Checkpointing in DataStage

Kevin Wei
kevinw@ibm.com

Scott Brokaw
slbrokaw@us.ibm.com

Checkpoint Feature

Goals of Checkpointing

- Automatically restart jobs when they fail
- Maintain strong performance while adding resiliency
- 2 types of checkpoints

Automatically added

- Automatically inserted into your flow
- You cannot set where checkpoints are performed
    - Implemented using environment variables
    - Checkpoints use compression by default

Intermediate Checkpoint

Inserted before sort operations

- Natural blocking point

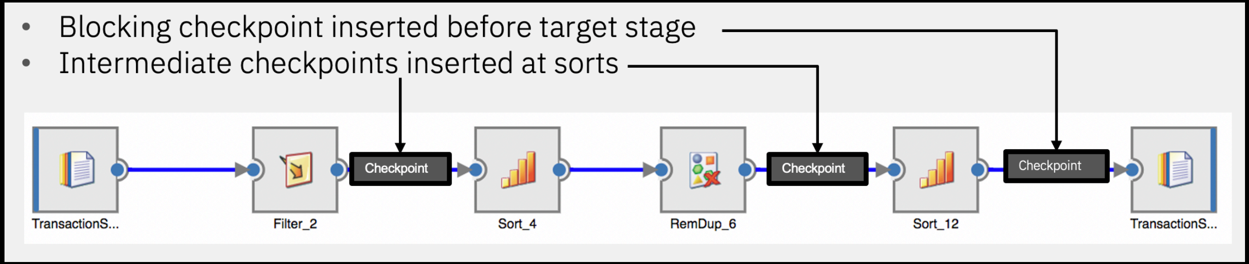
Only the last (most recent) checkpoint is persisted

- Previous checkpoints are purged from disk

Illustration: Intermediate Checkpoint

Blocking checkpoint inserted before target stage

- Sorts are naturally blocking so intermediate checkpoints are inserted before sorts
- Once all partitions receive EOD, the checkpoint is complete and usable for restart



- Blocking checkpoint inserted before target stage
- Intermediate checkpoints inserted at sorts

Target Blocking Checkpoint

Inserted before a target stage

- Prevent partial insertion of data (may leave the target in an unknown or corrupt state if the flow restarts)

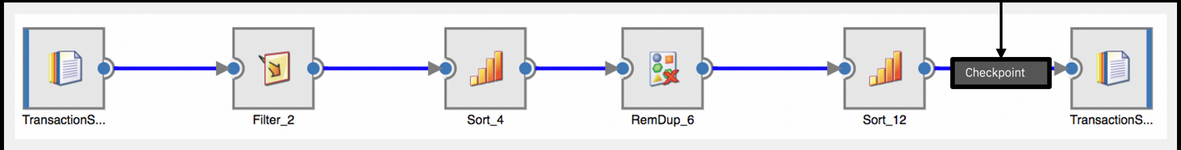
Buffers all data until ready to insert to target

- Holds all data until ready to go to target
- Will not auto restart if there is a failure as the job is writing to the target
- Once target blocking checkpoint is reached, any failure requires manual restart
  - Can restart from a persisted checkpoint

Illustration: Blocking Checkpoint

Blocking checkpoint inserted before target stage

- Buffers all the data until reach end of data
- If job fails before end of data, job will restart
- When checkpoint receives EOD, it writes to target
  - Once a target has processed a record, it will not be restarted



Restart

Checkpoint Restart

- Auto-triggered when a job fails
- Can be used with or without checkpoints
  - Without checkpoints, will simply restart from the beginning
- Environment variables to designate number of restart attempts

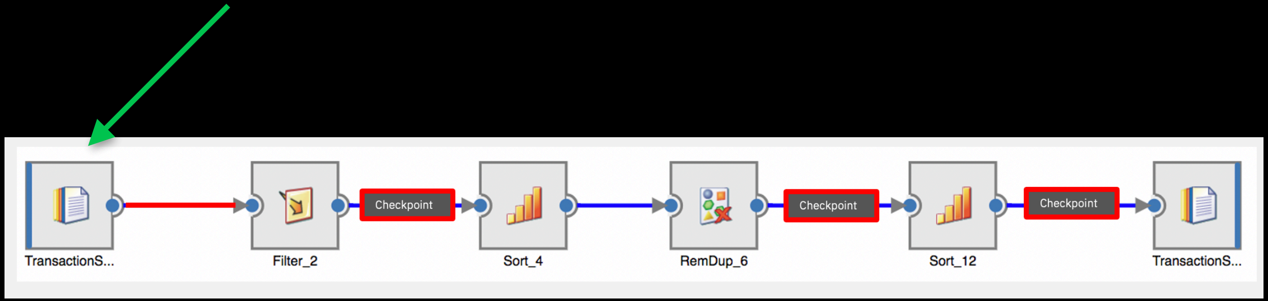
Restart Delay

- Set to a valid integer seconds to delay restart
  - Helpful to resolve network connection or timeout issues

Scenario: Restart from Source

Automatically restart from source stage

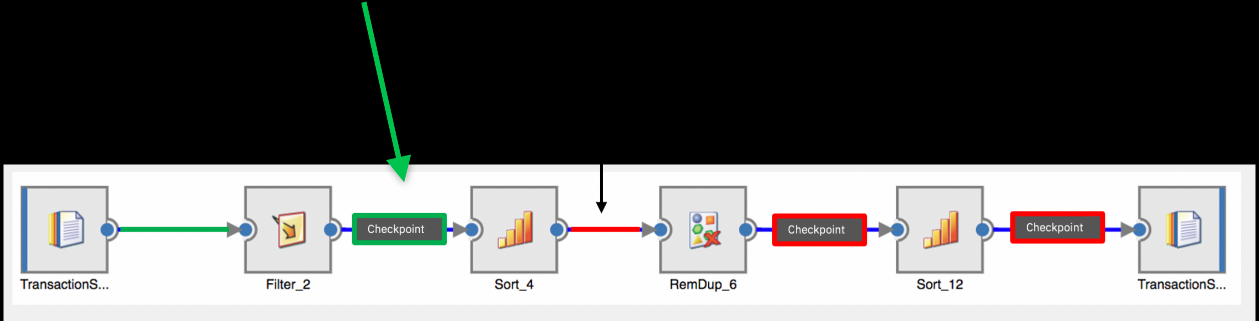
- Failure occurs in the first link
- None of the checkpoints are complete
  - Checkpoints are only complete when they receive EOD on all partitions



Scenario: Restart from a Complete Checkpoint

Automatically restart from a complete checkpoint

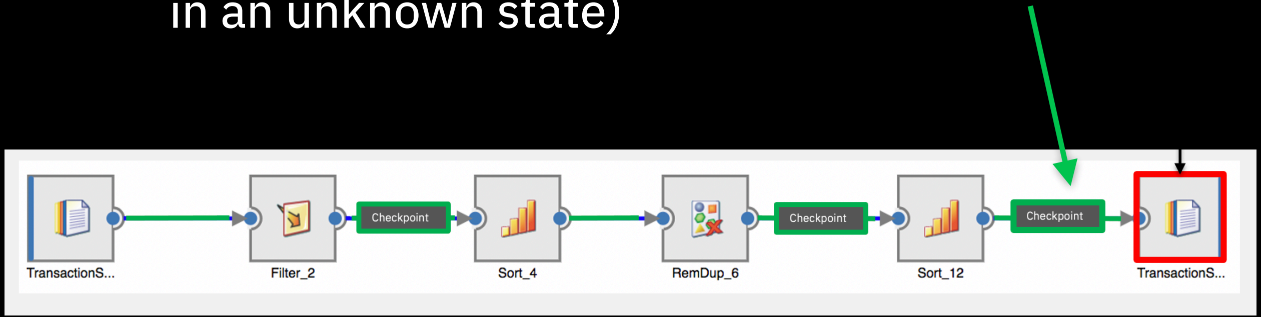
- First checkpoint is complete
  - Second and third checkpoints are not complete
- Failure occurs while this link in red is processing data
  - Auto restart from first checkpoint



Scenario: Manual Restart from a Persistent Checkpoint

Manually restart from a complete checkpoint

- All checkpoints are complete
  - Target stage has begun processing records
- Failure occurs while the target stage is processing data
  - Job will **not** automatically restart (target stage is in an unknown state)

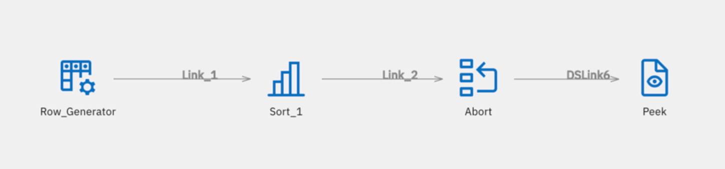


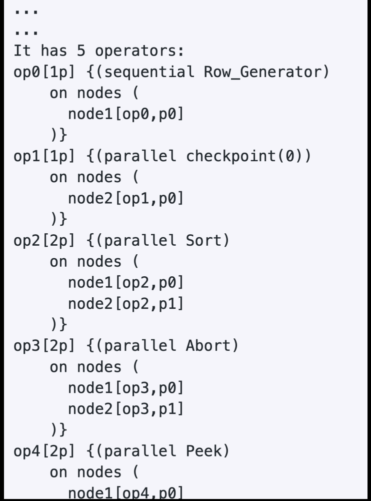
Reading The Score

Converts GUI DataStage Flow into an Execution Pattern

- Checkpoints are not inserted in the GUI
- Score report is useful to understand checkpointing

Checkpoint inserted before sort





```
...
...
It has 5 operators:
op0[1p] {(sequential Row_Generator)
    on nodes (
      node1[op0,p0]
    )}
op1[1p] {(parallel checkpoint(0))
    on nodes (
      node2[op1,p0]
    )}
op2[2p] {(parallel Sort)
    on nodes (
      node1[op2,p0]
      node2[op2,p1]
    )}
op3[2p] {(parallel Abort)
    on nodes (
      node1[op3,p0]
      node2[op3,p1]
    )}
op4[2p] {(parallel Peek)
    on nodes (
      node1[op4,p0]
```

Checkpoint Environment Variables

kevinw@ibm.com
slbrokaw@us.ibm.com

- **APT_CHECKPOINT_RESTART=n**
  - Defining this to n will allow failed jobs to automatically restart n times if a failure occurs. Recommend only setting this in production.
- **APT_CHECKPOINT_RESTART_DELAY**
  - If defined as a positive integer defines the number of seconds to delay before restarting a job
  - Use as needed on high volume, long running jobs:
- **APT_CHECKPOINT_ENABLED**
  - "targetonly" inserts only target checkpoints before targets (buffering)
  - "sortonly" inserts only intermediate checkpoints before sorts
  - Any other value inserts both types of checkpoints
- **APT_CHECKPOINT_ESTIMATE**
  - Doesn't create checkpoints but calculates the space each checkpoint will require to store on disk (eg. at every Sort Operation since these are natural blocking)
- **APT_CHECKPOINT_PERSISTENT=<path>/myjob.checkpoint.fs**
  - Retains complete checkpoint data files across job runs
  - Completed persistent checkpoints used on manual restart

Questions

kevinw@ibm.com
slbrokaw@us.ibm.com

IBM