

Bus modelling in Zoned disks RAID storage systems

Peter Harrison¹ and Soraya Zertal²

¹ Imperial College London, South Kensington Campus, London SW7 2AZ, UK
pgh@doc.ic.ac.uk

² PRiSM, Université de Versailles, 45 Av. des Etats-Unis, 78000 Versailles, France
Zertal@prism.uvsq.fr

Abstract

A model of bus contention in Multi-RAID storage architecture is presented. Based on an M/G/1 queue, the main issues are to determine the service time distribution that accurately represents the highly mixed input traffic of requests. This arises from the coexistence of different RAID organisations that generates several types of physical request (read/write for each RAID level) with different related sizes. The size distributions themselves are made more complex by the striping mechanism, with full/large/small stripes in RAID5. We show the impact of the bus traffic on the system's performance as predicted by the model and validated against the simulation of the hardware, using a common workload assumptions.

Keywords : Multi-RAID, zoned disks, M/G/1 queues, I/O and bus modelling, simulation.

1 Introduction

Storage systems have evolved from small collections of interconnected disks to large disk-arrays, shared by multiple applications serving a very large user community. RAID (Redundant Arrays of Independent Disks) [2] systems were the first widely accepted, large scale storage architecture, for which many propositions for enhancement have been made. Apart from a sequence of RAID variants to improve speed of access and reliability (from RAID0 to RAID6) and the HPautoRAID [6] to have two RAID organisations on the same storage system at separate space locations, these include the Multi-RAID system [4] where different RAID configurations coexist on the same disk devices without physical space boundaries, to jointly

fulfill dynamically different performance and space requirements.

On this architecture, with multiple schemes implementing the most used RAID organisations (RAID01 and RAID5), requests of different type and size are executed in parallel on asynchronous disks connected via a bus to the RAID controller. The bus data transfers occur, with a wide range of lengths, at different steps during the requests' execution. The bus is a critical shared resource for these essentially independent, asynchronous parallel disks and is subject to congestion. The delay so introduced has a significant impact on the whole system response time, especially for the RAID5 partial stripe writes. Modelling such a storage architecture and related mechanisms is useful in analysing the performance it can deliver and predicting its behaviour for different workloads. We have subsequently elaborated the disks array model and carefully validated it, using an approximate solution for its specific fork-join problem that arises from the independent operation of the disks [4]. We extended this work to modern zoned disks by finding appropriate seek distance distributions, linear and non-linear in [11, 12] respectively.

Other works in the area relate to single RAID levels and their performance in a specific working mode (normal or degraded) [1, 7, 9] and some of them focused on the delivered throughput [8] which can be limited by the shared bus bandwidth.

In this paper, we focus on the bus connecting device for unique RAID organisations as well as Multi-RAID systems. The very specific context characteristics which motivate this study are discussed in Section 2, the proposed model is described in Section 3 and its parameters as moments, are determined using the detailed description of the principles of operation of the system, in Section 4. Validation against simulation is shown in Section 5 and the paper concludes in Section 6.

2 Context and motivations

We model the RAID connecting bus using an M/G/1 queue, extracting its specific usage in terms of generated traffic in various contexts – before, during or/and after the disk's service period and its effect on the total IO response time.

In RAID storage systems, an IO request's execution varies according to its type (read/write) and size (data blocks), as well as the RAID organisation type¹ [2]. The resulting different combinations may lead to the transfer of data on the bus to a native, mirror or parity disk; before, during or/and after an actual disk service, on one or more different disks. The transfer before or after the service are well studied, especially in the networking community but also in parallel computer architecture community [10, 3] which is closer to our specific RAID storage system. However,

¹From RAID0 to RAID6, as defined in Berkeley's classification.

the presence of transfers at different steps in each overall disk access split this process into phases separated by bus delays, each of which has a different impact on the whole response time. In fact, the delay can change the execution of the following phase. This impact is more important in a Multi-RAID, when different RAID organisations coexist on the same disk drives, leading to more complex execution schemes and hence generating more complex bus traffic.

The simplest case is a read or write request on a RAID01 (striped and mirrored RAID). The bus transfer is performed once: after the disk service for reads and before the disk service for writes, as shown in Figure 1. However, the number

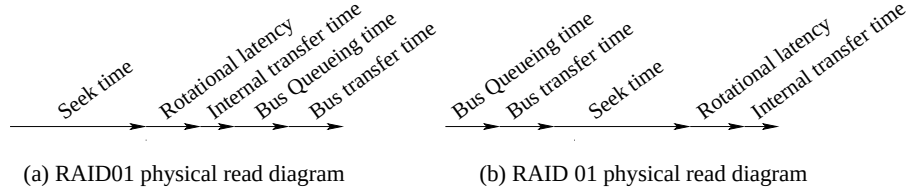
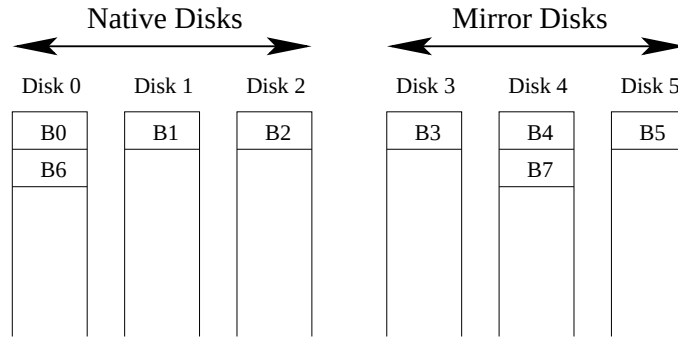


Figure 1: RAID-01 service time diagram

of data blocks (native or mirror) transferred to/from each disk is different in each case and depends on the request type and size as shown on Figures 2 and 3 for a 8 blocks request from B_0 to B_7 . On RAID01, we can read either the native block on a native disk or its mirror on the mirror one according to the selected scheduling policy². In the example shown on Figure 2³, we read native versions of blocks $\{B_0, B_1, B_2, B_6\}$ on native disks $\{disk0, disk1, disk2\}$ and we read mirror versions of the rest of blocks. This scheduling leads to a two consecutive blocks read request on $disk1$ (native) and $disk3$ (mirror) and a single block read on the other disks. The RAID01 write requests need an access to both native and mirror versions on the associated disks. The 8 blocks logical request generates then a three consecutive blocks request to $\{disk0, disk1\}$ to write the native version of blocks $\{B_0, B_3, B_6\}$ as well as their mirrors $\{disk3, disk4\}$ to write the mirror version of the same blocks. The native disk $\{disk2\}$ as well as its mirror $\{disk5\}$ is accessed by a two consecutive native blocks request $\{B_2, B_5\}$. RAID5 (distributed parity) is much more complex due to its data and redundancy patterns. On a N disks RAID5, a stripe is a collection of $(N - 1)$ data units with an associated one parity unit controlling them, each of which on a different disk. This is also called full stripe because it covers all the disks. A partial stripe is a stripe with a parity block and less than $(N - 1)$ data blocks. For every stripe and regardless to

²Random, shortest waiting queue, shortest seek distance,...etc.

³Only accessed blocks are indicated on disks for all Figures.

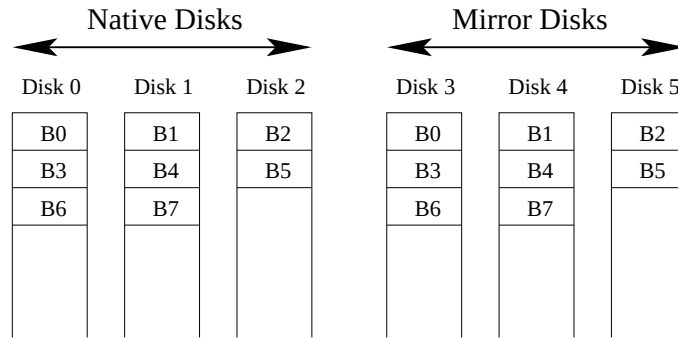


8 blocks logical read request
2 blocks on disks {0,4} and 1 block on the others

Figure 2: RAID-01 logical read

its width, the parity stripe unit (block) is calculated and associated to a disk on a round-robin manner.

The read on a RAID 5 is similar to the one on RAID01, with unique target disks rather than having the choice between a native and a mirror target. However, the write is completely different and much more complex. According to the logical request size and the stripe width, generating a full, large or small stripe when the request size covers all data units in the stripe, at least half of the data stripe units and less than half of the data stripe units respectively. A RAID5 write might generate a combination of these three write types : full write(s) followed by a large or a small one. Every write request among these three ones and their possible combinations has an appropriate parity calculation and pre-read operations and involves different numbers of disks. These are respectively none of the disks for the full stripe writes, the non-target disks (those not concerned by the write request) and the parity disk only for large stripe writes, and the target disks with the parity disk only for small stripe writes as shown on Figure 4. In a combination case (full followed by large or small request), the parity calculation is performed according to its two components. The transfer is performed once for reads (after the disk service) and according to the diagram in Figure 5 for writes. The Multi-RAID we considered is a combination of RAID01 and RAID5, which makes its generated data transfers quite complex combinations of the diagrams in Figures 1 and 5.



8 blocks logical write request
 2 blocks on disks {2,5} and 3 blocks on the others

Figure 3: RAID-01 logical write

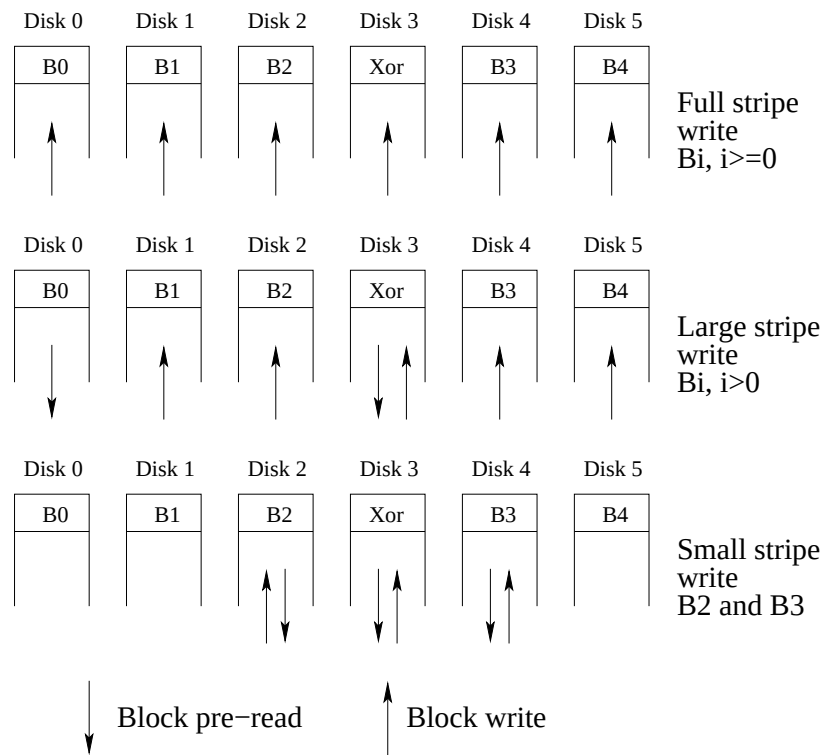


Figure 4: RAID5 logical writes

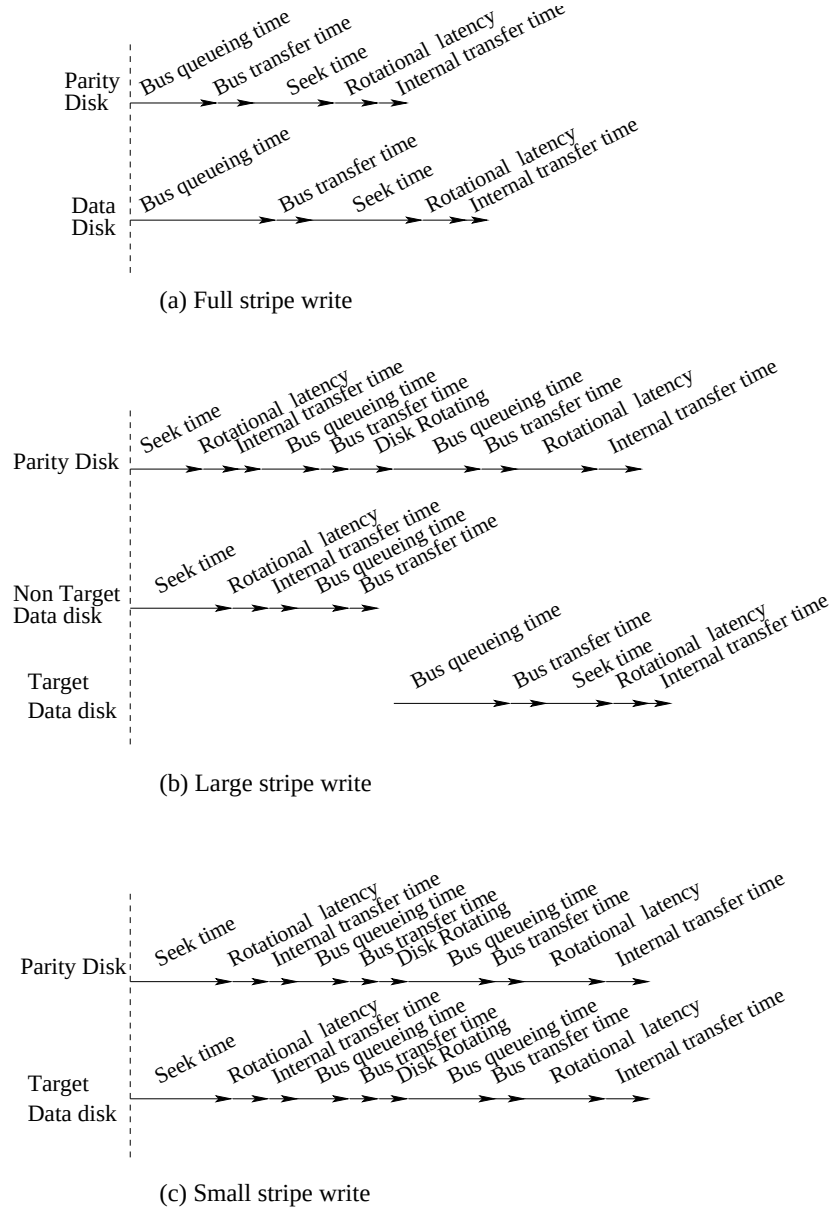


Figure 5: RAID5 service time diagram

3 Bus response time model

We analyse the bus response time of a logical request by considering its type (read/write), its size (number of blocks) and the system's workload intensity (ar-

rival rate in term of number of logical requests per second). We model the delay caused by contention at the bus by a conventional M/G/1 queue with a single workload class which is the aggregate of all IO transfer types and sizes for the different coexisting RAID organisations. The challenge is to estimate the probability distribution of these sizes – or at least its moments – and to calculate the aggregate arrival rate of such transfers, a rather easier task. The bus response time is then composed of the queueing time Q_{bus} and a service time S_{bus} that depends only on the size of the particular IO transfer in question :

$$T_{bus} = Q_{bus} + S_{bus} = Q_{bus} + K \times T$$

where K is the size of the IO transfer in term of blocks, related to a logical request, which we estimate in Section 3.2, and T is the time taken to transmit one block, the reciprocal of the bus bandwidth.

3.1 Bus Queueing time (Q_{bus}) :

We use the Pollaczek-Khinchin formula [5], as used for the queueing time at the disk level in [4], with bus-related parameters:

$$E[Q_{bus}] = \frac{\lambda_{bus} \overline{\overline{S_{bus}}}}{2(1 - \rho_{bus})} \quad (1)$$

The traffic load is now defined by:

$$\rho_{bus} = \lambda_{bus} \times \overline{S_{bus}}$$

where $\overline{S_{bus}}$ denotes the mean bus service time and, as we will use later, $\overline{\overline{S_{bus}}}$ denotes its second moment. In fact, we use n overbars to denote the n th moment of a random variable in general.

As we are considering a Multi-RAID storage system, the bus traffic intensity λ_{bus} derives from a mixture of RAID layout requests on top of the different types of IO transfer generated by the requests' pre-reads, native and mirror accesses, as shown in Figures 1 and 5. Considering the two most used RAID organisations, RAID01 (mirroring and striping) and RAID5 (distributed parity), the bus traffic can be detailed as below:

$$\lambda_{bus} = \lambda_{bus_RAID01} + \lambda_{bus_RAID5}$$

where

$$\lambda_{bus_RAID01} = \lambda \times P_{raid01} \times B(2 - p_r), \text{ and}$$

$$\lambda_{bus_RAID5} = \lambda \times P_{raid5} (p_r \times B + p_w ((c dt_{full} * N * [\frac{B}{(N-1)}]) + (c dt_{lg} * (N + 1)) + (c dt_{sm} * 2(B\%(N-1) + 1))))$$

B is the logical request's size (number of blocks), p_r and p_w are the probability of a request to be a read or a write respectively, P_{raid01} and P_{raid5} are the probabilities

of accessing the RAID01 area or the RAID5 one respectively and finally λ is the logical requests arrival rate.

“%” denotes the modulo function.

The boolean parameters cdt_{full} , cdt_{lg} and cdt_{sm} indicate respectively if the stripe is full, large or small [4]⁴. They are defined for RAID5 write requests and depend on the logical request size (B) and the RAID5 width (N) ::

$$\begin{aligned} cdt_{full} &= ([B/(N-1)] > 0), \\ cdt_{lg} &= (B \% (N-1) \geq [N/2]) \text{ and} \\ cdt_{sm} &= (B \% (N-1) < [N/2]) \end{aligned}$$

3.2 Bus service time (S_{bus}) :

A bus service consists of transferring K data blocks (native/mirror/redundancy) via the bus to/from the disk devices from/to the RAID controller. The number of blocks to transfer depends on the request type, request size and the associated RAID organisation as follows :

For the RAID01 read and write operations :

$$K_{R01} = \begin{cases} B & \text{w.p. } p_r \\ 2 \times B & \text{w.p. } p_w \end{cases}$$

For RAID5 operations, in addition to the read and write modes, we need to distinguish full stripes from partial (small or large) stripes. Hence we have:

$$K_{R5} = \begin{cases} B & \text{w.p. } p_r \\ K_{R5W} & \text{w.p. } p_w \end{cases}$$

Then the number of data (native and parity) blocks transferred via the bus for a RAID5 write (K_{R5W}) can be calculated for every possible case as:

$$K_{R5W} = \begin{cases} K_f & \text{if } (cdt_{full} \wedge \neg(cdt_{lg} \vee cdt_{sm})) \\ K_{lg} & \text{if } cdt_{lg} \wedge \neg cdt_{full} \\ K_{sm} & \text{if } cdt_{sm} \wedge \neg cdt_{full} \\ K_f + K_{lg} & \text{if } (cdt_{full} \wedge cdt_{lg}) \\ K_f + K_{sm} & \text{if } (cdt_{full} \wedge cdt_{sm}) \end{cases}$$

⁴We interpret “false” as 0 and “true” as 1 to simplify later calculations.

where $\wedge$ denotes the boolean AND operation, with :

$$K_f = \lceil \frac{B}{(N-1)} \rceil \times N$$

$$K_{lg} = N + 1$$

$$K_{sm} = ((B \% (N - 1)) + 1) \times 2$$

4 Moments of bus delay

We wish to calculate the mean and variance of the bus delay and consequently require the first *three* moments of the:

1. Number of blocks to transfer (K) :

The first moment of K is:

$$\overline{K} = P_{raid01} \times \overline{K_{R01}} + P_{raid5} \times \overline{K_{R5}}$$

The second moment of K is:

$$\overline{K^2} = P_{raid01} \times \overline{K_{R01}^2} + P_{raid5} \times \overline{K_{R5}^2}$$

where the second moments of K_{R01} and K_{R5} are:

$$\begin{aligned} \overline{K_{R01}^2} &= B^2 \times p_r + (2B)^2 \times p_w \\ \overline{K_{R5}^2} &= B^2 \times p_r + p_w((cdt_{full} \times K_f^2) + (cdt_{lg} \times K_{lg}^2) + (cdt_{sm} \times K_{sm}^2)) \end{aligned}$$

2. Service time (S_{bus}) :

The first three moments of the bus service time are simply:

$$\overline{S_{bus}} = \overline{K} \times T, \quad \overline{S_{bus}^2} = T^2 \times \overline{K^2} \quad \text{and} \quad \overline{S_{bus}^3} = T^3 \times \overline{K^3}$$

3. Bus queue (Q_{bus})

The first moment is calculated using the Pollaczek-Khinchin formula as in Equation 1. The second moment can be used to assess the accuracy of our model and is obtained as (see, for example, [4]):

$$\overline{Q_{bus}^2} = \frac{\lambda^2 \overline{S_{bus}^2}}{2(1 - \rho_{bus})^2} + \frac{\lambda \overline{S_{bus}^3}}{3(1 - \rho_{bus})} \quad (2)$$

5 Bus model validation

In order to validate our model, we developed a hardware simulator representing all the architecture components and their performed functions. The simulator is an event-driven one, written in C and composed mainly of 3 modules :the workload generator, the event (IO request) execution engine and the statistical analyser. For scalability safety, the execution engine uses hardware libraries for hardware characteristics such disks, connecting bus ...etc. The workload generator and the model use common assumptions : uniform requests' address distribution on the storage

space and poissonian requests' inter arrival time.

We focus on the analysis of the impact of three parameters on the bus queueing time: the request type (read/write) by varying the probability of a logical request to be a read (p_r), the logical request size by varying B , and the architecture configuration (i.e. RAID organisation here) by considering an exclusive RAID01, an exclusive RAID5 and a mixture of both; using the above model and an event-driven simulator for a storage system composed of 16 FujitsuMAN3367 disks connected to 40MB/s Wide Ultra SCSI bus.

However, we start by studying the relation between the request size B and the generated traffic (K) which shows the heavy traffic source of bus contention.

1. Request size Vs generated traffic

For RAID5 writes, Figure 6 confirms the significant effect of the requests' sizes on the generated traffic and the variations of this traffic according to the stripe width (full, large or small). As we are considering a 16-disk storage array, organised as a RAID5, full stripe writes are obtained for logical request sizes $B \in \{15 * i, i > 0\}$, small stripe writes are obtained for $B \in \{j + 15 * i, i > 0 \text{ and } j \in [1..7]\}$ and large stripes are obtained for $B \in \{j + 15 * i, i > 0 \text{ and } j \in [8..14]\}$. The generated traffic increases with B during small stripe phases, decreases as B increases during large stripe phases because the number of pre-read IOs decreases, and finally the generated traffic reaches its lowest values for full stripes. Figure 6 shows this effect for an exclusive write request stream ($p_r = 0$) and a mixed request stream with equal proportions of reads and writes ($p_r = 0.5$) against an exclusive read stream ($p_r = 1$) for which the generated traffic is linear in the logical request size.

For RAID01 writes, the generated traffic is proportionnel to the logical request size B with a factor of two.

Considering a Multi-RAID storage system with an equal proportion of RAID01 and RAID5 on Figure 7, we can see the explosion of the generated traffic in an exclusive small write mode. In this case, the generated bus traffic is composed of the double RAID01 logical requests and quadruple RAID5 ones.

2. Request type

In order to analyse the effect of reads and writes on bus queueing behaviour, we vary the probability of a request being a read p_r from 0, for an exclusive read workload, to 1 for an exclusive write workload, including 0.5 for a balanced workload. We can see the variation for RAID01 in Figure 8, the model and the simulation results showing good agreement and indicating the increase in the queueing time with the proportion of writes, doubling the

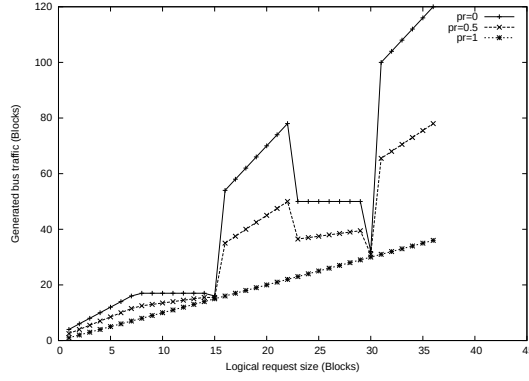


Figure 6: Generated transfer blocks on RAID5

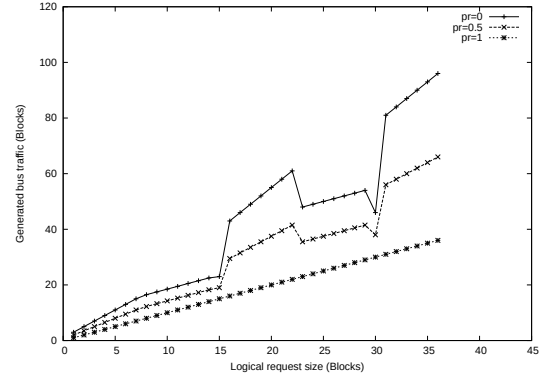


Figure 7: Generated transfer blocks on Multi-RAID

traffic for this category of RAID. This is confirmed for RAID5 as well, as shown in Figure 9 where the traffic is multiplied by four for small ($B = 1$) write requests.

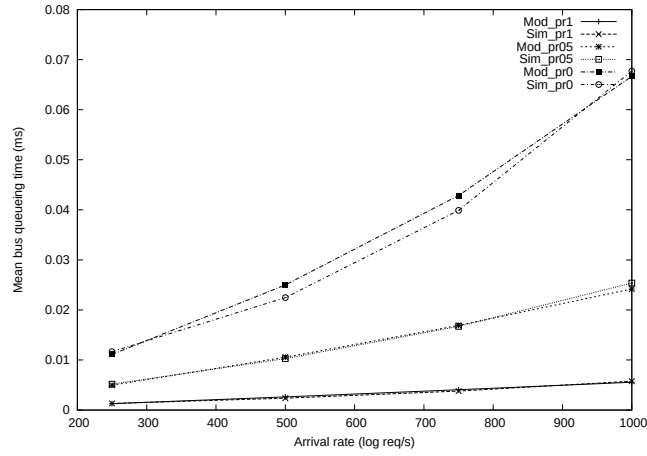


Figure 8: Request type impact on RAID01 small requests ($B = 1$)

3. Request size

We observed the behaviour of the bus under various request sizes, going from one block (4KB) to 4 blocks, then 8 blocks for RAID01. Figure 10 shows the impact of the logical request size on the generated bus traffic and hence on the bus queueing time, mutually validated by the analytical model and the simulation, which show good agreement. We choose the exclusive

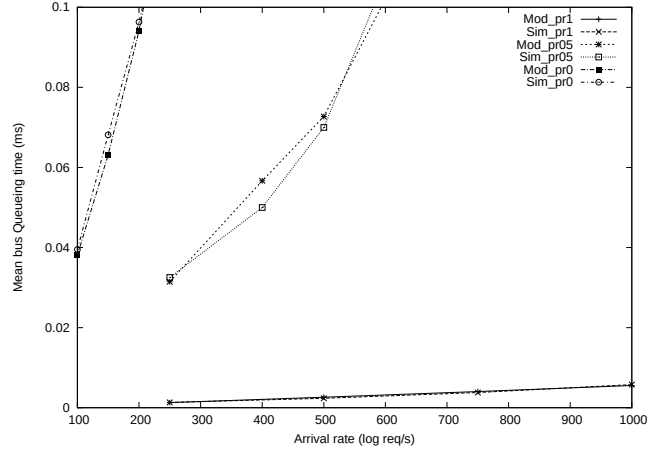


Figure 9: Request type impact on RAID5 small requests ($B = 1$)

read mode to isolate the size impact from any write operation effect. We can see that for 8 blocks requests, the queueing time explodes at relatively low arrival rates (reaches 9ms at 150 req/s). RAID5 reads are similar to the RAID01, thus Figure 10 is representative of both organisations.

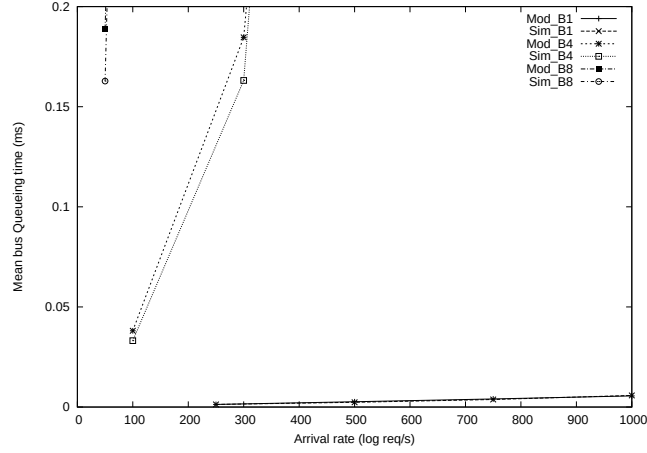


Figure 10: Request size impact on RAID01

4. Architecture configuration

We notice different traffic intensities generated by the same logical request stream on an exclusive RAID01 storage system, on an exclusive RAID5 storage system and on a mixture of both in a Multi-RAID with equal proportions.

Figure 11 shows clearly the effect of the architecture configuration on the generated traffic and the agreement of the model and the simulation results.

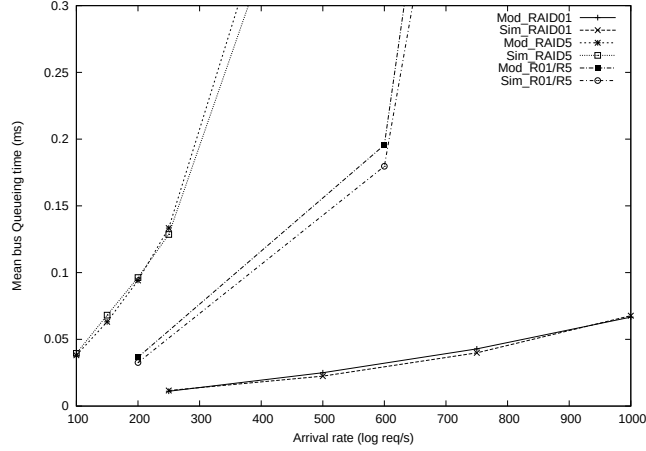


Figure 11: Architecture configuration impact on Q_{bus}

6 Conclusion

In this paper, we achieved an additional step of our hierarchical storage system model for disk arrays: the disks connecting component model ; which is one of our Intelligent Performance evaluation Of large Data Systems (IPODS) project's aims. The model showed good accuracy when compared with simulation in preliminary experiments. Further validation is required in which we consider request arrival streams with non-uniform addresses and higher moments of response times. Similarly, at greater computational expense, response time distribution functions themselves could be obtained from the Pollacek-Khintchine result for Laplace transforms, using a numerical inverter. This is likely to be much more sensitive to validation, especially in the tail. Extension of our storage architecture to a matrix of disks for very large systems can be achieved using this work, with adding new buses components and related connections in the simulator and extending our model to a multi-bus one using an M/G/n queue. We can then move to the final part of our overall model for a whole storage system, namely the controlling component with its request scheduling schemes and data caching policies, which influence heavily the net storage system performance.

References

- [1] S. Chen and D. Towsley. A performance evaluation of raid architectures. *IEEE Transactions on Computers*, 45(10), October 1996.
- [2] G. Gibson D. A. Patterson and R. H. Katz. A case for redundant arrays of inexpensive disks (RAID). In *Proceedings of SIGMOD Conference*, June 1988.
- [3] Michael J. Flynn. *Computer architecture-pipelined and parallel processor design*. Jones and Bartlett Publishers, 1995.
- [4] P.G. Harrison and S. Zertal. Queueing models of RAID systems with maxima of waiting times. *Performance Evaluation*, 2007.
- [5] Ng Chee Hock. *Queueing Modelling Fundamentals*. John Wiley Publisher, 1996.
- [6] C Staelin J. Wilkes, R. Golding and T. Sullivan. The hp autoraid hierarchical storage system. *ACM transactions on Computer systems*, 14, Feb 1996.
- [7] E.K. Lee and R.H. Katz. An analytic performance model of disk arrays. In *Proc. ACM SIGMETRICS*, May 1993.
- [8] G. A. Alvarez M. Uysal and A. Merchant. A Modular, Analytical Throughput Model for Modern Disk Arrays. In *Proceedings of the International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunications Systems (MASCOTS)*, August 2001.
- [9] A. Merchant and P.S. Yu. An analytical model or reconstruction time in mirrored disks. *Performance evaluation*, 20, May 1994.
- [10] A. van de liefvoort and N. Subramanian. A new approach for the performance analysis of a single-bus multiprocessor system with general service times. *IEEE Transactions on Computers*, 42, Mar 1993.
- [11] S. Zertal and P.G. Harrison. Multi-raid queueing model with zoned disks. In *High Performance Computing and Simulation*, 2007.
- [12] S. Zertal and P.G. Harrison. Non-linear seek distance for optimal accuracy of zoned disks seek time in multi-raid storage systems. In *High Performance Computing and Simulation*, 2008.