AN EMULATOR FOR TESTING THE PERFORMANCE OF FRAMED ACCESS SCHEMES

Nedo Celandroni N.Celandroni@cnuce.cnr.it
Erina Ferro E.Ferro@cnuce.cnr.it
Francesco Potortì F.Potorti@cnuce.cnr.it

CNUCE, Institute of National Research Council
Via S. Maria 36 - 56126 Pisa - Italy
Phone: +39-050-593111 (operator)
Fax: +39-050-904052

ABSTRACT

We present the architecture of an emulator, named FRACAS (*FRA*med *C*hannel *A*ccess *S*imulator) which is suitable for testing the performance of DA-TDMA access schemes. The main strengths of the FRACAS architecture are its speed, extendibility, and the existence of a working prototype whose portable source code is freely available to students and researchers in the field of framed allocation methods. FRACAS was designed to compare different satellite access schemes, in a quick and accurate way. Anyway, it can be employed for the performance evaluation of any TDMA access method. The modularity of FRACAS facilitates the inclusion of new features. FRACAS users can choose the network configuration, the whole traffic load of the network, the traffic load of each individual station, the request and the allocation policy, and the statistics to collect. The validation test of our implementation of FRACAS is presented, comparing the results with those obtained in the real experimentation.

INTRODUCTION

In the last few years many new methods for accessing a data satellite channel in TDMA have been proposed in the literature [1-5, for example], alongside other older methods [6-10, for example]. Few of these have actually been implemented as prototypes, and even less have come onto the market. It is therefore difficult to compare their performances, mainly because of the lack of a common testbed for the various solutions. Our past experiences during the test phase of satellite access schemes in real environments [11] convinced us of the need for a simulation tool designed ad-hoc for evaluating the performance of satellite networks access schemes, that would enable us to define the network topology, the traffic carried out by each individual station, the allocation request policy, the bandwidth allocation policy, and the statistics to be collected. Such a tool would facilitate changing the allocation policy in order to make comparisons and find the most suitable one. The existing simulators which can be found in the literature(1) are generally bulky and expensive. Most of them are general-purpose tools, so they do not exploit the particular structure of TDMA access methods. Moreover, they are generally discrete-event simulators, i.e. they model a certain system as it evolves over time by a representation in which the state variables change only at a countable number of points in time. These points in time are the ones at which an event occurs, where an event is defined as an instantaneous occurrence which may change the state of the system.

We present the architecture of a high speed, lightweight emulator, named FRACAS, which is useful for analysing the performance of demand assignment access schemes that have some sort of frame structure. Therefore it is well suited for satellite TDMA access schemes, which are always framed. We describe a proof-of-concept implementation, which is completely functional and easy to extend. The prototype is a portable software package, written in C, and has been used in research projects [12, 13]. The allocation methods considered by FRACAS may have either centralised or distributed control. The stations receive data traffic from a number of different simulated sources, which are representative of the actual traffic. Also, the architecture makes it possible to adapt the simulation time and the accuracy of results to the user's purposes, both for a preliminary network dimensioning, and when a precise exploration of the network performance is required. FRACAS produces some key performance figures, which are useful for studying the behaviour of an allocation policy, for example average and peak packet delay, traffic impulse delay, and packet loss due to insufficient buffer space.

FRACAS was conceived in a research environment, with the goal of being highly specialised, and therefore efficient, so that it can be used even on CPU bound machines. Technical

---

(1) OPNET, BONeS, RESQ, and AMS, for example.

details on the FRACAS prototypal implementation are not reported in this paper. Interested readers can find them in [14].

FRACAS FUNDAMENTALS

FRACAS is a *discrete-time* emulator, so it is not based on events. In our prototypal implementation, the time granularity is equal to the *frame duration*, which is the basic *time unit* of any action in FRACAS. This implies that everything that happens inside a frame appears to happen at the end of it, and time measurements with a resolution better than a frame time are not possible. We argue that, when emulating TDMA allocation schemes, event-based simulation is an overkill because, most of the time, all that is needed to get reasonably accurate emulations is to study what happens at the end of each frame. In order to obtain such a behaviour, it would certainly be possible to use an event-based simulator to generate regularly time spaced events, one per frame, but this would mean that the event-based simulator would not be used efficiently. Even if the architecture of FRACAS is maximally efficient when the time resolution is equal to one frame, any specification of the desired time granularity is also possible for each emulation run. Although our prototypal implementation does not allow it, the time granularity could be made as small as desired with respect to the frame duration. This feature would be useful when, after having run a series of tests and having tuned the system to their liking, experimenters require and get the maximum possible precision from the simulation, by reducing the discretisation effects at the expense of running time.

FRACAS assumes that a frame of fixed duration is accessed by a number of stations in conformance with the allocations they have received. At each frame, each station receives a number of *traffic units* (TRU, which is the unit of measure of the traffic in FRACAS) to be transmitted on the framed channel, then it makes a request for an allocation. When all the stations have made their requests, the allocations are computed. The frame where these allocations will be used by the stations is delayed with respect to the frame where the requests have been made by an *allocation delay*, which generally depends on the allocation policy. At each frame, each station queues the traffic units it has to transmit on the framed channel, and dequeues a number of traffic units equal to the allocation received. Each pair *requester-allocator* defines a specific *allocation policy*. The *requester* is the object responsible for computing the number of TRUs requested by each station (station request) in each frame. There is only one requester for all the stations, that is, all the stations use the same criterion for making their request. At each frame, each station calls the requester in order to compute its allocation request. There is a specific requester for each allocation policy. The *allocator* is the object, invoked once per frame, which computes the allocations for all the stations. It is invoked after all the stations have called the requester. The output of the allocator is an array of numbers representing the number of TRUs that will be allocated to each station after the allocation delay.

The policies currently included in our prototypal implementation are:

- •fixed TDMA — a fixed assignment to each station;
- •FODA/IBEA — a centralised control system developed at CNUCE in the framework of the Olympus project, validated by simulative and experimental results [1];
- •V2L-DA — a centralised control system which supports variable bit rate video traffic [15];
- •FEEDERS — a partially distributed control protocol derived from FODA/IBEA [2];
- •DRIFS — a fully distributed control protocol derived from FODA/IBEA [3];
- •CFRA — a slotted centralised control allocation policy developed at ENST-Toulouse [4].

Modelling a network with FRACAS involves choosing the allocation policy, a set of traffic *generators* for each station defined in the network, and, finally, a set of *workers* which will output the desired statistics. The traffic *generators* are responsible for computing the number of TRUs entering a station in each frame. All TRUs have the same length, which is chosen according to the granularity required. At each frame, a station queues the TRUs received from all its generators, and dequeues at most a number of TRUs equal to its allocation. The TRUs produced by a generator are collectively identified by how many they are, that is, they are not individually distinguishable. In order to queue them, the station at each frame simply sums the current number of computed TRUs (queue length) to the number of TRUs received as input from the generators. Each station is fed by an arbitrary number of generators, each defined by individual generation parameters. The traffic generators implemented in our prototypal implementation include constant-bit rate, Poisson, periodic impulsive with constant rate or Poisson bursts, Markov-modulated impulsive, and fractional Gaussian noise. The traffic generated belongs to one of the following four types: stream (real-time, fixed throughput data, with the highest transmission priority), vbr (variable bit rate data, with the second highest transmission priority), interactive data, and bulk data (with lowest transmission priority). All the parts of FRACAS keep different structures for each of the four types of traffic. Therefore, the input queue of each station is composed of four counters, the request issued by the requester is composed of four numbers, and the allocations computed by the allocator are expressed by four numbers per station.

The *workers* are used to gather the statistics of interest. An arbitrary number of workers can be defined, each of which has a specialised job. When the emulator runs, at each frame many quantities inside the emulator can be probed in order to compute statistics on them. The candidate quantities are termed *observables*. Each worker makes a single operation on the observables, such as listing them into a file, and computing their mean, variance, quantiles, mass distribution, and so on. All the observables can either be computed per station, or globally, considering the system as a whole. In our prototype implementation, observables include the traffic in input to a station, the station's input queue length and the input data dropped because of insufficient buffer space, the requests made and the allocations given to a station, the traffic sent, the allocation unused, and the traffic delay. The basic observables are expressed in *numbers of TRUs per frame* per station, and for each station it is possible to monitor the number of TRUs in *input* (that is, those produced by the generators), the number of TRUs *sent*, the available *allocation*, and the *queue* length. The *delay* is the only observable which denotes a measure of time, expressed in *number of frames*. It is computed as the difference between the frame number when a TRU is generated and the frame number when it is sent (*queuing time*), plus the satellite round trip time. The *delay* observable is also special because it can be computed either per frame or per TRU. In the former case, for each frame a delay is computed which is the mean of the delays of the TRUs sent in that frame; in the latter case a delay is computed for each TRU sent. Denoting by $I(f)$ the cumulative number of TRUs in input to a given station at frame f, and by $S(f)$ the cumulative number of TRUs sent by the same station, FRACAS computes the *queuing delay* D_n of the n^{th} TRU as $D_n = f_s - f_i$, where $S(f_s) = I(f_i) = n$.
Traffic generators and workers can be incorporated into FRACAS, or they can be built as external applications. Indeed, there is one ad hoc generator (*external generator interface*) which simply reads from an external program (or file) a list of numbers, which are interpreted as the number of TRUs generated per frame, and there is one worker (*external worker interface*) which writes a list of observables to an external program (or file). This feature is particularly useful if other programs already exist that can generate numbers representing traffic or that can make particular statistical computations on lists of numbers.

IMPLEMENTATION RELIABILITY

To check the reliability of FRACAS we made many comparisons between results obtained using a real satellite network and the emulator. Here we present the measures of the packet delays in two different situations. The traces in Figure 1 refer to an individual station whose input is loaded with a traffic step with a constant rate higher than the bandwidth of the satellite channel.
The allocation method used in this test is FODA/IBEA. While both the prototype emulator and the real system use the same allocation algorithm, the results are not identical, because the real system uses many implementation and hardware-dependent tricks, which are difficult to reproduce exactly in FRACAS. Moreover, the resolution of the emulated packet delay is one frame (20 ms in this case), which introduces a further error with respect to the measured delay. These effects notwithstanding, the emulated traces reflect the measurements well, considering that we are comparing an emulator with a real system, not merely with a mathematical formula.
The relative errors are in the range of [-5%; 5%]. This test was chosen because it is particularly severe, in that it highlights possible problems in the emulator. Indeed, the traffic that feeds the station saturates it, that is, the station is never allocated enough capacity to empty its input queue. In such a situation all the emulation errors accumulate.

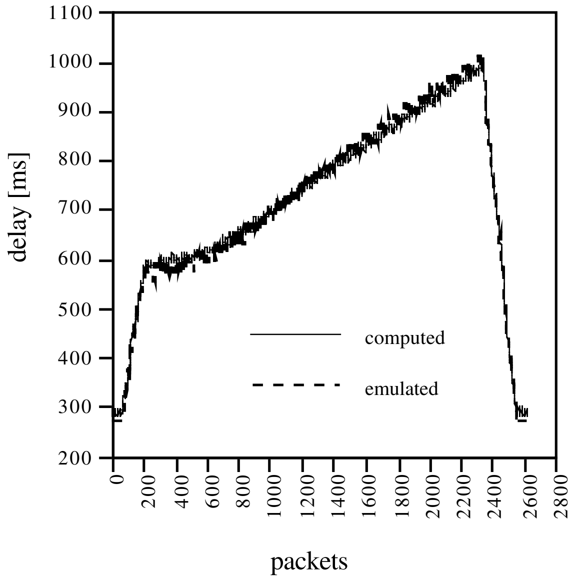

Fig. 1 Packet delay of a station during a transient of the traffic load. Measurement and emulation results.

Figure 2 shows another test comparing the performance of a real network composed of four stations running the FODA/IBEA protocol, and the emulated behaviour. All stations are fed with Poisson traffic, with stations from 1 to 4 receiving 42%, 32%, 20% and 6% respectively of the overall network load. Three runs are performed, with three different overall network loads, up to a load of 85%, excluding all the overheads (signalling, preambles and guard times), which saturates the allocation method and results in queues at station 4. The real data are collected over periods of 30 seconds, corresponding to 1500 frames. Even with this complex set-up and the rather short running period, the relative differences between measured and emulated delays are in the range of [-6%; 2%], with only one point where the difference exceeds one frame length (20 ms).

Since all the timings in FRACAS are multiples of one frame, no delay shorter than this quantization unit can be resolved. This implies, for example, that null delays in a real system are rounded up to a half frame delay. The delays obtained with FRACAS generally have an error in the range of [-1; 1] frames. This means that the mean packet delay has a maximum error of 1 frame, though the error is effectively halved for emulation runs of any significant length. When FRACAS runs an emulation with a time resolution set to one frame, as in our implementation, its concept of time is limited to the current frame number. The traffic generators in FRACAS compute how many packets are generated in the current frame, but no knowledge exists either about the exact instant in the frame when each packet is generated or about the instant in the frame when the packet is sent. FRACAS's idea of delay in this case is that, if a packet is generated in a frame and there is an allocation available after N frames the packet has a delay of N frames. In reality, the packet's delay would lie in the range of [N-1; N+1] frames, depending on the point in the frame when the packet arrives and the point when it leaves.

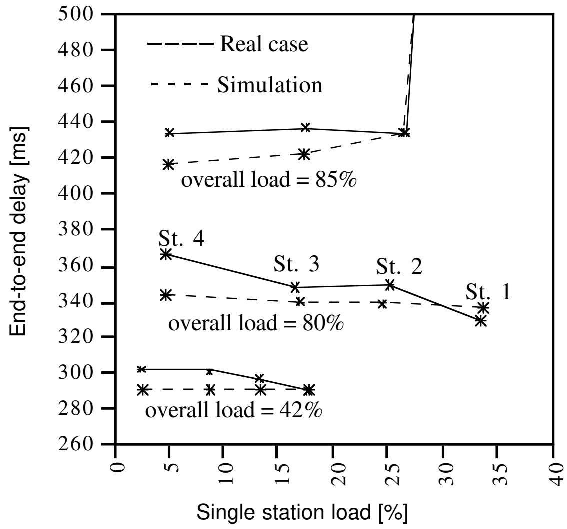

Fig. 2 Mean packet delay for four stations under medium and high loads. Measurement and simulation results.

The *delay error* is the sum of two errors: the error on the position of the arriving packet in the frame (*arriving error*), and the error on the position in the frame of the leaving packet (*leaving error*), each in the range [-0.5; 0.5] frames. The leaving error is not easily estimated, because it depends on the position of the allocation inside the frame, which can have any distribution (and thus any mean) inside the range. For a given allocation policy, it may be possible to compute a distribution of the allocation position, and thus get an estimate of the range of the leaving error which is more accurate than the generic [-0.5; 0.5] frames, as in the case of a fixed allocation. However, this is generally not possible. The arriving error, on the other hand, can be estimated with high accuracy if we assume that the instants when the packets arrive have no correlation with the frame repetition period, which is generally the case. Under this assumption, a packet's delay error has a uniform distribution in the error range. If we assume an emulation run lasting a number of frames k (say, k »100), the error of the mean is well approximated by a Gaussian distribution with a null mean and variance equal to $\frac{1}{12k}$ frames, which is negligible for any practical

purpose. For example, in an emulation run lasting 10000 frames, the 99.7% confidence interval of the mean of the packet delay would be less than ±1% of the frame size.
Thus, for all meaningful emulation runs for which the packet generation rate is not synchronised with the frame rate, we can assume that the error of the mean delay computed by FRACAS has an error in the range [-0.5; 0.5] frames, whose distribution depends on the allocation policy.

CONCLUSIONS AND FUTURE WORK

We have presented the architecture for a performance evaluation tool, which is especially suitable for research teams and students, to explore and compare different satellite access policies in TDMA. The source code of our prototype implementation is freely available to students and researchers for downloading at:

ftp://fly.cnuce.cnr.it/pub/fracas.tgz.(2)

The tool seems to be very useful for the academic community, as it is supported by the amount of positive feedback we have had from people who have used the source programs. Work is in progress towards enabling the prototype to run independent replication tests, in order to accurately assess the reliability of the results obtained. Another direction of improvement is the implementation of time resolutions smaller than one frame. Although we believe that smaller time resolutions generally add nothing to the significance of the results, it may be important in some corner cases, where the behaviour of the emulated system appears to be unstable, and changing the time resolution may provide a deeper insight into its response to traffic loads. Moreover, we are investigating the possibility of augmenting the FRACAS architecture with the capability to consider the behaviour of allocation policies when faced with severe casualties, such as signal fades. This would allow the analysis of recovery from disastrous events (power and component failure) or poor signal reception (deep signal fading).

REFERENCES

[1] N. Celandroni, E. Ferro, N. James, F. Potortì: "*FODA/IBEA a flexible fade countermeasure system in user oriented networks*", International Journal of Satellite Communications, Vol. 10, N. 6, pp. 309-323, November-December 1992.

[2] N. Celandroni, E. Ferro, F. Potortì: "*FEEDERS-TDMA: a distributed-control algorithm for satellite channel capacity assignment in a mixed traffic and faded environment*", International Journal on Satellite Communications, Vol. 15, No. 4, pp. 185-195.

[3] N. Celandroni, E. Ferro, F. Potortì: "*DRIFS-TDMA: a proposal for satellite access distributed control algorithm for multimedia traffic in a faded environment*", Vol. 15, No. 5, pp. 227-235, September-October 1997.

[4] T. Zein, G. Maral, T. Brefort, M. Tondriaux: "*Performance of the Combined/Fixed Reservation Assignment (CFRA) scheme for aggregated traffic*", Proceedings of the COST-226 Final Symposium, pp. 183-198, Budapest (H), 10-12 May 1995.

[5] N. Celandroni, E. Ferro: "*The FODA-TDMA satellite access scheme: presentation, study of the system, and results*", IEEE Trans. Communications, vol. COM-39, pp. 1823-1831, Dec. 1991.

[6] S.N. Crozier: "*Sloppy-Slotted ALOHA*", proceedings of the International Mobile Satellite Conference, Ottawa, 1990, pp. 357-362.

[7] S.S. Lam: "*Packet broadcast networks - A performance analysis of the R-ALOHA protocol*", IEEE transactions on Computers, vol. C-32, no. 8, pp. 717-726, Aug. '83.

[8] I.M. Jacombs, R. Binder, E.V. Hoversten: "*General purpose packet satellite Networks*", proceedings of the IEEE, vol. 66, no. 11, pp. 1448-1467, Nov. 1978.

[9] C.J. Wolejsza Jr., J. McCoskey, D. Taylor: "*The Random Access with Notification Multiple Access Protocol: performance and implementation*", ICDSC 8, pp. 131-138, Jan. 1989.

[10] Tho Le-Ngoc, V. Krishnamurthy: "*Performance of Combined Free/Demand Assignment Multiple-Access schemes in satellite communications*", International Journal of Satellite Communications, Vol. 14, 11-21 (1996)

[11] N. Celandroni, E. Ferro, F. Potortì, A. Bellini, F. Pirri: "*Practical experiences in interconnecting LANs via satellite*", ACM SIGCOMM Computer Communication Review, Vol. 25, No. 5, October 1995.

[12] N. Celandroni, E. Ferro, F. Potortì: "*Comparison between distributed and centralised demand assignment TDMA satellite access schemes*", International Journal on Satellite Communications, Vol. 14, No. 2, pp. 95-112, March-April 1996.

(2) Users who download the FRACAS code are kindly requested to notify any of the authors of this paper.

[13] N. Celandroni, E. Ferro, F. Potortì, G. Maral: "*Delay analysis for interlan traffic using two suitable TDMA access schemes*", International Journal of Satellite Communications, Vol. 15, No. 4, pp. 141-153.

[14] N. Celandroni, E. Ferro, F. Potortì: "*FRACAS: FRAmed Channel Access Simulator. User Manual*", CNUCE Report C95-27, September 1995.

[15] N. Celandroni, M. Conti, E. Ferro, E. Gregori, F. Potortì: "*A bandwidth assignment algorithm on a satellite channel for VBR traffic*", International Journal on Satellite Communications, Vol. 15, No. 6, pp. 237-246, November-December 1997.