

Performance Study of a Packet Load Balancer for High-Speed Data Links

prepared for the lecture “Computer System Performance Analysis and Benchmarking” by
Prof. Thomas Stricker, Laboratory for Computer Systems,
Swiss Institute of Technology (ETH), Zurich, Switzerland

Gero Dittmann

IBM Zurich Research Laboratory

Saeumerstrasse 4, CH-8803 Rueschlikon

Tel: ++41-1-724-8243, Fax: ++41-1-724-8955

ged@zurich.ibm.com

March 16, 2001

Chapter 1

Studied System

This study follows the performance analysis techniques described in [Jai91].

1.1 Load Balancer Architecture

The system under test (SUT) is a Load Balancer chip (also described in [BDE⁺01]) that resides in the middle between a high-speed SONET framer and four network processors (NP's). The Load Balancer distributes IP packets coming in from the optical link across the NP's (see fig. 1.1). It should do this in a flow preserving manner, i.e. packets with the same IP five-tuple, consisting of source and destination address, layer 4 protocol number, and—if present—layer 4 port numbers, should go to the same NP. This ensures in-sequence delivery of packets within flows and avoids the need for exchange of state-information between NP's. The ultimate goal is to still balance the load evenly over the four NP's.

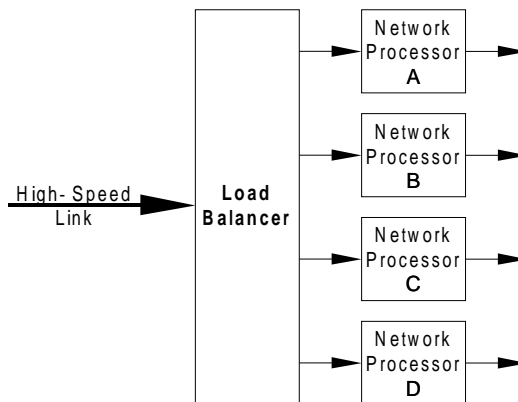


Figure 1.1: Load Balancer Application

The internal structure of the receive part of the chip is shown in fig. 1.2. Upon entering the Load Balancer a packet is first inspected by a header parser. Here the fields in the packet header that uniquely identify a flow are extracted. They are then compressed by

a hash function to an index that serves as an address into the look-up memory of the balancing unit. Hashing entails hash collisions, i.e. multiple flow ID's will be mapped to the same index. This is a desired effect since the look-up memory has to be much smaller than the number of possible flow ID's.

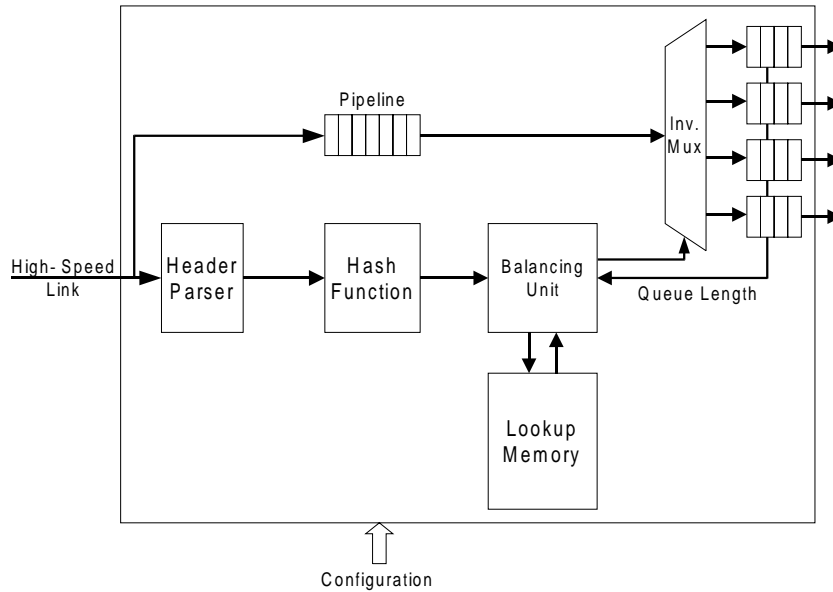


Figure 1.2: Load Balancer Schema

The balancing unit finally makes the decision to which NP a packet is to be sent based on information stored in the look-up memory and on the length of the individual queues towards the NP's. It controls the inverse multiplexer through which packets are enqueued. In parallel to the described decision making process, the according packet is delayed in a pipeline until the queuing decision is available. The four queues are implemented in a shared buffer.

The balancing unit implements several algorithms that optimize the load distribution. First of all, a flow is considered no longer to exist if no packet belonging to it has arrived within a **time-out** period. This time-out can be tuned. If a packet hits a time-out table entry then a new association between the table entry and an output queue is performed. This allows for perpetual revision of assignments in order to adapt to load differences between NP's.

Secondly, length of the four output queues is continuously monitored. If a queue exceeds a certain threshold then an overload condition for this queue is concluded. In consequence, some flows for this queue are permanently **reassigned** to other queues to achieve a better balancing. The queue threshold can be adjusted.

Furthermore, bandwidth consumption of individual flows is measured. If a flow exceeds a fourth of the incoming link capacity then a single NP would not be able to handle this flow. As a remedy, packets of these excessive flows are "**sprayed**" across the four queues. This way they do not effect other "well behaving" flows and are still being forwarded. The sensibility to bursts can be configured to allow for short term bursts of excessive data rate that are not being sprayed. Also, a recovery speed is adjustable which influences when

spraying stops after a flow has dropped under the excessive rate.

1.2 Analysis Set Up

The goal of this analysis is to find a memory size suitable for the shared buffer, depending on the settings of the system variables, and to find guidelines for optimal parameter settings to keep memory utilization low because it is a measure for latency and drop probability. Also, the number of reassignments and spraying should be minimal because they can lead to out-of-order delivery of packets which is not desirable. Therefore, **performance metrics** are:

- Maximum memory utilization.
- Number of reassignments per occupied lockup table entries.
- Percentage of sprayed packets compared to all transmitted packets.

If memory utilization is larger than 100% then a buffer overflow has occurred which means that packets had to be dropped. This situation must be avoided.

System parameters that affect performance are:

- Real shared buffer size.
- Flow time-out period.
- Queue length threshold for reassignments.
- Burst tolerance for spraying.
- Speed of recovery from spraying.

This parameter set is identical with the factor set, i.e. these system parameters will be varied in the course of the analysis. Other system parameters that will be kept constant are e.g. the *hash function* or the *size of the look-up memory*. **Workload parameters** are the traffic characteristics, e.g. flow ID variance, packet length distribution, flow shapes, etc.

Since the performance of the Load Balancer must be evaluated before it is actually built, measurement is not an option. State-of-the-art network traffic analysis does not allow for a complete picture of traffic characteristics and their parameter constraints. Thus, it seems impractical to cover all important application scenarios by analytical modeling. These considerations leave simulation as the only feasible **evaluation technique**. A model of the Load Balancer has been implemented in C.

As simulation **workload** traffic traces are used. These have been recorded at central routers of universities and commercial ISPs in the US and were publicly available from [NLA]. Since these traces only represent under-utilized OC-3 links (155 Mb/s), they have to be scaled up to the desired link speed under full utilization. A fully loaded link is

emulated by eliminating gaps between packets. In a second step the traces are then time compressed. Alternatively, it was also tried to combine several low speed traces to generate one high speed trace. Test runs with these merged traces showed that they do not present the Load Balancer with any problems but are in fact much easier to handle. Furthermore, several traffic synthesis methods were employed, such as self-similarity models, but these methods are based on traffic analysis which suffers from the above mentioned incompleteness of reflected parameters. This explains why synthetic traces are no challenge to the Load Balancer, either, as test runs revealed. The goal in selecting the workload is to test the Load Balancer under full load conditions with all sorts of extremely demanding traffic while still keeping traffic characteristics in realistic constraints. Consequently, only time compressed real traces are used for simulation. Preliminary experiments and traffic analysis have been used to compile a comprehensive workload suite of 63 traces representing a wide variety of traffic characteristics.

Chapter 2

Experiments

The system factors that are varied in the experiments and their levels are given in table 2.1.

Factor	Levels
Buffer size	66, 100 (arbitrary unit)
Flow time out	20 - 200 ms
Reassignment threshold	25 - 75%
Burst tolerance	25 - 50%
Spraying recovery	0 - 80%

Table 2.1: Load Balancer Factors and Levels

2.1 Estimation of Factor Relevance

In order to understand the influence of each factor on system performance while avoiding a complete 2^5 factorial design, a 2^{5-2} fractional factorial design experiment is carried out. It uses the symbol and level assignments shown in table 2.2. The sign table for this design is given by table 2.3. As an example, it also gives the measured results for one of the traffic traces (*Mem* denotes memory utilization, *Reas* represents the reassignment ratio, and *Spray* is the spraying ratio). In table 2.4 the translation to experiment set up can be found.

Symbol	Factor	Level -1	Level +1
A	Buffer size	66	100
B	Flow time out	20 ms	200 ms
C	Reassignment threshold	25%	75%
D	Burst tolerance	25%	50%
E	Spraying recovery	0%	80%

Table 2.2: Levels for First Fractional Design

It is assumed that experiments with different traffic traces cannot be considered simple

Exp.	I	A	B	C	AB	AC	D	E	Mem	Reas	Spray
1	1	-1	-1	-1	1	1	1	-1	0.648%	834	0
2	1	1	-1	-1	-1	-1	1	1	0.638%	120	2.3%
3	1	-1	1	-1	-1	1	-1	1	0.696%	601	0
4	1	1	1	-1	1	-1	-1	-1	0.675%	352	0
5	1	-1	-1	1	1	-1	-1	1	1.589%	201	0
6	1	1	-1	1	-1	1	-1	-1	0.923%	11	0
7	1	-1	1	1	-1	-1	1	-1	1.920%	140	0
8	1	1	1	1	1	1	1	1	1.371%	14	2.3%
Mem:											
Total / 8	1.06	-0.16	0.11	0.39	0.01	-0.15	0.09	0.02			
Variation		11.0%	5.3%	70.1%	0.1%	9.9%	3.4%	0.1%			
Reas:											
Total / 8	0.021	-0.011	-0.000	-0.014	0.005	0.006	-0.000	-0.004			
Variation		33.6%	0.1%	48.7%	5.7%	8.6%	0.1%	3.3%			
Spray:											
Total / 8	0.58	0.58	0	0	0	0	0.58	0.58			
Variation		33.3%	0%	0%	0%	0%	33.3%	33.3%			

Table 2.3: Sign Table and Example Results

Experiment	Buffer size	Flow time out	Reassignment threshold	Burst tolerance	Spraying recovery
1	66	20 ms	25%	25%	0%
2	100	20 ms	25%	25%	80%
3	66	200 ms	25%	50%	80%
4	100	200 ms	25%	50%	0%
5	66	20 ms	75%	50%	80%
6	100	20 ms	75%	50%	0%
7	66	200 ms	75%	25%	0%
8	100	200 ms	75%	25%	80%

Table 2.4: Experiments for Fractional Design

repetitions of experiments and the resulting variations between runs cannot just be counted as measurement errors. The individual traces represent completely different traffic scenarios at different points in the network that would be accommodated by different parameter settings. This view is supported by the wide spread of results.

Consequently, an alternative way has to be found to summarize results from different workloads. The chosen approach is to compute variations separately for each traffic trace and then, since variations are ratios, use the geometric mean of the variations per factor, normalized to 100%. For the computation of variation on reassignments and spraying only traces are considered for which the respective mechanism has been involved in at least one experiment, i.e. traces that showed reassignments or spraying. Table 2.5 gives the results and their standard deviation in percentage points. In addition to the effects from the sign table also the confoundings are indicated.

	I	A	B	C	AB	AC	D	E
	BCD	DE	CD	BD	CE	BE	BC	AD
	ADE	BCE	ACE	ABE	BDE	CDE	AE	ABC
	ABCE	ABCD	ABDE	ACDE	ACD	ABD	ABCDE	BCDE
Mem: Variation		3.7%	1.8%	90.6%	0.6%	1.1%	1.3%	0.9%
Standard Deviation		5.1	6.96	26.9	4.1	3.1	9.1	10.1
Reas: Variation		17.7%	1.8%	73.4%	2.9%	9.1%	8.0%	5.9%
Standard Deviation		9.0	5.1	23.9	4.6	5.9	9.8	10.7
Spray: Variation		18.8%	0%	0%	0%	0%	51.8%	35.0%
Standard Deviation		11.0	0	0	0	0	23.6	22.9

Table 2.5: Confoundings and Results

Chapter 3

Conclusions

Although the standard deviation of the resulting variations is very large, it is still possible to draw conclusions from the experiments because the most important variations are significantly larger than others. Moreover, while scales differ strongly between workloads, the order of variations is essentially consistent across workloads. With this in mind, the conclusions of this study are:

- The by far most important factor for **memory utilization** is the *reassignment threshold*. If this threshold is set low then already small queue imbalances are smoothed out by reassignments. Thus, queues hardly lengthen up. Investigation of single workload results showed only for traffic with the largest spraying rates *burst tolerance* and *spraying recovery* also have a big impact on memory. The earlier spraying kicks in, the earlier imbalances due to excessive flows are smoothed out.
- Not surprisingly, **reassignments** are dominated by the setting of the *reassignment threshold*. *Memory size* also has an influence because it gives more room for queue imbalances before reassignments occur.
- Interaction between the reassignment and the spraying mechanisms is much smaller than expected.
- **Spraying** is primarily impacted by *burst tolerance* and *spraying recovery*, as one would expect. From the numbers, no precedence of one over the other can be derived. If the traffic is very bursty but does not really have high-datarate flows then spraying can be avoided by burst tolerance setting. Flows with a long-term high datarate can profit from fast spraying recovery settings.
- A very interesting observation is the fact that *flow time out* does not seem to really have any influence on either of the performance metrics. This entails that it can be set to long periods without compromising load balancer performance. This way it can be avoided to disrupt flows that really have not ceased to exist, yet.
- The system can cope with the smaller of the two simulated memory sizes because memory influence on system performance is far from being dominant for all three metrics. Any negative impact can be compensated by appropriate setting of the other factors.

Bibliography

- [BDE⁺01] Werner Bux, Wolfgang E. Denzel, Ton Engbersen, Andreas Herkersdorf, and Ronald P. Luijten. Technologies and building blocks for fast packet forwarding. *IEEE Communications Magazine*, pages 70 – 77, January 2001.
- [Jai91] Raj Jain. *The Art of Computer Systems Performance Analysis*. John Wiley & Sons, New York, 1991.
- [NLA] NLANR. Measurement and operations analysis team. <http://moat.nlanr.net/>.