ON LOAD BALANCING IN DISTRIBUTED TELECOMMUNICATION SYSTEMS

Per-Oddvar Osland
Department of Telematics
Norwegian University of Science and Technology
peroo@item.ntnu.no

Peder J. Emstad
Department of Telematics
Norwegian University of Science and Technology
peder@item.ntnu.no

Abstract Distributed systems providing telecommunication services, such as IN and TINA, must satisfy strict real-time requirements. Load control and load balancing are two closely connected policies that may be applied to achieve low rejection probability for new calls, maintained response times for accepted calls, and balanced load on available resources. This paper presents requirements and performance measures for dynamic load balancing policies for telecommunication systems, as well as a brief review of the latest research in the field.

1. INTRODUCTION

Systems for provisioning of value added telecommunication services are subject to increasing use. With extensive implementation of mass services such as Number Portability (NP), Virtual Private Networks (VPN), Universal Personal Telecommunications (UPT), and Televoting, almost every phone call will require some processing. In architectures such as the Telecommunication Information Networking Architecture (TINA) and the Intelligent Network (IN), this processing will be performed at dedicated nodes. TINA provides a distributed architecture for service provisioning, and recent publications (Bonnet and Marchese, 1998; Arvidsson et al., 1999; Osland and Emstad, 1999) argue that IN should also be distributed in order to provide dependability and improved performance. Calls made may now be viewed as a flow of service

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35581-8_35

© IFIP International Federation for Information Processing 2000

requests, initiated by users throughout the network, towards the processing nodes. We assume sufficient network capacity, and hence these nodes will be potential bottlenecks in high load situations.

People are used to the telephone as a quick and reliable way to communicate. Frequent call rejection and/or long response times are perceived as indications of severe performance degradation. A major challenge when providing value added telecommunication services will be to meet customers demand for availability and QoS. By QoS we mean minimal rejection of new calls and, for admitted calls, to keep response times at an acceptable level. Optimal resource usage is needed to meet these requirements.

Load balancing and load control policies may be employed to fulfill the above stated demands. A *load control* (LC) policy regulates the admission of new service requests; it attempts to reject as few requests as possible while at the same time assuring sufficiently low response times for admitted requests. In order to optimise overall network performance, the policy should operate at a network level rather than on a node by node basis. A *load balancing* (LB) policy distributes the accepted service requests on the available processing nodes in order to provide fairness in terms of equal expected response-times, and to keep system utilisation up, i.e. avoid situations with some idle nodes while others are heavily loaded.

These policies are closely coupled, and are sometimes considered as two aspects of one policy. LC is a mature and well studied field, whereas LB for telecom systems has not received the same attention until quite recently. The scope of this paper is to render some basic concepts on load balancing, and to indicate cautions to be taken when designing LB policies for distributed telecommunication systems. The objectives of an integrated load control/load balancing policy will be addressed.

The paper is organised as follows: In Section 2. we present some issues to be considered when designing LB policies for telecom systems. Terminology, classification, requirements and performance measures for load balancing policies are addressed in Section 3. A literature review is given in Section 4. before concluding remarks and indications for further work which are found in Section 5.

2. LOAD BALANCING POLICIES FOR DISTRIBUTED TELECOM SYSTEMS

Today we observe a steady convergence between the computer science and telecommunication worlds, where a range of audiovisual services are offered through different networks. Some of these services are real-time

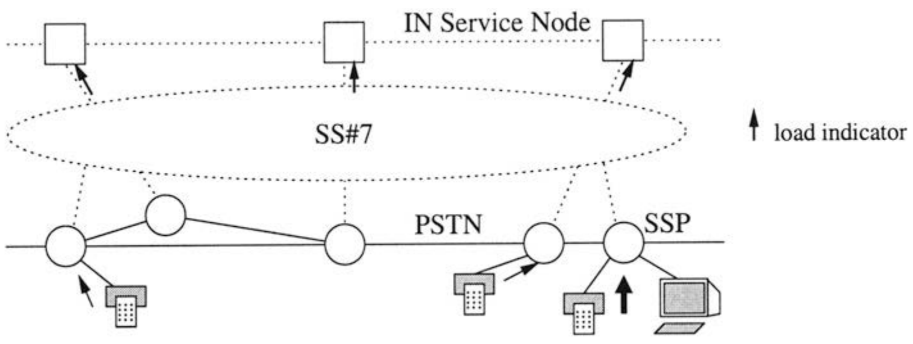

Figure 1 Load balancing improves IN performance.

telecom applications like NP, VPN, and UPT. For systems providing these services, such as IN and TINA, there must be specific requirements to policies for load balancing and control. In this section we present some issues that must be considered when formulating these requirements.

Load balancing and control in distributed systems is a vast and well studied area. As pointed out by Rumeswicz, 1999, the literature is dominated by two views. On the one hand, LB is often used to minimise response times for jobs (Eager et al., 1986; Kunz, 1991). This results in low response times, but if LC is not an issue, it also leads to severe performance degradation in overload situations. In other contributions, in the telecom literature in particular (see e.g. Berger, 1991; Rumsewicz and Smith, 1995), load control has received much attention. Here emphasis has been on keeping system throughput and response times at an appropriate level for different traffic load cases.

An obvious suggestion would be to apply LB results from the distributed systems literature to problems in the telecom world. To understand why this is not immediate, we need to review some important properties of architectures providing value added telecom services.

- As discussed above, most jobs (service requests) in the telecom world are subject to real-time requirements. Therefore emphasis should be on maintaining response times and minimising rejection probability rather than minimising response times and optimising throughput.

- A common situation that arises in distributed operating systems is when nodes receive jobs that can either be processed locally or migrated for remote processing. In a worst case scenario, this could result in a job being migrated several times before finding a node that accepts it for processing. The situation is somewhat different when dealing with services with strict real-time requirements. In IN and TINA, some nodes receive service request, and forward them to other nodes for processing. In an IN context, the SSPs will schedule jobs, while the Service Nodes will process them (see Figure 1). A service request can be scheduled for a processing node only once. Once scheduled, further migration or job rejection is intolerable.

- A computer network considered is usually a LAN or WAN, while a telecom network, often covering an entire country, is far more widespread. Network capacity is typically higher in the former case, e.g. compare a 10Mb/s WAN with 64 kb/s in Signalling System No 7. Therefore network delays should be taken into account when doing performance evaluation of load balancing policies in telecom networks. This aspect is seldom considered as crucial, and has been treated only by a few publications. Mirchandaney et al., 1989, account for delay when distributing jobs, but assumes negligible delays when distributing load information between the nodes. Kremien and Kramer, 1992, investigate several published load scheduling policies while taking network delays into account.

3. TERMINOLOGY, CLASSIFICATION AND QUANTIFICATION

The proper design of a load balancing policy depends on the target system. In this section we first clarify some terminology, then describe a general classification of LB policies. Furthermore a set of explicit requirements and performance measures for LB policies in distributed telecom systems is presented.

Load distribution is the migration of jobs to a set of nodes for remote processing. *Load scheduling*, more precisely called spatial load scheduling, is the act of selecting a specific node for job transfer. Temporal load scheduling, which means to permute jobs (e.g. according to job size in order to maximise throughput), is not within the scope of this article. The scheduling can be classified as deterministic (e.g. Round Robin) or probabilistic. In the latter case, *scheduling probabilities* indicate the probability for selecting a specific node for job transfer. According to Eager et al., 1986, the purpose of *load sharing* is to obtain high system utilisation, or to never let a node be idle when a job is waiting. There is no restriction on the number of jobs in each node, nor is there a requirement that all jobs should get the same response times. *Load balancing* means to strive for equal load on nodes, which

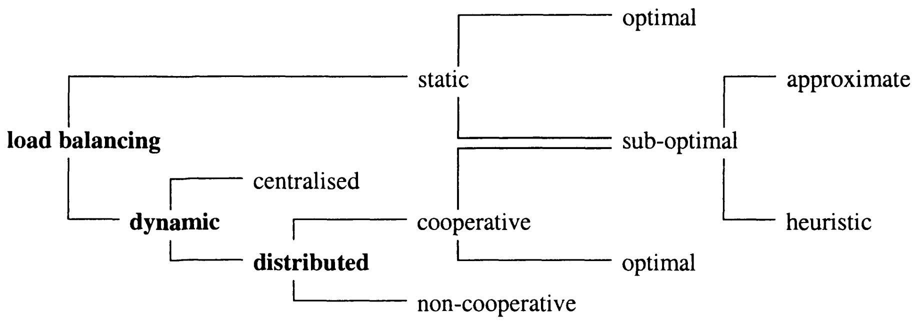

Figure 2 Casavant's classification of load balancing policies

in a homogeneous environment (equal capacity on servers) means equal queue lengths. A node is modelled as a queue plus a server, hence the response time (aka sojourn time) is the sum of queueing and processing time. In a non-homogeneous system we say that the load is balanced when a new job will have the same expected response time, independent of which node it will be sent to. This is equivalent to # of jobs in a node times expected service time being the same for all nodes.

3.1 Classification

Using the classification of Casavant and Kuhl, 1988 (see Figure 2), we learn from the so far sparse literature on LB in telecom systems (Kihl et al., 1999; Rumeswicz, 1999; Arvidsson et al., 1999; Osland and Emstad, 1999), that dynamic, distributed load balancing policies seem to be best suited. A *dynamic* (or adaptive) policy takes system state into account when scheduling new service requests. A *distributed* algorithm resides and acts in several nodes in the network. In particular, a policy is *symmetrically distributed* if the same policy resides in all nodes. In an IN context, the SSPs distribute load on the Service Nodes. Then the LB policy could be placed in the SSPs. A policy is cooperative if it allows nodes to migrate load rather than processing all jobs locally. Since the choice for a scheduling node in IN or TINA is either to migrate or reject a job, this classification is irrelevant for the scope of this article. A thorough classification and qualification of policies is also performed by Kremien and Kramer, 1992, who conclude that dynamic, symmetrically distributed LB policies are the most promising for computer systems.

An adaptive policy requires the distribution of load information from processing nodes to scheduling nodes. In a highly distributed telecom environment this could be a demanding process, and network delays introduce the problem of out-aged state information. Hence the frequency and amount of information to be exchanged is determined by a trade-off between overhead (transport and processing of state information) and the usefulness of the information.

3.2 Requirements

Most of these requirements are taken from Kremien and Kramer, 1992; Arvidsson et al., 1999; Rumeswicz, 1999.

1. The policy must be simple to implement.
2. The policy must be quick and require little system resources in terms of memory for algorithm construction and run-time cost.
3. The amount and exchange frequency of state information must be moderate enough to only constitute a small fraction of the entire traffic in the system. Policies that require frequent exchange and much system information must be significantly better than less demanding policies.
4. The hit-ratio, i.e. the fraction of correct decisions made by the policy must be better than that of a random policy.
5. Adaptability: The policy must work well for different traffic patterns.
6. Scalability: Change in size of the system should not affect policy performance significantly.
7. Stability: As a result of load balancing, the state in the system should gravitate toward the stable state of the system. Oscillating or diverging behavior indicates poor performance.

The following requirements are particularly important to a load balancing policy for telecom architectures:

1. The primary goal should be to satisfy customers demands, and may be formulated as the following optimisation problem: *Minimise rejection probabilities while maintaining response times at or below a given limit.*
2. Optimise resource utilisation, i.e. maximise utilisation while maintaining response times.
3. Balanced load on processing nodes.

4. Accepted calls must be given priority over fresh calls.
5. Fairness: Unless priority is an issue, different users and services should be treated fairly.

The demand for limited response times implies that processors may have to run at a load lower than the load that maximises throughput.

3.3 Performance measures

To validate a policy, performance measures for the following baseline policies can be compared: *no balancing, random*, and *ideal.* To be useful, a dynamic policy must be better than the random policy, and by far outperform a situation where no load balancing is used. The *ideal* policy always "sees" the best solution for every scheduled job. Comparison with an ideal situation serves as a measure for how close a policy is to an unachievable bound. Below we give some general indications for evaluation of LB policies with respect to the requirements stated in Section 3.2.

Response times: Let S be the response time for a job in a node, and s^{limit} the limit that S should not exceed. Since emphasis should be on maintaining S rather then minimising it, $P(S > s^{limit})$ is a more suitable performance measure than $E[S]$. A contribution that supports this choice may be found in Hvasshovd et al., 1995, where requirements for a telecom database are stated. They suggest $s^{limit} = 15$ms and require $P(S > s^{limit}) < 5\%$. The following inequalities may be used when comparing with the no balancing and random policies:

$$\frac{P(S > s^{limit} \mid \text{LB})}{P(S > s^{limit} \mid \text{no balancing})} \leq T_1$$

and

$$\frac{P(S > s^{limit} \mid \text{LB})}{P(S > s^{limit} \mid \text{random})} \leq T_2$$

As indicative values, we suggest $T_1 = 0.1$ and $T_2 = 0.5$. Furthermore, the maximum difference in expected response time at the servers must be small, say less than 20%.

Rejection probability: $P(rejection)$ for call attempts:

$$\frac{P(rejection \mid \text{LB})}{P(rejection \mid \text{no balancing})} \leq R_1$$

and

$$\frac{P(rejection \mid \text{LB})}{P(rejection \mid \text{random})} \leq R_2$$

As indicative values, we suggest $R_1 = 0.1$ and $R_2 = 0.5$.

Hit-ratio: When scheduling a job for a server, we define a correct hit as the scheduling decision made by the ideal policy. As a general requirement the hit-ratio for a LB policy should be between .5 an 1, and must be better than for the no balancing and random policies.

Scalability: A scalable policy works well for systems of different sizes, and is indicated by

$$\frac{\text{performance(LB(scalability-parameter A))}}{\text{performance(LB(scalability-parameter B))}} = 1 \pm \alpha$$

where $\alpha \ll 1$, and A $\gg$ B. Number of nodes, server capacity and call arrival intensity are typical scale parameters. Slight changes in policy settings may be an acceptable compensation for sensitivity to scale.

Fairness: For a mix of services, users requesting the same service should experience the same response time and rejection probability. Every service type should be entitled a fair share of the entire processing capacity. In particular, situations where some dominating services (e.g. Televoting) suffocate other services must be avoided.

Balanced load: The load on the processing nodes must be balanced. If utilisation differs significantly from node to node, higher utilisation at the fastest servers should show.

4. LOAD BALANCING: A LITERATURE REVIEW

Dynamic load balancing policies comprise two main activities; information exchange and decision making. The policies reviewed in this section mainly differ in the way these activities are implemented.

Information exchange strategies determine how to store, exchange, and update information such that the scheduling nodes keep a local view of the system's state. Queue length is a representative and much used metric for system state. Information exchange strategies can be characterised as either periodic or state-driven.

The *decision making* part of the policy indicates to which node a job should be scheduled for processing, or if it should be rejected. A typical strategy is adaptive, probabilistic scheduling, where scheduling probabilities are weighted in order to reflect the load status in the system. The scheduling probabilities indicate the probability for selecting a specific node for job transfer. In an adaptive policy the scheduling probabilities are updated regularly as new state information appears.

Kihl et al., 1999, use no explicit load information exchange when evaluating load balancing algorithms for TINA networks but, rather, count the number of jobs rejected when sent for remote processing. This simplifies the policies, and also makes them well suited for scalable networks. However, state information is exchanged only at high load since there will be no rejections when the load is low. The rejection counts are used to define scheduling probabilities in two policies. These are compared with the random and ideal baseline policies. Tests show that it is difficult to find a policy that works well for a variety of traffic situations. As an improvement, Kihl et al recommend adding explicit load information to the TINA protocol.

A neat way to obtain additional load information (still implicit) is presented by Hosein, 1999, who considers overload control for an IN database. He lets the response time of a job at a processing node indicate the load on the node. This works both for high and low load situations, and the policy shows good performance under varying server intensity. However, if network delay is significant, this method may provide inaccurate information.

Arvidsson et al., 1999, propose to use agent technology for the exchange of load information. They use a 'Market-Based Control' approach to distribute IN processing capacity, represented by tokens. The information dissemination is periodic; every 10 seconds an auction is held where agents representing the SCPs and SSPs sell and buy tokens. By letting the SCPs offer available processing capacity in terms of tokens, LC and LB is achieved. The agents acting for the SSPs express the SSPs' need for processing capacity by placing bids in the auction. Through mechanisms in the auction the following goals are achieved: SCP utilisation close to .9, maximum network profit, minimum response delays, fairness to services and users, and equally balanced load. A great advantage is that there is no need for synchronisation of the parts in the auction. The strategy turns out to work well for different traffic load scenarios. Still it remains to estimate cost for holding the auction and for the activity of the agents (signalling between agents, extra memory and processing required) in order to justify the policy.

Up to this date, the most comprehensive policy is probably presented by Rumeswicz, 1999. He models a call as composed of several jobs that can be processed at different nodes, and gives preference to calls which have received a significant amount of processing over recently commencing calls. The policy is scalable because it makes no a priori assumption about processor capacity or placement of functionality on processors. It also works well for different traffic mixes, hence it is adaptive. When it comes to information exchange, the processing nodes distribute a relative load indication, each node having its own frequency of emitting load information. The policy does not require synchronisation of controls taken, it attempts to maximise the number of calls completed, and it is simple to implement.

In a previous paper on performance evaluation of load scheduling policies for IN (Osland and Emstad, 1999), we describe a policy where an SSP explicitly asks for load information from the Service Nodes. This makes the load distribution event-driven and synchronises it in the sense that Service Nodes distribute their load simultaneously. Synchronised information dissemination makes the policy suitable for analytical performance evaluation with network delays taken into account. Upon reception of state information, this policy compares the expected response time in the Service Nodes with predefined threshold values. Scheduling and rejection probabilities are then set in order to minimise rejection probability while keeping response times below a given limit. Analytical and simulation results show that this policy satisfies it's goals, and the utilisation is rather good compared with an optimal policy. A weak point is that the policy does not scale well when system size changes, and that frequent update of state information is assumed.

5. CONCLUSIONS AND FUTURE WORK

In systems providing telecommunication services, such as IN and TINA, load balancing and control are two closely connected policies used to satisfy critical requirements such as maintained response times, low rejection probability and balanced resource usage. In this paper we have presented requirements and performance measures for dynamic load balancing policies for telecommunication systems. A brief review of the latest publications in the field has also been given. The review features scalable, adaptive, and efficient policies. Researchers do not agree if emphasis should be on maintaining response time and minimising rejection probability, or to maintain goodput while minimising response times. We support the former, which means optimal QoS from a customers point of view. The latter means optimal resource usage from a operator's point of view.

A topic for further work is to compare the algorithms to show their pros and cons with respect to relevant performance measures. Work is in progress to formulate and study more efficient policies that take the problem of network delays into account.

Acknowledgements

We would like to thank Professor Don Towsley of University of Massachusetts at Amherst for his helpful comments and suggestions.

References

Arvidsson, Å., Jennings, B., Angelin, L., and Svennson, M. (1999). On the use of agent technology for in load control. In Key, P. and Smith, D., editors, *Teletraffic Engineering in a Competitive World. Proceedings of the 16th International Teletraffic Congress*, volume 3b, pages 1093–1104, Edinburgh, Scotland. IEE, Elsevier.

Berger, A. (1991). Comparison of call gapping and percent blocking for overload control in distributed switching systems and telecommunications networks. *IEEE Transactions on Communications*, COM-39, 4:574–580.

Bonnet, M. and Marchese, P. (1998). A model of Intelligent Network distributed computing for Number Portability. In *Procedings of the 5th International Conference on Intelligence in Networks*, pages 378 – 383, Bordeaux, France.

Casavant, T. L. and Kuhl, J. G. (1988). A taxonomy of scheduling general-purpose distributed computing systems. *IEEE Transactions on Software Engineering*, 14(2):141–154.

Eager, D. L., Lazowska, E. D., and Zahorjan, J. (1986). Adaptive Load Sharing in Homogeneous Distributed Systems. *IEEE Transactions on Software Engineering*, 12(5):662–675.

Hosein, P. (1999). Overload control for real-time telecommunication databases. In Key, P. and Smith, D., editors, *Teletraffic Engineering in a Competitive World. Proceedings of the 16th International Teletraffic Congress*, volume 3a, pages 263–272, Edinburgh, Scotland. IEE, Elsevier.

Hvasshovd, S.-O., Torbjørnsen, Ø., Bratsberg, S., and Holager, P. (1995). The ClustRa Telecom Database: High Availability, High Throughput, and Real-Time Response. In *Proceedings of the 21st International Conference on Very Large Data Bases (VLDB'95)*, pages 469–477, Zurich, Switzerland.

Kihl, M., Widell, N., and Nyberg, C. (1999). Load balancing algorithms for TINA networks. In Key, P. and Smith, D., editors, *Teletraffic Engineering in a Competitive World. Proceedings of the 16th Interna-*

tional Teletraffic Congress, volume 3b, pages 999–1008, Edinburgh, Scotland. IEE, Elsevier.

Kremien, O. and Kramer, J. (1992). Methodical analysis of adaptive load sharing algorithms. *IEEE Transactions on Parallel and Distributed Systems*, 3(6):747–760.

Kunz, T. (1991). The influence of different workload descriptions on a heuristic load balancing scheme. *IEEE Transactions on Software Engineering*, 17(7):725–730.

Mirchandaney, R., Towsley, D., and Stankovic, J. A. (1989). Analysis of the effects of delays on load sharing. *IEEE Transactions on Computers*, 38(11):1513–1525.

Osland, P.-O. and Emstad, P. J. (1999). Performance evaluation of load scheduling policies for Intelligent Networks. In Key, P. and Smith, D., editors, *Teletraffic Engineering in a Competitive World. Proceedings of the 16th International Teletraffic Congress*, volume 3a, pages 283–292, Edinburgh, Scotland. IEE, Elsevier.

Rumeswicz, M. (1999). Load control and load sharing for heterogenous distributed systems. In Key, P. and Smith, D., editors, *Teletraffic Engineering in a Competitive World. Proceedings of the 16th International Teletraffic Congress*, volume 3b, pages 1083–1092, Edinburgh, Scotland. IEE, Elsevier.

Rumsewicz, M. P. and Smith, D. E. (1995). A comparison of SS7 congestion control options during mass call-in situations. *IEEE/ACM Transactions on Networking*, 3, 1:1–9.