Enhanced Redundant Scheduling Capability for Low Cost Ground Station Networks

Marco Schmidt and Klaus Schilling
University of Wuerzburg
Wuerzburg (Germany)

Abstract

Scheduling in ground station networks is an important issue to achieve a high utilization of resources. To satisfy the special needs of academic ground station networks, the CUSS scheduling system was implemented, which has the capability to schedule redundant contact windows utilizing a newly developed approach. We refer to the problem of scheduling redundant contact windows as the *Redundant Request Satellite Scheduling* problem (RRSS) (Schmidt and Schilling 2009b). It was shown, that the CUSS scheduling system is a reasonable tool for improving the efficiency of ground station networks by distributing contact time fairly between partipants. This work describes in detail the redundant scheduling mechanism of the CUSS system, realized through a tailored objective function for contact window assignment. The objective function contains a penalty value to avoid unfair distribution of contact windows. Beside the presentation of the objective function itself, the paper proofs how the penalty value guarantees a fair redundancy distribution.

Introduction

In the recent years many projects from the small satellite community started to built ground station networks with the aim to enhance satellite operation through resource sharing. A crucial issue in this respect is satellite scheduling to achieve a high utilization of the resources. This work deals with a special scheduling algorithm designed for the special needs of these ground station networks.

This paper is related to the redundant scheduling capability of the CUSS scheduling system. Redundant scheduling means, that the scheduler can assign more than one contact window to a request, as it is desired mainly in academic ground station networks (for further information to this specific aspect in academic ground station networks refer to (Schmidt and Schilling 2008)). The redundant scheduling algorithm was designed to equally distribute additional contact windows, this is guaranteed by the objective function γ. The γ function consists of two subfunctions γ_1 and γ_2, which are added to evaluate a certain schedule. While γ_1 is only a weighted sum of priorities, the γ_2 function (called penalty function) is a more complex. The γ_2 equation is responsible for adding a penalty value if the redundant contact windows are not equally distributed. This paper deals especially with the behavior of the penalty function and how an equally distributed redundancy is achieved.

The paper is arranged as follows: The next section introduces some basic definitions, which are important to understand the principle of the CUSS scheduling procedure. The penalty function itself is described main section, its behavior for a typical scheduling scenario is handled afterwards. In the last section a short conclusion is presented.

Definitions

Before a detailed description of the redundant scheduling mechanism of the CUSS system can is presented, a few definition need to introduced, to sketch the necessary background. The different concepts behind the following equations and parameters are only described cursory, for a more extensive description with more relation to the scheduling problem itself please refer to (Schmidt and Schilling 2009b). Scheduling in the scope of satellites is in general the problem to assign an amount of ground stations to an amount of satellites. In the case of the Redundant Request Satellite Scheduling (RRSS) problem, the participants submit requests for contact windows with satellites. These satellites are able to communicate with a number of ground stations, which are connected in a network. The complete mathematical description of these entities (users, ground stations, satellites) can be retrieved from (Schmidt and Schilling 2009b). Basic entities are the requests $R_i, i = 1...p$, representing a desired communication to a specific satellite. Furthermore, the users define in the requests the earliest start time and a latest end time for which this request is valid. Additionally, the user specifies the duration of a requested contact window (in minutes). In this work the term contact window C_{ij} is used to represent a time interval, which is available for a communication between a satellite and a ground station.

$$C_{ij} = \{R_i, t_{AOS}, t_{LOS}, G\} \tag{1}$$

The subscript i is a reference to the corresponding request R_i, the subscript j is numbers the contact windows in ascending order. Each contact window C_{ij} is furthermore associated with a ground station G and an start (t_{AOS}) and end time (t_{LOS}). It is important to note that normally a contact

window can be determined by its orbit geometry and the location of the ground station, which therefore is fixed with respect to start and end time of the contact window. In this work we refer to a contact window as a requested time interval from an user, which is placed inside the time period the satellite is in contact with the ground station, therefore it is more flexible and it can be shifted within a contact period, as the contact period can be greater than the requested contact window. For simplification we assume for the further text that the complete contact window is used for assignment. Additionally we define for each contact windows an attribute C_{ij}^b, it is defined as

$$C_{ij}^b = \begin{cases} 1, \text{ if } C_{ij} \text{integrated in final schedule} \\ 0, \text{ otherwise} \end{cases} \quad (2)$$

and can be used from the scheduler to indicate if a contact window is scheduled. A schedule is a set of contact windows, which are determined from the different requests R_i and from the satellites orbit geometry. This calculation is done automatically and determines a set of n contact windows, which should be integrated in a schedule. As it is not possible to integrate all contact windows due to overlappings into the schedule, a search algorithm tries to identify a subset of possible contact windows and marks these as "scheduled". So, if we use the term schedule, we normally refer to all scheduled contact windows. The CUSS scheduling system (Schmidt and Schilling 2009a) distincts between assigned and unassigned contact windows through the attribute "scheduled".
In classical scheduling problems the aim is to find a schedule which satisfies as many requests as possible (Barbulescu et al. 2002) (Barbulescu, Kramer, and Smith 2007). A request is satisfied, as soon as one contact window was assigned to the specific request. In the RRSS problem, a different approach is followed, as the requirements in academic ground station networks differ a lot. Here it is very often desired to have more than one contact window assigned to a given request. That means that a request is not automatically satisfied if it receives one contact window, it is possible to assign several redundant contact windows to a request (further information can be found in (Schmidt, Rybysc, and Schilling 2008)). This brings us to the definition of the term *equal distribution of redundancy*: If it is possible to assign more than one contact window to one request, the additional assigned contact windows should be distributed equally over the different requests. The redundant scheduling capability is the main contribution of this paper and is handled in depth in this work.

The penalty function

Core part of the scheduling system is the objective function γ, which is capable of redundant scheduling. This redundant scheduling capability is implemented by the penalty function γ_2, which is subtracted from γ_1 to achieve an equal distribution of redundancy. The penalty function γ_2 is formulated for a given set of request R_1,..,R_i as

$$\gamma_2 = \sum_{1 \leq k \leq i} \left(\lambda^{\kappa(R_k)} \right) \quad (3)$$

where λ is a positive integer > 1. In the further report λ is set to a value of 3, this value was determined in empirical tests as a reasonable choice (λ influences basically the weight of the penalty function on γ_1). The function $\kappa(R)$ is defined as

$$\kappa(R_i) = R_{max}^b - R_i^b \quad (4)$$

with

$$R_i^b = \sum_j (C_{ij}^b) \quad (5)$$

R_i^b describes the actual number of assigned contact windows C_{ij} for request R_i in a schedule. The value R_{max}^b describes the maximum number of assignable contact windows for one request, evaluated over the complete set of requests $R_1, .., R_i$. So R_{max}^b is a specific constant for a set of given requests: I.e from the set of given requests (R_1,..,R_i), one request has a maximum number of assignable contact windows, these amount of contact windows is defined as R_{max}^b. Therefore it is obvious that $\kappa(R_i)$ will be always ≥ 0 as the actual amount of assigned contact windows of R_i will never be greater than the maximal amount of assignable contact windows over all requests. As these definitions are used quite often all over the report, the meaning of them is explained in an example:

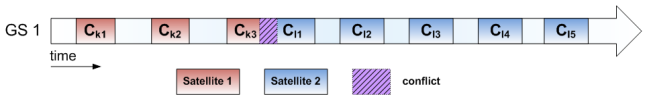

Figure 1: Example with two requests

Lets consider two request, R_k with 3 possible contact windows (C_{k1}, C_{k2}, C_{k3}) and another request R_l with 5 possible contact windows ($C_{l1}, C_{l2}, C_{l3}, C_{l4}, C_{l5}$). One conflict can be observed between R_k and R_l, their contact windows C_{k3} and C_{l1} are overlapping and only one of these two contact windows can be assigned on the final schedule (see figure 1). In this scenario the constant R_{max}^b is equal to 5, as request R_l has the maximal number of 5 assignable contact windows. For the final schedule two options are possible:
Case 1: R_k gets all of his 3 available contact windows assigned, so $R_k^b = 3$ and $\kappa(R_k) = 5 - 3 = 2$. Due to the conflict in one of the contact windows, only 4 windows can be assigned to R_l, which results in $R_l^b = 4$ and $\kappa(R_l) = 5 - 4 = 1$ (see schedule option 1 in figure 2). The penalty value for this schedule would then be calculated as $\gamma_2 = \lambda^2 + \lambda^1 = 12$.
Case 2: The conflict will be resolved with the other option, R_k receives only 2 contact windows, therefore 5 contact windows will be assigned to R_l (see schedule option 2 in figure 2). The parameter R_{max}^b is again equal to 5 (this

value is a constant for a given set of requests). Now we can calculate $\kappa(R_k) = 5 - 2 = 3$ (only 2 assigned contact windows for $R_k \Rightarrow R_k^b = 2$) and $\kappa(R_l) = 5 - 5 = 0$ (because 5 assigned contact windows for $R_l \Rightarrow R_l^b = 5$). As a result γ_2 is determined as $\lambda^3 + \lambda^0 = 28$.
As the penalty function γ_2 will be subtracted from γ_1, the smaller value determines the better schedule, here it is schedule option 1 (case 1 with penalty 12). The explanation for that is quite simple, the schedule option 1 is preferred because the distribution of contact windows is more equal. So in this scenario it is clear that schedule option 1 is preferred, due to its lower penalty value γ_2. The question is now, if the penalty function γ_2 is suitable in general to achieve an equal distribution of redundancy.

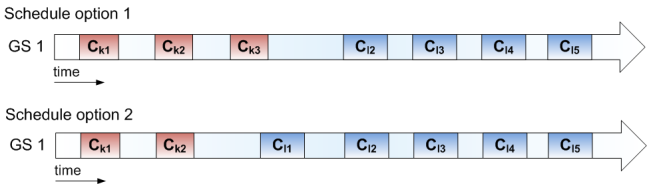

Figure 2: Schedule options for the example

In the next section it is shown, that the penalty function γ_2 can be used for any combination of two requests R_i and R_j, that γ_2 will be in any case minimal if the redundant contact windows are equally distributed.

Behavior of the penalty function for two arbitrary requests

The aim of this section is to prove that the penalty value γ_2 is minimal, if two requests R_i and R_j have equally distributed redundant contact windows. Two arbitrary requests R_i and R_j are assumed, with a number of n, respectively m, contact windows available. The proof can be divided in two distinct cases:
Case 1: Both requests R_i and R_j have no overlapping contact windows. This case can be handled quite simple, because the absence of conflicts means, that the maximum amount of contact windows can be assigned ($R_i^b = n$ and $R_j^b = m$, both are maximal), therefore $\kappa(R_i)$ and $\kappa(R_j)$ are both minimal. Thus, also the result of the penalty value γ_2 is minimal.
Case 2: Both requests R_i and R_j have at least one conflict, i.e. overlapping contact window. Thus, for a conflicting pair of contact windows, only one of these will be assigned in the final schedule. Lets assume w.l.o.g. that the request R_i gets in total k contact windows, which leads to $\kappa(R_i) = R_{max}^b - k = a$. Request R_j on the other side receives l contact windows, resulting in $\kappa(R_j) = R_{max}^b - l = b$, with $a, b \geq 0$. The penalty for this assignment is calculated as

$$\gamma_2 = \lambda^a + \lambda^b \tag{6}$$

When the assignment of contact windows is changed by transferring exactly one conflicting contact window to R_j instead of R_i, the penalty function will change to

$$\gamma_2 = \lambda^{a+1} + \lambda^{b-1} \tag{7}$$

The explanation is, that an additional contact window for R_j decreases the κ value by 1. Equations 6 and 7 are now used to derive how the penalty function behaves in general, when the contact window assignment is changed. The following inequality is the starting point:

$$\lambda^a + \lambda^b < \lambda^{a+s} + \lambda^{b-s} \tag{8}$$

It will be shown here, that for a given schedule the penalty increases, when the distribution of contact windows is changed in an unfair assignment. This is expressed in equation 8, where the penalty value of a schedule should be greater ($<$), if an unfair distribution of contact windows was created by removing a contact window from request R_i to R_j. Furthermore it will be proven, that equation 8 holds, if a fair distribution of redundant contact windows was integrated in the schedule. In the first step, the inequality in equation 8 is modified to

$$\lambda^a \cdot (1 - \lambda^s) < \lambda^b \cdot (\lambda^{-s} - 1) \tag{9}$$

furthermore holds

$$\lambda^{a-b} \cdot (1 - \lambda^s) < (\frac{1}{\lambda^s} - 1) \tag{10}$$

and finally is formed to

$$\lambda^{a-b} \cdot (1 - \lambda^s) < (\frac{1 - \lambda^s}{\lambda^s}) \tag{11}$$

the term $1 - \lambda^s < 0$, as the parameters $\lambda > 1$ and $s > 0$. The last step is

$$\lambda^{a-b} > \frac{1}{\lambda^s} \tag{12}$$

To show under which condition inequality 12 is satisfied, one has to distinguish three further cases:
Case 2.1: $a > b$. Then inequality 12 is always true, because $\lambda > 1$ and $s > 0$. This means, if request R_i has less contact windows assigned than R_j (which is equal to $a > b$), then a further decrease of contact windows for R_i (is equal to an increase of a) leads automatically to a higher penalty, due to the simultaneously increase of contact windows for R_j. Or in other words, if R_i has already less contact windows than R_j, the distribution will be even more unfair when additional contact windows will be assigned to R_j instead to R_i
Case 2.2: $a < b$. When we assume that a is smaller than b, equation 12 can be modified to

$$\frac{1}{\lambda^{b-a}} > \frac{1}{\lambda^s} \tag{13}$$

which holds for the case that $a - b < s$. This means, that the amount of s contact windows can be assigned to R_j instead R_i without an increase of the penalty value. $a < b$ means, that R_j has less contact windows assigned than R_i, i.e. the redundancy is not yet equally distributed. By moving one contact window from R_i to R_j, the schedule becomes more fair, the penalty value decreases.

Case 2.3 $a = b$: Then inequality 12 is again always true, as $1 > \frac{1}{\lambda^s}$. So, if R_i and R_j have an equal amount of contact windows assigned, any further rearranging of assigned contact windows will raise the penalty due to unfair distribution of contact windows.

It was shown in this section, that the penalty function γ_2 can be used to prevent an unfair distribution of redundant contact windows. A minimal value for γ_2 is always achieved, when redundant contact windows are equally assigned over two requests.

Redundancy distribution for more than two arbitrary requests

Before proving the correct behavior of the penalty function for more than two requests, a formal definition of *equal distributed redundancy* with respect to a schedule is necessary. When considering only two requests, an equal distribution of redundant windows is simply achieved by balancing the number of assigned, conflicting contact windows. But how can equal distribution be assessed for more than two requests? This is achieved by introducing the term *distance*: The *distance* of schedule σ (see equation 14) refers to the maximum number of assigned contact windows r_{max} (with respect to all requests from schedule σ) minus the minimum number of assigned contact windows r_{min}(considering all requests from schedule σ).

$$Dist(\sigma) = r_{max} - r_{min} \tag{14}$$

$$r_{max} = \max\left(R_i^b, \forall i\right) \tag{15}$$

$$r_{min} = \min\left(R_i^b, \forall i\right) \tag{16}$$

Using this equation makes it possible to define that the redundancy of schedule σ_1 is more balanced than schedule σ_2, if distance of schedule σ_1 is smaller than the distance of schedule σ_2

$$Dist(\sigma_1) < Dist(\sigma_2) \tag{17}$$

As this definition seems not so clear from the first impression, an example is given for illustration.

Equal redundancy distribution for an example scenario

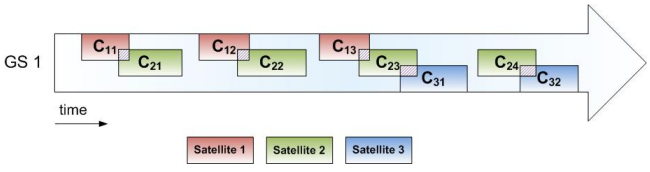

Figure 3: Example Scenario with 3 requests

The scenario in figure 3 depicts a situation with three requests. The conflicts between these requests are expressed with the overlapping, hatched areas. Therefore, it is not possible to assign all 9 available contact windows. Two schedules, which could be a result from the scheduling process are shown in figures 4 and 5. It can be seen quite easily, that the amount of assigned contact windows for schedule σ_1 and σ_2 are equal (5 in total). The question is now, which of these schedules better satisfies the criterion of equal distributed redundancy? From the upper definition of *distance* this is the case for schedule σ_2, because $Dist(\sigma_1)$ can be calculated as the maximum amount of contact windows $r_{max} = 3$ (satellite 2) minus the minimum amount of contact windows $r_{min} = 1$ (satellite 1 and 3), so $Dist(\sigma_1) = 3 - 1 = 2$. The same evaluation for schedule σ_2 leads to a value of $Dist(\sigma_2) = 2 - 1 = 1$ (satellite 1 and 3 have two contact windows assigned). Therefore, schedule σ_2 has a fairer distribution of redundant contact windows. It has to be shown now, that the penalty value is minimal, if the distance in a schedule with more than two requests is minimal.

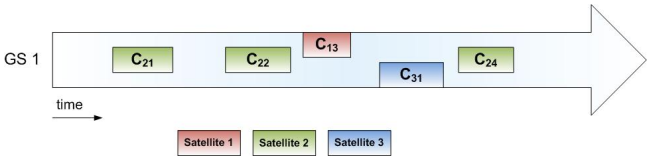

Figure 4: Schedule σ_1

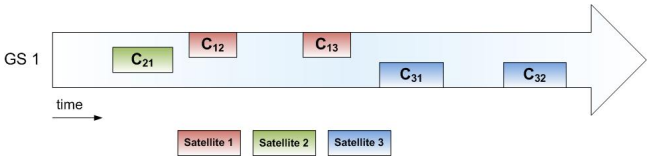

Figure 5: Schedule σ_2

Relation between the penalty function and distance

With the mathematical formulation of the term *distance* (see equation 14) it will be proven now, that the penalty function is minimal, if the *distance* of a schedule is minimal, i.e the redundancy is equally distributed for a given problem instance with more than two requests. In a real world scenario, each orbit results in an unique pattern of available contact windows, thus the number of available contact windows for each request is different. The interesting question is then, how the total amount of assigned contact windows interacts with the objective to equally distribute redundant contact windows. Therefore, two cases are distinguished to prove that the objective function behaves correct, i.e. it is minimal if the redundancy is equally distributed between the requests:

Case 1: Schedules σ_1 and σ_2 have the same total amount of n assigned contact windows. To prove that the penalty function behaves correctly, it has to be shown that the penalty value of schedule σ_2 is smaller than the penalty

value of schedule σ_1, if the distance of schedule σ_2 is smaller than the distance of schedule σ_1. Mathematically formulated:

$$\gamma_2(\sigma_2) < \gamma_2(\sigma_1) \tag{18}$$

$$\text{if} \quad Dist(\sigma_2) < Dist(\sigma_1)$$

These equations express the intuitive expectation, that the penalty value is smaller, if the redundancy is fairly distributed, i.e. the distance between the schedules is smaller.

Case 2: Schedule σ_1 has in total n contact windows assigned, Schedule σ_2 has m contact windows assigned, with $n \neq m$. For this case it has to be shown, that there exists at least one penalty value for schedule σ_2, which is smaller than the penalty value of schedule σ_2, if $n < m$. Expressed in equations:

$$\exists \gamma_2(\sigma_2) < \gamma_2(\sigma_1) \tag{19}$$

$$\text{if} \quad \sum_i \kappa(R_i) < \sum_j \kappa(R_j)$$

Note: Equation 19 expresses, that schedule σ_2 has more contact windows in total assigned than schedule σ_1. Intuitively one expects, that the penalty value for schedule σ_1 should be greater, as it has less contact windows in total assigned. But having more contact windows assigned does not necessarily imply a fair distribution of redundant contact windows, for example could many conflicts in the calculated contact windows result in a very unfair redundancy distribution. But when the conflicts are neglected, there should exist at least a schedule σ_2, which has a smaller penalty value then schedule σ_1, as it has more contact windows assigned than σ_1.

Proofs

This section proves the correct behavior of the penalty function for the two cases above:

Case 1: If $Dist(\sigma_2) < Dist(\sigma_1)$, then the penalty value $\gamma_2(\sigma_2) < \gamma_2(\sigma_1)$:

This proof uses the fact, that a decrease of $Dist(\sigma_1)$, automatically results in a decrease of the penalty $\gamma_2(\sigma_1)$. The penalty value of schedule σ_1 can be calculated by

$$\gamma_2 = \lambda^{a_1} + \lambda^{a_2} + \lambda^{a_3} + ... + \lambda^{a_n} \tag{20}$$

We assume without the loss of generality, that $a_2 = R^b_{max} - r_{max}$ and $a_1 = R^b_{max} - r_{min}$. That means $Dist(\sigma_1)$ is given by $a_2 - a_1$. If we reduce the distance of that schedule be removing one contact windows from R_2 to any other request, we obtain a new schedule σ_2, and the penalty is altered from $a_2 \rightarrow a_2 + 1$ and $a_n \rightarrow a_n - 1$. That means in equation 20 exactly two terms of the sum are altered, namely λ^{a_2} changes to λ^{a_2+1} and λ^{a_n} to λ^{a_n-1}. The inequality

$$\lambda^{a_2+1} + \lambda^{a_n-1} < \lambda^{a_2} + \lambda^{a_n} \tag{21}$$

is true when the redundancy is distributed fair. This means that also the penalty value for schedule σ_1 will decrease, as the following characteristic holds

$$\lambda^a + \lambda^b < \lambda^{a+s} + \lambda^{b-s} \tag{22}$$

Thus, $\gamma_2(\sigma_1)$ will be reduced by decreasing the distance by 1. Therefore, $\gamma_2(\sigma_2)$ is also smaller than $\gamma_2(\sigma_1)$, as $Dist(\sigma_2) < Dist(\sigma_1)$.

Case 2: For the second case it has to be shown now, that there exists at least one schedule σ_2, which has a smaller penalty value than schedule σ_1, if the total amount of assigned contact windows for schedule σ_2 is greater than the amount of assigned contact windows for schedule σ_1.
This is shown through proof by contradiction. If there would exist no $\gamma_2(\sigma_2)$, that is smaller than $\gamma_2(\sigma_1)$, the penalty value of schedule σ_1 is minimal and can be expressed as

$$\gamma_2(\sigma_1) = \lambda^{b_1} + \lambda^{b_2} + \lambda^{b_3} + ... + \lambda^{b_n} \tag{23}$$

The schedule is now modified to yield a new penalty value of

$$\gamma_2(\sigma_1') = \lambda^{b_1-1} + \lambda^{b_2} + \lambda^{b_3} + ... + \lambda^{b_n} \tag{24}$$

It can be seen easily, that $\gamma_2(\sigma_1') < \gamma_2(\sigma_1)$, because $\lambda^{b_1-1} < \lambda^{b_1}$. Furthermore is $\sum_i \kappa(R_i) < \sum_{i'} \kappa(R_i')$, as we changed the redundancy distribution from b_1 to $b_1 - 1$. This means a schedule σ_1' exists, which fulfills the requirements in equation 19 and is therefore a contradiction.

In summary show the proofs, that the proposed scheduling objective function γ_2 guarantees an equal distribution of redundant contact windows. While the γ_1 objective tries to maximize the number of contact windows included in the final schedule, prefers the γ_2 objective these schedules where a fair distribution of contact windows exists. This characteristic better satisfies the demands of the small satellite community.

Summary

So it has been shown in this section, that the penalty function γ_2 distributed equally the redundancy between the different requests. This behavior is very important, especially in the context of academic ground station networks, because the participants share their resources without commercial interest, therefore it is crucial to handle the different users equally. Of course the penalty function can not guarantee, that a schedule is always fair with respect to redundancy, as conflicts or orbit geometry affect the schedule in that way a lot. But the proofs have shown, that the aim to distribute the redundancy equally between requests is achieved by this objective function. The impact of the penalty function in the complete objective function can be changed by multiplying the result with a weight or by adjusting the parameter λ. Experiments with real world satellite data have shown, that the resulting schedules retrieved by the scheduler using the

objective function, satisfy the needs of academic ground station networks.

Conclusion

Satellite scheduling is an important issue in ground station networks to improve utilization and efficiency. The presented approach of redundant contact window scheduling can be used for scenarios where several participants have to share the resources from one ground station network. It was shown, that the objective function of the presented approach fully satisfies the requirements of the RRSS scheduling problem. Further work will concentrate on optimization of the search algorithms for the special case of redundant contact window scheduling.

References

Barbulescu, L.; A. Howe, A.; Watson, J.; and Whitley, D. 2002. Satellite range scheduling: A comparison of genetic, heuristic and local search. In *Seventh International Conference on Parallel Problem Solving from Nature*.

Barbulescu, L.; Kramer, L.; and Smith, S. 2007. Benchmark problems for oversubscribed scheduling. In *The 17th International Conference on Automated Planning & Scheduling*.

Schmidt, M., and Schilling, K. 2008. Satellite scheduling for educational ground station networks. In *IAC*, number IAC-08-B4.3.6.

Schmidt, M., and Schilling, K. 2009a. Evaluation of the cuss scheduling system with respect to real world scenarios. In *Proc. 60th International Astronautical Congress (IAC'09), Daejeon, Korea*.

Schmidt, M., and Schilling, K. 2009b. A scheduling system with redundant scheduling capabilites for ground station networks. In *IWPSS*.

Schmidt, M.; Rybysc, M.; and Schilling, K. 2008. A scheduling algorithm for ground station networks related to small satellites. In *SpaceOps*.