PanDA and panda-client News

Tadashi Maeno (BNL)

Outline

- Rebrokerage
- Express job
- New TAG analysis scheme on Panda
- Beyond-pledge resource management
- Output file merging

Rebrokerage (1/3)

- Rebrokerage
  - Re-assigns long waiting jobs to another site on the server side
    - When a site goes down or becomes too busy
    - In the context of PD2P
- Two cases
  - All subjobs in a job (the same jobDefinitionID) are waiting
    1. All subjobs are reassigned to another site with the same jobDefinitionID and jobsetID
    2. Dataset locations are changed accordingly

Rebrokerage (2/3)

- Some subjobs including buildJob have already run
  1. Creates new buildJob
  2. Creates new runAthena subjobs which correspond to waiting subjobs
  3. Sets new jobDefinitionID and the same jobsetID
  4. Creates new output dataset(s), changes ownership, and adds them to the output dataset container
  5. Assigns new subjobs to a site
  6. Kills waiting subjobs

Kill and retry

Rebrokerage (3/3)

- Brokerage Algorithm
  - Almost the same as normal analysis brokerage
    - Dataset availability
    - Site occupancy
    - Site status
    - Free disk space size
    - schedconfig.maxinputsize
  - Disabled when the user sets --site or --disableRebrokerage explicitly
- Automatic Rebrokerage
  - Triggered when the job is older than 3 days and some subjobs are waiting for 24 hours
  - Disabled for ganga jobs since it creates inconsistency in ganga's job repository due to new subjobs. pbook is fine since it synchronizes the local job repository with the pandaDB
- Once jobs are reassigned by rebrokerage, 'rebro' is added to job.specialHandling
  - e.g., reassigned twice → rebro,rebro
  - Montioring is required

Express job

- Gets very high priorities to have quick turn-around
- Quick tests before bulk submission
- Very tight limitation to prevent abuse
  - Each user can use 180min of execution time per 6 hours and can have 3 active subjobs in all sites
- The --express option of pathena/prun
- The flag is suppressed on the server side when the user doesn't have enough quota
  - E.g., if the user submits N (>3) subjobs with --express, the flag is suppressed except the first 3 subjobs
- Needs Monitoring

New TAG analysis scheme (1/2)

- Marcin Nowak has developed the TAG GUID count service
  - Input
    - Dataset name
    - EventSelector.Query (selection criteria)
    - EventSelector.StreamRef (stream name, ESD,AOD,RAW,…)
  - Output
    - TAG and parent AOD/ESD/RAW GUIDs for selected events
- Workflow
  1. Parent GUIDs are retrieved from the TAG GUID count service using TAG DS name + selection criteria
  2. GUIDs are converted to dataset and file names using DQ2
  3. job is split using parent file boundaries
     e.g. if TAGs point to 500 ESDs and --nFilesPerJob=50, there will be 10 (500/50) subjobs
  4. subjobs are sent to sites where both TAG and parent datasets are available
  5. Each subjob reads selected events in parent files through TAG files

New TAG analysis scheme (2/2)

- e.g,

Query='EventNumber=123 && RunNumber=456'
AND StreamRef='StreamESD'

→ One TAG GUID and one ESD GUID from the GUID svc

→ One TAG dataset/file and one ESD dataset/file from DQ2

→ The job is configured to read only one ESD file through the TAG file

Previously the job was configured to read all ESD files since it was unknown which ESD contained selected evetns

→ The job is sent to a site where both the TAG and ESD datasets are available

Previously the job was sometimes sent to a site where only TAG was available since relation between TAG and ESD datasets was unknown

→ Each subjob reads TAG/ESD files from SE at directIn sites, or the pilot copies TAG/ESD files to WN for copyToScratch sites

No need to generate an (potentially large) intermediate TAG file using ELSSI

Works for ARC as well

Beyond-Pledge Resources Management

- Regional CPU/Storage resources provided by using budgets beyond ATLAS MoU share
  - Each site/cloud/country can decide how those resources are used
- Existing policy
  - Runs jobs only for users who belong to a particular working or country group
    - Dedicated resources
    - Idle when the group doesn't use
- New additional policy
  - Runs jobs for users who belong to a particular country group if such jobs are waiting in the queue. Otherwise, runs any user's jobs
    - Resources are always being used effectively
    - Only one pilot stream
    - Ratio between Pledge/Beyond-Pledge can be defined in schedconfig
- Internal implementation will be explained on Tur.

Output Files Merging (1/2)

- Analysis jobs tend to produce many small files
  - FTS cannot transfer small files very efficiently
- It is difficult to make a general tool to merge output files since the required merging tool depends on output file format
  - Flat NTuple → hadd
  - Structured NTuple → hadd + dictionaries
  - DQ monitoring → DQHistogramMerge.py
  - POOL → Athena
  - XML → ?
- Users can submit appropriate merging jobs by using prun since they know how files should be merged → not sure if it is worthwhile to elaborate a general tool
- According to DDM team, small files are problematic for data transfer but not for SE since analysis output files are put on DISK

Output Files Merging (2/2)

- It is proposed to make a temporary zip file for transfer
  1. DQ2 or DaTRI makes two Panda jobs and one DQ2 subscription
  2. The first job runs at src site to create a zip file from analysis output files and registeres the zip file to src DQ2
  3. The subscription transfers the zip file to dest site and sends a callback to Panda to activate the second job
  4. The second job runs at dest site to expand the zip and registers expanded files to dest DQ2
- The zip file is hidden from users
  - Don't need to change their analysis codes
- Confirm the fundamental scheme works fine but more developments are required
  - Monitoring
    - Currently one has to look at DQ2 dash + Panda mon → common frontend
  - Error handling
  - Integration with DQ2
  - Priorities and quotas
  - etc
- Looking for someone to take over this issue
  - A separate service component