The Trade-Off between Power Consumption and Latency in Computer Networks

Matthias Herlich and Holger Karl

Universität Paderborn, Fachgebiet Rechnernetze, EIM-I
Pohlweg 47-49, 33098 Paderborn, Germany
{matthias.herlich,holger.karl}@uni-paderborn.de

Abstract. As the power consumed by computer networks is nearly independent from the load, networks consume much more power than necessary when the load is low. We analyze the influence of disabling network components on power consumption and latency of data transfers. We define two power consumption models and compare the power consumption necessary to achieve given upper limits of latency.

Our results show that a trade-off between power consumption and latency exists; it is e.g. possible to conserve 39% of power when accepting a 20% latency increase in the `nobel-germany` network. We conclude that it is possible for networks to adapt to changing demands to conserve power while the latencies increase only slightly.

Keywords: Power consumption, propagation delay, routing.

1 Introduction

Computer networks are currently designed to maximize throughput and minimize latency, while power consumption is ignored. As the power consumption is nearly independent from the demands the network has to serve [1], it consumes much more power than necessary when the demands are low.

Disabling lines is a simple adaption to consume less power, but it increases latency, e.g., by introducing detours. We formalize the problem of disabling lines to conserve power while fulfilling all demands. Then we define network configurations that minimize power consumption and configurations that minimize latency as well as trade-offs in between. Unfortunately the corresponding decision problem (capacitated fixed-charge multi-commodity minimum-cost flow network) is NP-complete [4].

To analyze the trade-off between power consumption and latency, we begin by formalizing the problem in the next section. Then we prove a bound between different possible metrics of latency and explain the relationship between the latency and bandwidth-delay product. After that we create an optimization model from the formalization of the problem and show the results we obtain with it. Our results show that it is possible to conserve considerable power by disabling line cards in networks while increasing the latency only slightly.

V. Casares-Giner et al. (Eds.): NETWORKING 2011 Workshops, LNCS 6827, pp. 273–280, 2011.
© IFIP International Federation for Information Processing 2011

2 Formalization

Network Graph. We formalize the network as a graph $G = (V, E)$ with the vertexes V representing routers and the edges E representing line cards and wirings that connect routers. For each edge e we define the capacity $c(e)$ as the maximal data rate, the latency $l(e)$ as the propagation delay, and the power consumption $p(e)$ under full load. The network has to serve unicast, static demands $D \subseteq V \times V$; the function $a : D \to \mathbb{R}^+$ maps the demands to the required data rates.

As we are analyzing propagation delay caused by *edges*, we consider only the power consumption of *edges*. Thus, to conserve power we disable edges and call the resulting states *enabled* and *disabled*. We represent the status of edges using configurations: a *configuration* C is a set of enabled edges together with routes for the demands using only enabled edges. A special configuration is a (not nesessarily unique) configuration C_{L} which enables every edge and routes every demand on its shortest path. Additionally, we define the flow of demand d on an edge e as $f_C^d(e)$ and the utilization of edge e as $u_C(e) = \sum_{d \in D} f_C^d(e)/c(e)$.

Power Consumption Model. The most important metric of our analysis is the total power consumed by the links of the network. This simplification assumes that future routing processors will be able to reduce their power consumption when line cards are disabled. We compare two models of power consumption: the binary model and its generalization, the linear model.

In the *binary model* a disabled edge consumes no power, and an enabled edge e consumes a fixed amount of power $p(e)$ – independent of its load. This model captures today's non-load-adaptive hardware well [1].

In the *linear model* an enabled edge e consumes $p_0 \cdot p(e)$ power when idle, where $p_0 \in [0, 1]$ is the percentage of maximum power consumed when idle. Power consumption scales linearly in utilization $u_C(e)$ from $p_0 \cdot p(e)$ when idle to $p(e)$ under full load. The binary model is a special case of the linear model with $p_0 = 1$. The linear model is motivated by techniques like adaptive link rates and burst transmissions that may reduce idle power consumption [8].

An algorithm which controls the topology to conserve power must consider the idle power consumption, lest it might consume more power [7]. The reason is that for low idle power, rerouting demands to disable edges can consume more power than is saved by disabling the edges.

Latency. As we restrict our model of latency to propagation delays, the latency of a demand d is the sum of the latencies of its consituing edges; we call it $l_C(d)$. To compare the latencies in two different configurations, we define the *stretch* of a configuration as the increase factor in all latencies caused by rerouting.

There are at least five different, intuitively reasonable ways to define the stretch of a configuration C compared to the latency-minimizing configuration C_{L}. We introduce each metric and give its value for a simple example: a ring-shaped network with 8 nodes where we disable one edge. Every edge $e \in E$ has latency $l(e) = 1$ and there is a demand of $a(d) = 1$ from every node to every other node.

The first two metrics express that *no* demand should suffer from a high latency due to power conservation. We define S^{MS} as the maximum stretch a *single* demand suffers in configuration C: $S^{\mathrm{MS}}_C(D) = \max_{d\in D}(l_C(d)/l_{C_\mathrm{L}}(d))$. In contrast, S^{SM} describes the stretch the maximum of *all* demands suffers in configuration C: $S^{\mathrm{SM}}_C(D) = \max_{d\in D} l_C(d)/\max_{d\in D} l_{C_\mathrm{L}}(d)$. While S^{MS} compares each latency to its *own* latency in the latency-minimizing configuration, S^{SM} compares the maxima of *all* latencies. In the 8-Ring, S^{MS} is 7, as the highest increase in latency is from 1 to 7 and S^{SM} is $7/4 = 1.75$ as the highest latency increases from 4 to 7.

To describe the tendency of the latencies, the next two metrics use the weighted arithmetic mean (weighted with the amount of transferred data $a(d)$) instead of the maximum. This is reasonable when it is acceptable for some demands to suffer from a high latency as long as the average stays low. Analogous to S^{MS} and S^{SM}, we define (a) S^{AS} as the weighted arithmetic mean of the stretch and (b) S^{SA} as the stretch of the weighted arithmetic mean latency:

$$S^{\mathrm{AS}}_C(D) = \operatorname*{avg}_{d\in D} \frac{l_C(d)}{l_{C_\mathrm{L}}(d)} = \frac{\sum\limits_{d\in D} \frac{l_C(d)}{l_{C_\mathrm{L}}(d)} a(d)}{\sum\limits_{d\in D} a(d)}; S^{\mathrm{SA}}_C(D) = \frac{\operatorname*{avg}\limits_{d\in D} l_C(d)}{\operatorname*{avg}\limits_{d\in D} l_{C_\mathrm{L}}(d)} = \frac{\sum\limits_{d\in D} l_C(d)a(d)}{\sum\limits_{d\in D} l_{C_\mathrm{L}}(d)a(d)}$$

For the ring example S^{AS} is $10/7 \approx 1.43$ and S^{SA} is $21/16 = 1.3125$.

One of the problems with the first four metrics is that they give different values when the order of combining (avg and max) and stretch calculation is reversed. The idea to use the weighted geometric mean S^{GS} is that this order is irrelevant for S^{GS}. It is defined by

$$S^{\mathrm{GS}}_C(D) = \operatorname*{geo}_{d\in D} \frac{l_C(d)}{l_{C_\mathrm{L}}(d)} = \left(\prod_{d\in D} \left(\frac{l_C(d)}{l_{C_\mathrm{L}}(d)} \right)^{a(d)} \right)^{\frac{1}{\sum_{d\in D} a(d)}}$$

and is approximately 1.22 in the 8-Ring.

The values for the different metrics show that most metrics do not differ by much in this example. In the next section we analyze the relationships between them in general and give a bound for their ratio.

3 Analysis of Metrics for Latency

3.1 Relationships between the Latency Metrics

First note that S^{MS} is always larger than or equal to any of the other four metrics. Between any two of the other metrics, only one statement holds in general: $S^{\mathrm{GS}} \leq S^{\mathrm{AS}}$. This is a direct implication of generalizing the inequality of arithmetic and geometric means to weighted means.

We define the *skewness* of configuration C $\mathrm{skew}(C)$ as the ratio between maximal and minimal latency: $\mathrm{skew}(C) = \max\limits_{d\in D} l_C(d)/\min\limits_{d\in D} l_C(d)$. The bound

$$A \leq \mathrm{skew}(C) \cdot \mathrm{skew}(C_\mathrm{L}) \cdot B$$

holds for all combinations of A and B from the five metrics, which is easily proven for each pair. It implies that all five metrics yield similar values when the ratio between maximum and minimum latency in both the latency-minimizing configuration C_{L} and in configuration C is small. For some pairs of metrics tighter bounds are possible, but as examples exist that are arbitrarily close to this bound it is the tightest possible general bound. Note that this bound depends neither on the number nor size of demands and thus the number of demands can be arbitrarily large and all metrics still fulfill the inequality above.

We conclude that the metrics are similar for low skewness and are bounded independed of the number and size of demands. We analyze how much the metrics differ for a practical scenario in Section 5. Next we will show a connection between bandwidth-delay product and latency.

3.2 Bandwidth-Delay Product and Latency

We define the *used* bandwidth-delay product (BDP) of an edge e as the BDP which is filled with data. In contrast, the *unused* BDP is the BDP which is *not* used to transfer data and *disabled* BDP is the BDP of disabled edges. Theorem 1 shows that if the used BDP is low, the average latency will be low and so will S^{SA}.

Theorem 1. *The total used bandwidth-delay product (BDP) of all edges is proportional to the weighted arithmetic mean latency of all demands.*

Proof. Recall the definition of the weighted arithmetic mean latency $\operatorname{avg}_{d\in D} l_C(d) = \sum_{d\in D} l_C(d)a(d) / \sum_{d\in D} a(d)$ and two ways to calculate the BDP each demand introduces in the network: (1) multiplying the latency of a demand $l_C(d)$ with the amount of demands $a(d)$: $l_C(d)a(d)$ and (2) calculating the sum of the BDP demand d introduces to each single edge: $\sum_{e\in E} f_C^d(e)l(e)$. As both of them are equal for every demand we derive the following:

$$\operatorname*{avg}_{e\in E} \overbrace{u_C(e)c(e)l(e)}^{\text{used BDP on edge } e} = \frac{1}{|E|}\sum_{e\in E}\sum_{d\in D} f_C^d(e)l(e) = \frac{1}{|E|}\sum_{d\in D} l_C(d)a(d)$$

$$= \frac{\sum_{d\in D} a(d)}{|E|} \operatorname*{avg}_{d\in D} l_C(d) \sim \operatorname*{avg}_{d\in D} \overbrace{l_C(d)}^{\text{latency of demand } d}$$

□

Figure 1 illustrates how bandwidth-delay product (BDP), power consumption, and latency interact when edges are disabled. Each of the two vertical bars represents the total BDP that is available in a network and shows how it is used. Rerouting traffic and disabling edges will increase the used BDP, but also allow to disable edges. When the power consumption of each edge is proportional to its total BDP and the binary model for power consumption is assumed, the BDP of disabled edges is proportional to the conserved power.

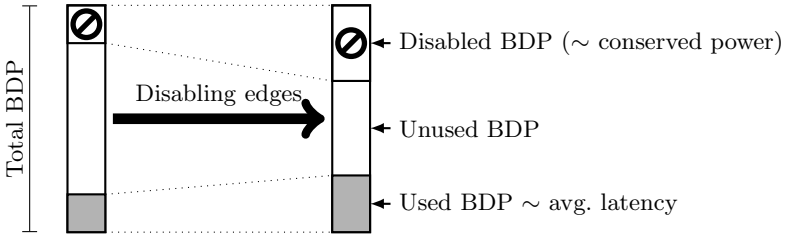

Fig. 1. Disabling edges and rerouting demands conserves power, but increases latency

4 Optimization Model

To get a general understanding of the power consumption of practical networks we formulate our problem as a mixed-integer linear problem (MILP). It is based on the following assumptions: A single algorithm controls routing and topology, the algorithm has global knowledge of end-to-end flows, the demands are static, and power consumption follows the binary model (unless stated otherwise).

$$\forall e \in E : \sum_{d \in D} \max(0, f_C^d(e)) \leq c(e) u_C(e) \tag{1}$$

$$\forall e \in E : u_C(e) \leq x_C(e) \tag{2}$$

$$\forall d \in D; u, v \in V : f_C^d((u,v)) = -f_C^d((v,u)) \tag{3}$$

$$\forall d \in D, u \in V : \sum_{v \in N(u)} f_C^d((v,u)) - \sum_{v \in N(u)} f_C^d((u,v)) = \begin{cases} -a(d), \text{ if } u = \text{src}(d) \\ a(d), \text{ if } u = \text{dest}(d) \\ 0, \text{ else} \end{cases} \tag{4}$$

$$\sum_{e \in E} u_C(e) c(e) l(e) / \operatorname*{avg}_{d \in D} \text{minLat}(d) \leq B^{\text{SA}} \cdot \sum_{d \in D} a(d) \tag{5}$$

$$\sum_{d \in D} \sum_{e \in E} l(e) \max(0, f_C^d(e)) / \text{minLat}(d) \leq B^{\text{AS}} \cdot \sum_{d \in D} a(d) \tag{6}$$

$$\min \sum_{e \in E} p(e)((1 - p_0) u_C(e) + p_0 x_C(e)) \tag{7}$$

We use a network flow model to calculate allocation of demands to edges. To express whether an edge is active or not we use the binary variable $x_C(e)$. Using the notation introduced in Section 2, we define the capacity constraint in Equation 1. We use $\max(0, f)$ to model full duplex links. Equation 2 guarantees that only active edges can transfer data. Equation 3 is the skew symmetry. We define the flow conservation in Equation 4 so that the allocation of flows meets all demands. $N(v)$ denotes the set of vertices incident to the vertex v. Note that these definitions allow multi-path routing. To specify an upper bound B^{SA} (B^{AS}) on the S^{SA} (S^{AS}) metric we use Equation 5 (6). Here $\text{minLat}(d)$ is the minimal latency necessary to route demand d. We minimize the power consumption specified in Term 7 for $p_0 = 1$ (unless stated otherwise).

Using the MILP we are able to specify upper bounds for the average stretch metrics S^{AS} and S^{SA}. As the geometric upper bound cannot be written as a

linear constraint, we are not able to use it in the optimization model. As our network model is based on the idea of flows, which allow multi-path routing, we cannot calculate the maximum latency in the linear program either. Hence, we cannot bound the maximum stretch metrics S^{SM} and S^{MS}.

5 Results

We use our model to calculate the power consumption in different networks: a hypercube, a two-dimensional Grid, and as a practical network we use the `nobel-germany` network from the Survivable fixed telecommunication Network Design library (SNDlib) [6]. The SNDlib includes power consumption (cost), capacity and a demand pattern, while we estimate the latencies from the physical distances of the routers (locations are present in the model). For both theoretical networks we set all capacities, latencies, and power consumption values to 1 and assume a uniform demand pattern. We scale the network load with a single scalar factor and solve the optimization problems with the GNU Linear Programming Kit (v4.44) and the Gurobi Optimizer 4.0. The error bars show the upper and lower bounds.

Figure 2 compares both the most power-efficient configuration and the latency-minimizing configuration as well as the power models. As the difference between the two power models is small, we use the simpler binary model for further analysis. Additionally, Figure 2 shows that it is possible to conserve 45% of power. The amount of conserved power depends on the type of network considered, as a minimum-power spanning tree is needed for connectivity between all nodes. As the load of the links is inhomogeneous and we use a single scale-factor, it is possible to disable links even at the highest load the network can transfer.

Figure 3 illustrates the power necessary to keep the stretch under a given factor. Allowing the latency to increase by 20% saves up to 39% of power. Although both average metrics, S^{AS} and S^{SA}, are generally different, they produce similar results in this scenario. Figure 4 shows the difference when bounding the metrics by 1.2 for

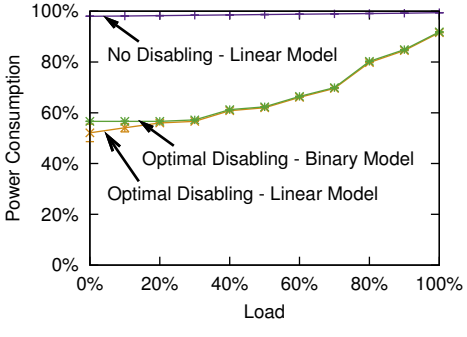

Fig. 2. Power consumption with different models using the `nobel-germany` network, $p_0 = 0.98$ in the linear model

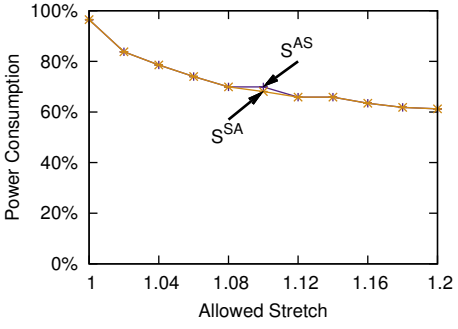

Fig. 3. Power consumption for different upper bounds for the stretch metrics S^{AS} and S^{SA} in the `nobel-germany` network

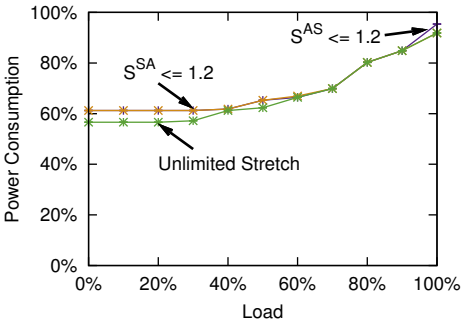

Fig. 4. Power consumption for S^{AS} and S^{SA} bounds of 20% increase in latency in the `nobel-germany` network

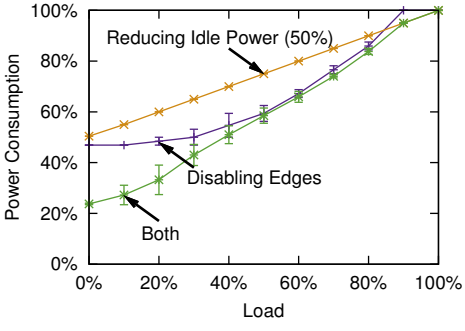

Fig. 5. Different ways to conserve power in the Hypercube of dimension 4

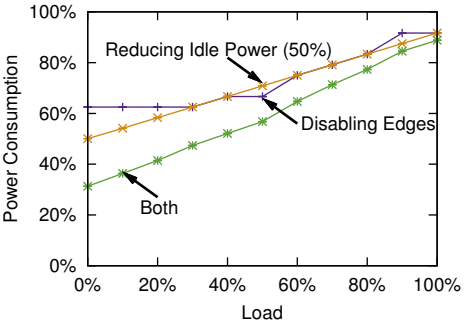

Fig. 6. Different ways to conserve power in the 4×4-Grid

different load levels. Again both metrics do not differ much and allowing a stretch of 1.2 consumes less than 5% more power than not limiting the stretch at all.

Figure 5 (6) shows the power consumption of different methods to conserve power in a 4-dimensional hypercube (4×4-Grid). We compare the optimal disabling of edges to a method of reducing the idle power consumption p_0 of all edges to 50% and the combination of both. The results show that both approaches reduce the consumed power in both networks and the power consumption is further reduced by combining both approaches.

6 Related Work

Many papers present work on minimizing the cost in fixed-charge networks [5,3], but we are not aware of any work that analyzes the implications on latency.

Chiaraviglio et al. [2] describe several centralized heuristics to approximate the most power-efficient topology for the binary power-consumption model. They

assume that line cards as well as routers can be disabled and demands of end-to-end flows are known. The heuristics provide results that are close to minimal power consumption, but they do not analyze latency.

Vasić and Kostić [9] describe a distributed algorithm that uses adaptive link rates to reduce power consumption by modifying both topology and multi-path routing. Their idea is to distribute the load so that lines can be set to low-speed low-power operating modes. Their work provides a possible approach to actually implement power-saving measures, but does not focus on its effects on latency.

Revirigo et al. [8] analyze how adaptive link rates and burst transmissions can reduce the idle power consumption of single links. This method considers each link individually while we consider the whole network.

7 Conclusion

We formalized the problem of conserving power in networks, applied our model to different networks, and analyzed power consumption and latency. Assuming a binary power-consumption model, we can reduce the consumed power by 39% and increase the latency by only 20% in the `nobel-germany` network. We conjecture that for most networks and demand patterns "sweet spot" configurations exist that conserve large amounts of power and increase latency only slightly.

In the future more complex models are possible, e.g. arbitrary functions from load to power consumption. Another approach is to consider not only propagation delays, but to include queuing delays into the model and test how they interact with power conservation methods.

References

1. Chabarek, J., Sommers, J., Barford, P., Estan, C., Tsiang, D., Wright, S.: Power awareness in network design and routing. In: IEEE INFOCOM (2008)
2. Chiaraviglio, L., Mellia, M., Neri, F.: Reducing power consumption in backbone networks. In: Proc. of the ICC, Dresden (2009)
3. Hochbaum, D.: Analysis of a flow problem with fixed charges. Networks 19(3), 291–312 (1989)
4. Johnson, D.S., Lenstra, J.K., Rinnooy Kan, A.H.G.: The complexity of the network design problem. Networks 8(4), 279–285 (1978)
5. Khang, D.B., Fujiwara, O.: Approximate solutions of capacitated fixed-charge minimum cost network flow problems. Networks 21(6), 689–704 (1991)
6. Orlowski, S., Pióro, M., Tomaszewski, A., Wessäly, R.: SNDlib 1.0–Survivable Network Design Library. In: Proc. of the 3rd INOC, Belgium (April 2007)
7. Puype, B., Vereecken, W., Colle, D., Pickavet, M., Demeester, P.: Power reduction techniques in multilayer traffic engineering. In: Proc. of the ICTON, Portugal, vol. 1 (2009)
8. Reviriego, P., Maestro, J., Hernadez, J., Larrabeiti, D.: Burst Transmission in Energy Efficient Ethernet. Internet (2010)
9. Vasić, N., Kostić, D.: Energy-aware traffic engineering. In: Proc. of the 1st Int'l Conf. on Energy-Efficient Computing and Networking, pp. 169–178. ACM, New York (2010)