Modelling cache expiration dates policies in content networks.

Pablo Rodríguez-Bocca, *Universidad de la República,Uruguay. prbocca@fing.edu.uy*
Héctor Cancela, *Universidad de la República,Uruguay. cancela@fing.edu.uy*

Keywords: Peer-to-peer networks, mathematical programming, optimization.

Introduction

Content networks are usually virtual networks based over the IP infrastructure of Internet or of a corporative network, which use mechanisms to allow accessing a content when there is no fixed, single, link between the content and the host or the hosts where this content is located. Even more, the content is usually subject to re-allocations, replications, and even deletions from the different nodes of the network [6][7][9]. In the last years many different kinds of content networks have been developed and deployed in widely varying contexts: they include peer-to-peer networks, collaborative networks, cooperative Web caching, content distribution networks, subscribe-publish networks, content-based sensor networks, backup networks, distributed computing, instant messaging, and multiplayer games.

As a general rule, every content network is actually a knowledge network, where the knowledge is the information about the location of the nodes where each specific content is to be found: this is "meta-information", in the sense of being the information about the information contents themselves.

The objective of the network is to be able to answer each content query with the most complete possible set of nodes where this content is to be found; this corresponds to discover the content location in the most effective and efficient possible way.

As both nodes and contents are continuously going in and out of the network, the task of maintaining updated the network meta-information is very difficult and represents an important communication cost. In this context, cache nodes are used to hold the available meta-information; as this information is continuously getting outdated, the cache nodes must decide when to discard it, which means increasing communication overhead for the sake of improving the quality of the answers. The policy employed for determining these cache expiration dates have a large impact in the performance of a network; this problem is related but at the same time different to other problems involving caches, as in this case the cached information does not take too much place, but gets outdated very quickly (usually in caching problems the cached information uses up significant space, and is valid for more extended time periods).

In this work, we model different policies for fixing the cache expiration dates for the information about the contents' location; in particular, we give mathematical programming formulations which represent the case where the cache expiration dates are equal for all contents, the case where the cache expiration dates are proportional to the query frequency for a content, and the case where all cache expiration dates can be fixed independently in order to maximize the correct answers to the queries received. We give also a numerical illustration of the application of these policies, employing data available from empirical studies of other aspects of content networks, and we look at the difference between the solutions obtained in each case. Finally, we give some conclusions and directions for future work.

Content Caching Problem Formulation

We consider a content network composed of source nodes and querying nodes, of cache nodes (also called aggregation nodes), and of a backbone (which will not be modeled in detail). Figure 1 presents a graphical representation of such a network, where it is important to note that this conceptual division, not necessarily has a correspondence at the equipment level, as the same computer may act at the same time as a source node, a querying node, a cache node, and a backbone node.

Querying nodes connect to at least one cache node in order to route their queries. Each cache node concentrates all queries of its connected nodes and consults the backbone when it is not able to directly answer the queries received. The behavior of cache nodes is simple: when a query over content *k* arrives, if the answer is present in the cache it is returned; otherwise, the cache node starts a search in the backbone to obtain the information and answer the query; this information is then stored in the cache, for a prefixed time d_k, afterwards it expires.

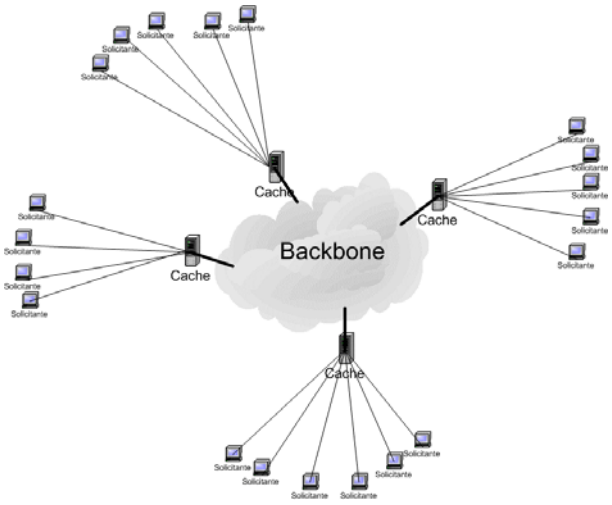

Figure 1: Simplified view of a content network

In two previous works [11][12] we have developed a formulation for the problem of caching meta-information in a content network in order to maximize the number of correct answers to the queries, while respecting the bandwidth constraints. Even if space reasons preclude giving here the modeling details, we include the main points of the model, which is formulated in terms of the following parameters (whose dimensional units are given between brackets):

- C set of network contents, partitioned into K content classes, such that if two contents belong to the same class, their parameters are identical ($C = C_1 \cup C_2 ... \cup C_K$).
- l_k: number of contents belonging to class k ($[l_k] = 1$).
- f_k: query rate for class k contents k ($[f_k] = 1/\text{sec.}$).
- λ_k: rate for source arrival for class k contents ($[\lambda_k] = 1/\text{sec.}$).
- μ_k: rate for content deletion in sources for class k contents. ($[\mu_k] = 1/\text{sec.}$).
- α_S: size per location answered in response to a content query ($[\alpha_S] = bytes$).
- α_B: size per location answered in response to a backbone search ($[\alpha_B] = bytes$).
- β_S: size of a content query packet ($[\beta_S] = bytes$).
- β_B: size of a backbone search packet ($[\beta_B] = bytes$).
- BW_{IN}, BW_{OUT}: input and output bandwidth restrictions in the cache node ($[BW_{IN}] = [BW_{OUT}] = bytes/\text{sec.}$).
- d_k: cache expiration times for class k contents. ($[d_k] = \text{sec.}$)

With some additional hypothesis about the query arrivals and the content storage times, the model leads to Poisson and birth-and-death processes, which support direct computation of the used bandwidth and of the distribution of the number of valid content locations stored at the cache nodes. The network primary objective is to be able to give the most complete and correct information to the queries received. To formalize this objective, we develop an expression for the expected number of correct answers taking into account all contents, leading to this expression:

$$\sum_{k\in C}\frac{l_k\lambda_k}{\mu_k{}^2 d_k}\left[\mu_k\left(1-e^{-f_k d_k}\right)+ f_k\left(1-e^{-\mu_k d_k}\right)-\frac{1}{d_k}\left(1-e^{-f_k d_k}\right)\left(1-e^{-\mu_k d_k}\right)\right].$$

Cache nodes have input and output bandwidth constraints, which can limit the number of queries they can receive, process, answer and eventually pass on to the backbone. Using the previous hypothesis, the bandwidth constraints can be formulated as follows:

$$\beta_S\sum_{k\in C} l_k f_k+\alpha_B\sum_{k\in C}\frac{l_k f_k}{1+d_k f_k}\overline{A_k}\le BW_{IN}\,,\quad \alpha_S\sum_{k\in C} l_k f_k\overline{A_k}+\beta_B\sum_{k\in C}\frac{l_k f_k}{1+d_k f_k}\le BW_{OUT}\,.$$

Policies for fixing Cache expiration dates

Based on the previous model, we now discuss three alternative ways to fix the cache expiration dates.

Single expiration time policy (SETP)

The simplest option for solving the problem is setting the same expiration time for all contents. In this case, the aggregation node defines a fixed expiration time d; for every content location query, the information is stored in the cache during this time, and then deleted. This amounts to having all variables $d_k = d$ in the previous formula, leading to the following mathematical programming formulation:

$$\max_{d\in\Re^+}\left\{\frac{\sum_{k\in K}\frac{l_k\lambda_k}{\mu_k{}^2 d}\left[\mu_k\left(1-e^{-f_k d}\right)+ f_k\left(1-e^{-\mu_k d}\right)-\frac{1}{d}\left(1-e^{-f_k d}\right)\left(1-e^{-\mu_k d}\right)\right]}{\sum_{k\in K}\frac{\lambda_k}{\mu_k}l_k f_k}\right\}$$

s.t.

$$\beta_S\sum_{k\in K} l_k f_k+\alpha_B\sum_{k\in K}\frac{l_k f_k}{1+df_k}\frac{\lambda_k}{\mu_k}\le BW_{IN}$$

$$\alpha_S\sum_{k\in K} l_k f_k\frac{\lambda_k}{\mu_k}+\beta_B\sum_{k\in K}\frac{l_k f_k}{1+df_k}\le BW_{OUT}$$

$d\in\Re^+$ decision variable,

$$l_k, f_k, \lambda_k, \mu_k, \alpha_S, \alpha_B, \beta_S, \beta_B, BW_{IN}, BW_{OUT}\in\Re^+\ \ \forall k\in K \qquad \text{SETP (1)}$$

Expiration time proportional to query rates policy (PETP)

The query rate has an important impact in the behavior of the different contents. It can be reasonable to assume that high query rates correspond to contents with high impact and that their expiration times must then be kept longer; the easiest way is to impose a linear

dependency. In this case, there is a single coefficient *e* such that for every content *k*, $d_k = ef_k$. Then, the mathematical programming formulation is as follows

$$\max_{e\in\Re^+}\left\{\frac{\sum_{k\in K}\frac{l_k\lambda_k}{e\mu_k{}^2 f_k}\left[\mu_k\left(1-e^{-ef_k{}^2}\right)+ f_k\left(1-e^{-e\mu_k f_k}\right)-\frac{1}{ef_k}\left(1-e^{-ef_k{}^2}\right)\left(1-e^{-e\mu_k f_k}\right)\right]}{\sum_{k\in K}\frac{\lambda_k}{\mu_k}l_k f_k}\right\}$$

s.t.

$$\beta_S\sum_{k\in K}l_k f_k+\alpha_B\sum_{k\in K}\frac{l_k f_k}{1+ef_k{}^2}\frac{\lambda_k}{\mu_k}\le BW_{IN}$$

$$\alpha_S\sum_{k\in K}l_k f_k\frac{\lambda_k}{\mu_k}+\beta_B\sum_{k\in K}\frac{l_k f_k}{1+ef_k{}^2}\le BW_{OUT}$$

$e\in\Re^+$ decision variable,

$l_k, f_k, \lambda_k, \mu_k, \alpha_S, \alpha_B, \beta_S, \beta_B, BW_{IN}, BW_{OUT} \in \Re^+ \quad \forall k\in K$ PETP (2)

Optimal policy (OP)

If all values d_k are free, unrelated variables, solving the optimization model will give the theoretical optimum for the cache expiration policy problem. The formulation is then the most general one, namely:

$$\max_{d_k\in\Re^+}\left\{\frac{\sum_{k\in K}\frac{l_k\lambda_k}{\mu_k{}^2 d_k}\left[\mu_k\left(1-e^{-f_k d_k}\right)+ f_k\left(1-e^{-\mu_k d_k}\right)-\frac{1}{d_k}\left(1-e^{-f_k d_k}\right)\left(1-e^{-\mu_k d_k}\right)\right]}{\sum_{k\in K}\frac{\lambda_k}{\mu_k}l_k f_k}\right\}$$

s.t.

$$\beta_S\sum_{k\in K}l_k f_k+\alpha_B\sum_{k\in K}\frac{l_k f_k}{1+d_k f_k}\frac{\lambda_k}{\mu_k}\le BW_{IN}$$

$$\alpha_S\sum_{k\in K}l_k f_k\frac{\lambda_k}{\mu_k}+\beta_B\sum_{k\in K}\frac{l_k f_k}{1+d_k f_k}\le BW_{OUT}$$

$d_k\in\Re^+$ decision variables, for $k\in K$

$l_k, f_k, \lambda_k, \mu_k, \alpha_S, \alpha_B, \beta_S, \beta_B, BW_{IN}, BW_{OUT} \in \Re^+ \quad \forall k\in K$ OP (3)

The three problems are non-linear optimization problem, both in the restrictions and in the objective function. We can see that both the feasible solution space and the objective function are convex. As the problem is stated as a maximization one, a convex objective function will in general lead to multiple local optima.

Numerical Illustration

In this section we present a numerical illustration over a case study, where the data was generated with information available in different literature sources especially referring to Gnutella or similar peer-to-peer (P2P) file sharing networks [1][6][8] [10] [13].

The instance we discuss here was generated using a random number generator; including the data for 878691 different contents (which correspond to the number of Gnutella

contents in the study by Chu [1]), where the distributions for the query frequency follow a modified Pareto distribution law taking into account the "fetch-at-most-once" effect [6]. The frequency of arrival of new storage locations for each content is linearly related to the query frequency, following the hypothesis mostly used in the literature. The main parameters of the case study are average content query rate 0.037938251 hr^{-1}, maximum content query rate 1000 hr^{-1}, average content storage rate 11.09749966 hr^{-1}, average content location validity rate 1 hr^{-1}, maximum allowed number of locations answered in response to a content query 200, size of a the answer to a content query 100 bytes, size of the answer of a backbone search 310 bytes, size of a content query 94 bytes, size of a backbone search packet 291.4 bytes, input bandwidth 921600000 bytes/hr, output bandwidth 460800000 bytes/hr (an ADSL 2/1 Mbps connection as reference value).
The problem was programmed using the AMPL modeling language; AMPL [4] is a software with a algebraic modelling language for doing mathematical programming, which can be used to easily represent a non-linear mathematical programming problem such as the ones discussed here. We solved the problem using different solvers available at NEOS [3][5]. The results presented here have been computed using the SNOPT solver.
We compare the three optimization policies above described. As the full problem generated has a large number of contents (878691), we classify the contents into 16 classes, and we solve this reduced problem. The results are then cast back in terms of the original, 878691 contents, problem (which could not be solved directly due to file size limitations in the used optimization interfaces).
Table 1 shows the objective values and the computing times for the three policies, in the 16 class problem. The execution times are short, even if it can be seen that the OP policy takes longest to compute to optimality. Table 2 shows the objective funtion values, as computed in the original, 878691 contents, case. As expected, it can be seen that the OP policy obtained the best results, with $\varepsilon = 0.99995405$. All the same, the other policies resulted in values not far away from this optimum.

OP		*SETP*		*PETP*	
Objective Function ε	Execution Time (s)	Objective Function ε	Execution Time (s)	Objective Function ε	Execution Time (s)
0.99995470	0.640	**0.99995210**	0.140	**0.99993000**	0.050

Table 1: SNOPT slutions. For the policies: OP, SETP and PETP.

OP		*SETP*		*PETP*	
Objective Function ε	Original Objective Function ε	Objective Function ε	Original Objective Function ε	Objective Function ε	Original Objective Function ε
0.99995470	**0.99995405**	0.99995210	**0.99995200**	0.99993000	**0.99993040**

Table 2: SNOPT solutions transformed to the original problem. For the policies: OP, SETP and PETP.

Conclusions

This paper studies how to model different alternative cache expiration time policies for content network cache nodes. The three models presented correspond to non-linear mathematical programming models, with different number of decision variables.

We also generate a test example, based on data available from the literature; using standard modelling languages (AMPL) and software available at the NEOS webpage, we compare the different policies, which in this case show an small advantage of the optimal policy with respect to the two simplest ones.
Future work includes studying more advanced non-linear programming solution methods which could be used to solve directly the problem with a large number of variables; improving the model to take into account other features of content networks; and implementing the caching policies at a cache node in a real network, in order to study the benefits in practice (and the possible side effects) of these policies with respect to current implementations.

References

[1] Chu J., Labonte K., and Levine, B., *Availability and locality measurements of peer-to-peer file systems*, ITCom: Scalability and Traffic Control in IP Networks. Proc. of SPIE, Vol. 4868, July 2002.

[2] J. Czyzyk, M. Mesnier, and J. Moré, *The NEOS Server*. IEEE Journal on Computational Science and Engineering, 5 (1998), pages 68-75

[3] Fourer, R., Gay, D.M, and. Kernighan, B.W. *AMPL: A Modeling Language for Mathematical Programming*. Duxbury Press / Brooks/Cole Publishing Company, 2002.

[4] W. Gropp and J. Moré, *Optimization Environments and the NEOS Server*. *Approximation Theory and Optimization*, M. D. Buhmann and A. Iserles, eds., pages 167-182, Cambridge University Press, 1997.

[5] Gummadi K., Dunn R., Saroiu S., Gribble S., Levy H., and Zahorjan J. *Measurement, Modeling and Analysis of a Peer-to-Peer File-Sharing Workload*. Proceedings of the 19th ACM Symposium of Operating Systems Principles (SOSP), Bolton Landing, NY, October 2003.

[6] H. T. Kung, et al. *MotusNet: A Content Network*. Technical report. Harvard University. 2001. http://citeseer.nj.nec.com/443175.html.

[7] Kung, H. T., and Wu, C. H. (2002). *Content Networks: Taxonomy and New Approaches*. Chapter in The Internet as a Large-Scale Complex System, Kihong Park and Walter Willinger (Editors), Oxford University Press. 2002.

[8] Lv Q., Cao P., Cohen E., Li K., and Shenker S.. *Search and replication in unstructured peer-to-peer networks*. In Proceedings of the 16th annual ACM International Conference on supercomputing, 2002.

[9] D. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, Z. Xu. *Peer-to-Peer Computing*. Techical report HPL-2002-57, HP Labs. 2002. http://www.hpl.hp.com/techreports/2002/HPL-2002-57.html.

[10] Qiu, L., Padmanabham, V. N. , and Voelker, G. M. *On the placement of web server replicas*. In Proc. 20th IEEE INFOCOM, 2001.

[11] Rodríguez-Bocca P., Cancela Bosi H., *A mathematical programming formulation of optimal cache expiration dates in content networks*, LANC 2005, IFIP/ACM Latin America Networking Conference. Cali, Colombia, October 10-14, 2005.

[12] Rodríguez-Bocca P., Cancela Bosi H., *Optimization of cache expiration dates in content networks*, QEST 2006, 3rd International Conference on the Quantitative Evaluation of SysTems. University of California, Riverside, United States. September 11-14, 2006.

[13] Sen, S. and Wong, J. *Analyzing peer-to-peer traffic across large networks*. http://citeseer.nj.nec.com/sen02analyzing.html