Developing alert calculations with snewpdag

J Tseng

SNEWS CM, 12 May 2021

UNIVERSITY OF OXFORD

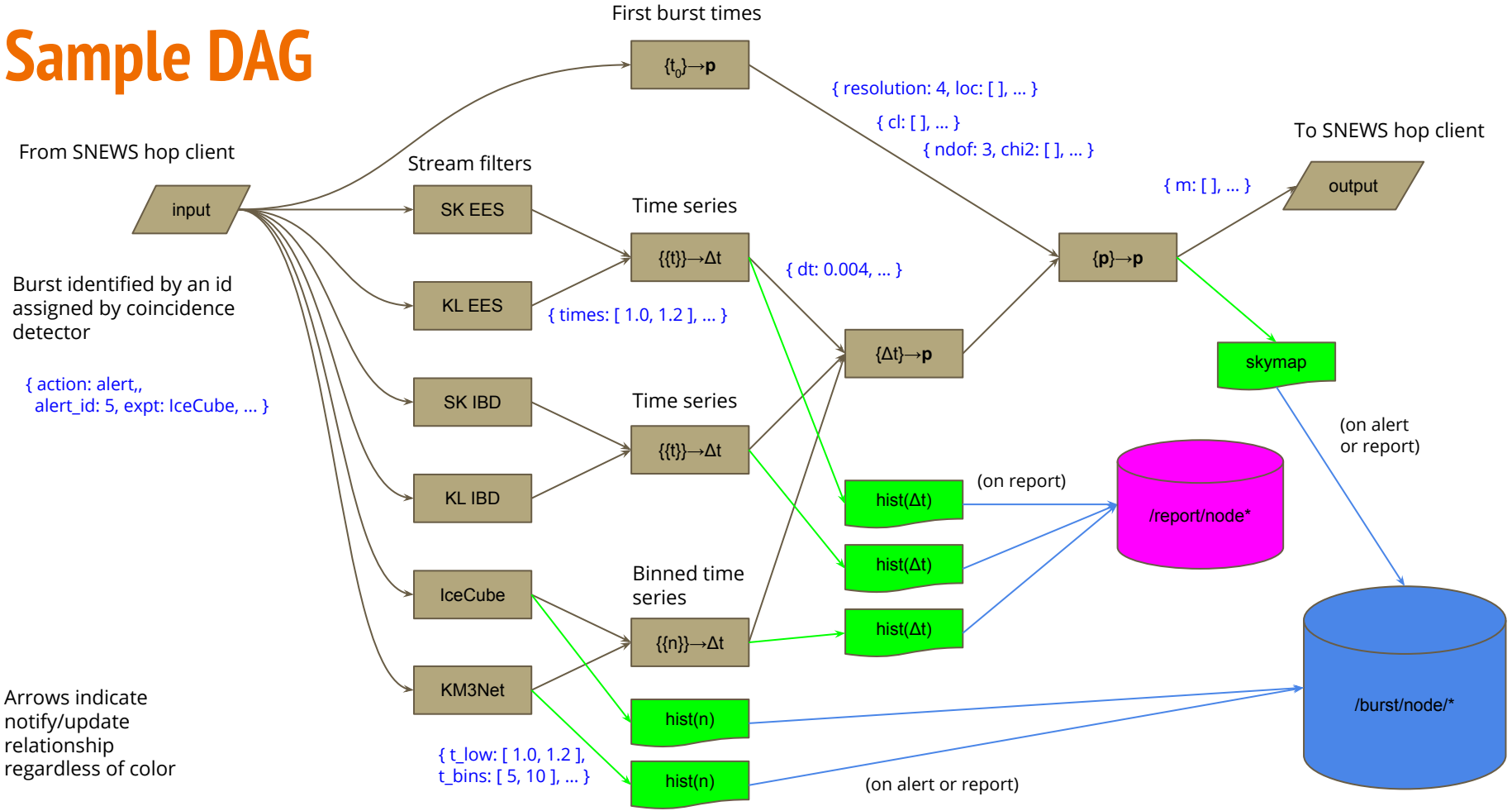
Outline

- How snewpdag is supposed to work in hopskotch
- Using snewpdag in MC trials
- Development, plugin projects, hackathon

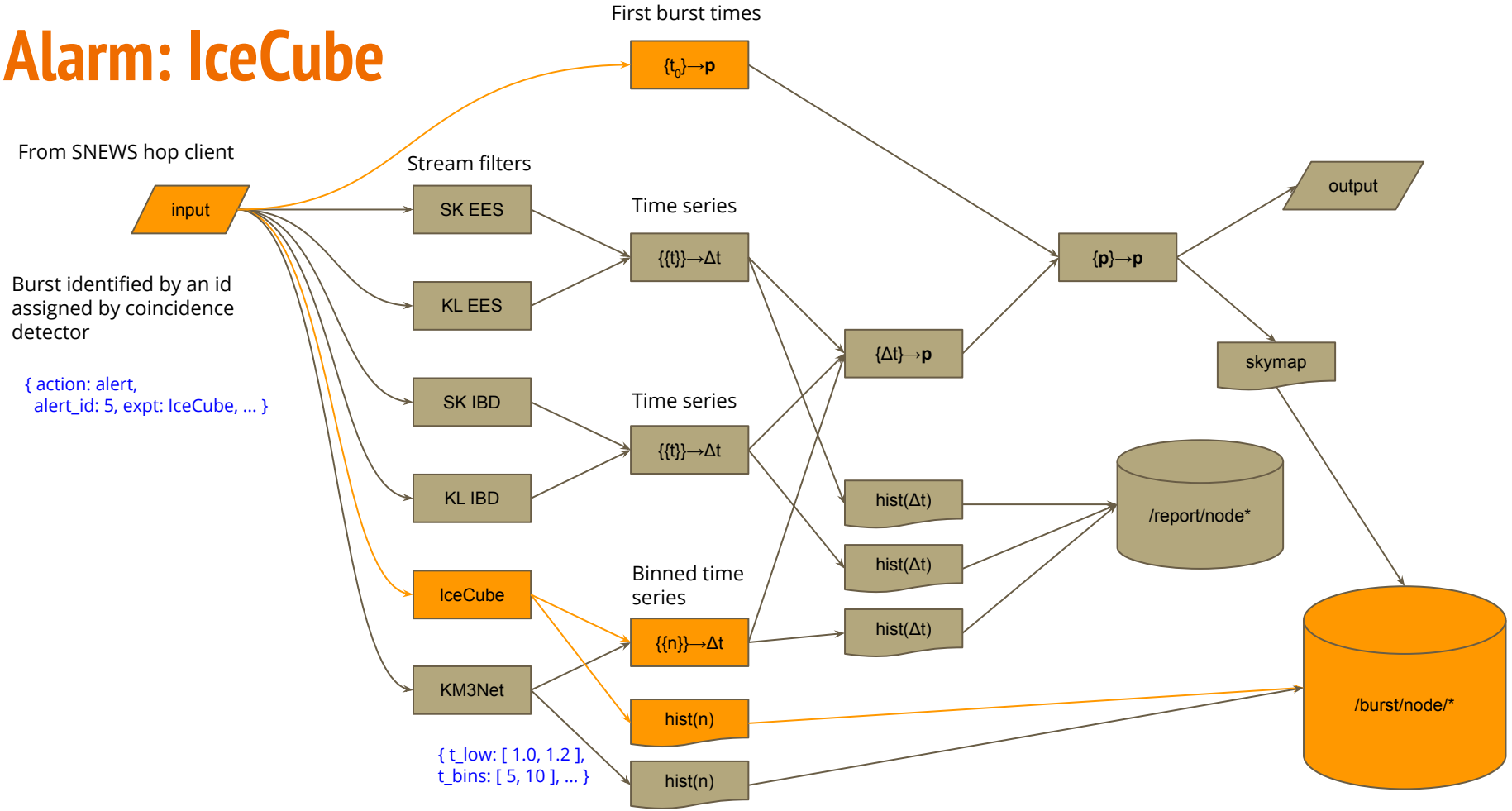
Hopskotch input/output: working assumptions

- Alert formed from a coincidence determined by hopskotch logic
- Hopskotch assigns an alert identifier
- Hopskotch also starts new snewpdag instance with that id
  - Snewpdag instance may persist for up to 48 hours
  - Data is sent to snewpdag in the form a python dictionary
- Subsequent alarms which satisfy coincidence will be sent through snewpdag instance with the assigned id
  - Alarms for a given coincidence can dribble in at any time (within 48 hours)
  - Use DAG to update only those parts affected by new alarm input
  - If snewpdag terminated early, state can be recreated by running latest versions of alarms through a new instance
- Results returned to hopskotch via callback with python dictionary

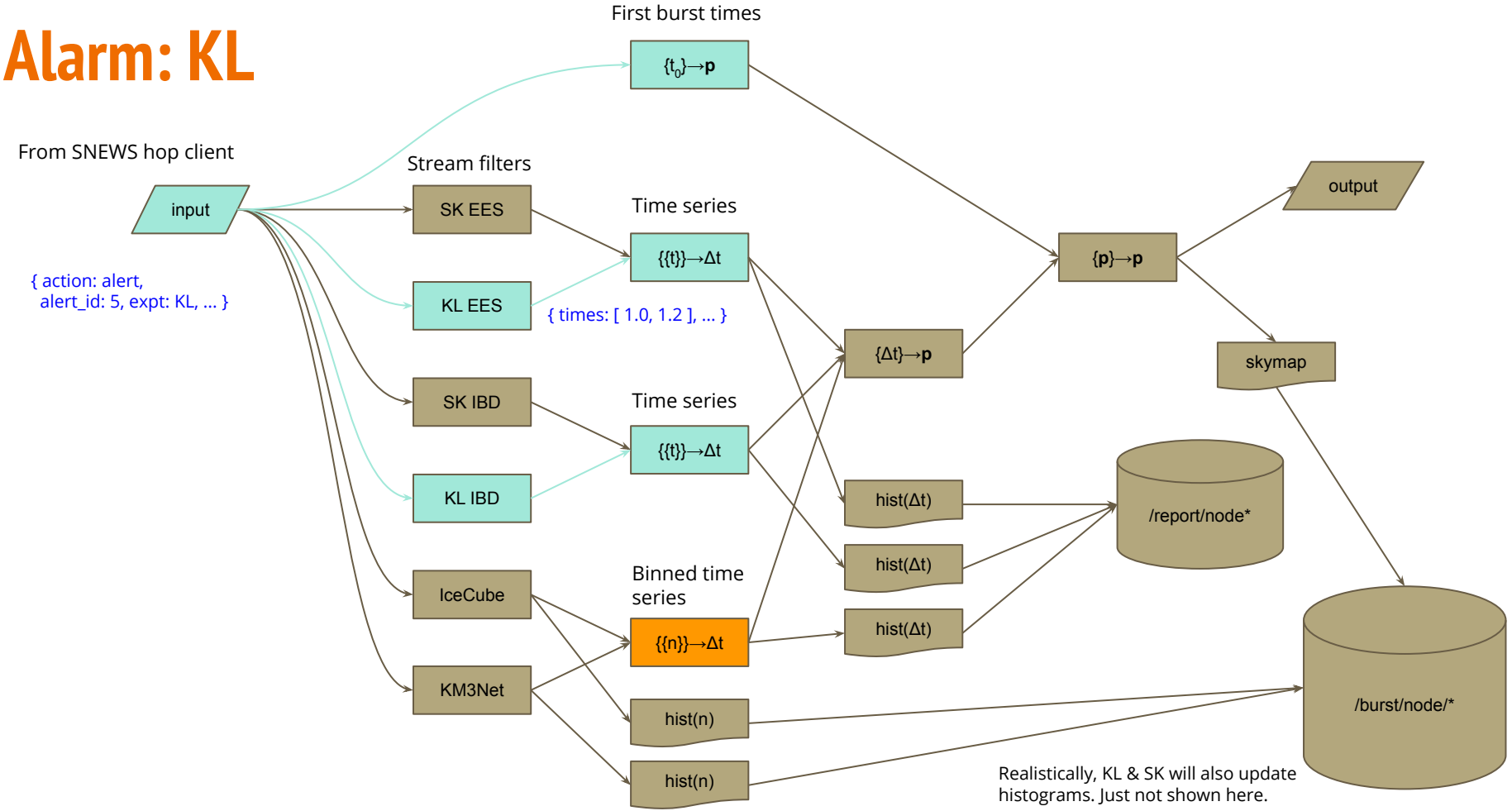
Sample DAG


First burst times
{t0}→p
{ resolution: 4, loc: [ ], ... }
{ cl: [ ], ... }
{ ndof: 3, chi2: [ ], ... }
From SNEWS hop client
input
Stream filters
SK EES
KL EES
SK IBD
KL IBD
IceCube
KM3Net
Time series
{{t}}→Δt
{ times: [1.0, 1.2], ... }
Time series
{{t}}→Δt
Binned time series
{{n}}→Δt
{ t_low: [1.0, 1.2],
t_bins: [5, 10], ... }
{ dt: 0.004, ... }
{Δt}→p
hist(Δt)
hist(Δt)
hist(Δt)
hist(n)
hist(n)
{p}→p
{ m: [ ], ... }
To SNEWS hop client
output
skymap
(on alert
or report)
(on report)
/report/node*
/burst/node/*
(on alert or report)
Burst identified by an id
assigned by coincidence
detector
{ action: alert,,
alert_id: 5, expt: IceCube, ... }
Arrows indicate
notify/update
relationship
regardless of color

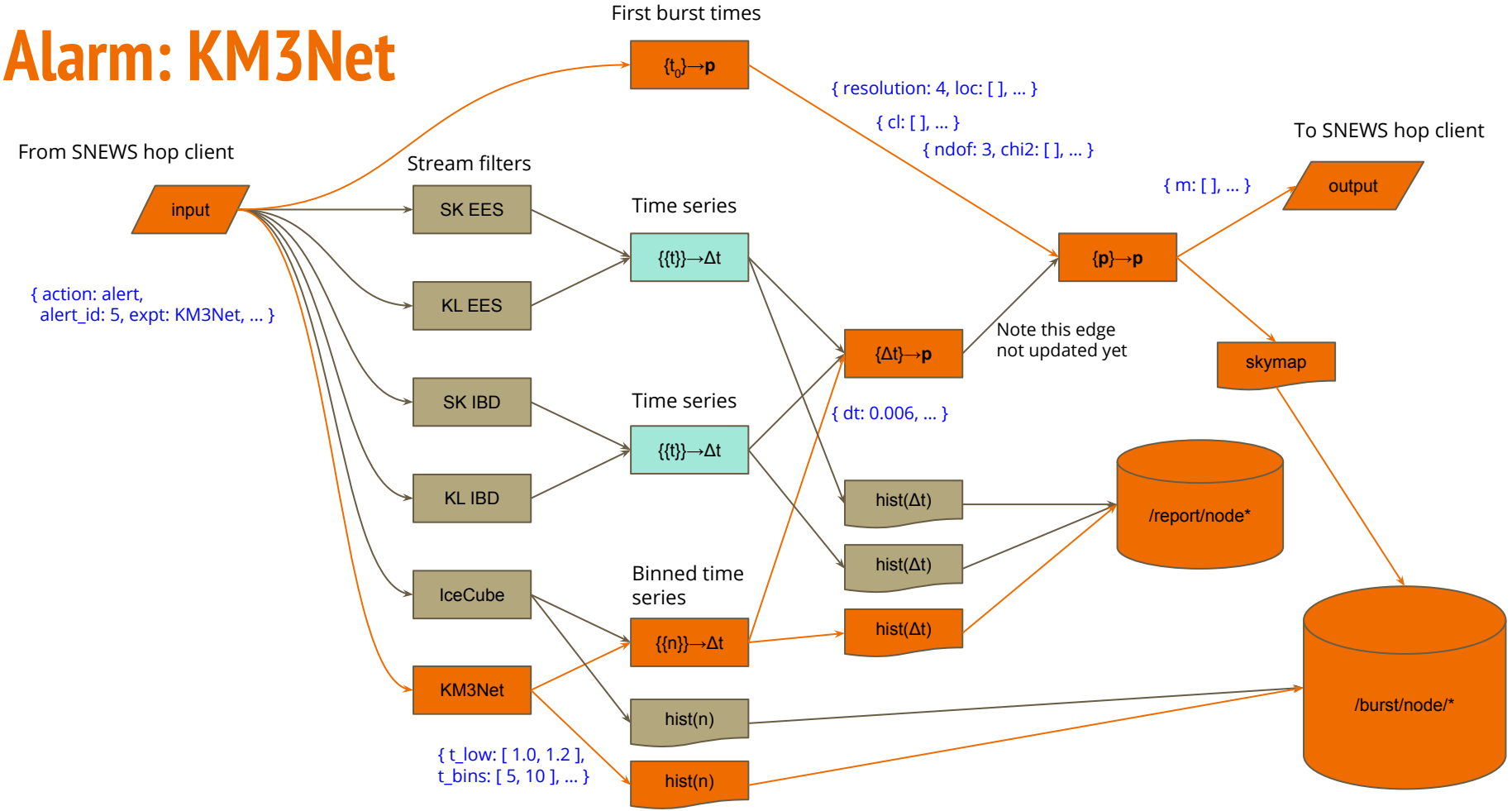
Alarm: IceCube



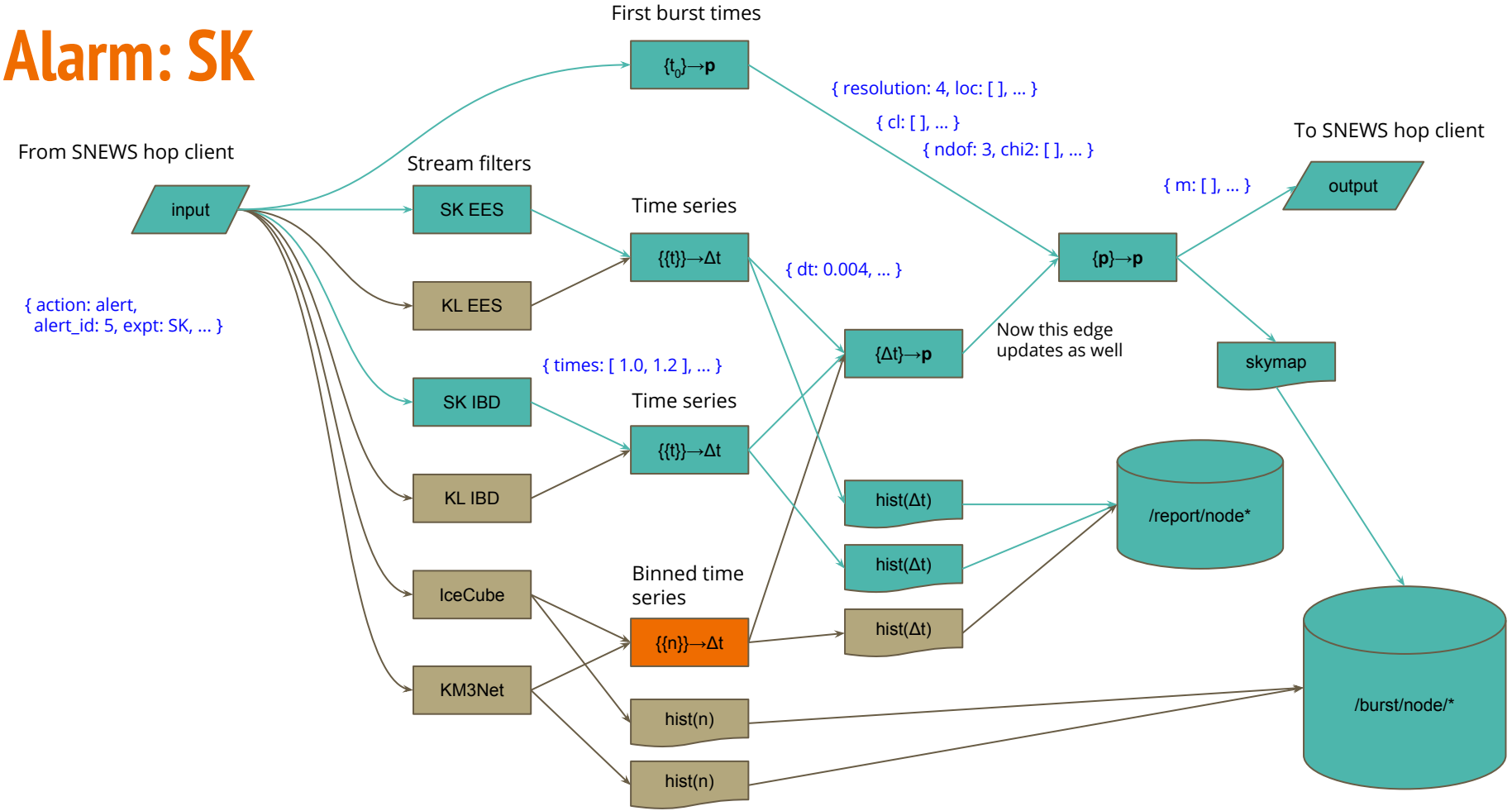
Alarm: KL


First burst times
{t0}→p
From SNEWS hop client
Stream filters
input
SK EES
Time series
{{t}}→Δt
output
{p}→p
KL EES
{ times: [1.0, 1.2], ... }
{ action: alert,
alert_id: 5, expt: KL, ... }
{Δt}→p
skymap
SK IBD
Time series
{{t}}→Δt
hist(Δt)
/report/node*
KL IBD
hist(Δt)
IceCube
Binned time series
hist(Δt)
{{n}}→Δt
KM3Net
hist(n)
/burst/node/*
hist(n)
Realistically, KL & SK will also update histograms. Just not shown here.

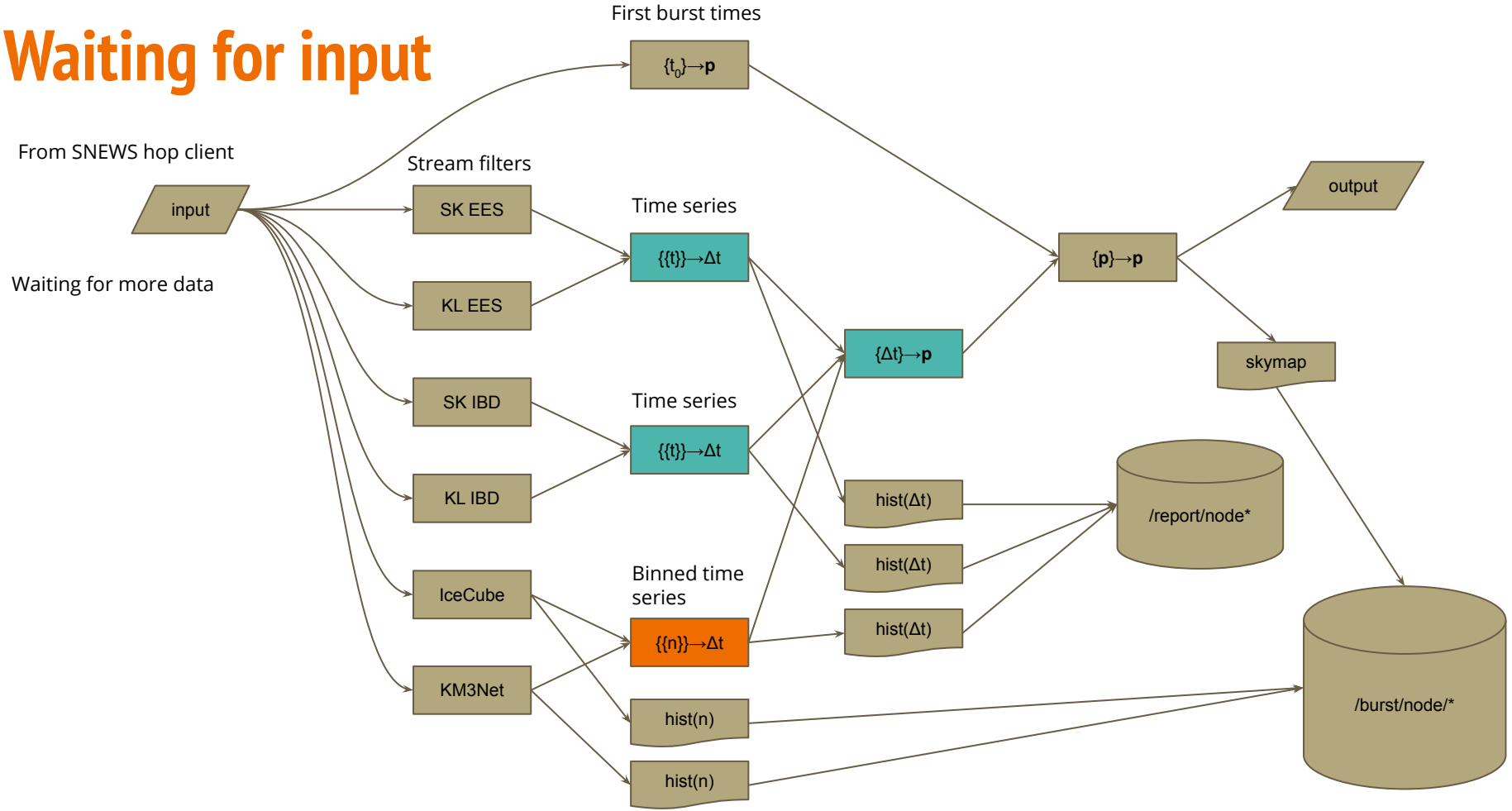
Alarm: KM3Net



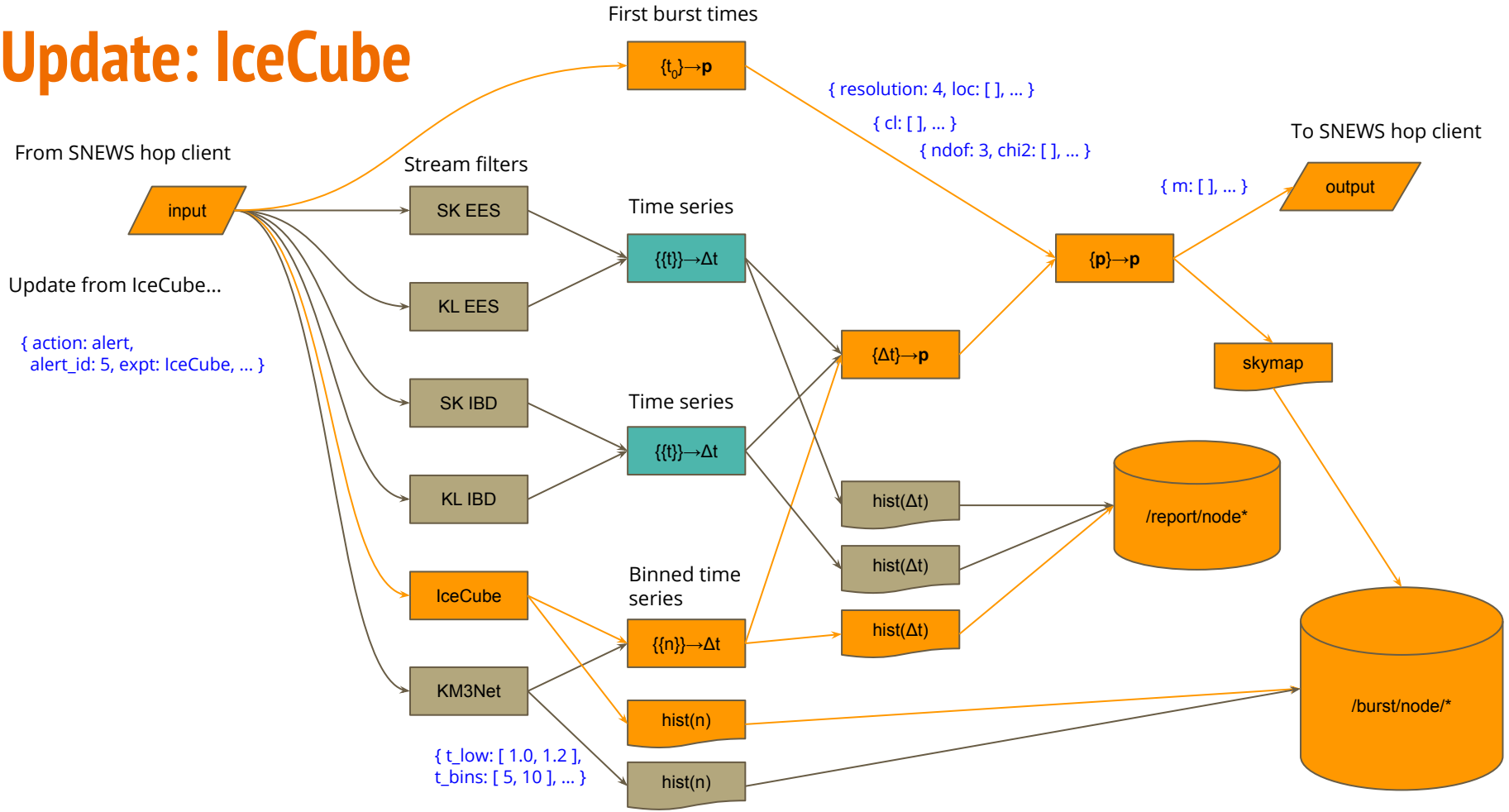
Alarm: SK


First burst times
{t0}→p
{ resolution: 4, loc: [ ], ... }
{ cl: [ ], ... }
{ ndof: 3, chi2: [ ], ... }
From SNEWS hop client
Stream filters
To SNEWS hop client
input
SK EES
KL EES
SK IBD
KL IBD
IceCube
KM3Net
Time series
{{t}}→Δt
{ dt: 0.004, ... }
{ times: [1.0, 1.2], ... }
Time series
{{t}}→Δt
{ action: alert,
alert_id: 5, expt: SK, ... }
{p}→p
{ m: [ ], ... }
output
{Δt}→p
Now this edge
updates as well
skymap
hist(Δt)
hist(Δt)
hist(Δt)
/report/node*
Binned time
series
{{n}}→Δt
hist(n)
hist(n)
/burst/node/*

Waiting for input

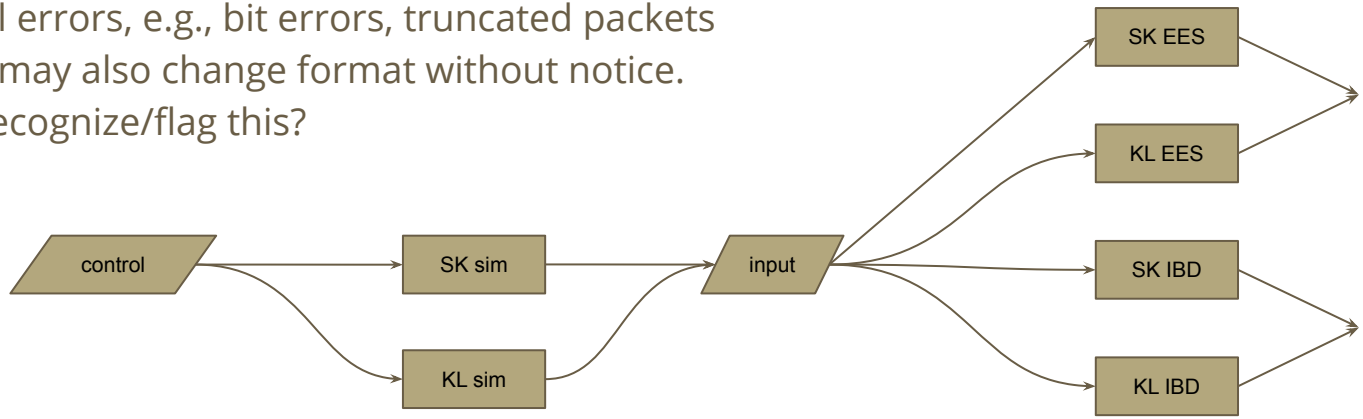


Update: IceCube



MC trials

- Aim to use same snewpdag logic for MC trials
  - Test algorithms with pseudo-experiments
  - Measure resolutions, biases, timing
- Method: same core DAG, replace input and output modules
- Also need to test snewpdag for robustness
  - Unintentional errors, e.g., bit errors, truncated packets
  - Experiments may also change format without notice.
    How do we recognize/flag this?



Configuration for pseudo-experiments (stdin)

- snewpdag takes one “alert” per line

```
python -m snewpdag --jsonlines config.py < input.py
```

- Input for MC trials:
  - Each pseudo-experiment consists of several “alert” actions
  - Each action is one python dictionary terminated by EOL
    - No EOL (line breaks) within the alert
  - Last action should be a “report” action to prompt accumulator plugins to write output
- Renderer plugins take a filename in configuration
  - The filename should be a pattern which can take name and id as arguments
  - Renderers can generate output on alert or report actions
  - Can also generate summary statistics

Development

- Github: https://github.com/SNEWS2/snewpdag
- See Development wiki page: https://github.com/SNEWS2/snewpdag/wiki/Development
  - Easiest to use pyenv (virtual environment) with pip, numpy, healpy
- All modules (plugins) subclass snewpdag.dag.Node
  - Override **__init__** and action methods (if needed)
  - Simplified to make development easier, less fiddly
- Plugin wiki pages:
  - Documentation for plugins already implemented
  - List of plugin projects
  - We suggest developing a utility or renderer first, then your favorite algorithm
- Overall making use/development easier

Action handlers

- **alert**: update to data
  - Only notify downstream if revised calculation is complete (i.e., to update downstream calculations)
  - If calculation is not yet complete, don't forward anything
- **revoke**: revoke some data
  - If revocation changes calculation, forward revoke
- **report**: trigger aggregate calculations
  - Should be forwarded
- **reset**: clear state between alerts (like revoke on all sources)
  - Used in MC trials
  - In data, we expect to have a different DAG instance per alert
  - Should be forwarded

To do: a sample of plugin projects from the wiki

Usually easier to start on this side

- Utilities
  - Input validation
  - Histogram rebinning
  - Action filter
- Renderers
  - Nicer looking plots with labels!
  - Scatter plots, 2D histograms, etc
  - Skymap
  - HTML output
  - SNEWS output

Then move on to this side

- Generators for MC trials
  - Analytical signals and backgrounds
  - Flat background
  - Random numbers (delays, event yields)
  - Delays based on detector locations
- Algorithms!
  - Pointing, obviously
  - Distance estimate
  - Feature detection: cut-offs, second peaks, long accretion phases

Hackathon

- Tentatively (?) scheduled to give a snewpdag tutorial in week 3 (8 June)
- Depending on participants and interests, could aim for a basic DAG by end of week 4 using just burst times of connected experiments
    - Soon afterwards: MC trials for physics prospects paper (SNEWS burst calculations as we'd like it to be)
- We are cleaning up documentation, rationalizing names, and overall making snewpdag more usable in preparation
- Beyond hackathon: evaluating algorithm performance and biases over the summer → input to paper!