Optimisation of the SALT schedule

Brent Miszalski brent@saao.ac.za

Optimising SALT

- A very complex combinatorial optimisation problem (e.g. travelling salesman)
- Five scientific priorities:
 - P0 (time critical or ToO)
 - P1 (highest), P2, P3 (lowest)
 - P4 (poor weather) not charged
- Restricted object visibilities (some wiggle room)
- Changing observing conditions (e.g. seeing, cloud)
- Current approach: Observe best 'now'. Cannot plan ahead easily (may miss high priority future blocks)



UT = 21:00 SAST = 23:00 LST=14:11 ** P0 * P1 * P2 * o P3 * o P4 Hour Angle

<----targets move to left with increasing time

block length lines include acquisition

time critical targets are in pink and time windows are *not* yet considered

Loaded MOS masks have RSS mode in GREEN, missing ones are in RED.

Objects < 25 deg from Moon are not showed (if not time critical) and MD = Moon Distance (deg).

Multiple-visit objects now have date last obs (YYYY-MM-DD) and wait time.

CCAS blocked objects now have orange boxes and **CCAS** warning on this page (azimuth=54.5-65.0 deg).

Wiki page

SALT Hour Angle Investigation Tool

Shows targets visible to SALT now

Helps SA make an informed decision

Moon: 20:08 09:53 (72%)

P0

Select Proposal PI	Instrument	WM	ObsTime	East (SAST)	West (SAST)	Target	Rank	Coords	Visits	Seeing	Moon	MD	MP	Transparency	Progress
28805 2014-1-RU-004 McCully	RSS LS PG900	28805	3487	18:01 to 19:14	22:17 to 23:30	TOO-RUSN-PSNJ1157	н	11:57:44.44 -10:10:15.7	3 / 5 Last: 2014-05-17 Wait: 4.6 days	3.0	Any	118.1	100.0	Scattered Clouds	P0: 59.8% [10461/17500]

P1

Select	Proposal	PI	Instrument	WM	ObsTime	East (SAST)	West (SAST)	Target	Rank	Coords	Visits	Seeing	Moon	MD	MP	Transparency	Progress
28308	2014-1-IUCAA-001	Dutta	RSS LS PG3000	28308	3168	19:58 to	to 00:16	J131938.76-004940.0	Hi	13:19:38.76 -00:49:40.0	0/1	2.0	Durk	101.5	14.6		P1: 10.6% [3253/30600] P2: 0.0% [0/5400] P3: 0.0% [0/16200]
28322	2014-1-IUCAA-001	Dutta.	RSS LS PG3000	28322	2668	19:58 to	to 00:16	J131938.76-004940.0	Hi	13:19:38.76 -00:49:40.0	0/1	2.0	Durk	101.5	14.6		P1: 10.6% [3253/30600] P2: 0.0% [0/5400] P3: 0.0% [0/16200]
28310	2014-1-JUCAA-001	Dutta	RSS LS PG900	28310	2953	21:17 to	to 00:33	J140731.99+044949.0	Н	14:07:31:99 +04:49:49:0	0/1	2.0	Durk	91.3	14.6	Scattered Clouds	P1: 10.6% [3253/30600] P2: 0.0% [0/5400] P3: 0.0% [0/16200]
28384	2014-1-IUCAA_UW-001	m	RSS LS PG900	28384	3553	20:17 to	to 00:26	J133356.02+001229.1	Н	13:33:56.02 +00:12:29.1	0/1	1.5	Dark	98.3	14.6	Thin Clouds	P1: 0.0% [0/42640] P2: 0.0% [0/24160] P3: 0.0% [0/28750]
28611	2014-1-RSA_OTH-002	Vaisanen	RSS LS PG900	28611	3331	22:42 to	to 01:20	CGCG 049-057	Ні	15:13:13.10 +07:13:32.0	0/1	2.0	Dark	76.2	14.6	Thin Clouds	P1: 11.3% [3131/27600] P3: 0.0% [0/24000]
28676	2014-1-RSA_OTH-009	Viljoen	RSS LS PG900	28676	3173	21:34 to	to 01:11	Candidate 24726	Med	14:35:03.30 +03:06:40.0	0/2	2.0	Durk	84.2	14.6	Thin Clouds	P1: 0.0% [0/28800]
28673	2014-1-RSA_OTH-009	Viljoen	RSS LS PG900	28673	3173	20:36 to	to 00:27	Candidate 431	н	13:44:07.70 +01:56:53.7	0/1	2.0	Dark	96.3	14.6	Thin Clouds	P1: 0.0% [0/28800]
28678	2014-1-RSA_OTH-009	Viljoen	RSS LS PG900	28678	3173	21:39 to	to 01:41	Candidate 5680	Med	14:52:23.90 +00:57:03.3	0/2	2.0	Dark	79.5	14.6	Thin Clouds	P1: 0.0% [0/28800]

Fundamental requirement

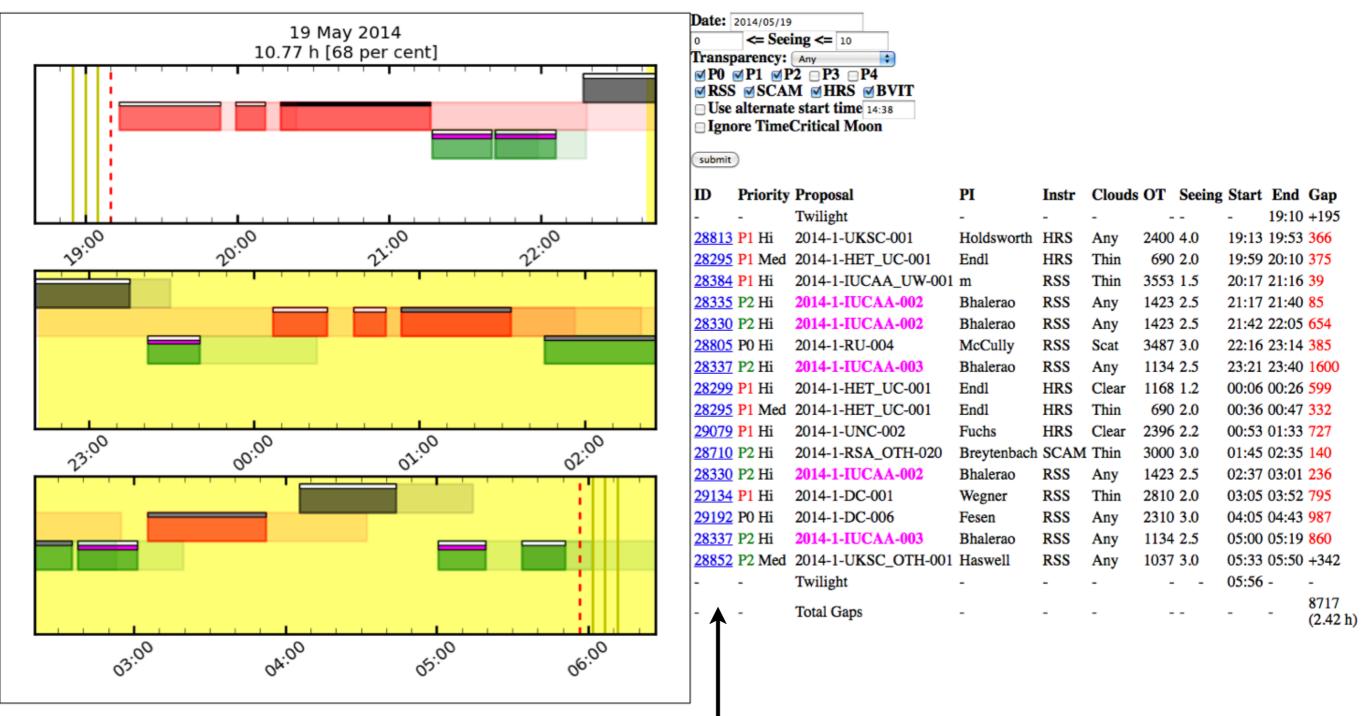
- Queue based telescopes must deliver high completeness for top-ranked science programs
 - Maximise completion of PI and P2 blocks
 - Take observations in most suitable conditions based on requirements specified by PI
- Solution: Create software to optimise the SALT queue, producing an automated schedule for the SA

SALT scheduling software

- Live query to science database
- Object-oriented Python code (using datetime)
 - SubBlock class
 - A schedulable unit west or east track
 - Queue class
 - Gathers the SubBlocks together
 - Allows a queue to be manipulated and optimised
 - Web interface (installed on web.salt.saao.ac.za)
 - Choose observing conditions
 - Display queue with matplotlib and HTML

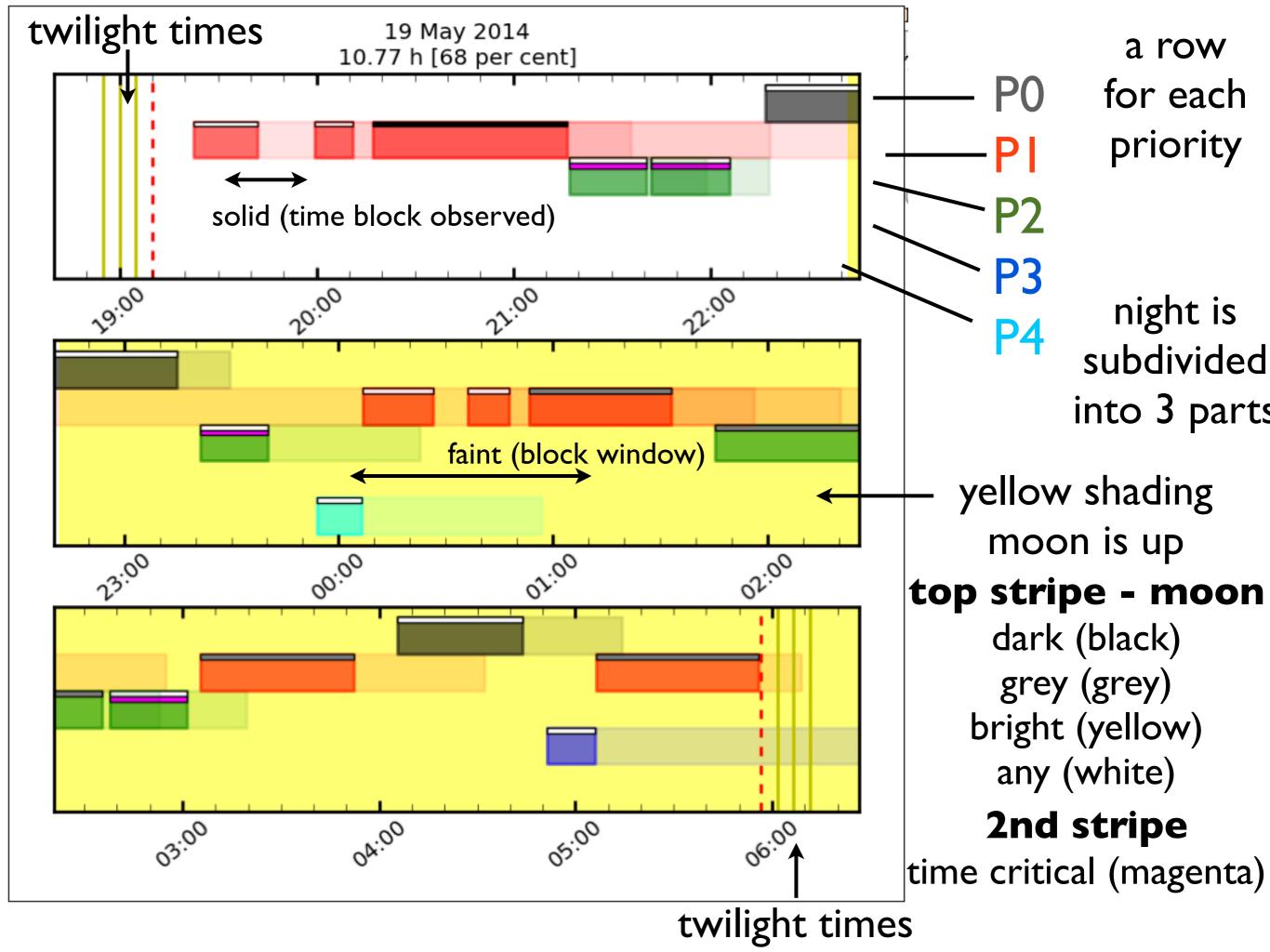
timeline

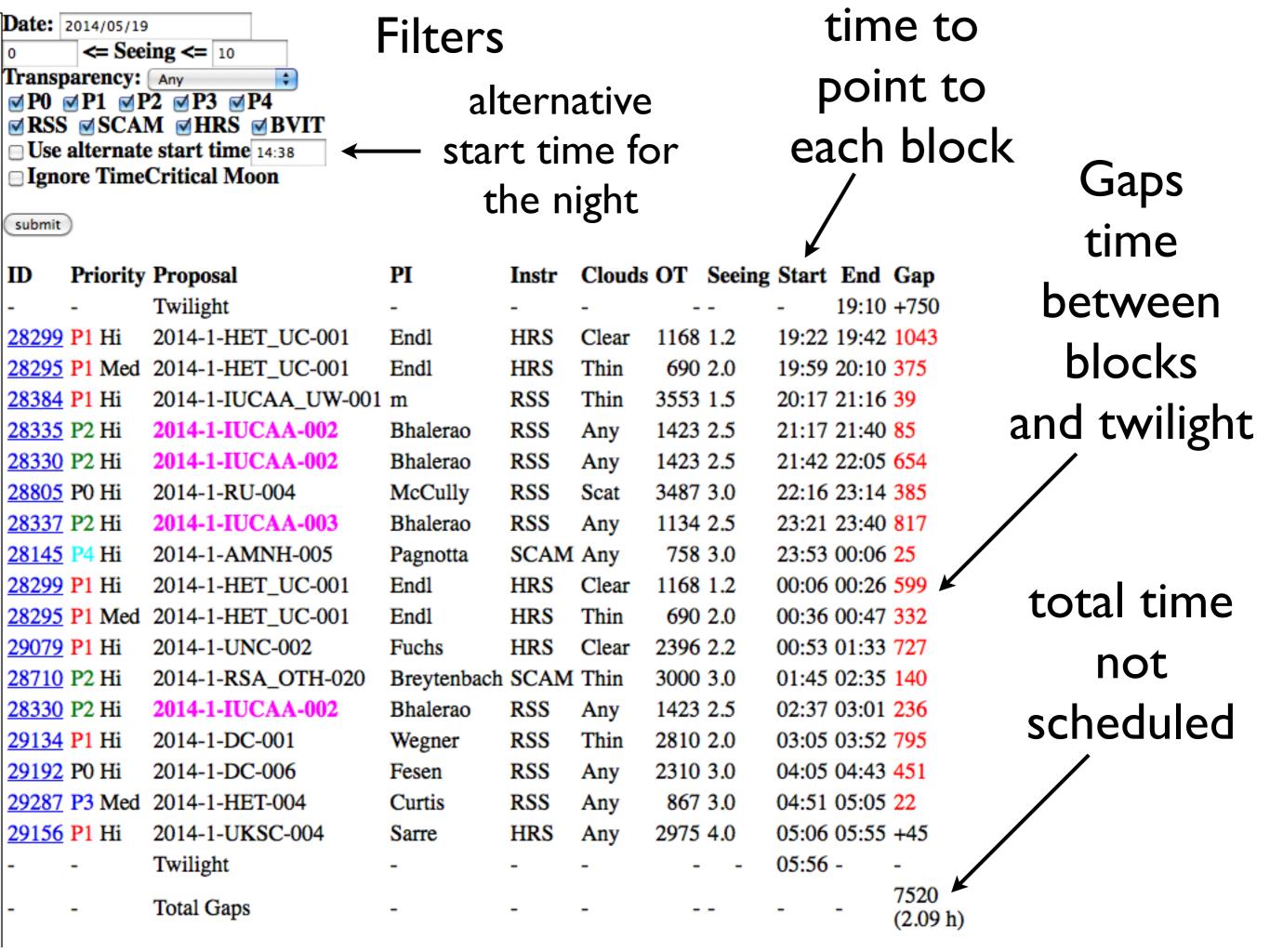
filters



block details

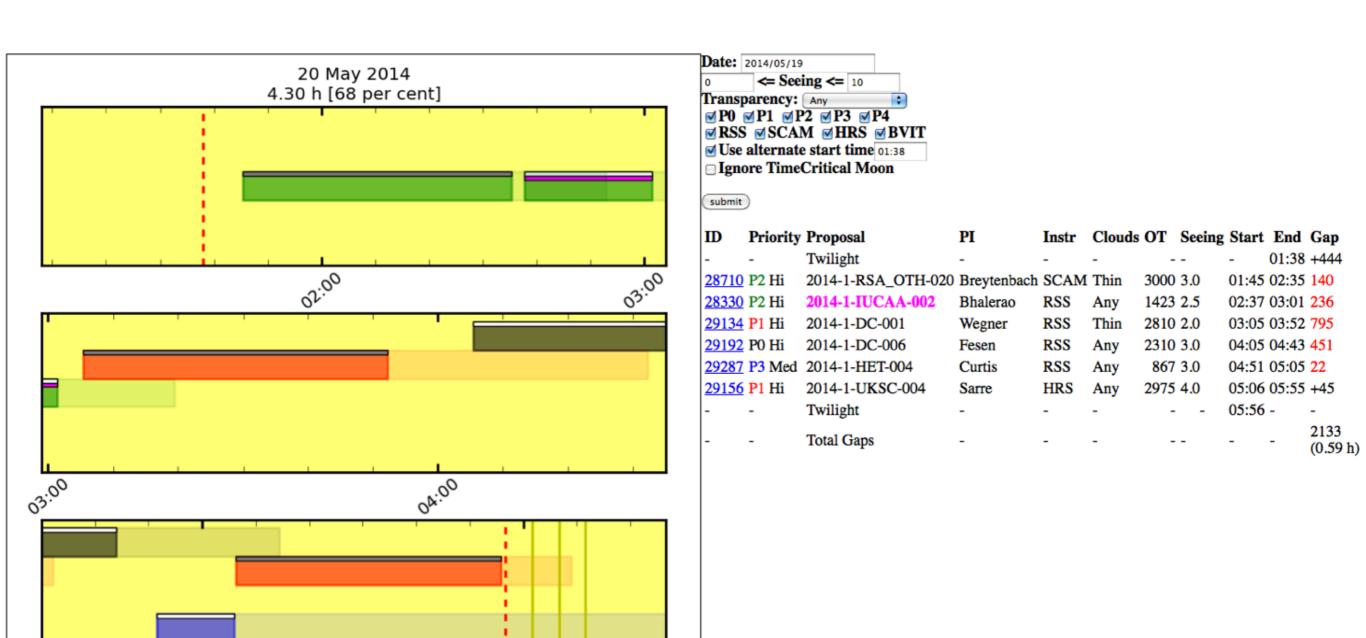
links to WM





Start time = 01:38 am

05:00



Features (fine tuning)

- Select start time (to restart the queue in changed conditions)
- Lunar conditions
 - Blocks use rise and set times to check if observable
 - Ignore acquisition time
 - Moon distance from target (min = 30, Pl limit)
 - Loosened max lunar phase from PI (e.g. grey in bright time)
- Checks for loaded MOS masks (check for slit too?)
- Checks for active time critical windows
- 5 (dark), 10 (grey) and 15 (bright) minute buffers for twilight
 - Could increase bright window for HRS targets (bright stars)

Features (fine tuning)

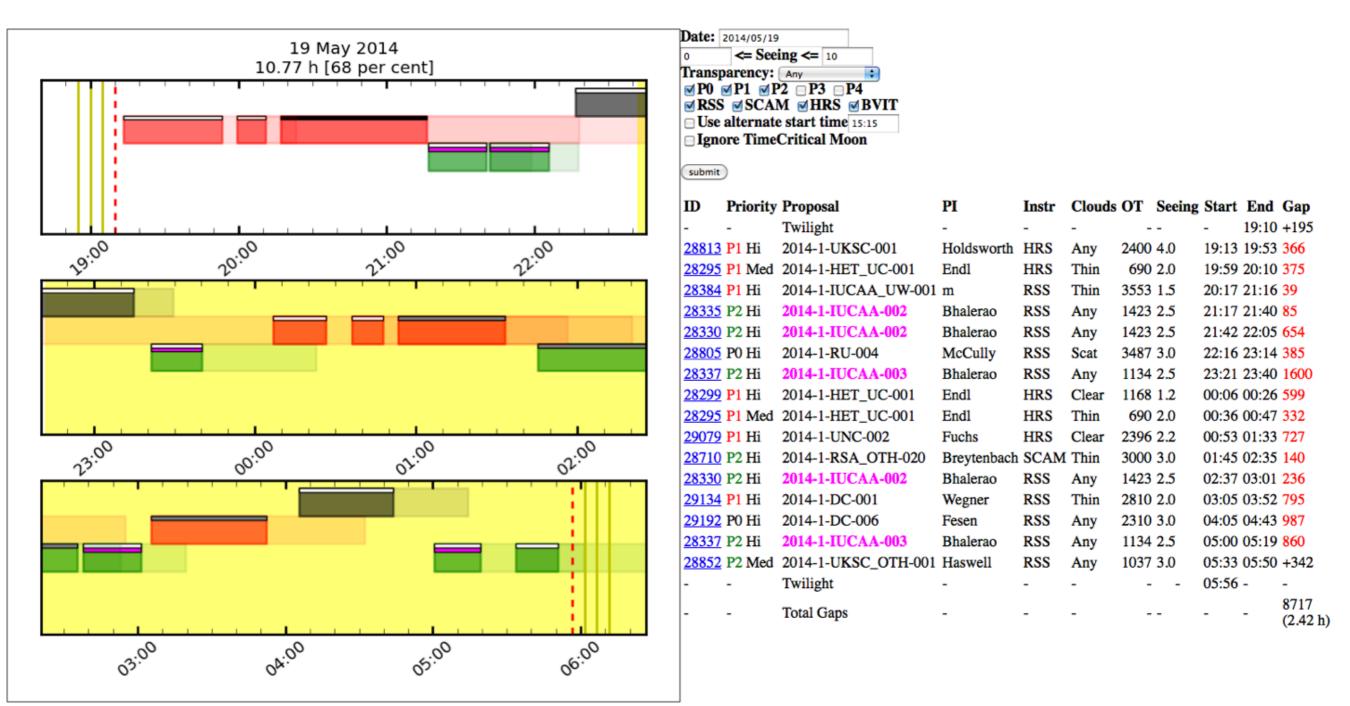
end

- Randomise start time of blocks
 - Checks for twilight, lunar and time critical boundaries (cannot change to an invalid start time*)
 - Essential for randomisation of the queue and squeezing in more blocks
- Still to add:
 - *Pointing windows (do not go to a block if not enough track time is left to complete the observation)
 - Support for pools (groups of targets) and blocks with multiple visits
 - Extending infrastructure to simulate a realistic semester
 - Non-sidereal target support

Optimising the queue

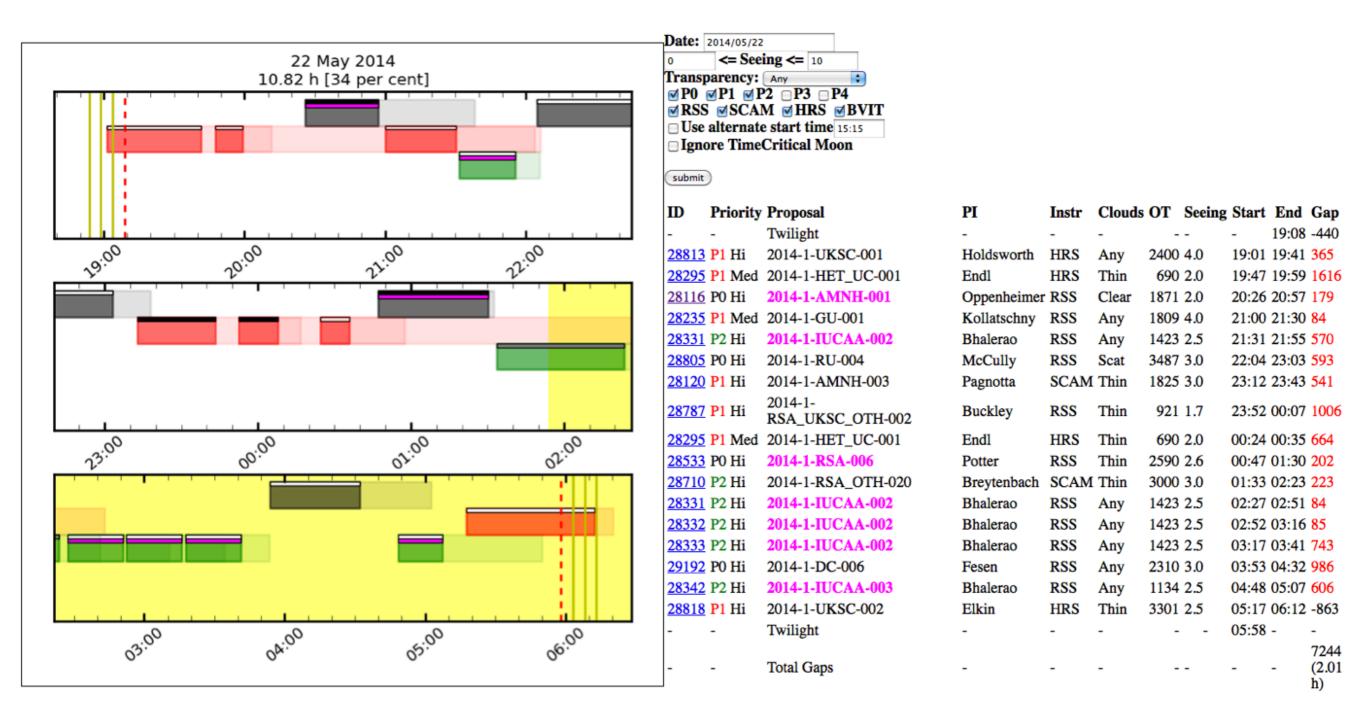
- Turns out to be a standard problem in computer science
 - Weighted interval scheduling problem
- Relatively fast O(n log n) divide and conquer algorithms exist to solve it
 - Already implemented in SALT scheduler software (adapted from http://farazdagi.com/blog/2013/weighted-interval-scheduling/)
- Unfortunately it does not incorporate randomisation of block start times (within their wiggle room)
- Can use as part of an algorithm that does include randomisation (still to be explored)
- Need to run simulations (simulated blocks?) to assess performance of adopted optimisation algorithm

Weighted Interval Scheduling (no randomisation)



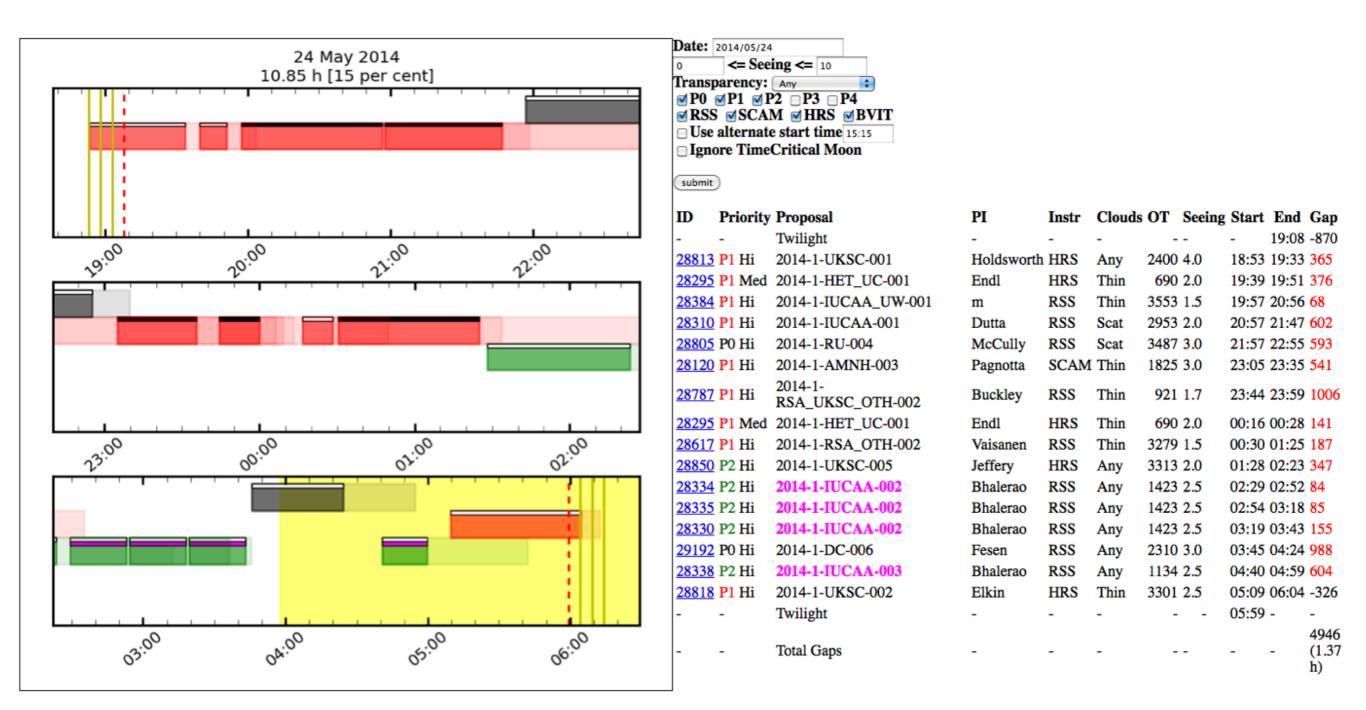
2.42h gaps

Weighted Interval Scheduling (no randomisation)



2.01h gaps

Weighted Interval Scheduling (no randomisation)

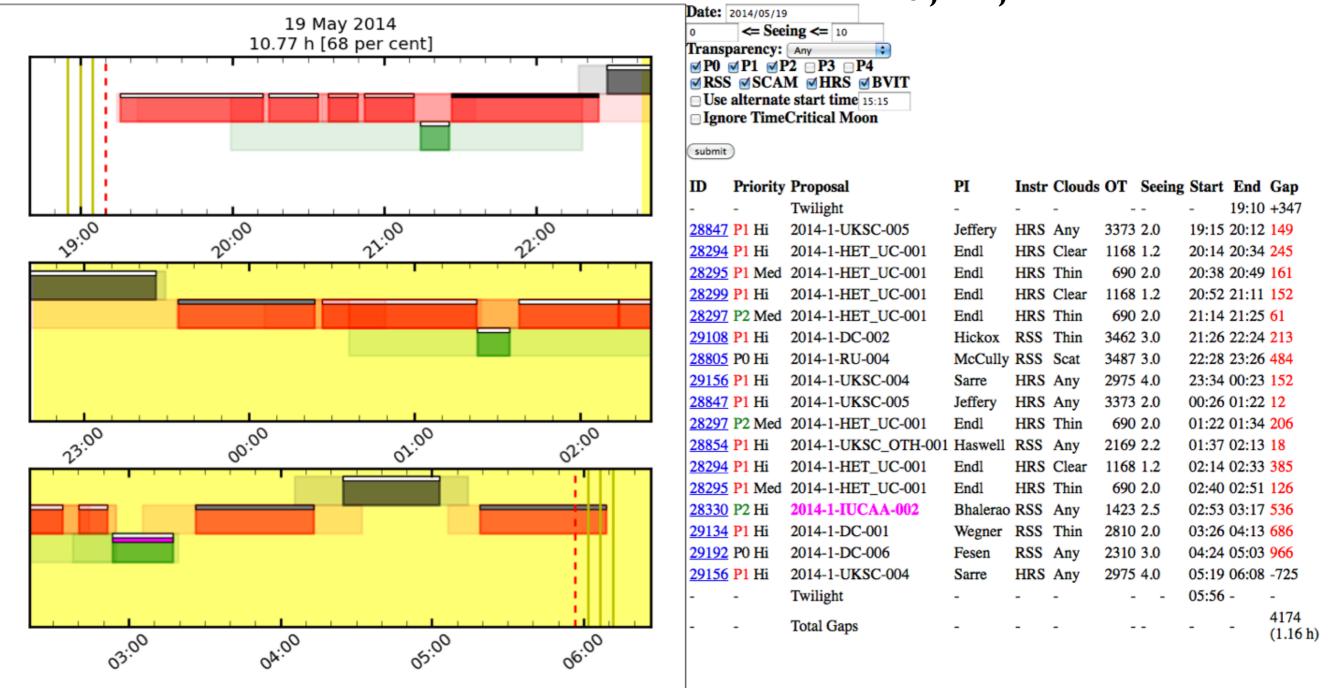


1.37h gaps

Current approach

- Aim to randomise block start times while swapping blocks around to explore the parameter space - works fairly well in filling gaps, but is a little slow and can be improved
- Current approach is to swap between blocks amongst two priority levels only (P0 and P1, P2 and P3, P3 and P4), only accepting moves that are favourable according to...
- Weighting scheme
 - 2**(10-priority) or if time critical, 2**(10-priority-0.15)
 - i.e. two P2 blocks == one P1 block
 - Can be extended to include other effects (partner share, observing conditions, etc)
- More experimentation required once BlockPointWindows are implemented in the database (will allow for 'physical' solutions)

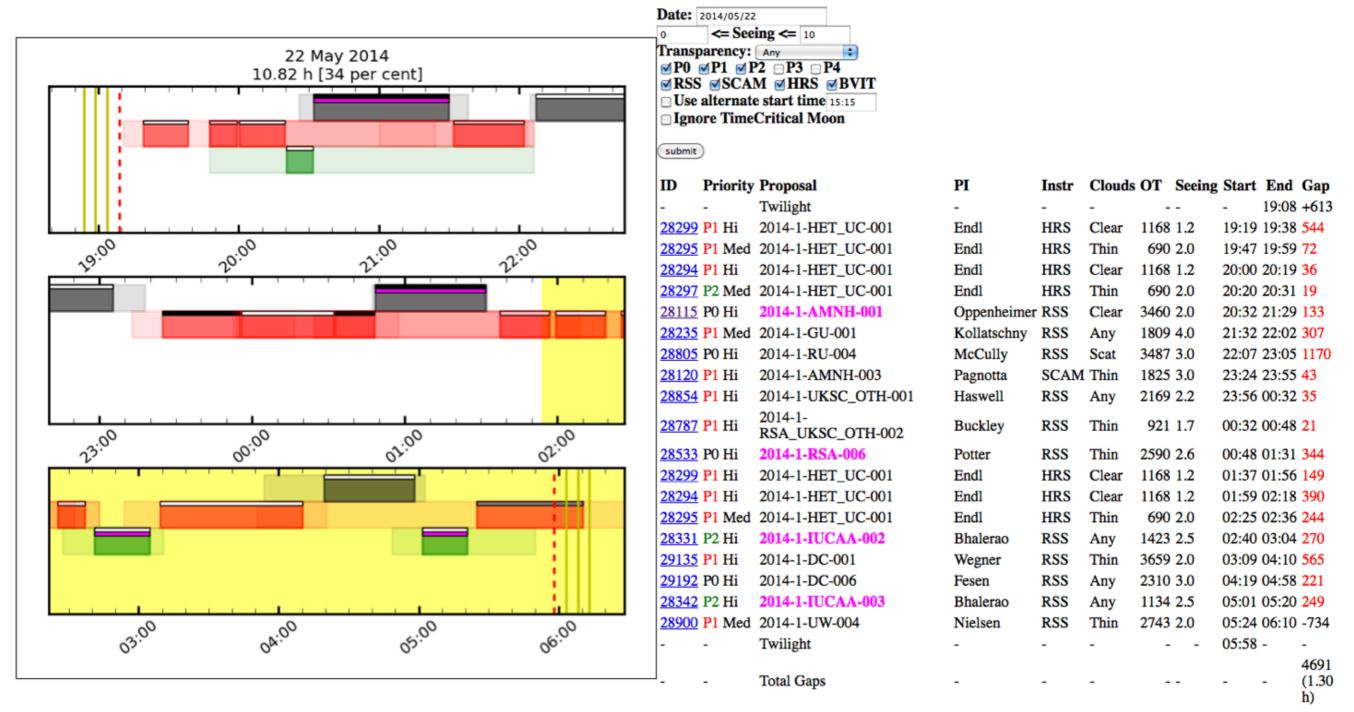
Randomisation (brute force)



1.16h gaps

Randomisation (brute force)

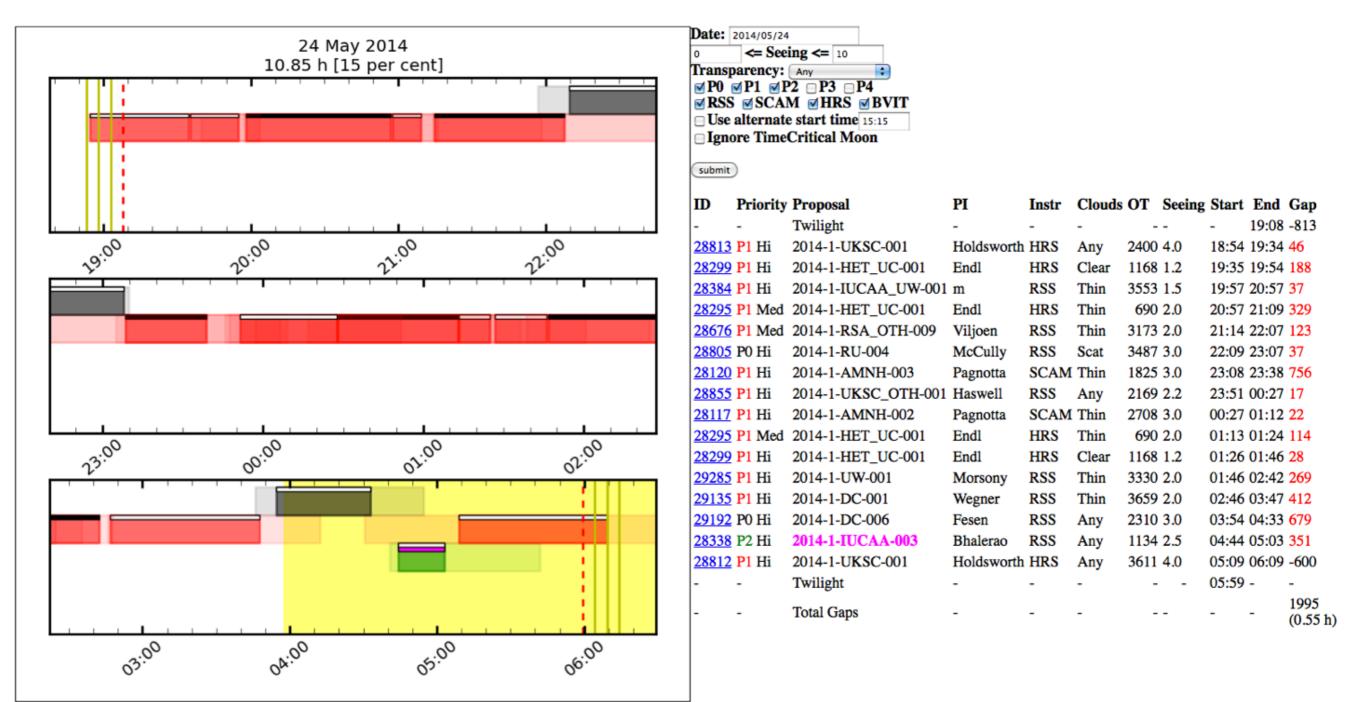
22 May P0,P1,P2



Notice blocks starting after their windows to allow other blocks into the queue!

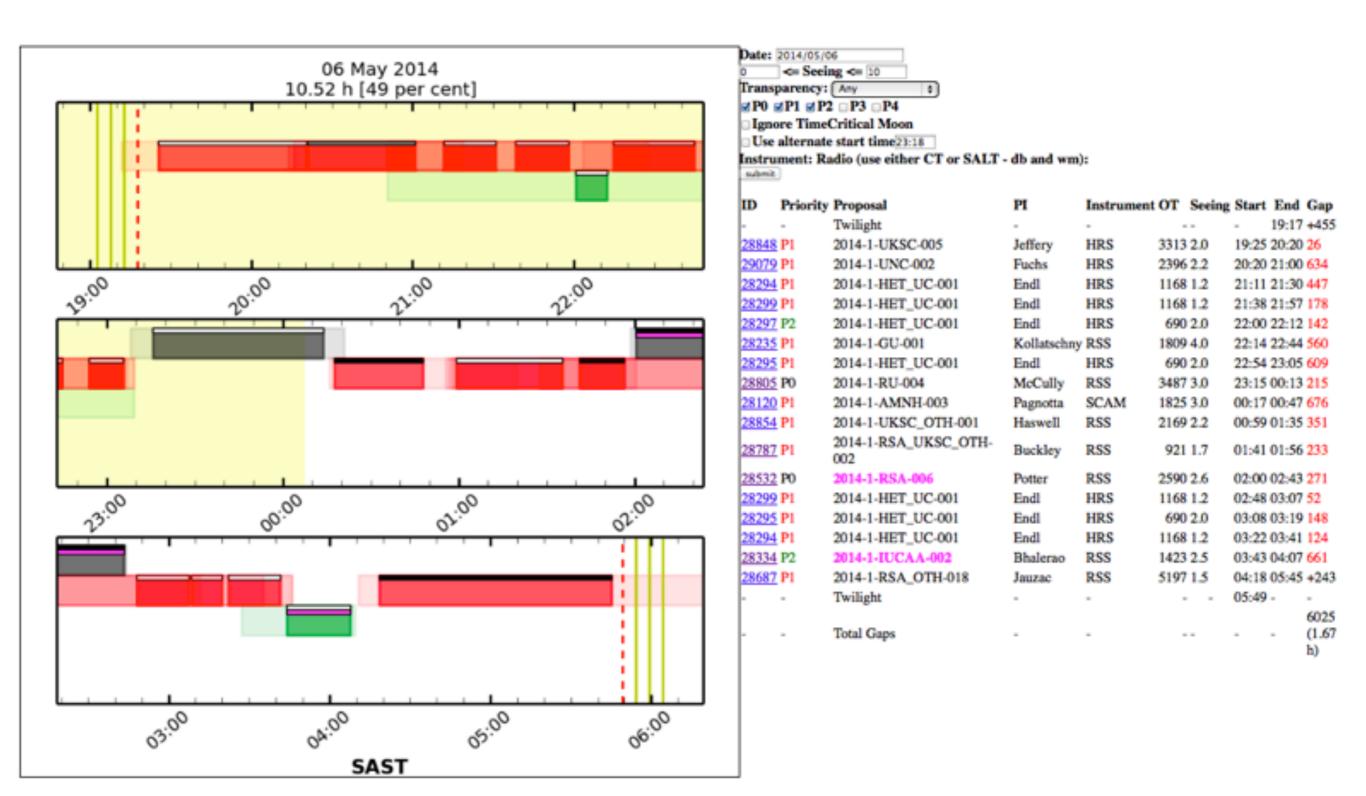
1.16h gaps

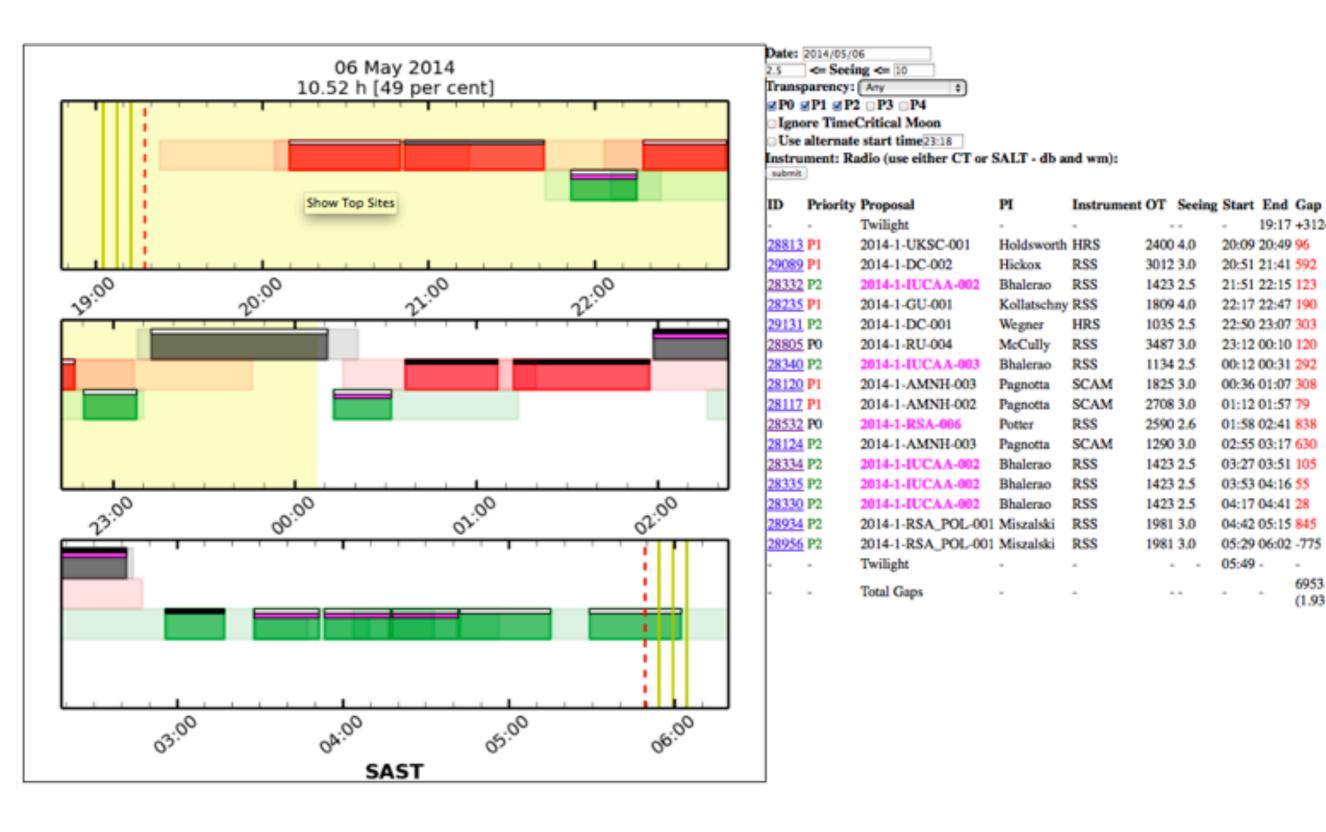
Randomisation (brute force)



0.55h gaps

More brute force randomisation examples

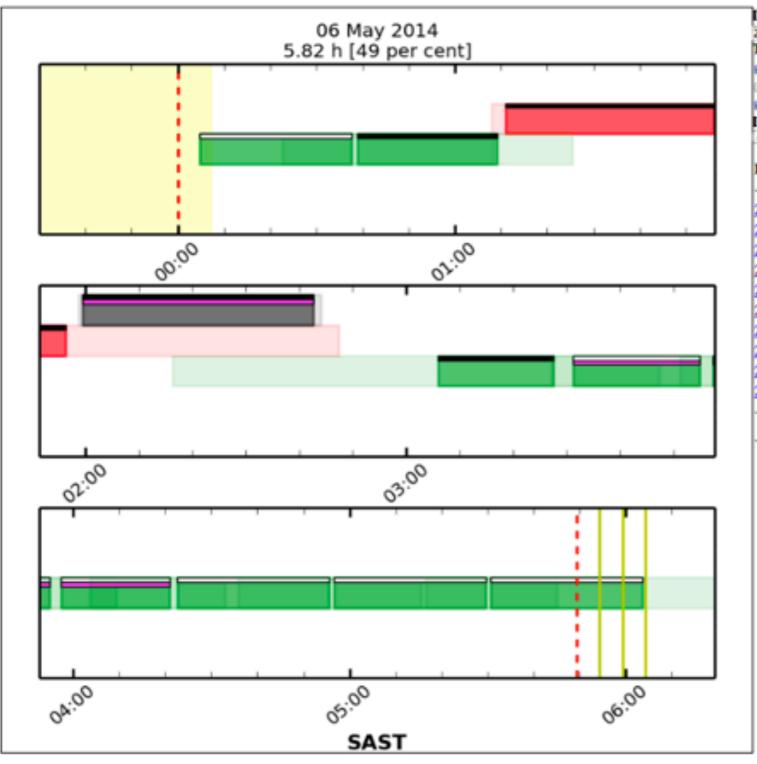




19:17 +3124

6953

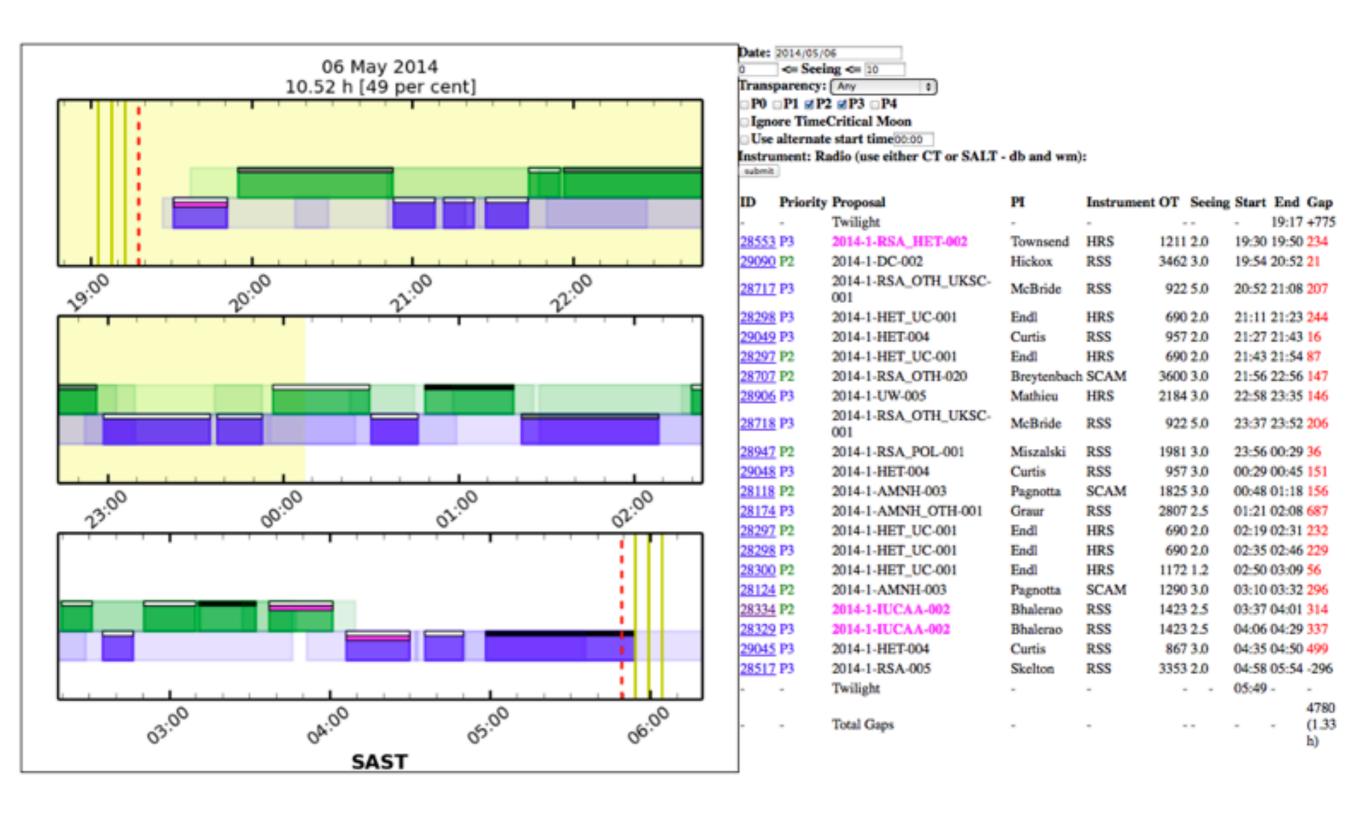
(1.93 h)



Instrument: Radio (use either CT or SALT - db and wm):

submit

ID	Priority	Proposal	PI	Instrument	OT	Seeing	Start	End	Gap
-	-	Twilight	-			-		23:59	+282
28925	P2	2014-1-RSA_POL-001	Miszalski	RSS	1981	3.0	00:04	00:37	68
28118	P2	2014-1-AMNH-003	Pagnotta	SCAM	1825	3.0	00:38	01:09	107
28117	P1	2014-1-AMNH-002	Pagnotta	SCAM	2708	3.0	01:11	01:56	194
28532	P0	2014-1-RSA-006	Potter	RSS	2590	2.6	01:59	02:42	1402
28124	P2	2014-1-AMNH-003	Pagnotta	SCAM	1290	3.0	03:05	03:27	220
28334	P2	2014-1-IUCAA-002	Bhalerao	RSS	1423	2.5	03:31	03:54	151
28335	P2	2014-1-IUCAA-002	Bhalerao	RSS	1423	2.5	03:57	04:21	92
28934	P2	2014-1-RSA_POL-001	Miszalski	RSS	1981	3.0	04:22	04:55	57
28935	P2	2014-1-RSA_POL-001	Miszalski	RSS	1981	3.0	04:56	05:29	57
28956	P2	2014-1-RSA_POL-001	Miszalski	RSS	1981	3.0	05:30	06:03	-855
-	-	Twilight	-			-	05:49		-
-	-	Total Gaps	-	-		-			1775 (0.49 h)



Resources on interval scheduling problem

- In graph theory, equivalent to finding the weighted maximum independent set
 - http://en.wikipedia.org/wiki/Independent_set_ (graph_theory) and
 - http://en.wikipedia.org/wiki/Interval_scheduling.
- http://farazdagi.com/blog/2013/weighted-interval-scheduling/
- http://pages.cs.wisc.edu/~shuchi/courses/787-F09/scribenotes/lec3.pdf
- http://www.cs.uiuc.edu/class/sp08/cs473/Lectures/lec10.pdf