

A multichannel Satellite Scheduling Algorithm

J.S. Gilmore and R. Wolhuter
Department of Electronic Engineering
University of Stellenbosch
7600 Stellenbosch, South Africa
Email: jgilmore@dsp.sun.ac.za, wolhuter@sun.ac.za

Abstract—During the design process of the South African Sumbandilasat satellite, a requirement was identified for a store-and-forward type on-board communications payload. This will allow for connectivity with remote rural ground stations. Due to other systems design constraints, these communications were limited to a single, half-duplex channel with 9600 baud capacity. This experience highlighted the requirement for a multichannel payload, to cater for the rising need for higher bandwidth data transfer to- and from remote parts of Africa. To this effect, a new payload is being designed for a next generation satellite, offering a total of 23, 9600 baud communication channels, allowing multiple, simultaneous store-and-forward connections to ground stations. Due to the possible excess of ground stations over available channels, a scheduling problem occurs. This paper presents a scheduling algorithm to do satellite on-board scheduling for the ground station links. The aim of the algorithm is to schedule all ground stations in the available communication time window as fairly as possible and in accordance with individual ground station requirements.

I. INTRODUCTION

Some previous multi-channel satellites only acted as transmitters, where data could not flow from a ground station to the receiver. Ground stations would only tune to the specific channel's transmission frequency and receive the data transmitted [1]. This is used for broadcasts, where many receiving ground stations exist, but only one transmitting ground station. No previous multi-channel communication satellites are known to have been developed in Africa. What makes the African context different from other parts of the world, is how isolated some regions of the country are.

When there are many ground stations communicating with a satellite, these ground stations have to be scheduled. This paper researches a scheduling scheme by which all ground stations that connect, are allowed to communicate before the satellite passes out of communications range.

The satellite system is characterised to obtain the run-time parameters and boundary values of the system. A simulator is designed to simulate the satellite-ground station model. Scheduling algorithms are formulated and investigated in the simulator to benchmark their performance against some basic scheduling algorithm. These algorithms are compared to discover which algorithm performs the best.

Section II describes the details of the position and velocity of the satellite, and how these characteristics may be used to compute the ground station acquisition rate of the satellite. Section III defines the multichannel satellite scheduling problem in scheduling terms. Section IV describes the simulation model used to evaluate the scheduling algorithms on. Section V

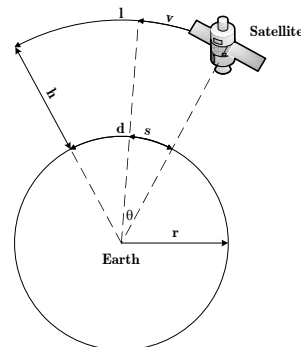


Fig. 1. Diagram of satellite orbit properties

describes a basic scheduling algorithm that may be used to solve the satellite scheduling problem and then goes on to describe possible improvements to this algorithm. Section VI presents the results of the simulations and compares the different scheduling algorithms. Section VII describes what work has been done in this paper and provides some future improvements that will be made.

II. SATELLITE SYSTEM CHARACTERISTICS

The satellite is in a polar sun-synchronous orbit [2] at a height of $h = 500$ km above sealevel. Such an orbit ensures that a satellite will pass through a certain horizontal line at approximately the same local time for each crossing. A satellite might pass over the equator twelve times a day, each time passing the equator at around 12:00, *local time*. The sun-synchronous orbit of the satellite ensures that the satellite travels over South-Africa at around the same time each day. The satellite completes one earth rotation in 100 min. The average radius of the earth is taken as $r = 6371$ km [3]. Figure 1 shows a diagram of the satellite orbiting the Earth.

From these values, the approximate velocity of the satellite may be calculated. If the earth is assumed to be circular, the distance the satellite travels in a day is given by the formula for the circumference of a circle:

$$\begin{aligned} p &= 2\pi(r + h) \\ &= 2\pi(6371 + 500) = 43\,172 \text{ km} \end{aligned} \quad (1)$$

Equation (1) gives the path length (p) of the satellite. The velocity of the satellite (v) may now be calculated using

equation (2).

$$\begin{aligned} v &= \frac{p}{t} \\ &= \frac{43\,172 \text{ km}}{100 \text{ min}} = 25\,903 \text{ km/h} \end{aligned} \quad (2)$$

Next, the velocity of the satellite's footprint, travelling along the earth's surface, can be calculated. This is done by projecting the satellite's velocity down to the surface of the earth. From figure 1, it can be seen that the arc the satellite makes when travelling through space, is subtended by an arc on the surface of the earth. The velocity of the satellite's footprint is then given by:

$$s = v \frac{d}{l} \quad (3)$$

Equation (3) gives the footprint's velocity (s) as the satellite's velocity (v), scaled by the lengths of the arcs l and d . Again looking at figure 1, from geometry:

$$\begin{aligned} \theta &= \frac{d}{r} = \frac{l}{r+h} \\ \text{therefore, } \frac{d}{l} &= \frac{r}{r+h} \end{aligned} \quad (4)$$

Substituting equation (4) into equation (3):

$$\begin{aligned} s &= v \frac{r}{r+h} \\ &= 25\,903 \left(\frac{6371}{6371 + 500} \right) = 24\,018 \text{ km/h} \end{aligned} \quad (5)$$

Equation (5) gives s in terms of v and the height of the satellite above the earth, where the radius of the earth is taken to be constant. This shows that the velocity of the satellite's footprint is not dependent on the length of the path it travels, which is to be expected. It also shows that the velocity of the satellite's footprint is not much smaller than that of the satellite itself. The reason for this is the low altitude of the satellite compared to the radius of the earth.

Using s , the ground station acquisition time (t_{aq}) may be calculated with

$$\begin{aligned} t_{aq} &= \frac{d_{RSA}}{s} \\ &= \frac{1000 \text{ km}}{24\,018 \text{ km/h}} \approx 2,5 \text{ min} \end{aligned} \quad (6)$$

Equation (6) gives the time which the satellite's footprint will take to travel $d_{RSA} = 1000 \text{ km}$. This is approximately the length the footprint will travel across South Africa.

The satellite-ground station negotiation time can be calculated using equation (7).

$$\begin{aligned} t_{x,min} &= \frac{h}{c} \\ &= \frac{500}{3 \times 10^5} = 1,67 \text{ ms} \end{aligned} \quad (7)$$

where $t_{x,min}$ gives the minimum travel time of a packet between the satellite, and a ground station directly below the satellite, and c is the speed of light in a vacuum. Satellite-ground station communications are line-of-site, i.e. the distance from the satellite to a ground station may be more than 500 km. Next

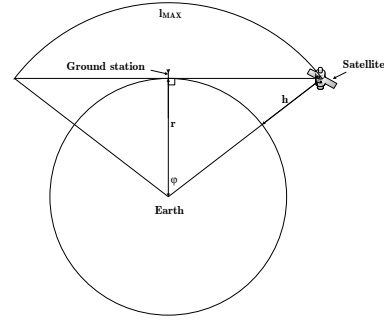


Fig. 2. Diagram depicting the line of site parameters

we calculate the maximum distance the satellite can be from a ground station.

Figure 2 shows the point at which a ground station will be able to initiate communications with the satellite. This is where the satellite intercepts the horizontal line, drawn from a ground station on the Earth's surface. This line is tangential to the circle. If this line is tilted, it will intercept the circle a second time, showing that there is no line-of-site to an object below the tangential line. The earliest point where there may be communication, is where the satellite intercepts the tangential line. Equation (8) now gives the angle ϕ , using trigonometry.

$$\begin{aligned} \frac{r}{r+h} &= \cos \phi \\ \frac{6371}{6371 + 500} &= \cos \phi \\ \phi &= 21,99^\circ = 0,3838 \text{ rad} \end{aligned} \quad (8)$$

Using the maximum communications range (δ_{max}) obtained from equation (9), the maximum packet travel time ($t_{x,max}$), can now be calculated using equation (10).

$$\begin{aligned} \delta_{max} &= r \tan \phi = 2573 \text{ km} \\ t_{x,max} &= \frac{\delta_{max}}{c} \\ &= 8,58 \text{ ms} \end{aligned} \quad (9) \quad (10)$$

The maximum round-trip-time (RTT) is then $RTT = 2 \times t_{x,max} = 17,16 \text{ ms}$.

The communication time window length (T_C), is the length of time that a ground station is in line of site contact with the satellite. To obtain this, the length the satellite travels through space in visible range of the ground station is used. This distance, combined with the velocity of the satellite gives T_C .

$$l_C = (r+h)\phi \quad (11)$$

$$\begin{aligned} T_C &= \frac{(r+h)\phi}{v} \\ &= \frac{5275 \text{ km}}{25\,903 \text{ km/h}} \approx 12 \text{ min} \end{aligned} \quad (12)$$

In equation (11), l_C represents the visible path length of the satellite. Equation (12) uses the visible path length of the satellite, along with the velocity of the satellite v , to calculate the maximum time that is possible for a ground station to communicate with the satellite. This is assuming the link budget is adequate.

III. MULTICHANNEL SATELLITE SCHEDULING PROBLEM

A. Scheduling problem notation

The scheduling problem notation given in [4] will be used. The notation has the form:

$$\alpha|\beta_0, \beta_1, \dots, \beta_i|\gamma$$

where α is the machine environment, β_i is the i th job characteristic and γ is the optimality criteria. The combination of α , β_i and γ in the notation uniquely describes a specific scheduling problem. For more information on sections III-B, III-C and III-D, see chapter 1 of [5].

B. Machine environment

A machine or server is the entity in a scheduling problem that services the job. Throughout this article the word “machine” is used interchangeably with “server”. The machine environment describes the characteristics of the servers in the scheduling problem. In single stage machine environments, jobs are only processed once. This is not to say that jobs may not be processed by more than one machine, but that no job, after being processed, needs to be processed again.

Two common single-stage machine environments are: “1”, there is only one machine and “ Pm ”, there are m parallel identical machines.

In the satellite system there are twenty-three channels, where all channels have equal capacity. In scheduling terms, this is a parallel machine problem where the channels are the machines and all machines are identical. The channels in which GSL’s must be scheduled are the servers of the system. The number of parallel machines is also a constant at $m = 23$. This shows that the machine environment is $P23$.

C. Job characteristics

Jobs are units of work that need to be performed. Jobs are usually entities that need to be serviced/scheduled by some server or machine. To understand how to better schedule jobs, certain job characteristics are defined to characterise different types of jobs. β defines the job characteristics in the scheduling notation.

Some common job characteristics are: “ r_i ”, each job has a specific release time, before which it cannot be scheduled. “ d_i ”, each job has a specific due date or deadline, after which it cannot be scheduled. “ $size_i$ ”, each job requires a certain number of servers or machines on which it must be scheduled simultaneously. “prec”, a precedence relation is given for each job, for example job i may only be scheduled after job x has completed. “pmtn”, job preemption is allowed, meaning execution of jobs may be halted and resumed later.

Ground stations are discovered by the satellite and these ground stations have certain properties. GSL properties include name, average connection time and the number of channels required. The ground stations that have to be allowed to communicate are the jobs. These ground station links are the jobs that need to be scheduled on channels.

The ground station links arrive over time and the order in which links arrive as well as the properties of these links are

assumed to be unknown before the actual arrival of the links. This assumption is made due to the fact that the orbit and position of the satellite is never precisely known by remotely situated ground stations.

This assumption creates the need for an on-line scheduler. An on-line scheduler learns of new jobs piece by piece. An on-line scheduler will continue to execute as long as there are new jobs arriving to be scheduled. In most real-life examples this is an ongoing process. An on-line scheduler has no knowledge of future jobs and thus cannot be made optimal [6]. In an on-line scheduler, there is always a release date r_i , before which time a job may not be scheduled.

A GSL may require more than one channel to communicate on. In scheduling terms it may be said that each job may require more than one machine. Problems of this type are called multiprocessor tasks. See ch. 11 in [5]. For the satellite scheduling problem, a ground station may require anything from one station to twenty-three stations, $size_i = \{1 \dots 23\}$.

Although it is possible to implement preemptive scheduling, the overhead switching costs will be too high and a lot of complexity will have to be added to the satellite’s administration communication system. In a satellite project, where radiation is a problem, adding over-optimised code to a system and thereby adding unnecessary complexity, is not considered good practice.

D. Optimality criteria

For every scheduling problem, some objective function must be minimised. How well the scheduling function is able to minimise the objective function is a measure of the performance of the scheduling function. It is possible to achieve an absolute minimum value for some combinations of objective functions and scheduling algorithms. When this can be done the scheduling scheme is said to be optimal.

In objective functions, C_i denotes the finishing time of job J_i and $f_i(C_i)$ denotes the associated cost. The three most common objective functions are:

$$\max\{C_i | i = 1, \dots, n\} \quad \sum_{i=1}^n C_i \quad \sum_{i=1}^n w_i C_i$$

The first one is the maximum completion time, also called the “Makespan”. The second one is the total completion time of all jobs, also called the “Total flow time”. The third one is the total weighted completion time of all jobs, also called the “Weighted total flow time”. Minimisation criteria for the satellite problem may be the flow time or the makespan.

E. Scheduling problem statement

All sections of the scheduling problem are now known. The full scheduling problem is then:

$$P23|r_i, size_i = \{1 \dots 23\}|C_{max} \quad (13)$$

Equation (13) is a summary of the different sections of the scheduling environment. Now that the problem is known, a solution may be found.

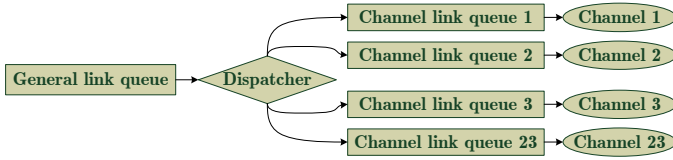


Fig. 3. Simulation diagram

IV. SIMULATION MODEL DESCRIPTION

The simulation model was implemented in a Java simulation environment called Desmo-J [7]. This environment contains a full suite of simulation structures to assist in implementing a simulation model. Desmo-J is a discrete simulation environment [8] where the user can use either event driven or process based simulation techniques to implement a simulation model. Process based simulation was used to implement this simulator in. This type of simulation consists out of various entities, all communicating with each other, and is best suited for multi-processor testbeds.

Figure 3 shows the implemented simulation model. The entities in the model are the channels, the dispatcher and the ground station links. Another entity, which does not appear in the model, is a GSL generator, to simulate GSL arrivals. The queues in the model are the general link queue, the channel link queues, the channel queues and the dispatcher queue. The channel queues and the dispatcher queue do not show up on the diagram, but these queues are used to record statistics on the idle times of the dispatcher and channels.

GSL's are created by the GSL generator and placed on the general GSL queue. When the GSL generator creates a GSL, it also defines its properties. These properties were described in section III-C. The values of these properties are sampled from some distribution, see section IV-A. The GSL generator only generates GSL's for a certain period of time, after which it switches off. This is to simulate the finite amount of time, a ground station is able to communicate with the satellite for, in one pass. The dispatcher removes the first item from the queue and places it in a channel queue, according to some scheduling algorithm. The dispatcher has to select both the channel queue, as well as the position in that channel queue. The time that the satellite requires to inform the ground station of the assignment is also taken into account. A channel is assigned to one specific channel queue. It selects a GSL from its queue and services that GSL. After a GSL has been serviced, it leaves the system.

If a GSL requires more than one channel, in the simulator, this is handled by copying the GSL process and scheduling all of these duplicate processes on different channel queues.

A. Random number streams

There are four random number streams (statistical distributions) used by the simulator. These are the interarrival time of GSL's, the average connection time of GSL's, the GSL negotiation time and the required number of channels by each GSL.

The GSL generator creates GSL's, at the times sampled from the "GSL interarrival time" time stream, with a required number of channels sampled from the "channel requirement"

time stream. When a GSL is removed from the general GSL queue, the dispatcher waits for an amount of time sampled from the "negation time" time stream, to simulate the time required for the satellite to inform the ground station of the channel assignment. When a channel services a ground station, the channel waits for the amount of time sampled from "average connection time" time stream, to simulate ground station communications.

The interarrival time stream is an Erlang distribution. If events occur at some average rate, that rate can be modelled as a Poisson distribution. The interarrival time of these events are modeled as an Erlang distribution [9]. The properties of this distribution were varied during simulation to increase or decrease the total amount of GSL's able to connect to the system.

The channel requirement stream is an integer exponential distribution, because the majority of the ground stations will require few channels, with only a small number of stations requiring more than one to three channels. The mean of this distribution was set to $\bar{X} = 4,5$.

The channel negotiation time stream is a Gaussian distribution, because there is a certain expected value. This expected value is the time it takes for a message to be transmitted down to earth and another message to be returned to the satellite. In section II, this value was computed as $t_{RTT} \approx 17$ ms. To obtain this type of response, a Gaussian distribution with $\bar{X} = 0,017$ and $\sigma = 0,0001$ was chosen.

The average connection time is a real exponential distribution, because ground stations are expected to usually only communicate for a short period of time. The mean of this distribution was set to $\bar{X} = 80$.

For more information on the types of distributions, see [10].

V. THE SCHEDULING ALGORITHM

To develop a scheduling algorithm, a base is needed to start from. The idea is to define a simple algorithm and then optimise all aspects of the algorithm to increase algorithm performance.

The main measures of performance for the algorithm are the average queue lengths, the maximum queue lengths, the average waiting time of GSL's and how well the algorithm avoids starvation of GSL's

To deal with multiple machines, there are in fact two scheduling schemes in the system. One scheme deals with channel assignments (level 1 scheduler) and the other scheme deals with determining the order in which GSL's assigned to the same channel will execute in (level 2 scheduler). There are twenty-three level 2 schedulers, one for each channel. There is one scheduling scheme for the level 1 scheduler and another for the twenty-three level 2 schedulers.

A basic level 1 scheduling technique that can minimise the flow time, if the correct level 2 scheduler is chosen, is to schedule the next GSL on the first available machine [11]. Usually this would mean having one queue of customers and when a server becomes available, another job is scheduled onto that server. In an effort to improve upon this basic technique, queues were placed before each channel. Now the

level 1 scheduler does not schedule the next job directly onto a channel, but rather hands that job to the correct level 2 scheduler. The level 2 scheduler then determines where to insert the GSL onto the channel specific queue. This helps in distributing the work and complexity of scheduling evenly.

The level 1 scheduler determines the next channel queue to schedule on, according to shortest length, removes the first element from the general GSL queue and hands it to the chosen channels's level 2 scheduler. If a GSL requires more than one channel, it duplicates the GSL's information, and hands all the copies to different level 2 schedulers. When there are no GSL's available, the level 1 scheduler blocks to avoid using unnecessary system resources.

A basic form of the level 2 schedulers, is to place all GSL's received from the level 1 scheduler into the channel specific queues, as if they were FIFO queues.

Improvements were made to both the level 1 and level 2 schedulers. The first improvement was to have the level 1 scheduler use the sum of average completion times, instead of the total length of the queue, to select the next queue. This improvement makes the level 1 scheduler more accurate in selecting the shortest channel queue. The reason being that the number of GSL's in the queue does not actually give the true length of the queue. The true length of the queue is measured in total communication time.

The second improvement was to have the level 2 scheduler sort the channel specific queue in a smallest communication time first order. The level 2 scheduler's job is to maintain the order of the queue after every insertion. This in effect implement a "shortest communication time first" scheduling scheme in the level 2 scheduler. This, along with the "first available machine" scheduling scheme of the level 1 scheduler solves the "Total flow time" optimally for an off-line scheduler [11]. As can be seen from the simulation results given in table I, it also improves the on-line scheduling scheme.

The third improvement was to change the channels' selection method. Instead always selecting the first element in the channel queue, they select the i th element in the queue according to an exponential distribution. This means that the first GSL will still be selected with the highest probability, but other GSL's may also be selected. This avoids starvation of large tasks.

VI. SIMULATION RESULTS

As stated in section II, the satellite acquires all ground stations in $t_{aq} = 2,5$ min of its pass over South Africa. Every ground station has $T_C = 12$ min of maximum communication time. The Erlang distribution modelling the GSL arrival rate was set so as to generate all GSL's in 2,5 min and then switch off.

The simulated channel queue length during one pass of the satellite is shown in figure 4. During the first 2,5 min, all the ground stations are discovered. The rate at which ground stations are serviced, may be obtained from the slope of the declining number of ground stations after the 2,5 min mark. The steeper this slope is, the better the scheduling algorithm performs.

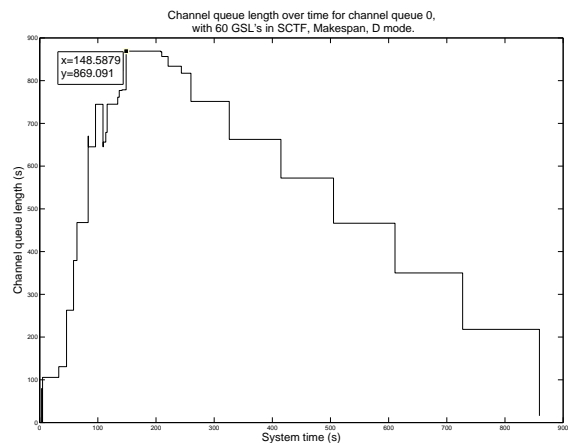


Fig. 4. Channel queue 0 length, with 60 GSL's in SCTF, Makespan, D mode

Table I displays the statistics for all the implemented scheduling schemes. These values were obtained using a maximum of 60 GSL's or $\lambda = 2$ and $\theta = 2,25$ for the Erlang distribution.

The maximum wait time is the average of the maximum wait times, over all the channel queues, that a GSL had to wait for. The average wait time is the average of the average wait times, over all the queues, that a GSL had to wait for. The completion time is the time it takes, from when the first GSL connects, to the last GSL being serviced. The simulation time was set as 12 min after the discovery of the last GSL. The stations still waiting to be serviced at the end of the simulation shows how many GSL's will still be waiting for service, on average on each channel queue, when the satellite goes out of range of the last ground station.

The modes are: "FIFO", the level 2 scheduler always puts ground stations at the end of the channel queue. "SCTF", the level 2 scheduler maintains a sorted queue of GSL's in the channel queue. "Q length", the level 1 scheduler uses the physical queue length to determine the channel queue to schedule a GSL on. "Makespan", the level 1 scheduler uses the total completion time of all GSL's on the queue to determine the channel queue to schedule a GSL on. "Deterministic", a channel always chooses the first GSL in the channel queue to service. "Stochastic", a channel chooses the next GSL to service from an exponential distribution that spans the average channel queue length.

From table I, the best time in which all the GSL's can be scheduled in is $950\text{ s} = 15,8\text{ min}$. This is longer than $T_C = 12\text{ min}$. This means there is not enough time to schedule all the ground stations in one pass of the satellite. To be able to schedule all the ground stations, the capacity of the scheduling algorithm should be determined. This is done by varying the parameters of the GSL arrival rate distribution.

All ground stations must be serviced at the end of $2,5\text{ min} + 12\text{ min} = 870\text{ s}$. From table I it is seen that the best performing algorithm is the "SCTF/Makespan/D" algorithm, with a average waiting time of 265 s. This algorithm has a completion time of 950 s. In other words, the time it would take for the

Mode			Maximum queue time (s)	Average queue time (s)	Completion time (s)	GSL's not serviced
FIFO	Q length	D	885	445	1340	2
FIFO	Q length	S	856	439	1230	2
FIFO	Makespan	S	849	430	1130	2
FIFO	Makespan	D	829	430	1080	2
SCTF	Q length	S	780	269	1240	1
SCTF	Q length	D	763	259	1120	1
SCTF	Makespan	S	778	277	1070	0
SCTF	Makespan	D	752	265	950	0

TABLE I
SIMULATION RESULTS FOR VARIOUS SCHEDULING SCHEMES USING 60 GSL'S

satellite to process 60 GSLs, is longer than the communication time window allowed by the orbit ($950\text{ s} > 870\text{ s}$). To be able to satisfy the required communication time window criteria, the number of ground stations allowed to connect, i.e. the capacity of the satellite, has to be reduced. By running simulations, the maximum number of stations that still satisfy the communication time window for the best performing algorithm, was found to be 47. The number of stations connecting were varied by varying the parameters of the Erlang distribution controlling the arrival rate of stations. At 47 GSLs, the Erlang distribution has the parameters $\lambda = 2$ and $\theta = 2,9$. The maximum queue time is then 622 s and the average queue time is 235 s .

When the results are compared, the simplest form of scheduling (FIFO and Q length) produce the schedule with the longest average and maximum queue times. Using makespan instead of queue length improves the maximum queue time by $6,3\%$, the average queue time by $3,4\%$ and the completion time by $19,4\%$. Using SCTF instead of FIFO improves the maximum queue time by $13,8\%$, the average queue time by $41,8\%$ and the completion time by $16,4\%$. All the improvements to the basic algorithm improves the maximum queue time by $15,0\%$, the average queue time by $40,4\%$ and the completion time by $29,1\%$.

No real improvement benefit can be seen from using a stochastic scheduling scheme, instead of a deterministic one. In this system, starvation cannot occur. All ground stations are discovered early on in the system life cycle and then they are serviced. No new ground stations enter the system after a certain point and thus a GSL requiring longer communication time will eventually be serviced. If it is possible to schedule all GSL's in the time window given, the GSL's with longer communication times will also be scheduled.

VII. CONCLUSION AND FURTHER WORK

It was shown that the ground station allocation problem can be written in scheduling terms and modelled in a Java simulation environment. The final scheduling algorithm can efficiently schedule GSL's on the satellite's communication channels. The simulation results look promising, with every improvement made in the algorithm showing visible improvements in the simulation. It also shows that it is possible to schedule ground station links in real-time, on a satellite.

Further work includes migration of GSL's from one channel queue onto another, to better distribute the load over all channels. Other distributions should also be investigated and measurements from actual satellite communication systems

can be used to determine the distribution of GSL arrivals. The scheme should also be changed to a weighted scheduling scheme, where station priority is taken into account. The metric to optimise would then be the product of the connection time and the priority. More exotic scheduling schemes should also be investigated.

REFERENCES

- [1] R. Goss, "Multichannel satellite communication and control system," U.S. Patent 5 121 409, 1992. [Online]. Available: <http://www.google.co.za/patents?hl=en&lr=\&vid=USPAT5121409>
- [2] R. J. Boain, "A-b-cs of sun-synchronous orbit mission design," in *AAS/AIAA Space Flight Mechanics Conference*, 2004.
- [3] W. M. Mularie, "Department of defense world geodetic system 1984, its definition and relationships with local geodetic systems," National Geospatial-Intelligence Agency, Tech. Rep., 2000.
- [4] R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan, "Optimization and approximation in deterministic sequencing and scheduling: A survey," *Annals of Discrete Mathematics*, pp. 287–326, 1979.
- [5] P. Brucker, *Scheduling Algorithms*, 5th ed. Springer-Verlag Berlin, 2007.
- [6] J. Y.-T. Leung, Ed., *Handbook of Scheduling: Algorithms, Models, and Performance analysis*. Chapman & Hall/CRC, 2004, ch. 15.
- [7] University of Hamburg. Desmo-j: Discrete-event simulation and modelling in java. [Online]. Available: <http://asi-www.informatik.uni-hamburg.de/desmoj/>
- [8] J. M. Garrido, *Object-Oriented Discrete-Event simulation with Java*. Kluwer Academic/Plenum Publishers, New York, 2001, ch. 4–5.
- [9] I. Angus, "An introduction to Erlang B and Erlang C," *Telemanagement*, no. 187, pp. 6–8, 2001.
- [10] P. Z. Peebles, *Probability, random variables and random signal principles*, 4th ed. McGraw-Hill, Inc., 2001, ch. 2.5.
- [11] V. T'Kindt, J.-C. Billaut, and H. Scott, *Multicriteria Scheduling: Theory, Models and Algorithms*. Springer, 2006, ch. 1.9.1.

John Gilmore was born on 17 June, 1985. He is currently a masters student at the University of Stellenbosch. He obtained his Electrical/Electronic Engineering (with Computer Science) degree, cum laude, from the University of Stellenbosch in 2007. His research interests include: Wireless communication networks and hardware system design.

Riaan Wolhuter has a B.Sc. B.Eng, M.Eng and a Ph.D. from Stellenbosch University, and a B.Sc(Eng)(Hons) from the University of Pretoria, in Electronic Engineering.

He is a senior researcher at the Faculty of Engineering at Stellenbosch University, South Africa.