Prioritization and Oversubscribed Scheduling for NASA's Deep Space Network

Caroline Shouraboura[*], Mark D. Johnston[†] and Daniel Tran[†]

[*]Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh PA 15213
cshourab @ andrew.cmu.edu

[†]Jet Propulsion Laboratory, California Institute of Technology, 4800 Oak Grove Dr., Pasadena CA 91109
{mark.d.johnston, daniel.tran} @ jpl.nasa.gov

Abstract

NASA's Deep Space Network (DSN) is a unique facility responsible for communication and navigation support for over forty NASA and international space missions. For many years, demand on the network has been greater than its capacity, and so a collaborative negotiation process has been developed among the network's users to resolve contention and come to agreement on the schedule. This process has become strained by increasing demand, to the point that oversubscription is routinely as high as 40% over actual capacity. As a result, DSN has started investigating the possibility of moving to some kind of prioritization scheme to allow for more automated and timely resolution of network contention. Other NASA networks have used strict static mission priorities, but if this were applied in the same way to the DSN, some missions would fall out of the schedule altogether. In this paper we report on analysis and experimentation with several approaches to DSN prioritization. Our objectives include preserving as much of each each mission's requested contact time as possible, while allowing them to identify which of their specific scheduling requests are of greatest importance to them. We have obtained the most promising results with a variant of Squeaky Wheel Optimization combined with limiting each mission's input based on historical negotiated reduction levels.

1. Introduction

NASA's Deep Space Network (DSN) consists of three communications complexes, located in Goldstone, California; Madrid, Spain; and Canberra, Australia. Each complex contains one 70-meter antenna and three or four 34-meter antennas. These ground antennas are responsible for communications and navigation support for a wide range of scientific space missions, from those in highly elliptical earth orbits, to some beyond the solar system. In future years, DSN will also support human missions to the moon and beyond.

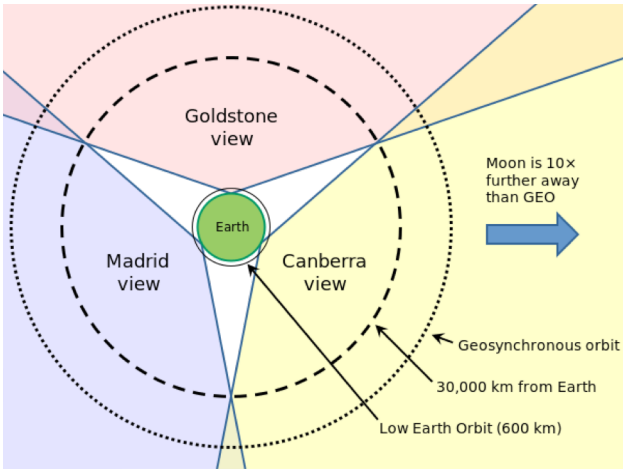

Fig. 1: Fields of the view of the DSN complexes, showing overlapped coverage for distant spacecraft.

The placement of the three DSN complexes allows at least one of them to be in view of any distant spacecraft at all times (Fig. 1); see (Imbriale, 2003).

At this time there are approximately forty missions using the Deep Space Network. There is a natural cycle of missions ending and new missions starting up in their *prime* mission phase, but the majority of DSN users are in their *extended* mission phases. The distinction between prime and extended missions plays a role in some prioritization suggestions, as will be discussed below.

The DSN has seen an increasing level of oversubscription in recent years. In 2011/2012 there was roughly 300 hours per week total difference between initial and final mission time allocations (how much tracking time was requested vs. how much could actually be scheduled.) However, in late 2015, this had grown to as much as 500 hours per week, an

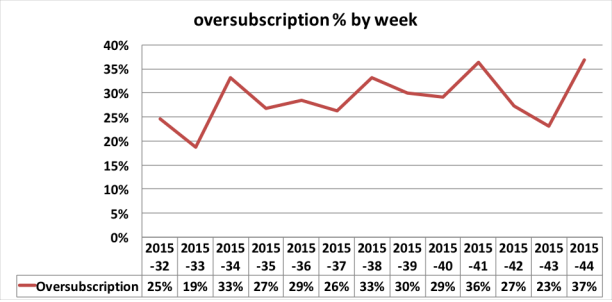

Fig. 2: Oversubscription percentage by week for a range of 13 weeks in late 2015. The average is about 30% with a standard deviation of 5%

increase of over 50% (Fig. 2). To place this in context, the weekly oversubscription amounts to about *4 additional antennas* worth of activities, over and above the 13 actual (34m and 70m) antennas in the DSN.

Along with oversubscription, the number of conflicts in the schedule has also increased. Most conflicts are due to oversubscribed resources, i.e. antennas or other assets at the DSN complexes. Resolving these conflicts takes increasingly high levels of human effort, since they are phrased as irreducible time requirements.

In the following (Section 2) we first briefly describe the DSN scheduling process, highlighting where oversubscription impacts the scheduling software and processes that generate and manage DSN schedules. We then describe some of the factors that come into play in evaluating priority schemes for the DSN (Section 3). Results of a series of experiments with different algorithms are then presented and discussed (Section 4), followed by conclusions and directions for future work (Section 5).

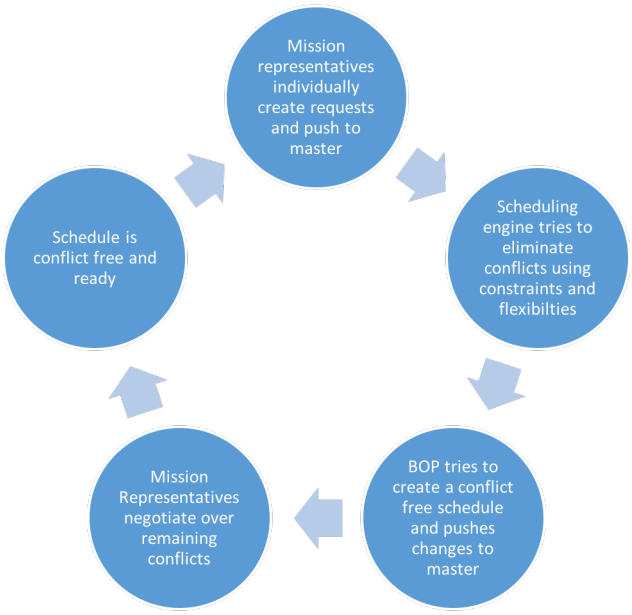

Fig. 3: Schematic diagram of the DSN mid-range scheduling process: for each week, the process starts at the top.

2. DSN Scheduling Process Overview

The DSN scheduling process (Johnston et al., 2014) operates on a rolling weekly basis (Fig. 3): as the deadline for a week approaches (roughly four months before the start of the week), mission scheduling representatives enter the requirements for that mission into the Service Scheduling Software (Johnston et al., 2012, 2012). Unlike other networks, many DSN user missions have changing requirements from week to week, reflecting mission events and phases, including a wide range of pre-planned science activities. Due to the long light travel time to many DSN spacecraft, spacecraft are sequenced with command loads that are generated many weeks ahead, and the DSN schedule is a critical input to this process.

Once all inputs for a week are in, they are integrated into a single schedule and the DSN Scheduling Engine (DSE, (Johnston et al., 2010)) is run to deconflict as much as possible, given any specified flexibilities in the input requirements from each mission. In practice, little flexibility is allowed, and the net oversubscription level means that many conflicts necessarily remain in the schedule.

Each requirement has a specified priority on a scale from 1 to 7. The default value is 7, nominal mission operations, and nearly all activities are assigned this priority level. Exceptions are made for elevated criticality events like launches, planetary landings and orbit insertions, and other high-risk mission events or unique major science opportunities, but in general these are rare. Note that priorities are on specific requirements, not on missions: there is no intrinsic priority distinction from one mission to another. Priority is used by the DSE to place higher priority activities on the schedule at their preferred times and antennas, and then to place lower priority activities where they least conflict with higher priority ones. However, given the high levels of oversubscription, many lower priority activities are placed in conflict with higher priority ones, since it is not permitted to drop them out of the schedule at this stage.

Once the scheduling engine has been run, and conflicts reduced as much as possible based on specified flexibilities, a human scheduler called “Builder of Proposal”, or BOP, starts to work on the schedule and makes further changes based on experience and background knowledge of each mission’s requirements. These changes include: deleting some activities, shortening tracks below their specified minimums, splitting tracks flagged as unsplittable and placing the (now shorter) segments into gaps in the schedule. This a time-consuming and labor-intensive process, requiring a great deal of familiarity with the entire DSN mission set and their typical requirement patterns. The BOP can generally

eliminate ~200 conflicts, but at the end there usually remain 10-20 conflicts. At the conclusion of the BOP phase, the week is released to the full set of mission scheduling representatives to negotiate the remaining conflicts and to make any adjustments to changes introduced by the BOP.

The second part of the interactive scheduling phase is when individual mission representatives collaboratively negotiate peer-to-peer, to resolve remaining conflicts and make additional changes (Carruth et al., 2010). In this process, one user will propose a set of changes, to which all affected users must concur before it becomes the new baseline. If any user disagrees with the changes, it falls on him or her to counter-propose an alternative (where just undoing a previous proposal is not allowed!). This process continues until the deadline is reached, at which point conflicts are either cleared or (rarely) waived, and the schedule is considered baselined and published. From the completion of the automated scheduling run to the baseline conflict-free schedule is typically 2-3 weeks. The overall duration of this process means that multiple weeks are being worked on in parallel.

3. Priority Considerations for DSN

As noted above, DSN currently uses a 7-level priority scheme strictly for categories of *events*: at the top are safety- and mission-critical activities, and at the bottom are normal science operations. Because nearly all activities (except essential maintenance) are considered "normal science" and thus at the lowest event priority level, the current priority scheme provides virtually no guidance for addressing oversubscription.

In terms of the current DSN scheduling process (Section 2), the greatest leverage for process improvement comes from the pre-BOP automated scheduling step: if oversubscription could be addressed *prior* to the BOP process, then both the BOP effort and the collaborative negotiation process phases could be drastically reduced. Missions would have their schedules baselined earlier, and could start work earlier to plan their onboard activities and generate their command loads. Prioritization could also play a role in later schedule changes, but these changes are of a much smaller magnitude. In the remainder of this paper, we focus entirely on the pre-BOP scheduling phase.

A variety of factors could be incorporated into a more fine-grained prioritization, including the following:

1. *Prime vs extended* missions: only about 25% of DSN mission users are still in their prime mission phase: the rest are in their extended missions (some have been flying for nearly 40 years). More than half of all requested time comes from extended missions. While prime vs extended could be used as a prioritization factor, would not help with addressing oversubscription, which would still be a problem even it there were no prime missions at all.
2. *NASA vs. non-NASA* missions: as DSN is a NASA asset, one option would be to give NASA missions priority for its use. However, high level agreements with partners provides for use by non-NASA missions on the same footing as NASA missions. As a result, this is not a factor that can help with oversubscription.
3. *Intra-mission priority tiers*: this concept calls for missions to divide their requested DSN time into priority tiers, rather than submitting all at the same event priority. This can provide explicit information about what each mission could possibly "do without" as being of lower priority. This information is implicit in the cuts that missions accept each week, accounting for the hundreds of hours of antenna time that is reduced by the BOP or negotiated away.
4. *Enforced reduced input levels*: this notion is based on the observation that missions ultimately accept reductions to deal with oversubscription, and so constraining their input levels to historically accepted output levels would be one way to enforce a request pool that would eliminate or drastically reduce oversubscription. For example, if mission X routinely states a requirement for 80 hours of tracking time, and then routinely accept 55 hours, their input could be constrained to 55 hours in the first place. This would add an additional check/enforce step to the process, but could shorten all the downstream steps. A drawback of this approach is that some mission requirements tend to vary from week to week and so a constant cut-off would be a problem for some missions.
5. *Time-dependent priority*: most DSN users require a time spread in their activities, to reflect the accumulation of scientific data and subsequent transmittal to Earth, and for regular measurements for navigation updates. Most also have a check for communication with Earth, such that if they have not been in touch for some configurable time, the spacecraft goes into "safemode". As a result, the time since last contact comes into play when considering the priority of each mission, so that no mission can be "starved" and drop out of the schedule, thus threatening spacecraft health and safety.

4. Experiments and Results

Experimental setup

To define a uniform basis for experiments, we used a 16-week period in 2012 consisting of just over 4,000 requirements for 31 missions. This particular dataset is only modestly oversubscribed, but reflects a realistic mix of typical requirement types. We used a modified version of the DSN Loading Analysis and Planning Software (LAPS) (Johnston et al., 2012) being developed for long-range planning and forecasting. This software allows for plugging in different algorithms for prioritization, thus making it easy to experiment. It can also be configured to drop requirements that can't be satisfied without conflicts.

As a baseline, we used a greedy algorithm that works as follows: for each priority tier from highest to lowest, order requirements by most constrained first, and schedule in their most preferred place (time/antenna). If there are no feasible places left, add the requirement into the unscheduled set. For our sample dataset, the overall total scheduled/requested is 88%. In this requirement sample, all missions are at the same event priority level, that of routine normal science priority.

From the perspective of any individual mission, it is not the total scheduled/requested that matters, but their own individual mission's level. In this baseline scenario, 4 of the 34 missions received 80% or less of their requested time, while one received less than half. So one of the questions we address is how to keep some missions from a proportionally greater impact, while satisfying all mission's requirements to the greatest degree possible.

To see the effect of the 'most constrained first' aspect of the baseline strategy, we removed that and scheduled all requirements at equal priority (breaking ties randomly). The net effect is as would be expected: some requirements with lots of flexibility consume places needed by more constrained requirements, and so more of the latter remain unscheduled. The overall total scheduled/requested drops to 82%, and 5 missions receive 80% or less of what they requested, with three receiving less than 50%.

We also looked at the impact of separating out prime missions from extended ones in the prioritization, assigning prime missions a higher priority. The three prime missions represented 12% of the total time requested, and when given a higher priority, they received virtually all that they requested (99.9%). However, there was a larger impact on the overall total scheduled time (reduced from 88% to 83%): of the extended missions, 5 received 80% or less of their requested time, and two receiving 50% or less. Therefore, the impact of prime vs. extended missions is not clear-cut, since including them at higher priority significantly drives down the total time scheduled, to the detriment of all.

Priority Tiers

From the observed behavior of mission users to accept less time than originally requested, it is clear that what is submitted by many users as *required* is not truly required: it represents a *desired* level of time allocation, and can be reduced as circumstances warrant to fit with everyone else in the same week. This is borne out by the BOP's strategy when working on a week: the first step is to cut virtually all missions back to "typical" levels and then to tweak and optimize the resulting schedule. In general, all users accept these cuts without complaint, and spend the negotiation period fine-tuning the resulting allocations, and attempting to horse-trade with other users to make incremental improvements. The fact that the required submission is, in fact, flexible, is not specified by users in their inputs to the scheduling process. We will return to this point later as it has a major impact on potential solutions.

Based on this observation, we considered how to modify the process if users did specify at least the *relative* priority of their own inputs. This suggested a tiered input approach, with users dividing their requirements up into tiers as illustrated in Fig. 4. We chose a simple scheme where levels 1 and 2 reflected elevated priority requirements, level 3 corresponded to normal priority, and levels 4 and 5 were desirable if possible. As an example, one mission had requirements on certain days of the week that were essential in order to upload commands to the spacecraft for the following week. For this mission, meeting those specific requirements was much more important that others in the same week, which could be more readily reduced or even occasionally dropped.

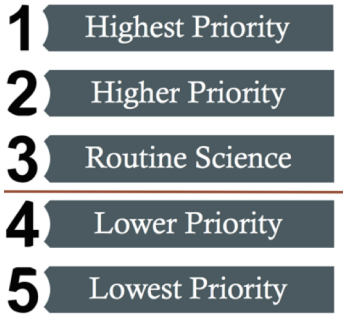

Fig. 4. A tiered priority scheme for user-specified requirement priority. Levels 1-3 are considered "must have", levels 4 and 5 are "desired if possible".

Generating an accurate dataset with this information is very difficult, so in the absence of real user inputs, we arbitrarily divided each mission's inputs into these 5 levels, with an even distribution of time across the levels. Each level was scheduled in priority order for all missions, with the results fixed when lower priority levels were considered (i.e., level 1 was scheduled for all missions, then level 2 added, and so on). The results showed that nearly all missions received 90-

100% of their level 1 and level 2 inputs, and 75% received 90-100% of the level 3 inputs. The levels that received less that half of the time requested were almost all from levels 4 and 5. However, the overall total time scheduled was still reduced to about 84%. It remains an open question whether that fact that more missions got more of their "highest priority" requests would balance out this reduced overall scheduling efficiency.

We explored some alternatives to the 5-level tiers, e.g. a 2-level tier with 80% of the time in a higher priority level, and the remaining 20% as lower priority. The results showed a slight improvement on the overall total scheduling efficiency, to 85.5%. On the other hand, the 5-level tiers provide more granular information as to how users place a relative value on the time they are receiving, and so could be the most useful in improving the automation process.

It is worth noting that while much more information about alternative requirements could be asked of users, it would place a significant additional workload on them to provide information that might not be used. For example, users could specify preferences for shrinking or dropping certain requirements, depending on which other requirements are satisfied in the schedule. Specifying these inputs could become complicated and time consuming, and would only be used if the dependency circumstances were realized. The notion of simply adding one additional relative priority field to a requirement would certainly be manageable, as well as being directly usable by the software, the BOP, and other schedulers during the negotiation phase.

Enforced Input Reduction

Another tactic to reduce the high level of oversubscription is to require users to reduce their inputs to a level that they have historically been shown to accept. We tried to simulate this by taking historical reduction results, and then arbitrarily cutting mission inputs to correspond to what each mission had found acceptable over a 6-month period around the time of our experiment 16-week time range. We only reduced the heaviest users (those requesting 40 or more hours of tracking time per week, a total of 14 missions). Reduction percentages varied over a significant range, with 7 missions receiving reductions over 20%, with the the largest reductions of about 35%. The results showed 26 of the 31 missions receiving over 90% of their requested time, with only one mission receiving less than 50%. The total input time was reduced by about 13% overall, and so the overall time scheduled was reduced accordingly.

Because our experimental reduction was arbitrary, it does not reflect the specifics of what each mission might consider essential. This would be very difficult to determine post facto to make the experiment more realistic. On the other hand, it would not be a large burden on users to ask them to fit their input requirements within an overall time cap. Furthermore, the option would remain to add back in additional requirements if it turned out that there remained opportunities to do so during the negotiation phase.

Squeaky Wheel Optimization

An optimization technique that has been used with good results on other oversubscribed scheduling problems is that of Squeaky Wheel Optimization (Joslin and Clements, 1999); see also (Barbulescu et al., 2006a, 2006b; Laura Barbulescu et al., 2004; L. Barbulescu et al., 2004; Kramer et al., 2007). In this approach, requirements are assigned an initial priority and then scheduled in priority order, then the priorities are adjusted until no further improvement in the objective function are observed. We applied this technique to the DSN scheduling problem in the following way. The objective we used was the overall total scheduled time for all missions. Note that this does not take into account that some missions might perform relatively poorly, even though the overall total scheduled time is better. This is an area for further work.

For the initial priority assignment, we tried different techniques, including: random; smallest requested time; and largest requested time. We found the best results using largest request time, likely because the larger users tend to dominate the objective function if they can be scheduled earlier and get a larger fraction of the time they request.

We adjusted the priority after each iteration by looking at which mission had the worst ratio of unscheduled to requested time, and swapping places with the next higher priority mission (Fig. 5). If the overall schedule did not improve, we restored the swap and tried instead the next worst, and so on. We terminated a run when there are no places left to swap without making the schedule worse.

For two missions, we found that unless they were excluded from the iteration process, they would invariably receive no time. Both were relatively small and so were left fixed as first and second priority in the list.

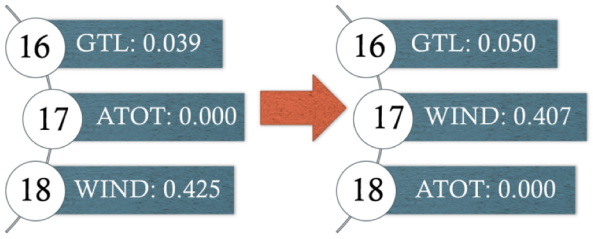

Fig. 5. Illustrative behavior of Squeaky Wheel Optimization in exchanging the priority of two missions (here ATOT and WIND) in order to achieve an improvement in the overall total time scheduled. Each mission is assessed using the ratio of unscheduled to requested time.

The results were encouraging, in that the best runs with SWO were able to schedule 91% of the total requested time, an improvement over the 88% found by greedy least constrained first.

5. Conclusions

The results of the experiments reported above are very encouraging in suggesting several promising directions to help address the DSN oversubscription problem:

1. Time reduction: use of this mechanism would require users to submit less "required" time in order to define a weekly scheduling pool that more nearly matches the antenna time available to be allocated. This requires a policy change, and leaves open the thorny question of how to set the appropriate restriction level per mission. Depending on the mission phase and the occurrence of various science and engineering events, there can be a significant variation from week to week, and so setting and policing this constraint could be burdensome. A further complication is the alignment of mission visibility periods at certain times during the year, which leads to some times being oversubscribed while others are unusable. Thus the target time to reduce to meet the objective of managing oversubscription is difficult to evaluate.

In spite of these considerations, some form of required time reduction is likely, and the most promising approach is to use the DSN long-range planning and forecasting software (Johnston et al., 2012) to set appropriate limits on how much time can be specified as "required" in the top priority tiers. This software will work from a long-range specification of requirements and can look ahead multiple years to assess oversubscription and contention due to overlapping critical events. Resolving contention far in advance could lead to "fair" and agreed input levels for the mid-range scheduling process.

2. Tiered relative priorities: this would allow users to explicitly specify how important are their different categories of requirements, knowledge which currently resides only in textual descriptive material or in the schedulers' heads. This approach could be readily combined with (1) time reduction, in that the total time in the top tiers could be restricted, while the lower priority requirements could be provided to take advantage of opportunities if they are available.

3. Squeaky Wheel Optimization with internal priorities: the use of a mission-level priority scheme does not lend itself to the DSN due to the high level of oversubscription and to the time variation in mission activity and corresponding requirements. However, the use of an *internal* and *dynamic* priority list does work well to improve the overall schedule efficiency while avoiding starvation of any mission due to being stuck in a low position on a static list. SWO could be combined with (2) tiered relative priorities to define an objective per mission to reflect the importance of meeting each mission's designated top priority activities, while attempting to fit in lower priority activities. It could also be combined with (1), time reduction to a fair level, in that the top tiers could be constrained to fit within agreed up (historical or forecast) limits on the available antenna time.

Currently, DSN is evaluating potential policy changes that would enable implementation of an approach like that described above. Further investigations will address how best to combine these approaches in a flexible but effective manner.

Acknowledgements: the authors are grateful for comments and suggestions from a wide range of participants in the DSN scheduling process, including mission personnel, scheduling team members, and the DSN scheduling office. We also acknowledge very useful suggestions from anonymous referees.

References

Barbulescu, L., Howe, A.E., Whitley, L.D., Roberts, M., 2006a. Understanding algorithm performance on an oversubscribed scheduling application. J. Artif. Intell. Res. 27, 577–615.

Barbulescu, L., Howe, A.E., Whitley, L.D., Roberts, M., 2004. Trading Places: How to Schedule More in a Multi-Resource Oversubscribed Scheduling Problem.

Barbulescu, L., Howe, A., Whitley, D., 2006b. AFSCN scheduling: How the problem and solution have evolved. Math. Comput. Model. 43, 1023–1037.

Barbulescu, L., Whitley, L.D., Howe, A.E., 2004. Leap before you look: An effective strategy in an oversubscribed scheduling problem.

Carruth, J., Johnston, M.D., Coffman, A., Wallace, M., Arroyo, B., Malhotra, S., 2010. A Collaborative Scheduling Environment for NASA's Deep Space Network. Presented at the SpaceOps 2010, Huntsville, AL.

Imbriale, W.A., 2003. Large Antennas of the Deep Space Network. Wiley.

Johnston, M.D., Tran, D., Arroyo, B., Call, J., Mercado, M., 2010. Request-Driven Schedule Automation for the Deep Space Network. Presented at the SpaceOps 2010, Huntsville, AL.

Johnston, M.D., Tran, D., Arroyo, B., Sorensen, S., Tay, P., Carruth, J., Coffman, A., Wallace, M., 2014. Automated Scheduling for NASA's Deep Space Network. AI Mag. 35, 7–25.

Johnston, M.D., Tran, D., Arroyo, B., Sorensen, S., Tay, P., Carruth, J., Coffman, A., Wallace, M., 2012. Automating Mid- and Long-Range Scheduling for NASA's Deep Space Network. Presented at the SpaceOps 2012, Stockholm, Sweden.

Joslin, D.E., Clements, D.P., 1999. Squeaky Wheel Optimization. J. AI Res. 10, 353–373.

Kramer, L.A., Barbulescu, L.V., Smith, S.F., 2007. Analyzing basic representation choices in oversubscribed scheduling problems. Presented at the 3rd Multidisciplinary International Conference on Scheduling: Theory and Application (MISTA-07), Paris, France.