed by another worker.

When a manager starts, if a main manager is already running, the manager will assume a “non-main” role.

Main Manager Election

A single main manager node must always exist, and so when a manager starts, or when a manager node fails, the managers co-ordinate to ensure that an active main manager exists. On start-up, a manager will check that an active main manager exists. If it does, then the manager simply joins the cluster. If an active main manager does not exist, the oldest manager alive is elected as main manager.

Seed Allocation

When a seed crawl starts, it is allocated to a manager. If the starting seed has a throttle policy, and any in-progress crawl has the same policy, the main manager will allocate the seed to the same manager as the previous seed (to maintain the assertion that seeds with the same throttle policy run on the same manager).

If the seed does not have a throttle policy, or no other seed with the same policy is running, then the main manager will choose a manager, trying to balance the number of seeds across managers.

In the case of a manager failure, the main manager will reallocate the seeds allocated to the failed manager to other managers. In the case of a new manager being added, the main will try to allocate seeds from other managers to maintain a balance. It should be able to reallocate seeds without impacting crawls (i.e. no pause required) by “un-allocating” the seed from manager one, releasing the seed on manager one (to remove the seed from any unsent batches) and then allocating the seed to manager two.

Seed Manual Rebalance

A user with administrator role is allowed to trigger a manual seed rebalance by calling the re-balance endpoint.

Node Failover

Node failover is split into three areas – identification of a failed main manager, identification of failed managers and identification of failed workers. Monitoring is done by examining the node heartbeat entry in the settings NoSQL database and “detecting” a failure when that heartbeat is out of date by more than a given period. Any clean-up work (setting in progress items to be available for instance) will be executed in the main manager.

Once a node is marked as "failed", it shouldn't reappear. All node types will do the following:

- Consider consistent failures to send a heartbeat to the NoSQL database as a fatal error and shut itself down.

- Monitor it's own status in the settings NoSQL database and shut itself down if it sees itself marked as "failed".
- A node is given a unique id at start up, so that when a node is restarted, its recognized as a new node.

Failed Main Manager Identification

All non-main manager nodes monitor the main manager for a failure and try to become the main manager if that is the case. Once the new main manager has been “elected”, its operation will change to reflect the new role.

Failed Manager Identification

Only the main manager monitors for failed manager nodes. If a manager fails, the main manager does the following:

- Allocate all the seeds from the failed manager to other managers.
- Mark the failed manager as such in the settings NoSQL database.

Failed Worker Identification

Only the main manager node monitors for failed worker nodes. If a worker node is detected as failed, the main manager does the following:

- Mark all items noted as in progress by that worker as available, so that the items can be added to new batches.
- Signal all managers to “reset” the status of any in memory batches that were sent to that worker but had not been acknowledged, so that the batch can be fetched by another worker.
- Mark the failed worker as such in the settings NoSQL database.