

# Crossmatching Developments

DS3 Report

VOtech Stage 6 Planning Meeting,  
8 October,  
Royal Observatory Edinburgh

Mark Taylor, Bristol University

`m.b.taylor@bristol.ac.uk`

# Outline

- Review matching algorithms
- Summarise suitability for different regimes
- Discuss recent progress
- Suggest future work

# Matching Algorithms

- Consider *on-demand* crossmatching
  - *Astronomer* asks: which objects in catalogue A match which in catalogue B?
- Different approaches required for different regimes of table size
  - Naïve
  - Tile-based
  - Multiple cone search
  - Pre-indexed

## Small matches (any algorithm)

- Naïve crossmatch algorithm:

```
for i = 1, N {  
  for j = 1, M {  
    if (distance(i,j) < rmax) then match()  
  }  
}
```

- Scales as  $N \times M$
- Hopeless for large tables, but for small ones it doesn't matter

# Tile-based methods

- Algorithm is roughly:
  - Divide parameter space (e.g. sky) into tiles
  - Perform naïve ( $N^2$ ) crossmatch on contents of each tile
  - . . . and worry about objects near tile boundaries
- Performance is around  $N \log(N)$  (but pathological cases exist)
- Generalises easily to any multi-D parameter space (not just sky)
- Implemented in STIL
  - Use from TOPCAT, STILTS, **STILTS/CEA**
  - Position→tile mappings calculated during match, no pre-processing used/required
  - Memory usage scales as total row count — fails for  $\gg 1$  Mrow
- Similar implementation in Aladin?
- Implementation scalability and efficiency could be improved
  - Recursive tile size refinements
  - Store intermediate data structures on disk
  - Smarter intra-tile tests

# Multiple Cone Search

- Cone search is a specialised instance of a crossmatch
  - 1-row local table vs. many-row remote table, sky distance match criterion
- Generalise: any sky crossmatch can be done by multiple cone searches
- Not very elegant
  - Scales as  $N$  = row count of smaller table
  - . . . but scaling factor may be large
    - ▷ dependent (hopefully sublinearly) on  $M$  = row count of larger table
    - ▷ latency associated with each query
- Ways to improve performance:
  - Decrease time of single query (tile-type indexes, spatial indexes)
  - Address latency of multiple queries (parallelise, iterate on server rather than client)
- Several existing implementations using Cone Search (now *IVOA SCS*) protocol: send many HTTP queries
  - STILTS `multicone`
  - GAVO MCMCS
  - Eduardo Gonzalez-Solares Python scripts

# Pre-indexing methods

- Matching between 2 or N huge catalogues cannot be done to order
  - The match itself will take hours or days
  - The datasets are far too large to move
- Such matches must be done at data centres containing both datasets
- Matching may use plane-sweep or tile-based methods (or others?)
- Match done once and results stored in database in some form
  - as new tables describing inter-table links
  - as indexed tile-index columns in existing tables
  - . . . . ?
- Provide some mechanism for users to access crossmatch results

# Algorithm Summary

(Oversimplified) Summary of 2-table crossmatch by row count:

row count	small (1k)	medium (1M)	large (10M)	huge (1G)
small (1k)	any	tile	multicone	multicone
medium (1M)	tile	tile	tile++	??
large (10M)	multicone	tile++	tile++	pre-index
huge (1G)	multicone	??	pre-index	pre-index

## Algorithms:

any: Doesn't really matter

tile: Tiling

tile++: Enhanced tiling

multicone: Multiple cone search

pre-index: Pre-indexed sky match

# Recent Progress: STILTS/CEA

with Guy Rixon

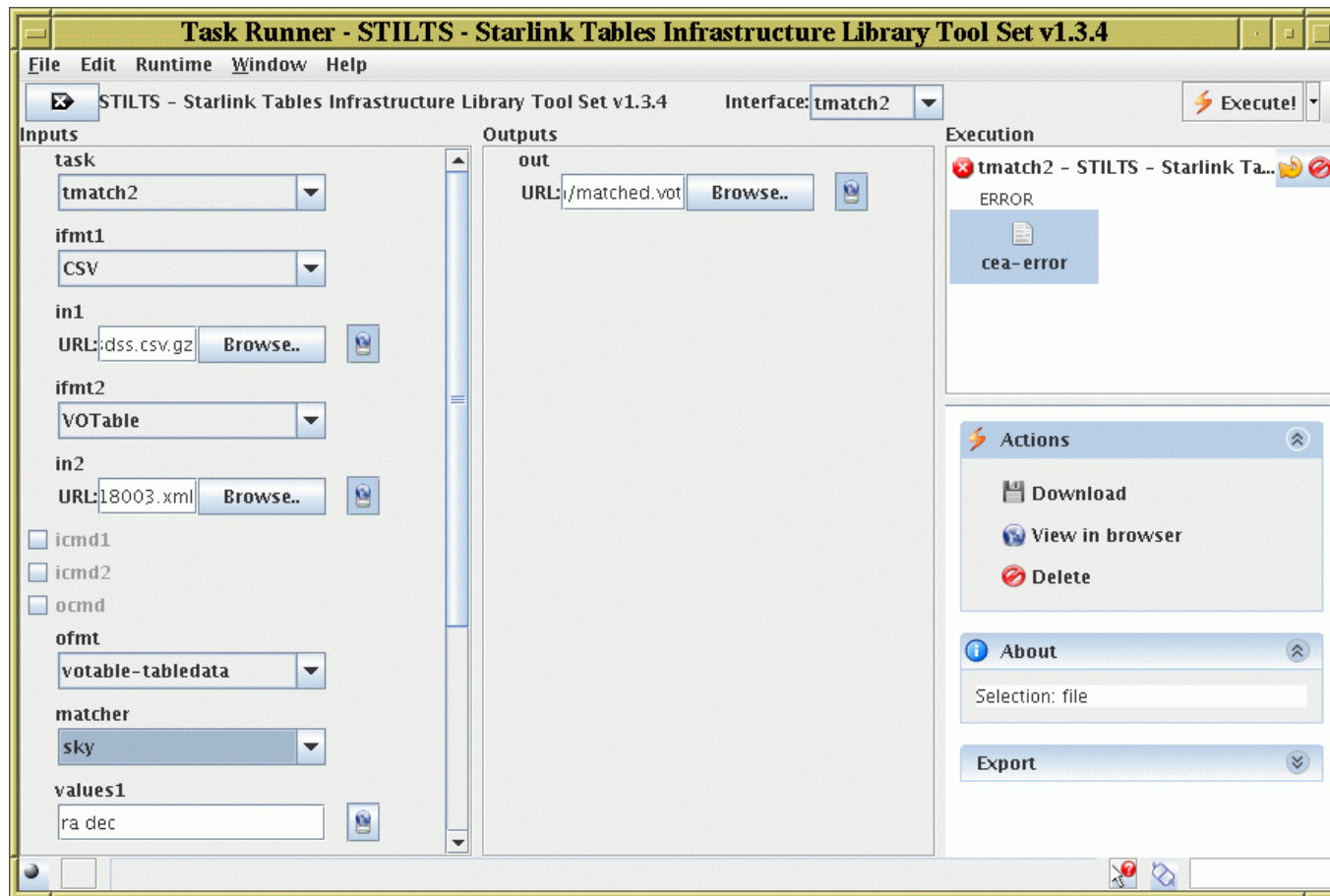
- CEA: framework for running applications on a remote server
- STILTS: suite of table analysis tools, including xmatcher `tmatch2`
- CEA interface for all STILTS tasks:
  - Has been in place for some time
  - Improvements for several reasons:
    - ▷ Updates to STILTS (new tasks etc, in response to user requests)
    - ▷ Fixes to STILTS CEA configuration
    - ▷ Improvements in AG task runner
  - Usability enhancements:
    - ▷ Where possible parameters use selection lists not free text (e.g. format types)
    - ▷ Standard output & error can now be seen by user
  - Issues with rapidly-evolving application
    - ▷ Need to keep `app-description.xml` in sync with deployed application
    - ▷ STILTS generates its own (complex) app-description file automatically
  - Still work in progress

# STILTS/CEA – Example

with Guy Rixon

## STILTS `tmatch2` run via CEA

- Interface details determined by auto-generated `app-description.xml` file



## Recent Progress: STILTS Multicone

- Generalised existing `multicone` command to use pluggable backends
- Anything with cone-search-like semantics can be used for cone step
- VOTables are streamed on input and output
- Current working implementations:
  - Submit cone searches to remote server (STILTS `multicone`, as before)
  - SQL SELECTs on local/remote database using JDBC (STILTS `sqlcone`) — *experimental*
  - *Make requests to database within DSA (see below. . . )*
- Other implementations are possible (useful for experimentation)

# Recent Progress: DSA Multicone

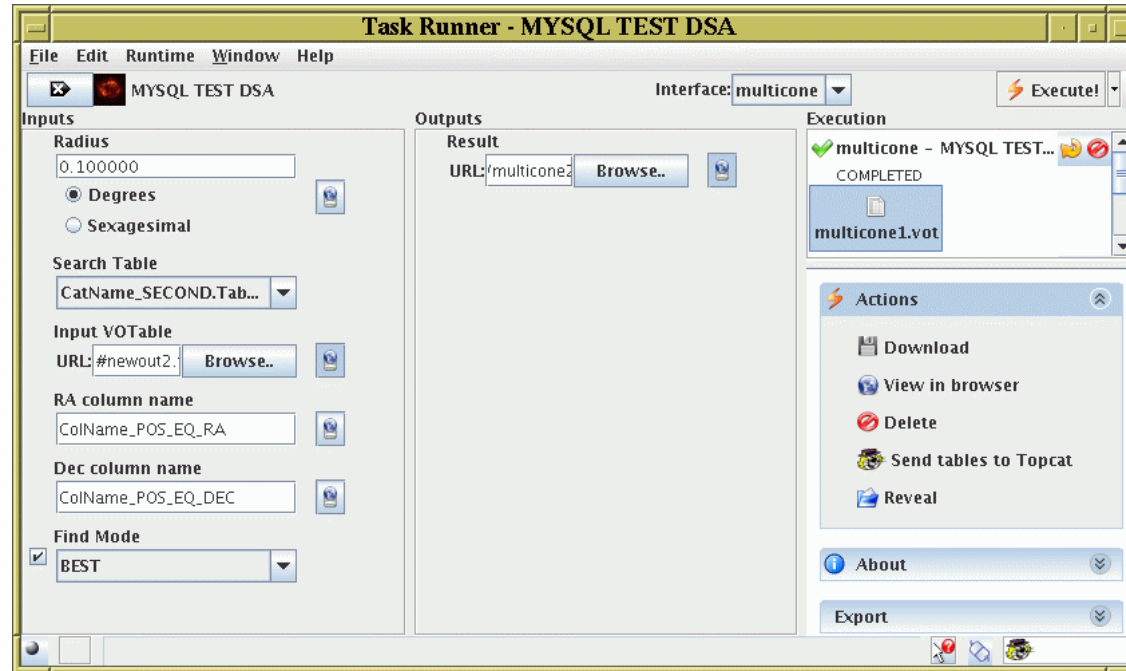
with Kona Andrews

- DataSet Access is the AG component to VO-enable databases
  - Allows DB admins to provide Cone Search, ADQL etc with minimal effort
- DSA now provides crossmatching using multi-cone search algorithm
  - Crossmatch using multi-cone search algorithm:
    - ▷ Submit a VOTable with RA, Dec columns and choose a DSA table
    - ▷ Get back a VOTable which is a join of the submitted and remote ones
  - Provided automatically by any DSA service which provides cone search
  - Two interfaces provided:
    - ▷ `multicone` interface to DSA CEA application
    - ▷ HTTP POST endpoint (non-standard → not currently advertised in registry)
  - Moves *iteration* part from client to server
    - ▷ Should lead to performance improvements over client-side iteration (eliminate latency & overheads)
    - ▷ Initial tests suggest modest improvements — but highly dependent on DB setup
    - ▷ The faster each *search* is, the better the improvement (try spatial indexes?)
  - Prototype/early implementation of multi-cone (etc) services being discussed in IVOA/DAL
  - Somewhat experimental

# DSA Multicone — Examples

with Kona Andrews

- DSA multi-cone search run via CEA interface



- DSA multi-cone search run using HTTP POST

```
curl --data-binary @messier.xml  
'http://server.org:8080/dsa/MultiCone?DSACATTAB=first  
&SR=0.01&RA=ra2000&DEC=de2000'
```

# Future Work??

- STILTS tool improvements

- Provide intra-table and multi-table match commands (already available in TOPCAT)
  - ▷ New `tmatch1` and `tmatchn` commands to go with existing `tmatch2`
- Provide easy-to-use sky matcher

```
skymatch2 radius=0.01 in1=t1.vot ra1=ra2000 dec1=de2000  
in2=t2.vot ra2=_RA dec2=_DEC
```

- Tile-based algorithm improvements

- Increase max size (=decrease memory requirements) of STIL crossmatching
  - ▷ Recursive tiling with decreasing tile size
  - ▷ Disk-based storage during algorithm
  - ▷ . . .

- Multi-cone improvements

- Server-side multicone gives incentive to speed up DSA cone search queries
  - ▷ spatial indexes?
  - ▷ indexes using sky tiling schemes?
- Provide parallel switch for STILTS multicone?
  - ▷ Will this make enemies of data centres?

- Multi-cone facility in VOExplorer

- Could use server-side (DSA) multicone if available, else client-side