Deadline Constrained Minimum Cost Data Transfer over Heterogeneous Connections

Jay Chen[1], Ghulam Murtaza[2], Srikanth Adimulam[3], Gursharan Singh[4], Kentaro Toyama[5]

[1]*New York University, 715 Broadway, New York, 10003, United States*
Tel: +1 (212) 203-3533, Email: jchen@cs.nyu.edu
[2]*Lahore University of Management Sciences, DHA, Lahore, 543000, Pakistan*
Tel: +92 (32) 14514465, Email: murtaza@lums.edu.pk
[3]*Microsoft Research, Tel: +91 (80) 66586002, Email: adi_sriku366@yahoo.co.in*
[4]*Microsoft Research, Tel: +1 (267) 432-1813, Email: gursharan@alumni.upenn.edu*
[5]*Microsoft Research, 196/36 2nd Main Sadashivnagar, Bangalore, 560 080, India*
Tel: +91 (80) 66586002, Email: kentoy@microsoft.com

Abstract: New wireless communications technologies are integrated into mobile devices such as smartphones every year. These technologies each have unique performance and cost characteristics. This begs the question: What is the minimum cost of transferring data using any combination of connections while meeting deadlines as defined by applications? The problem has an optimal linear programming solution, but a naive implementation encounters problems leading to sub-optimality and in some cases very poor performance. Connection quality variability, task changes, and the time to perform the linear programming computation all increase data transmission cost. We address these challenges to provide a complete solution to the problem of minimum cost deadline constrained data transfer over heterogeneous connections.

1. Introduction

The link between telecommunications and growth of sustainable development in rural areas is fairly well established [1]. Mobile phones have seen tremendous adoption rates in the developing world [7], and as the number of wireless technologies continues to grow (WiFi, WiMax, GPRS, 3G, Satellite, etc.) they are eventually integrated into mobile devices. These technologies each have different performance and cost associated with them. Their coverage area is also unevenly distributed. While this is an issue for both stationary and mobile networking applications, the network options available may change frequently for mobile devices. This is of particular importance in the developing world since the availability and cost of networks varies dramatically. Therefore, mobile devices should be aware of the connections that are available and the tradeoffs between them in order to minimize the cost of transferring data.

This problem was inspired by a typical data-collection scenario that occurs in non-profit work in developing countries. Non-profits frequently have personnel collecting data electronically in remote rural areas. This data must be synchronized with a central server, but connectivity of various forms is intermittent. On the one hand, non-profits have small budgets and must conserve what costs they can (and data channels can be expensive); on the other hand, they can tolerate some delays in data synchronization. Thus, inexpensive synchronization that occurs with some delay may be preferable to real-time synchronization occurring at high cost. Depending on the urgency and cost of the transfer, and availability of different channels, the data transfer cost can be minimized.

There is related work in radio resource management, but these are very low level solutions that are not focused on the particular application requirements and their affordability [8, 9, 10, 11]. Radio resource management typically attempts to optimize the resources used across the communications channel, not their underlying cost. This has been studied within individual networking technologies such as 3G networks [8], WLANs [9], and MANETs [11]. Quality of Service has also been a point of focus in wireless environments [9]. While the handoffs we advocate are akin to those in cellular networks [12], the focus of our work is cost minimization, and not attached to any particular handoff scheme or network. We are not aware of any other work that has addressed this deadline driven cost minimization problem across multiple communications networks. The most similar in the spirit of minimization of cost relate to routing algorithms within individual wireless networks [2,3], and handoffs across heterogeneous networks without the notion of affordability [4,5].

In this work we begin by formulating the theoretical problem of transferring data with deadlines across these heterogeneous links. We initially simplify the problem with some assumptions so that linear programming(LP) naturally produces an optimal solution. We then relax these assumptions to arrive at our final algorithm. We simulate our solution under a range of possible scenarios to explore its performance, and illuminate possible issues. Finally, to evaluate the practicality of our algorithm we implemented it as a part of an application on a smartphone. We find that the performance of our implementation generally reflects what was expected from our simulations.

2. Problem Formulation and Theory

In this section we describe the assumptions and problem formulation. We walk through an example problem and describe the difficulties of naively using linear programming to solve the problem. Finally, we discuss relax some assumptions to generalize our solution making it more practical.

2.1 Problem Formulation

The fundamental problem is that blocks of data are given to the connection manager with certain deadlines by which they must be delivered. The links available to transfer data have different bandwidths (bytes/sec) and costs (dollar/byte). The goal is to meet all delivery times while minimizing the total cost to deliver all chunks. There are several simplifying assumptions we make:

1. Data comes in blocks with hard deadlines.
2. Data blocks have equal priority.
3. Schedule fatter channels first, channels used at most once.
4. Only one channel used at a time.
5. All tasks known in advance.
6. Channels have fixed bandwidth and cost.

Assumptions 1 and 2 are a part of the data transfer problem formulation. The notion of deadlines is easily understood and compatible with the minimization of the cost of transferring data. Assumption 3 is an optimization we use to simplify the problem. Assumption 4 is a simplification that is easily relaxed in our final algorithm. Assumptions 5 and 6 cause our algorithm to have to be re-run each time the tasks change or the available connection bandwidth changes beyond a specified threshold. We discuss the implications of re-running LP in our evaluation.

We begin with an example to illustrate the problem. In this example two jobs π_1 and π_2 have to meet deadlines τ_1 and τ_2 (Figure 1). Figure 2 shows the four available connections (channels) have different bandwidths b_1-b_4 and costs c_1-c_4. Finally, Figure 3 illustrates the format of the output, a *schedule* where jobs π_1 and π_2 are sent across the different connections in order to both meet their deadlines and to minimize the cost of transferring the data. The output takes the form of a schedule: $\{t_1, t_2, \ldots t_{m*n-1}\}$. Each connection is used one at a time until the deadline is met for each deadline.

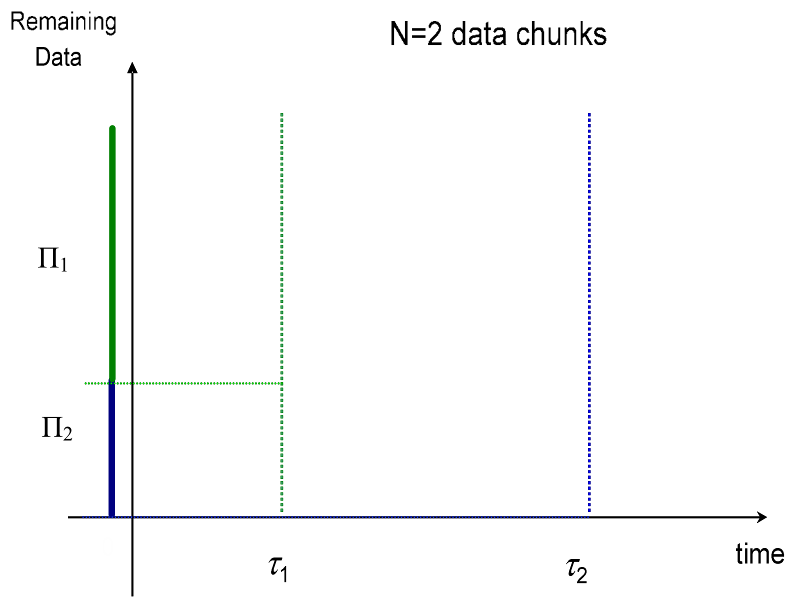


Figure 1: Data chunk i has size Π_i (bytes), and deadline τ_i (time).

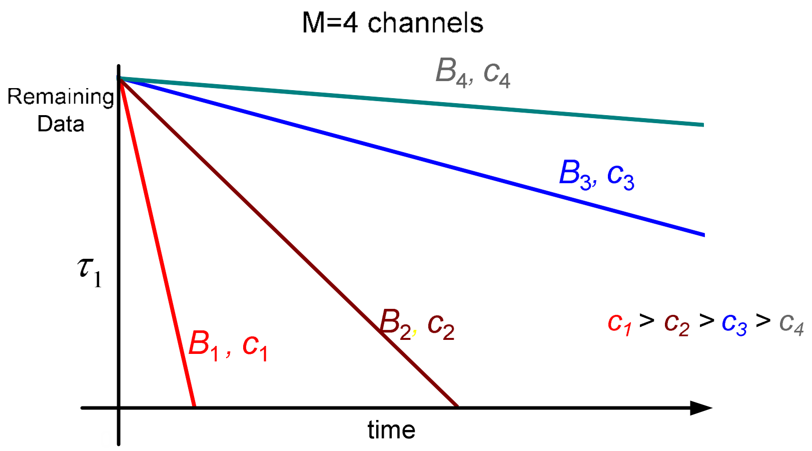


Figure 2: Channel j has bandwidth B_j (bytes/time), and cost c_j (\$/byte).

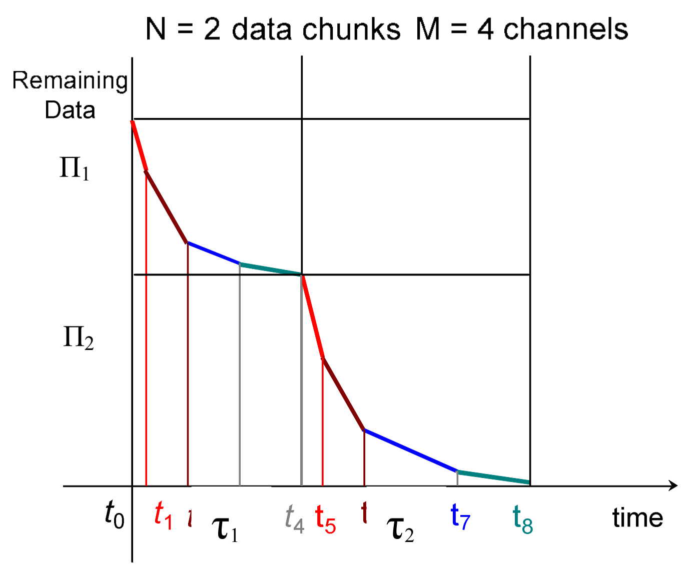


Figure 3: Output takes the form of a schedule: $\{t_0, t_1, t_2, \ldots, t_{m*n-1}\}$.

In this example each connection is used multiple times and while this is satisfactory, an optimization that we can do is to use each connection only once. The result is an output schedule of the form shown in Figure 4. Each connection is used only once, and all deadlines are still met, and the solution is still optimal in minimizing cost. The benefit here is that some additional slack is available if there are slight fluctuations in connection bandwidth or latency.

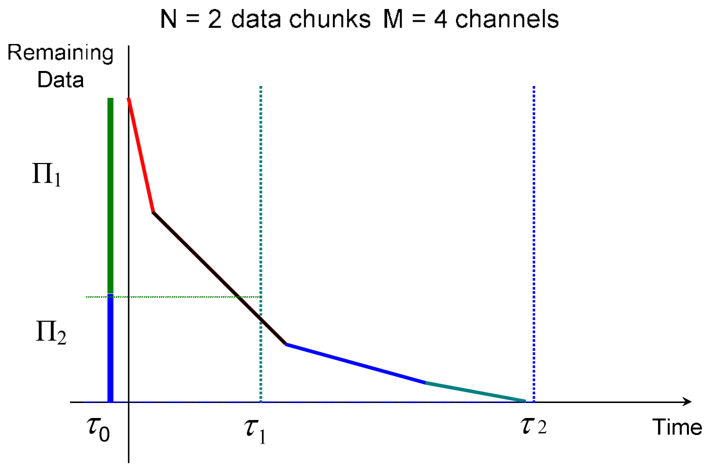


Figure 4: Optimal solution meets all deadlines, while minimizing cost.

2.2 Linear Programming

Since there are m hard constraints (deadlines), and one soft constraint (cost) linear programming appears to be the obvious solution. However, we observe that no formulation of a *single* linear programming problem can capture changes in the "topology" of a solution in a straightforward manner (Figure 5).

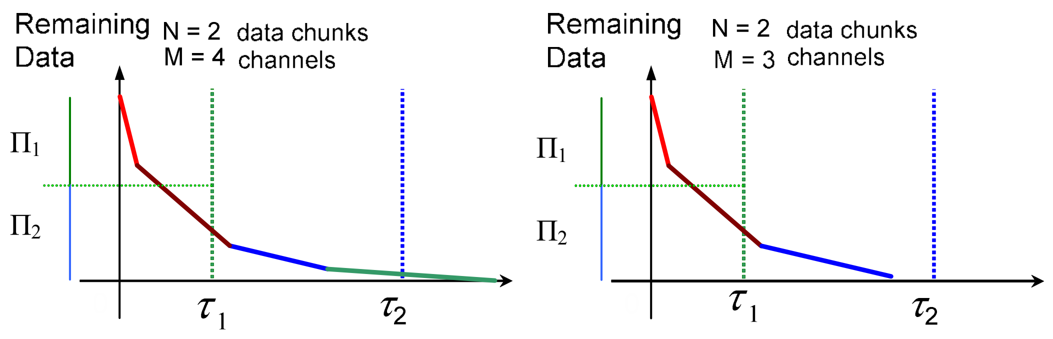
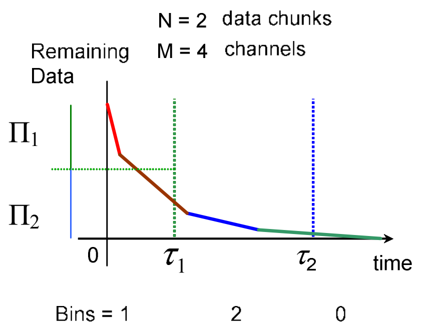


Figure 5: Different "topologies" of a solution. Transitions may occur between different pairs of deadlines, and some channels may be used at all.



Count transitions between $t = 0$ and τ_1.

...

Count transitions between τ_i and τ_{i+1}.

For each series of transition counts, run linear programming separately.

Lowest cost solution among all feasible solutions is *the* solution.

Figure 6: Brute-force approach to solving via linear programming.

However, solving this problem using this simple brute-force method requires $O(m^n)$ instances of linear programming. This solution suffers from a combinatorial explosion as the number of data chunks increases. To simplify the problem we assume that all channels are being used for each deadline. The final, general formulation of the linear programming problem constraints is shown in Figure 7.

$$Minimize\ \ Cost = \sum_{i=1}^{N*C} c_i(t_i - t_{i-1})$$

$$\forall\, n = 1\ to\ C \qquad \sum_{i=1}^{N*n} b_i * (t_i - t_{i-1}) \geq \sum_{1}^{n} \pi_i$$

$$\forall\, i = 1\ to\ C, t_{N*i} \leq\ \tau_i$$

$$\forall\, i = 1\ to\ N * C, \qquad t_{i-1} \leq\ t_i$$

Figure 7: General formulation of problem as a linear programming constraints.

3. Results

To evaluate the practicality of our solution we began by running our algorithm with a range of simulated input values and channel scenarios. We then implemented our algorithm on a smartphone to compare the actual performance with our simulation results.

3.1. Simulations

We implemented a simplex solver for linear programming to explore how our algorithm performs in simulation. We found that when all tasks are known in advance, and channels are static the algorithm works as expected if the input is "satisfiable", that is the deadlines can be met for all jobs. Unsurprisingly, we found that this was not the case when deadlines cannot be met with the available resources, or when the schedule must be recomputed due to changes in the channels available or the jobs to be sent. When there are not enough available resources to successfully transfer all jobs by their respective deadlines, incorporating priorities would be a reasonable solution. However, in the usage scenario we are designing for, deadlines are easily met using the resources available. The difficulty is only in finding the most cost effective means of doing so. We therefore focused our attention on the problem of recomputing the LP solution given new input parameters.

The recomputation is a significant issue because each iteration of the simplex solver takes some time. If the recomputation takes longer than the rate of fluctuations then the problem will have changed by the time the solution has been found. We explored this more by implementing our algorithm into an application and evaluating it in a real world setting.

3.2. Implementation and Application

We implemented our algorithm in C# as a connection manager prototype on an HTC 3300 smartphone with a 200 MHz processor. The client side architecture is shown in Figure 9.

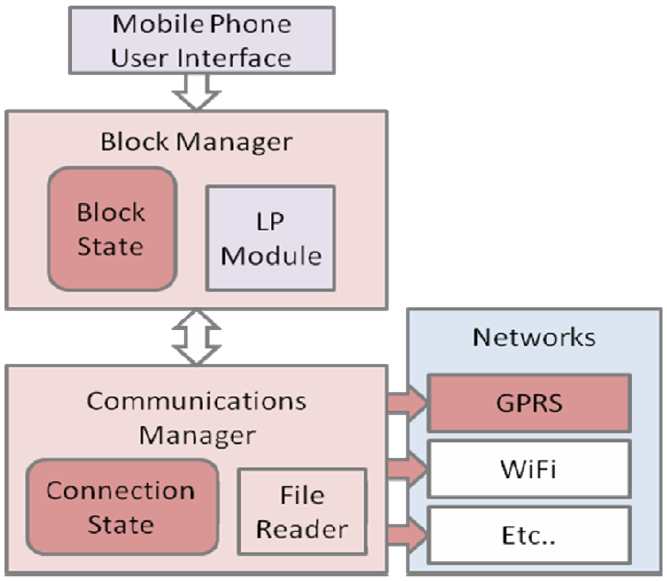


Figure 9: Client-side Architecture

The block manager keeps track of the blocks (tasks) requested to be transferred by the user interface, retrieves connection state information from the communications manager, and submits new schedules to the communications manager. The communications manager takes the schedule and divides the blocks up into chunks to be sent out on the assigned network interfaces. Connection fluctuations and block changes cause the system to obtain a new schedule from LP module. The implementation details are found in our technical report [6] which includes both client and server implementations, communication protocol, loss recovery, crash recovery, and logging. The performance of our application mirrors that of our simulation, but we did run into slight imperfections due to bandwidth estimation inaccuracy, data losses due to interrupted threads and dropped packets. We consider these to be areas of improvement for future work.

5. Conclusion and Future Work

In this work we have developed an algorithm for transferring deadline constrained data while minimizing cost across a heterogeneous set of connections. We formulated a simple version of the problem with several assumptions as a linear programming set of constraints. We gradually relaxed these assumptions until we arrived at a more general algorithm. Through simulations we found that the main drawback of the problem was the long amount of time it took to rerun the LP module. Recall that our algorithm re-runs LP whenever tasks change or bandwidths change beyond certain thresholds. Finally we implemented our solution on a smartphone to explore the practicality of our solution. We found that the actual performance is nearly optimal except in certain corner cases. The general imprecision of our implementation were due to bandwidth estimation, computation time of our algorithm, and data loss and retransmits across the connections. If job changes occur at a rate which is less than LP computation time which is typically under 10 seconds on our smartphone our application performs well. For higher rate fluctuations which typically are due to rapid bandwidth changes, we intend to use some heuristics to either improve the computation time or smooth out the fluctuations using better bandwidth estimators (currently a moving average of perceived bandwidth). Although this framework for minimizing data transfer costs is functional, we are working on generalizing our algorithm to incorporate slack to avoid missed deadlines and a failure model for missed deadlines. In the future we intend to optimize our solution by taking into account the overhead of retransmits due to lossy connections and the underlying networking protocols.

References

[1] World Bank. 2006. Information and Communications for Development 2006: Global Trends and Policies.
[2] Stojmenovic, I. and X. Lin, Power-aware localized routing in wireless networks, IEEE Int. Parallel and Distributed Processing Symp., 2000.
[3] Cheng, X.Z. and Du, X.Z. Virtual backbone-based routing in multihop ad hoc wireless networks. Technical Report 002, University of Minnesota, June 2002.
[4] Broch, J., Maltz, D.A., and Johnson, D.B. Supporting Hierarchy and Heterogeneous Interfaces in Multi-Hop Wireless Ad Hoc Networks. In Proc. of the Workshop on Mobile Computing, June 1999.
[5] Wang, H. J., R.H. Katz, and J. Giese. Policy-Enabled Handoffs Across Heterogeneous Wireless Networks. In WMCSA 99, 1999.
[6] Chen, J., Murtaza, G., Adimulam, S., Singh, G., Toyama, K. Cost-Aware Deadline-Driven Data Transfer Across Heterogeneous Networks. Technical Report Microsoft Research India 2008.
[7] Canalysis.com. Worldwide Smartphone Growth Metrics.
mirror: http://www.dailywireless.org/2008/11/05/smartphone-growth-metrics/.
[8] R. Michael Buehrer. Radio Resource Management in 3G CDMA, 12th Virginia Tech/MPRG Symposium on WIRELESS PERSONAL COMMUNICATIONS June 5-7, 2002.
[9] Qunfeng Dong, Suman Banerjee, Benyuan Liu. Throughput Optimization and Fair Bandwidth Allocation in Multi-Hop Wireless LANs, IEEE INFOCOM 2006.
[10] A. Aguiar, A. Wolisz, and H. Lederer, "Utility-based Packet Scheduler for Wireless Communications", In Proc. of The 6th IEEE Workshop on Wireless Local Networks, Tampa, FL, USA, November 2006.
[11] Francisco J. Ros, Pedro M. Ruiz, Antonio F. Gómez-Skarmeta: Performance Evaluation of Existing Approaches for Hybrid Ad Hoc Networks Across Mobility Models. MSN 2005: 886-896
[12] Dongmei Zhao, C Dongmei Zhao. Radio Resource Management for Cellular CDMA Systems Supporting Heterogeneous Services IEEE Transactions on Mobile Computing (2003).