HST Planning and Scheduling Lessons learned

Mark Giuliano

3/17/15

Space Telescope Science Institute

My Background

- PhD in computer science
- 24 years experience at STScI
  - Developing operational planning and scheduling applications for HST and JWST
  - Managing teams creating software supporting astronomical missions
    - Astronomers Proposal Tool
    - Grants Management system
  - Processes engineering to work with project stakeholders to determine integrated software human processes that are efficient and effective
  - Supporting operations
  - Developing code for other missions OPTIX, WFIRST, FUSE, CHANDRA.

Lessons Learned

- I present a series of lessons learned:
  - Start with generic concepts for any component of the ground system
  - Move towards items specific to planning and scheduling
- I will spend much time on human centered processes in addition to scheduling technology
  - These are equally important in creating a scheduling system

Software Engineering

- Utilize best practice software engineering techniques:
  - Revision control
  - Independent user and acceptance testing
  - Design and code reviews
  - Iterative development with plenty of prototypes
  - Utilize continual build and test software (e.g. Jenkins)
  - Build software encapsulation for current requirements
    - Refactor the code to handle new capabilities
      - Don't try and crystal ball hooks for new requirements
    - When your encapsulation adds complexity don't be afraid to rewrite
    - Minimize technical debt
- Starting with generic steps tailor the processes to meet the needs of your mission
  - Tailor at the component level
    - Uniformity is not required across all system components
    - The level of formality required for a component that can damage the telescope is much higher than a component that cannot

Stakeholder Focus

- Build trust and open communication for all stakeholders
  - Developers, testers, users, project management, science community
- Build trust early in the process through involvement, prototypes,
  - Hone your active listening skills
- However, sometimes users will present a solution without defining the actual problem
  - Always define the problem and then determine potential solutions
  - Need to balance between giving stakeholders what they want and what they need

Mixed Initiative Software

- Your planning/scheduling software
  - Needs to not only
    - Produce quality plans and schedules automatically
  - But also to
    - Allow in house experts to guide the software in order to:
      - Add additional constraints to a schedule
        - » Remove or override constraints
      - Pre specify parts of the schedule
        - » Provide the ability for software to check constraints
    - Allow in house users to understand why the software did or did not do a particular action
- Planning/scheduling system can not only make decisions but can callaborate

What Makes an AI Application Successful?

- Good technology is necessary but not sufficient for an application to be successful
  - Additional human and software factors often are more important than the optimal performance of the application
  - Often the technology only has to be good enough to make it work
- From David Waltz pioneer in computer vision:
  - AI in an successful application is like the raisins in Raisin Bran cereal. They are only 2% but its not Raisin Bran without them.

Plan for Evolving the Mission

- As the mission evolves users will learn how to take advantage of observatory capabilities
  - Users will push tolerances for observatory capabilities
    - Tighter tolerances, larger collections of linked observations
    - Software needs to be better, faster, and cheaper to use and maintain
- Plan for process improvements
  - Development and user resources
  - The code should be malleable
- Be forward in terms of technology
  - Users expect better interfaces and faster response
  - While not having to relearn whole new paradigms

Scheduling Granularities

- There is no single scheduler
  - Plan and schedule at multiple levels
- For HST and JWST
  - Long range planning
    - Assigns observations for a cycle to 56 day long least commitment plan windows
      - This is not a long term schedule
    - Concerned with *resource balancing, plan stability*
    - *Prevents short term scheduler from making locally optimal but globally sub-optimal decisions*
  - Short term scheduling
    - Creates week long second-by-second schedules using plan windows as input
    - Concerned with *schedule efficiency*
- Motivation:
  - The precise HST orbit model is known only a few weeks in advance
    - Uncertainties in the orbit prevent the creation of second-by-second schedules.
  - Separation of concerns:
    - Change is the norm
    - Even if we could create long term schedules (i.e. precise assignments to time) they would rapidly become invalid as program inputs change
    - The long range plan can remain stable while the short term schedules change

Stability and Money

- Planning for grant money is a strong motivation for plan stability
  - Grant money for observers typically only starts when the first observation of a program is executed
  - Observers typically hire graduate students to help reduce data
- Observers need to know roughly when their observations will execute in order to plan for hiring
  - Do not need a precise schedule and just need to know when observations will be scheduled within the order of a month

LSST Scheduling Granularities?

- How many levels of planning and scheduling are there for LSST?
  - Mission level – Over multiple years
    - Not needed for HST or JWST as the mission is GO driven
  - Yearly cycle
    - Based on yearly GO cycle and progress of long term survey
  - Mid-term - Plans for the next month? Week?
    - Based on forecasted weather
  - Nightly – Plans the current night based on weather conditions
    - Replans activities as the weather changes
- How do the scheduling granularities integrate?
  - What is a plan or schedule at each level of granularity?
  - What entities are scheduled?
    - While a short term scheduler will schedule exposures longer term planning engines can schedule visits
  - What is the primary concern at each level?
  - What constraints are modeled at each granularity how are they modeled
    - E.g. Slew time in a long range plan is statistical while slew time in a scheduler is known

Planning Use Cases

- Understand the requirements for your use cases for each granularity of scheduling
- Long range planning:
  - Cycle ingest
    - Plan newly approved science on top of existing plan
  - Cycle maintenance
    - As the cycle executes adjust the plan based on new observations and executions of past observations
  - What if engineering scenarios
    - What will the plan look like with an alternate weather model or a new and improved slew capability?
- Short term scheduling (= ? Nightly planning)
  - Plan a night based on long range plan input and weather forecast
  - Replan a night based on a changed weather forecast

Science Specification

- In service mode observing
  - Observer creates an observation specification which provides the science goals
    - Submits specification to observatory
  - Observatory staff carry out exposures which meet the science goals of the specification
    - Observers do not have to know the details of how to operate the observatory
    - Can map specification to different scientifically valid instances depending on the circumstance

Make Observers Responsible for Efficiency

- In early HST observing cycles observers were allocated time in terms of seconds.
  - In low earth orbit a target is typically visible for ~50 minutes out of the 96 minute orbit
- There was no motivation to make series of exposures that would utilize orbits efficiently
  - So allocated 90 minutes of visibility a PI might make three 30 minute exposures requiring three orbits each with 50 minutes of visibility
- By cycle 5 observers were allocated time in orbits and were thus motivated to fill every second of visibility
- *But observers were often frustrated by having to iterate to use up all of their visibility*
  - We had to provide tools to automatically fill orbits
- **Make observers responsible for efficiency but provide the tools required so they can achieve their goals without being frustrated**
- **Science policy can drive scheduling efficiency**

Constraint Types

- Constraints can be categorized along multiple dimensions:
  - Physical versus observer specified
    - Physical constraints are those required by the capabilities and tolerances of the observatory
      - Sun avoidance, Earth avoidance, Moon avoidance, Guide stars
      - Cannot be relaxed while observer constraints can be relaxed
  - Constraints on a single observation versus constraints on one or more observation
    - Absolute constraints versus relative constraints
  - Hard constraints that cannot be violated versus soft constraints or preferences
    - Some single features will belong to multiple categories
      - For example background noise is a preference up to a limit where it is a hard constraint

Calculating Scheduling Constraints

- Need to provide tools to observers which verify that their science specification is schedulable in isolation of other programs
  - Calculate constraints independently
    - So they know what to relax to gain schedulability
  - Calculate constraints quickly so users get feedback
    - This often conflicts with computing constraints independently
  - When possible allow constraint creation without specifying a full observation
    - Makes it easier to explore the schedulable space
  - Provide visual tools display constraint windows
    - Interactive tools that allow parameters to be adjusted with visual feedback

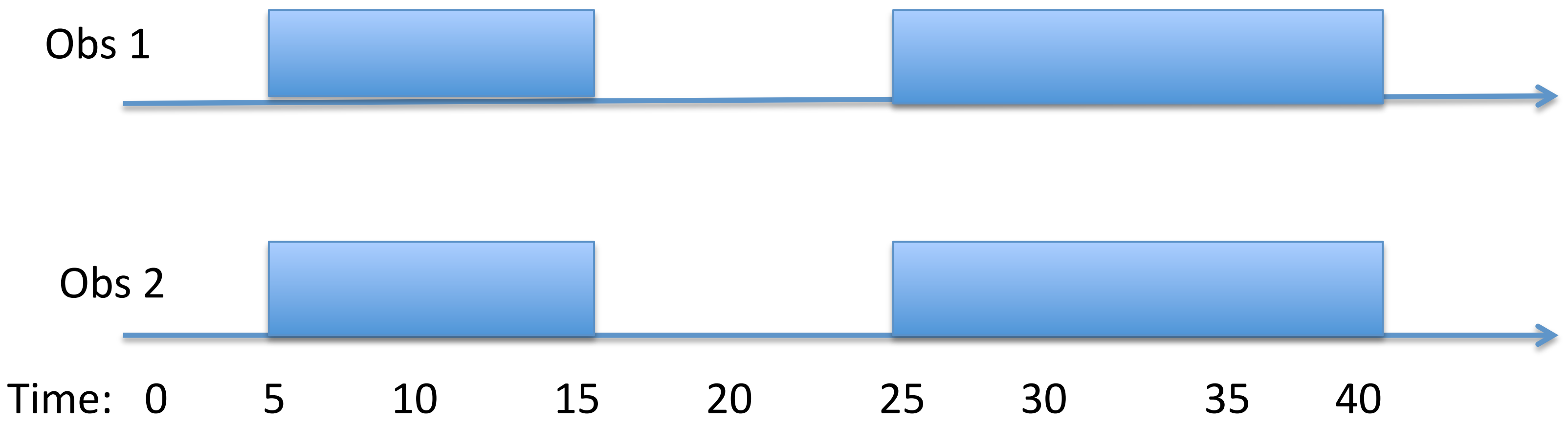
Temporal Constraint Propagation

- Given a set of pre computed absolute constraints and relative constraints
- Want to be able to propagate constraints so that:
  - legal scheduling windows can be made available to observers and to the planning/scheduling software
- Constraints are beyond simple temporal networks
    - Absolute constraints can have multiple intervals
      - E.g. the sun constraint can be satisfied in different intervals over the year

PI Specified Constraints Examples

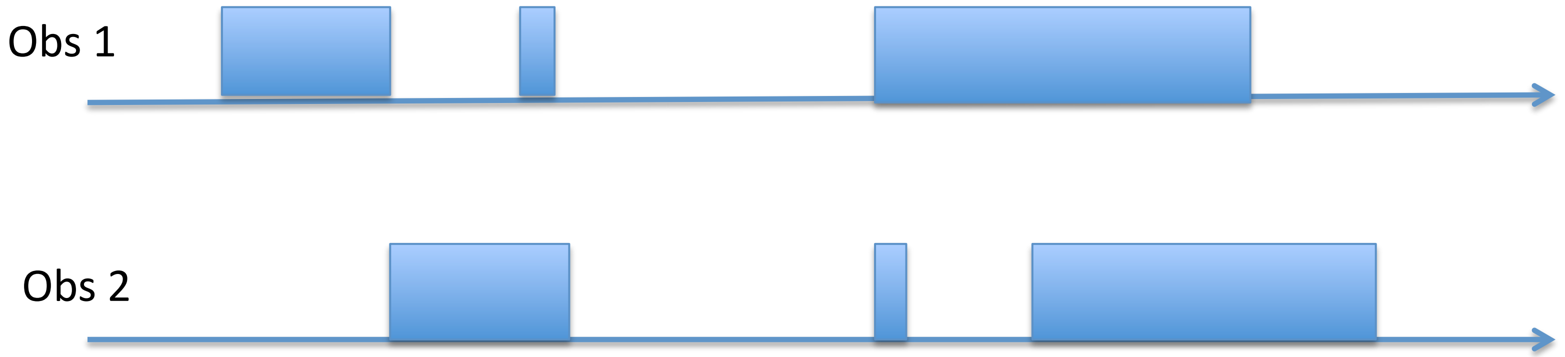
- Absolute Constraints
  - Observation 01 between June 1 2016 July 1 2016
  - Observation 02 After August 15 2017
  - Phase: Plan Observation 01 in between 0.25-0.5 of 4 day period starting September 1 2018 at 11 am
  - Telescope roll constraints: schedule Observation 02 at roll 40-60.
- Relative constraints
  - Observation 02 after 01 by 10-20 days
  - Group observations 05, 06, 07 within 24 hours
    - This makes propagation NP complete
  - Sequence observations 08,09, 10 within 48 hours
  - Roll links: Orient OB02 10-20 degrees from OB01
- Group within make the problem NP complete
  - Can approximate full propagation

Constraint Propagation Example



These plots show intervals that are good for scheduling two different observations.

Now suppose that Obs 2 is after Obs 1 by 5-12 time units



Features of the Search Domain

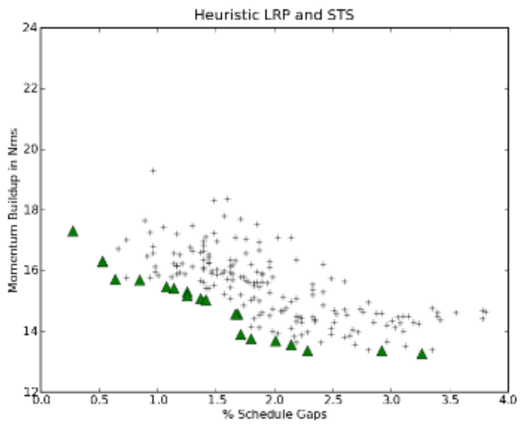
- What is the hard part of the search domain?
- Is the problem highly constrained so the problem is finding any feasible solution
  - That is a solution that violates no hard constraints
  - Primary search is in the constraint satisfaction space
- Or is it easy to find a feasible solution but hard to find a highly preferable solution
  - Primary search is in the preference space
- Often it is hard to find a feasible solution and you still need to satisfy preferences

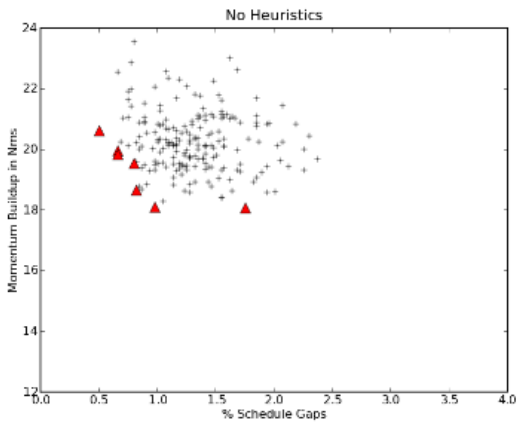
Multi-objective perspective

- Effective scheduling of astronomy missions requires the ability to make trade-offs among competing mission objectives:
  - Time on target, minimizing the use of critical mechanisms, preferring the higher priority science, Survey versus GO programs, …..
- Objectives are often competing in that improving one objective means making another objective worse
- Objectives have different constituents lobbying for them
  - e.g. mission science community versus engineering
- The traditional approach is to combine the weighted average of separate objectives, e.g. $\sum \alpha_i f_i(x)$
  - But: combining objectives loses information and pre-determines the trade-offs among them!

Multi-Objective Solution Approaches

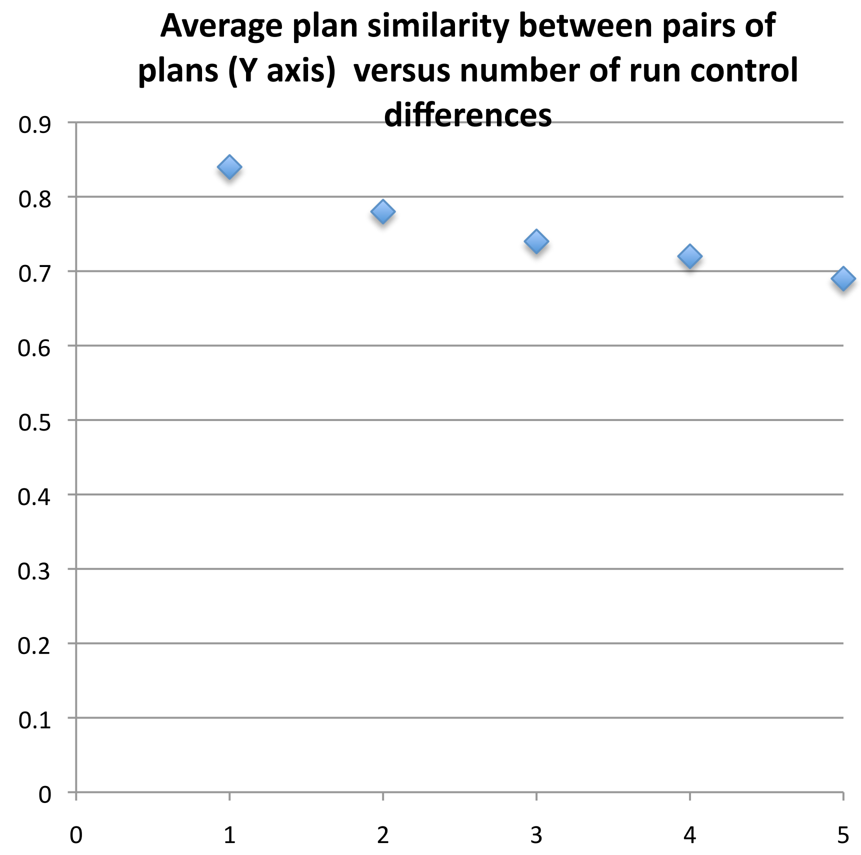
- Multi-Objective Scheduling:
  - Explicitly maintain and exploit multiple objectives during scheduling
  - Algorithms build up approximate Pareto optimal frontier from a population of candidate schedules
    - i.e. "non-dominated" solutions, such that no other candidate is better, considering all objectives.
    - Utilizing evolutionary algorithms (e.g. GDE3)





Collect Metrics on your scheduler

- SPIKE has many runtime parameters with discrete settings and we can compare runs the different by 1 parameter, 2 parameters, ….
- Take all pairs that differ by N parameters and compute the average LRP difference
- The results are as hoped for as the number of run controls increases the amount of plan similarity decreases
- In other words the run controls have an impact



Conclusions

- Building a planning and scheduling system requires paying attention to generic factors:
  - Software engineering, human factors, teaming
- As well as technical areas within planning and scheduling
  - Scheduling granularities, use cases, constraint and preference modeling, ….
- I hope you found this useful
  - When I did my brain storming session with my team for lessons learned we had not finished after an hour
    - I did not cover all the topics that we covered…..