Balancing Peer and Server Energy Consumption in Large Peer-to-Peer File Distribution Systems

Lachlan L. H. Andrew, Andrew Sucevic, Thuy T. T. Nguyen
Centre for Advanced Internet Architectures, Faculty of ICT
Swinburne University of Technology, Melbourne, Australia
Email: {landrew, tnguyen}@swin.edu.au

***Abstract*—Network induced energy consumption is a significant fraction of all ICT energy consumption. This paper investigates the most energy efficient way to distribute a file to a large number of recipients. It is shown that using peer-to-peer and naively minimizing the transfer time results in energy consumption that is an order of magnitude larger than simply distributing directly from a server, but that with careful management peer-to-peer systems can reduce the server's cost without increasing overall energy consumption.**

I. Introduction

As in most industries, there is a push in the communications industry to improve energy efficiency. However, the energy consumption of communications infrastructure itself is only part of the story; there is a considerable amount of *network induced* energy consumption, in which devices such as personal computers (PCs) or set-top boxes are left on simply to maintain network connectivity [1]. Since PCs and their monitors have been estimated to consume a quarter of all IT energy [2], this provides significant potential for savings.

One form of network induced energy consumption consists of leaving PCs on to download files. Peer-to-peer (P2P) technology can reduce the time required for file distribution, but little attention [1], [3] has been paid to its energy consumption.

This paper considers large scale file distribution, such as distributing an update to a popular software package, or perhaps the distribution by a television station of content via P2P transmission among set-top boxes [4]. This differs from the more common file sharing in terms of the scale (possibly millions of peers), the asymmetry (software houses having multi-gigabit connections to the Internet) and the increased potential for centralized management. Peer-to-peer distribution is a promising means for content distributors to reduce their infrastructure and networking expenses.

Our objective in this paper is to investigate the implication of this for overall energy consumption. To this end, we compare the energy consumption of four schemes: the four combinations of with or without P2P transmission, and of naive or energy-optimized strategies.

Previous work [5], [6] shows that "optimal" schedules are excessively complex, even for simple optimization tasks such as the sum of peer download times under a model ignoring peers' download constraints. With download constraints and the ability to turn peers OFF, they become even more complex. For the comparison of this paper, it is sufficient to find a simple, scalable strategy that asymptotically minimizes peer and server energy, rather than the actual optimal schedule. Deriving such a scalable strategy in Section III, and proving its asymptotic optimality in Section IV, is the main technical task of this paper.

Following [7]–[10], only upload and download constraints are considered, while network congestion is ignored. It is assumed that a set of peers simultaneously start downloading a given file, and that as soon as a peer has downloaded some data, that data can be forwarded to other peers.

In this model, a simple symmetric strategy minimizes the time for the last peer to receive the file [7], [8]. This strategy causes all peers to finish simultaneously, and is the "naive P2P" scheme used in our comparisons. However, some peers can finish much earlier with minimal or no increase in the finish time of the last peer. This led to a search for smaller average finish times [5], [11]–[13]. It was conjectured in [5] that sequentially minimizing finish times also minimizes the sum of finish times, but a counterexample was presented in [6]. For a given finishing order, [10] derives the polytope of all possible combinations of finish times.

However, these strategies assume that peers will remain on to continue uploading until the last peer has finished downloading. This need not minimize the energy consumed by the peers. This paper instead considers the case that the central algorithm can turn peers ON or (if they are not in use for some other purpose) OFF at any point in the schedule, to minimize the total energy consumption.

It is shown that it is optimal for most peers to be ON only while they are downloading, and to download at the fastest rate possible, while a small subset of energy-efficient peers may be required to be ON for the entire duration of the transfer.

The rest of this paper is organized as follows. After describing the model in Section II, a class of strategies is described in Section III, with the aim of getting good performance in large systems. Section IV derives an upper bound on the cost of any strategy in this class, and shows that this bound approaches the lower bound on the cost of any strategy, which demonstrates the asymptotic optimality of the class. These results are numerically validated in Section V, where it is shown that turning peers OFF can result in significant energy savings over systems aiming to minimize the total download time. Implementation issues are briefly discussed in Section III-A.

II. Model

The model used here is a fluid model, similar to those of [7]–[10] which were used for determining the minimum total time (makespan) required for peers to download a file, and the minimum sum of download times. The novelty in our model is the different objective of minimizing the energy consumption instead of the download time, by allowing peers to turn ON and OFF during the download.

In particular, the system has a single server which contains a file of size F which is distributed to N peers. All nodes (server and peers) are assumed to be able to communicate to all other nodes with no congestion in the core network, so that the only constraint being the upload and download capacities of each node. The network is also assumed to be static, in that no peers can arrive or leave. The file is broken up into infinitely small pieces, which allows a peer to immediately forward any data it receives without delay to another peer.

The following notation is used in this model:

- F: size of the file
- N: the set of peers (not including the server)
- C_s: upload capacity of the server
- $U_i \in [U_{\min}, U_{\max}]$: upload capacity of peer i
- $D_i \in [D_{\min}, D_{\max}]$: download capacity of peer i
- $P_i \in [P_{\min}, P_{\max}]$: The power consumption of peer i.
- P_s: The power consumption of the server.

When peer i is ON, it can download at rate D_i, it can upload any data it has received to any other peers, at rates not exceeding U_i in total, and it consumes power P_i. When it is OFF, it cannot download or upload, and consumes no power. A peer is "finished" when it has downloaded all data in the file.

For most of the paper, it is assumed that the server must be ON, consuming power P_s, until all peers have finished. The numerical results section considers the case of an energy-proportional virtual server; the transmission rate can be varied up to the maximum of C_s, and P_s is proportional to the transmission rate, not to C_s.

The objective is to choose the rates and the sequence of ON peers to minimize the total energy consumption, which is the integral over all time of the power consumption.

III. Strategy

In the above model, the strategy to minimize the total energy is significantly more complicated than the strategy proposed in [5] to minimize the sum of the download times. Instead of deriving the optimal strategy, we present a much simpler strategy, which still performs well for large systems.

It is based on two observations: that there is a minimum amount of time for which the peers must be ON, and that there is a minimum additional energy per bit of data not uploaded during that time.

Since the server must remain ON until all peers have finished, the minimum energy per bit delivered is achieved by augmenting the server with a (possibly empty) set E of energy-efficient peers. These peers will remain ON for essentially the entire time, effectively providing the server with additional capacity.

Let the set E of extra peers satisfy

$$E \in \arg\min_{E} \frac{C_s + \sum_{i\in E} U_i}{P_s + \sum_{i\in E} P_i} \tag{1}$$

This can be found by sorting peers in descending order of U_i/P_i and greedily adding peers to E until the next U_i/P_i is less than the fraction in the right hand side of (1). To simplify notation, let

$$C_s' = C_s + \sum_{i\in E} U_i \qquad P_s' = P_s + \sum_{i\in E} P_i.$$

The set E will be empty if and only if C_s has a higher power efficiency (upload capacity per watt) than any of the peers. This is likely if the server has high energy efficiency and the peers are all personal computers connected by residential broadband connections. However, E is likely to be non-empty if some of the peers are power-efficient notebooks connected to the local area network of a well-connected corporation.

A broad class of schedules is described in Figure 1. It leaves many degrees of freedom, but contains enough structure that all feasible schedules in this class are asymptotically optimal, provided that no peer receives the same data twice.

The left hand side of inequality (2) is the total upload capacity given that only peers in $E \cup K$ are ON, and the right hand side is the download capacity required to keep all peers in K downloading at full capacity. Initially, $\eta(0) \geq 1/k$, with equality if and only if $\sum_{i\in E} U_i/|K| \leq D_{\min}$; subsequently, $\eta(t)$ will often be 0.

Within this class, the recommended choices are:

1) In step 2, low-capacity peers start early enough that the only peers left by step 8 have high capacity.
2) In step 2, the first phase consists of the lowest-capacity peers, so that each peer in E can send data to the largest possible number of peers; this minimizes the amount it needs to download from the server.
3) In (2), η is the minimum amount of data that each peer $j \in E$. must download in order to upload at full rate to the peers $i \in K$.
4) If (2) is not met with equality, the excess upload capacity is used to upload to peers in E that have not yet downloaded the entire file, starting with the one with the smallest download capacity.

The specific instance used for our numerical results is described in Section V.

A. Implementation considerations

This strategy is intended, in the spirit of [7]–[10], to be an idealized benchmark against which more practical decentralized strategies can be compared. However, some comments on implementation issues are called for.

As with all centralized schemes, this assumes that the capacity of the access link is known. This could be estimated by an agent on the peer using active probing techniques such as [14] for determining the capacity of a particular link on a path. Note also that U_i and D_i need not be the entire link capacities; the system may for example limit data transfers to at most a certain percentage of the physical link capacity, at the expense of suboptimality.

Schedule:

1) Determine E by (1).
2) Select an ordering for $N \setminus E$.
   Transmission rates:
3) Download enough to peers $j \in E$ to be able to upload at rate U_j in step 6.
4) Maintain a list K of peers downloading at full capacity D_i.
5) Whenever a peer i finishes downloading, or at the start, remove i from K and greedily add peers to K, subject to, for some $\eta(t) \in [0,1]$,
$$C_s + \sum_{i\in K} U_i + \sum_{j\in E} U_j \geq \sum_{i\in K} D_i + \eta(t) \sum_{j\in E} U_j. \quad (2)$$
6) Upload to each peer $i \in K$ at D_i.
7) If not all peers in $N\setminus E$ have finished, go to step 5.
8) Finish all $j \in E$, with the maximum possible number downloading at rate D_j at each time.
   Power control:
9) At any given time, only the server and peers $i \in K \cup E$ are ON.
10) In step 3, only enough peers in E are ON such that the upload capacity of all but one is fully used.
11) Once all peers in $N\setminus E$ have finished, the only ON peers in E are those that are either downloading, or without whose upload capacity the remainder could not download at their full rate.

Figure 1. Class of schedules

The strategy assumes that P_i is known for each peer. This will typically not be known. The strategy can be used instead with surrogate data, such as assuming that each peer has some "average" power consumption, or some given correlation between access link rate and power consumption.

IV. Performance bounds

The above class of schedules is quite simple. It makes no effort to perform "packing" of the sort that makes the knapsack problem hard [15]. This section derives an upper bound on the cost of the schedule, and a corresponding lower bound on the cost of any schedule. The ratio of these bounds approaches 1 in a natural scaling regime in which $N \to \infty$, $C_s \to \infty$, $C_s = o(N)$ and $P_s = O(C_s)$. This demonstrates that the schedule is asymptotically optimal, despite its simplicity.

The bounds are based on the same observations as the strategy itself: that there is a minimum amount of time for which the peers must be ON, and that there is a minimum additional cost per bit of data not uploaded during that time.

A. Lower bound

To find a lower bound on the energy cost of distributing the file to all of the peers, notice that each peer must be ON while it is downloading, giving a cost $\sum_{i\in N} FP_i/D_i$. While it is ON, there is no additional cost for uploading at rate U_i, but power must be used to upload the remaining $NF - \sum_{i\in N} FU_i/D_i$. Given that the server must be ON until all peers are finished, the cheapest way to supply data is from a combination of the server and all peers in E. This gives a lower bound on the optimal total cost OPT of
$$\frac{OPT}{F} \geq \sum_{i\in N} \frac{P_i}{D_i} + (N - \sum_{i\in N} \frac{U_i}{D_i}) \frac{P'_s}{C'_s} \quad (3)$$
Note that network coding [6] cannot overcome the need to have an integer number of peers ON.

B. Upper bound

We now find an upper bound on the energy cost of the class of schedules of Section III. The total energy consumption of any schedule in the proposed class is bounded above by
$$\begin{aligned}\frac{cost}{F} \leq & \sum_{i\in N} \frac{P_i}{D_i} \quad (4)\\ & + \frac{P'_s}{C'_s}\left(N - \sum_{i\in N} \frac{U_i}{D_i} + \frac{N \max_i(D_i - U_i)}{k_{\min} D_{\min}}\right)\\ & + \frac{P_s}{C_s}\frac{C_s + U_{\max}}{D_{\min}} + \frac{\max_{j\in E}(D_j - U_j)\sum_{j\in E} U_j}{k_{\min} D_{\min}} \frac{P_s}{C_s}\end{aligned}$$
where
$$k_{\min} := C_s / \max_{i\in N}(D_i - U_i) - 1. \quad (5)$$

The justification is as follows.

The total energy consumption is bounded above by the sum of the energy consumed by the peers while they are downloading, plus the energy required by the server and the peers in E to supply the data that was not uploaded by peers in $N \setminus E$. To bound the latter term, we bound the aggregate upload capacity that is unused, and add this to the lower bound $F(N - \sum_{i\in N} U_i/D_i)$ on the amount of data that cannot be supplied by peers while they are downloading. This unused capacity contains three components: a *truncation* component that occurs at the end when there are no more peers to add to K in (2), a *packing* component that occurs because there are more peers, but the next peer in sequence requires more capacity than is available, and possibly an *initialization* component that occurs in step 3 when the peers in E are receiving enough data to start assisting the server.

First we find an upper bound on the packing component. The maximum unused upload rate during this time is $\max_{i\in N}(D_i - U_i)$; if more capacity were available then (2) would allow the next peer to be added to K. It remains to bound the amount of time for which this rate is unused.

Divide the download time into "windows"; a new window starts at the time of completion of all peers that were in K at the time the previous window started. The duration of a window is the maximum of the times to finish each of the peers in K, which is bounded above by $F/D_{\min}$. Again, the maximality of the set K subject to (2), and the fact that $\eta \leq 1$, implies that $k_{\min}$ given by (5) is a lower bound on the number of peers in K during this phase. Since the sets K at the start of distinct windows are disjoint, the number of windows is at most $N/k_{\min}$. Thus, the packing component is at most
$$\frac{N}{k_{\min}} \frac{F}{D_{\min}} \max_{i\in N}(D_i - U_i). \quad (6)$$

The truncation component, which starts from the time the last node begins downloading, is bounded above by

$$\frac{F}{D_{\min}}(C_s + U_{\max}). \tag{7}$$

This is the time taken to download by the peer with the smallest download capacity, multiplied by the unused rate of upload capacity. The latter contains two terms; The first is the upload capacity of the server, which must be ON. The second is the upload capacity of either a downloading peer, since that peer cannot upload to itself and may not have another peer to upload to, or alternatively one member of E which may be required to be ON to ensure all other peers are downloading at D_i, but may not be fully utilized. The average cost per unit data during this period is at most P_s/C_s, since all peers in E have $P_j/C_j \leq P_s/C_s$.

Finally, we bound the filling component of wasted upload. Since peers in E that are not being filled at full rate can be turned OFF, the waste is at most $\max_{j\in E}(D_j - U_j)$ per unit time. To provide enough capacity during a subsequent stage, it is sufficient to download a total amount $(\sum_{j\in E} U_j/k_{\min})(F/D_{\min})$. This is supplied either by the server or peers in E, and so has a cost of at most P_s/C_s per unit of data. Thus, the total cost for making up for the wasted capacity during this phase is

$$F\frac{\max_{j\in E}(D_j - U_j)\sum_{j\in E} U_j}{k_{\min}D_{\min}}\frac{P_s}{C_s}. \tag{8}$$

The upper bound (4) for the server ON time can now be formed by adding the upper bounds of wasted upload capacity in (6), (7) and (8) to the total amount of data that needs to be uploaded, minus the maximum each peer can upload during its minimal download time F/D_j.

C. Asymptotic tightness

Consider now the case that there is a constant β such that $\beta = U_i/D_i$ for all $i \in N$.

Theorem 1. *In a scaling regime where $C_s \to \infty$ and $N \to \infty$, bounds* (3) *and* (4) *imply*

$$\frac{cost}{OPT} = 1 + O\left(\frac{1}{C_s} + \frac{P_s}{N} + \frac{P_s}{C_s^2}\right). \tag{9}$$

Proof: Dividing (4) by (3) gives

$$\begin{aligned}
\frac{cost}{OPT} \leq & 1 + \frac{\frac{P'_s}{C'_s}\left(\frac{N\max_i(D_i-U_i)}{k_{\min}D_{\min}}\right) + \frac{P_s}{D_{\min}}(1 + U_{\max}/C_s)}{\sum_{i\in N}\frac{P_i}{D_i} + (N - \sum_{i\in N}\frac{U_i}{D_i})\frac{P'_s}{C'_s}} \\
& + \frac{\frac{\max_{j\in E}(D_j-U_j)\sum_{j\in E}U_j}{k_{\min}D_{\min}}\frac{P_s}{C_s}}{\sum_{i\in N}\frac{P_i}{D_i} + (N - \sum_{i\in N}\frac{U_i}{D_i})\frac{P'_s}{C'_s}} \\
\leq & 1 + \frac{N\max_i(D_i - U_i)}{k_{\min}D_{\min}(N - \sum_{i\in N}\frac{U_i}{D_i})} \\
& + \frac{P_s/D_{\min}}{\sum_{i\in N} P_i/D_i}(1 + U_{\max}/C_s). \\
& + \frac{\max_{j\in E}(D_j - U_j)\sum_{j\in E}U_j}{k_{\min}D_{\min}\sum_{i\in N}P_i/D_i}\frac{P_s}{C_s}.
\end{aligned} \tag{10}$$

Since β is a constant, substituting (5) into the second term above gives

$$\frac{D_{\max}^2(1-\beta)}{C_s D_{\min}} + O(C_s^{-2}) = O(1/C_s).$$

For $C_s > U_{\max}$, the second line of (10) is bounded above by

$$\frac{P_s}{N}\frac{2D_{\max}}{D_{\min}} = O\left(\frac{P_s}{N}\right).$$

Finally, the last term of (10) is $O(P_s/C_s^2)$ since the numerator of the first factor is $O(N)$ and the denominator is $\Omega(C_s N)$ (that is, increases at least as fast as $C_s N$).

This shows that (9) is an upper bound. Since $cost \geq OPT$, (9) follows. ∎

Given that it is possible to get $P_s = O(C_s)$ simply by replicating servers, this theorem implies that for large systems, it is optimal to use the proposed schedule, with the server capacity increasing without bound but slower than the number of peers. In particular, the bound is minimised if $C_s \propto \sqrt{N}$.

V. Numerical results

We can now compare energy-aware P2P with alternatives such as server-only distribution and energy-unaware P2P. These alternatives were tested under the following conditions.

The distribution of download speeds was based on that measured by Akamai, a large content distribution network, for the US state of California in August 2009 [16]. The maximum and minimum speeds were not specified, and so $D_{\min} = 128\,\text{kbps}$ and $D_{\max} = 100\,\text{Mbps}$ were used.

The server was assumed to be a standard volume server, with a $C_s = 1\,\text{Gbps}$ connection to the Internet[1].

The server power was taken to be $P_s = 218\,\text{W}$, the average power of a volume server in 2005 [17]; this value was not adjusted to consider the data centre Power Utilization Efficiency (PUE). It was assumed that 90% of peers are PCs using power $P_i = 100\,\text{W}$, while the other 10% are set-top boxes consuming $P_i = 15\,\text{W}$. The upload and download speeds are independent of the node's power.

Unless otherwise stated, the ratio of upload to download capacity was $\beta = 0.2$, which is typical of asymmetric digital subscriber line (ADSL) connections.

The energy-aware strategy evaluated was as described in Figure 1, with the following implementation decisions.

1) In Step 2, peers were ordered by increasing capacity.
2) In (2), the coefficient η for the rate at which peers in E download was as described in the appendix, which is sufficient to perform the required uploading.
3) Any unused upload capacity is directed to the peers in E, starting with the one of lowest capacity.

A. Benefit of optimization

To determine whether P2P increases or decreases the overall network-induced energy consumption, Figure 2 plots the total energy consumption of the server and peers for four schemes.

[1]To make the first and third $O(\cdot)$ terms of (9) vanish, C_s should also grow, but constant C_s demonstrates the decrease in the upper bound.

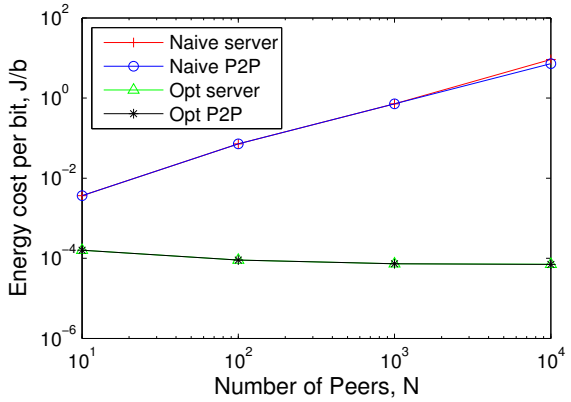

Figure 2. Total energy consumption as the number of peers N increases and the server capacity is $C_s = 1\,\text{Gbps}$. Upload to download capacity ratio $\beta = 0.2$. The difference between P2P and server-based is small compared to that between naive and optimized.

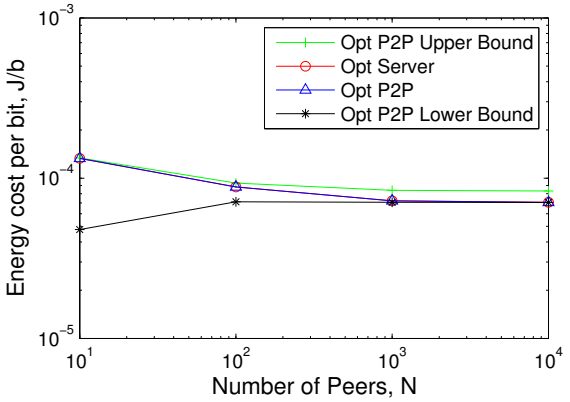

Figure 3. Total energy consumption as the number of peers N increases. Upload to download capacity ratio $\beta = 0.2$. The measured performance of the proposed schedule, and its provable upper bound, approach optimality for large systems.

In first, "naive server", all data is sent from the server, but all clients are ON for the entire duration. The second, "naive P2P", is the P2P scheme [7] that minimizes the total download time of all peers, which again requires that all peers be ON during the download. In "Opt server", all data is sent from the server, but peers are either OFF or being uploaded to at full rate D_i. "Opt P2P" is the scheme of Section III.

This shows clearly that the difference in energy consumption between P2P schemes and purely server-based schemes is small compared with the savings that are possible by optimizing for energy consumption. Henceforth, we consider only the optimized strategies.

B. Scalability

Next, we investigate the optimality of the "Opt P2P" scheme. A central contribution of this paper is to demonstrate that a simple strategy can obtain nearly optimal energy performance in large systems. To validate this claim, Figure 3 shows how the proposed strategy performs as the number of users grows from 100 to 10 000, relative to the upper and lower bounds, and the "Opt server" strategy.

Although the upper bound (4) is sufficient to establish the asymptotic form (9), it is clearly loose, largely because it does not depend on the order in which peers are served. Tighter bounds could be obtained by noting that typically $K \gg k_{\min}$, and that most peers that are unfinished in the final stage have download capacities well above $D_{\min}$.

The performance of "Opt P2P" and "Opt Server" are identical except for very large N, not merely because of the scale of this graph. To see why, note that the total peer energy consumption is fixed at $F\sum_{i\in N} P_i/D_i$ for any scheme in which peer i is only ON when downloading at its capacity D_i. Similarly, the total server capacity is at least $FP_s/D_{\min}$, the energy it expends while the slowest peer is downloading. When $C_s/N > D_{\min}$, both of these bounds are tight. This is typically the case when a standard volume server is dedicated to distributing a single file.

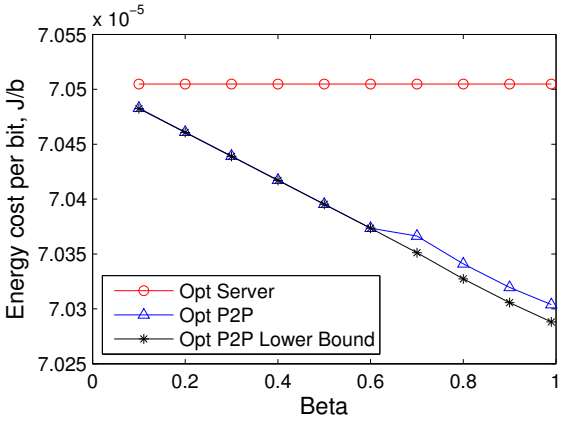

Figure 4. Total energy consumption as the ratio of upload to download capacity β increases. Total energy saving is negligible, even though saving in download time is appreciable. The kink at $\beta = 0.7$ is due to a change in E.

C. Effect of upload asymmetry

Peer-to-peer technology clearly has the biggest potential benefit when peers have high upload capacity. Figure 4 shows the total cost for $N = 100\,000$ users when the asymmetry ratio β ranges from 0.1 to 0.8.

Just as increasing β reduces the download time in typical P2P systems, energy use decreases when the upload capacity is significant compared with the download capacity. However, the gains are not large, because the energy cost is dominated by the time that the peers are ON, which is $F\sum_{i\in N} P_i/D_i$ in both cases.

D. Cost-optimization by the server

The motivation for a file distributor to use P2P is typically to reduce its own costs, rather than altruistically to minimize energy consumption. The two goals can be traded off by placing a greater weight on the server energy than the peer energy, causing an increase in E.

Figure 5 shows the total energy consumption against server energy consumption, as the set E is forced to contain an increasingly large fraction of the most energy-efficient peers.

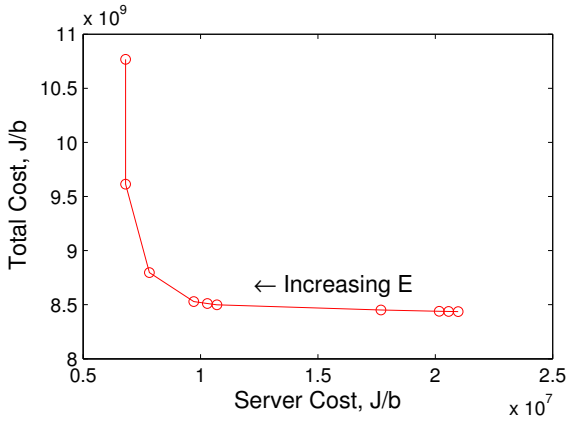

Figure 5. Total cost versus server cost, as the set E of peers that remain ON to assist uploading increases from empty to 10% of all nodes. Further increasing E increases the total cost dramatically, for minimal decrease in server cost.

Increasing E can result in a very large increase in total energy consumption for a very small reduction in the total cost to the server. This shows the importance of providing a suitable incentive for the file distributor to seek to minimize total energy consumption, rather than merely its own cost.

VI. Conclusion

This paper has presented both a study of the energy efficiency of P2P file distribution in a typical setting, and also a new P2P schedule that is asymptotically optimal for energy efficiency.

Specifically, it has shown that a simple schedule can be asymptotically optimal, which eliminates the need to determining the exact schedule as done in a related context in [5].

However, the structure of the policy raises several concerns. Most notable is the issue of fairness [18], [19]. The optimal scheme sometimes forces a small subset of peers to carry a large fraction of the burden of uploading. It remains to determine how nearly optimal a scheme may be while introducing limited unfairness.

The results of the numerical study suggest that there is little difference in the total energy consumption between an optimized P2P file distribution system and an optimal server-only system. However, the optimal P2P strategy for minimizing energy consumption is significantly different from the strategy for minimizing the total download time, since the latter causes too many peers to remain ON for long periods.

The results of Section V-B also demonstrate that, when some peers have low download rates, it is very inefficient to dedicate a server to serving a single file, even when 10 000 peers want to download that file.

The numerical study makes simplifying assumptions, such as that peers are only ON for the purpose of downloading the file, and that all peers are available at any time. However, we expect the above qualitative insights to hold more broadly.

Going forward, it will be important to determine a practical schedule, considering issues such as congestion within the network, that allows the cost savings of peer-to-peer to be achieved without causing excessive energy consumption.

Acknowledgement

This work was supported by ARC grants DP0985322 and FT0991594.

References

[1] Y. Agarwal, S. Hodges, R. Chandra, J. Scott, P. Bahl, and R. Gupta, "Somniloquy: Augmenting network interfaces to reduce PC energy usage," in *Proc. Usenix NSDI*, 2009.

[2] Australian Computer Society, "Carbon and computers in Australia," 2010.

[3] J. Blackburn and K. Christensen, "A simulation study of a new green bittorrent," in *Proc. ICC Workshops*, 2009.

[4] M. Cha, P. Rodriguez, S. Moon, and J. Crowcroft, "On next-generation telco-managed P2P tv architectures," in *Proc. Int. Conf Peer-to-peer Systems*, 2008, p. 5.

[5] G. Ezovski, A. Tang, and L. L. H. Andrew, "Minimizing average finish time in P2P networks," in *Proc. IEEE INFOCOM*, 2009.

[6] C. Chang, T. Ho, M. Effros, M. Medard, and B. Leong, "Issues in peer-to-peer networking: A coding optimization approach," in *IEEE Int. Symp. Network Coding (NetCod)*, June 2010, pp. 1–6.

[7] R. Kumar and K. Ross, "Peer-assisted file distribution: The minimum distribution time," in *Proc. IEEE HotWeb*, 2006.

[8] J. Mundinger, R. Weber, and G. Weiss, "Optimal scheduling of peer-to-peer file dissemination," *J. Scheduling*, vol. 11, pp. 105–120, 2008.

[9] M. Mehyar, G. WeiHsin, S. H. Low, M. Effros, and T. Ho, "Optimal strategies for efficient peer-to-peer file sharing," in *Proc. ICASSP*, 2007.

[10] Y. Wu, Y. C. Hu, J. Li, and P. A. Chou, "The delay region for P2P file transfer," in *Proc. Int. Symp. Info. Th. (ISIT)*, 2009, pp. 834–838.

[11] M. Lingjun and K.-S. Lui, "Scheduling in P2P file distribution – on reducing the average distribution time," in *IEEE Consumer Commun. Netw. Conf. (CCNC)*, Jan. 2008, pp. 521–522.

[12] M. Lingjun, P.-S. Tsang, and K.-S. Lui, "Improving file distribution performance by grouping in peer-to-peer networks," *IEEE Trans. Netw. Serv. Manag.*, vol. 6, no. 3, pp. 149 –162, Sept. 2009.

[13] P.-S. Tsang, X. Meng, and K.-S. Lui, "A novel grouping strategy for reducing average distribution time in P2P file sharing," in *IEEE Int. Conf. Commun (ICC)*, May 2010, pp. 1–5.

[14] K. Harfoush, A. Bestavros, and J. Byers, "Measuring capacity bandwidth of targeted path segments," *IEEE/ACM Trans. Netw.*, vol. 17, no. 1, pp. 80–92, Feb. 2009.

[15] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1990.

[16] Akamai Technologies Inc., "Observed average internet speeds for U.S. network connections," Oct. 2009. [Online]. Available: http://fjallfoss.fcc.gov/ecfs2/document/view?id=7020243366

[17] J. Koomey, "Estimating total power consumption by servers in the U.S. and the world," 2007. [Online]. Available: http://enterprise.amd.com/Downloads/svrpwrusecompletefinal.pdf

[18] P. Tsiaflakis, Y. Yi, M. Chiang, and M. Moonen, "Fair greening for DSL broadband access," in *GreenMetrics*, 2009.

[19] L. L. H. Andrew, M. Lin, and A. Wierman, "Optimality, fairness and robustness in speed scaling designs," in *Proc. ACM Sigmetrics*, 2010.

Appendix

To calculate η in (2), the evaluated strategy finds the maximal subset T of K such that

$$\sum_{i\in T} D_i \le \sum_{j\in E} U_j. \tag{11}$$

Then

$$\eta = \max\left(\frac{\max_{i\in T} D_i}{\sum_{j\in E} U_j}, \frac{1}{|K\setminus T|}\left(1-\frac{\sum_{i\in T} D_i}{\sum_{j\in E} U_j}\right)\right). \tag{12}$$

This provides sufficient data for the peers in E to send at aggregate rate D_i to each $i \in T$, and to use any remaining capacity to send equally to the remaining peers in $K \setminus T$.