Beyond flow networks: Internet flow routing with clock nets

Abstract

Internet flow routing refers to the problem of finding the best route for transmitting a large data set to a receiver who may be located across the globe. The problem constraint is that the capacity of the network between the sender and receiver changes during the day depending on traffic intensity. A time-varying flow network is the state-of-the-art model underlying flow routing algorithms. The fastest universal max flow algorithm for time-dependent flow routing is NP-complete. Since flow networks do not incorporate features inherent to the internet, the paper develops a new model for internet flow routing called clock nets. Using clock nets, the paper develops a new max flow algorithm with polynomial time complexity.

1 Introduction

Internet flow refers to the problem of finding optimal capacity paths between a sender and receiver for transmission of data. This is the problem underlying applications that involve transmission of bulk data; examples include inter-data-center transfers and transmission between cloud providers and their clients. The objective is to find the cheapest and fastest route for transmission of a given bulk data set. This problem falls under the domain of dynamic max flow algorithms. The algorithms for dynamic max flow have high complexity; the fastest universal maximal dynamic flow algorithm is NP-complete [12]. *This paper develops a polynomial time algorithm for dynamic max flow.* The **contribution** of this paper is the development of a new simple solution technique for a well established hard problem, namely, time synchronized max flow. This problem underlies internet bulk transmission. The **unique** feature of this paper is that we have solved the problem of developing a simpler routing algorithm by focusing on the model - instead of using flow networks as the underlying model, we develop a new model, clock net, more suitable to encapsulating internet max flow. The computational simplicity of the algorithm is a consequence of the suitability of our model.

A flow network is a graph theoretic model, and is the underlying data structure for flow routing algorithms. The sender and receiver are connected by a network of nodes and edges (links). Each edge has flow capacity which is defined by the rate of flow along the edge. The available flow capacity along an edge may change with traffic intensity. The nodes may have storage capacity where flow can be temporarily stored until capacity opens up in the outgoing edges.

The study of flow networks is an established area, and there are several algorithms for finding the route with maximum flow from sender to receiver. There are other objective functions of relevance such as finding the route that transmits a given flow in the shortest time from sender to receiver. The complexity of flow routing algorithms is dependent on the number of nodes and edges in the underlying flow network, the transmitted data size, and the flow time allowed by the application. Global systems such as the internet have a large number of nodes and links, so existing flow routing algorithms do not scale well.

This paper evaluates flow routing from the perspective of one representative application, namely, bulk transmission via the internet. The bulk data sets sizes fall in the range of tens of gigabytes to petabytes. Bulk transmission is an application whose time has come with the establishment of cloud computing and big data. Cloud providers transfer big data sets between data centers around the globe; bulk transfers also occurs between cloud providers and their clients. The objective is to find the cheapest, quickest route between the sender and receiver.

Standard internet routing algorithms find the shortest hop paths, but the shortest hop path is not necessarily the best route for bulk transmission. For example, consider links with capacity of 20 Mb/s and 2 Gb/s: a 100 MB file is transmitted in less than a minute over either link, while a 100 GB file takes 11.36 hours over the 20 Mb/s link and less than 7 minutes over the 2 Gb/s link, a reduction of 99% of transmission time. For bulk transmissions, routes with maximum capacity are optimal.

The study of transmission of very large data sets via the internet is fairly new. The key objective of bulk routing is on availing maximum capacity using parallel transmission protocols such as GridFTP, BitTorrent, and Slurpie. Since accessing large bandwidth can have negative consequences on the performance of other internet applications, bulk transmission should minimize this negative impact. The Qbone protocol [10] achieves this objective by lowering the priority of bulk packets. Lowering priority of standard TCP packets has also been presented as a viable option for bulk transmissions [2, 11]. Another approach is to find the least congested paths from sender to receiver. The most promising of these approaches is OpenFlow [7].

Bulk routing algorithms can combine the objective of accessing maximum bandwidth by parallel techniques and minimizing negative impact on other applications by taking advantage of bandwidth availability distribution. In most networks, high bandwidth is available during early morning hours when users are sleeping [4]. Since high bandwidth times vary by time of day which is dependent on global positioning, store-and-forward transmission is recommended for high throughput bulk transmission at low cost [6, 9]. Bulk data are stored in transit storage hubs until bandwidth opens up in the receiver.

A time-varying flow network is the state-of-the-art model underlying bulk store-and-forward routing. The study of flow networks is an established area, so there exist algorithms for generating optimal routes satisfying various objectives [12]. A couple of papers [3, 5] have developed flow network routing algorithms specifically for bulk transmission. Routing algorithms fall under the category of search algorithms, so the complexity depends on the search space which is a function of the number of nodes, links, file size, and range of flow time. The global scale, bulk file sizes, and diurnal time range of bulk transmissions results in a large search space. Consequently, by its very nature, bulk routing algorithms have high complexity [3, 5, 12]. Prior papers have generated the routing algorithm with flow networks.

We first develop a flow model of bulk transmission and show why flow networks are a poor routing model for global systems. Next, the paper identifies characteristics required of a time dependent flow routing tool. This is done in conjunction with identifying the level of modeling detail required to develop meaningful, practical and valid routing algorithms. The paper then develops a new data structure, clock net, for flow routing in the internet. Finally, the paper uses clock net to develop a bulk routing algorithm with polynomial time complexity.

2 Modeling bulk transmission

Bulk transmission, also referred to as big data transmission, is the routing of a bulk number of packets from a sender's network to a receiver's network via one or more intermediate transit networks. Each network along the path is an independent unit managed by individual ISPs, so a network is often called an Autonomous System (AS). Each AS has a negotiated bandwidth capacity linkage with one or more ASs. The underlying routing model is a flow network with the sender, receiver, and transit ASs represented by nodes. An edge connecting two nodes represents the inflow/outflow bandwidth between the two nodes. Prior papers [3, 5] have developed routing algorithms by assuming that the nodes and edges along with their bandwidth capacity are given inputs to the problem of routing. For interconnected networks on the scale of the internet, the construction of the model is a significant part of the dynamic flow routing problem. This paper starts with the construction of the underlying flow model.

Between a sender and receiver's AS, there are several paths on the internet. If every single path is modeled, then the underlying flow network would be a hodgepodge of nodes and edges. A routing algorithm has to search for an optimal path from this jumble of paths. However, if every path is not included, then the route selected by the algorithm need not be optimal. The task of a modeler is to determine the minimum number of nodes and edges that need to be included in the underlying routing model in order to be confident that the route selection is optimal. Here, we first make the minimal assumption that the only inputs are the sender and receiver's global positioning and bandwidth. The transit nodes, network of edges, and the edge capacities must be determined.

The edge bandwidth is constant when a fixed bandwidth is purchased or leased for bulk transmission. In most cases, bulk transmission is another application vying for bandwidth on the internet, and the bandwidth available for bulk transmission is dependent on internet traffic. The higher the bandwidth usage by other applications, the lower the bandwidth available for bulk transmission. The bandwidth usage has a diurnal wave pattern that mimics the sleep-wake cycle of internet's users [4]. During early morning hours when users are sleeping, there is low internet traffic leaving maximum available bandwidth; during the day, internet traffic increases reaching peak usage around 8:00 PM, leaving minimal available bandwidth. Thus, the available bandwidth distribution, shown in Figure 1, inversely mimics the sleep-wake bandwidth usage cycle. Table 1 tabulates free bandwidth; the distribution has been deliberately simplified since it is used in examples through the paper.

Table 1: Simplified example highlighting the diurnal wave distribution of bandwidth availability. LTC = Local Time Clock; BW/hr = Bandwidth per hour; * = available base bandwidth; C = Constant value

LTC	BW/hr	LTC	BW/hr
12:00 AM	* + 10C	12:00 PM	* + 8C
01:00 AM	* + 14C	01:00 PM	* + 7C
02:00 AM	* + 16C	02:00 PM	* + 6C
03:00 AM	* + 17C	03:00 PM	* + 5C
04:00 AM	* + 18C	04:00 PM	* + 4C
05:00 AM	* + 18C	05:00 PM	* + 3C
06:00 AM	* + 17C	06:00 PM	* + 2C
07:00 AM	* + 16C	07:00 PM	* + 1C
08:00 AM	* + 14C	08:00 PM	*
09:00 AM	* + 12C	09:00 PM	* + 1C
10:00 AM	* + 10C	10:00 PM	* + 4C
11:00 AM	* + 9C	11:00 PM	* + 8C

The flow network representation of bulk transmission that incorporates the shared nature of the internet has time-varying bandwidth capacity along the edges. Since available bandwidth distribution follows a diurnal wave pattern, the *cycle time* of the flow network is 24 hours. The capacity along each edge of the flow network varies for a period of 24 hours from the start of transmission. The capacity distribution along an edge is determined

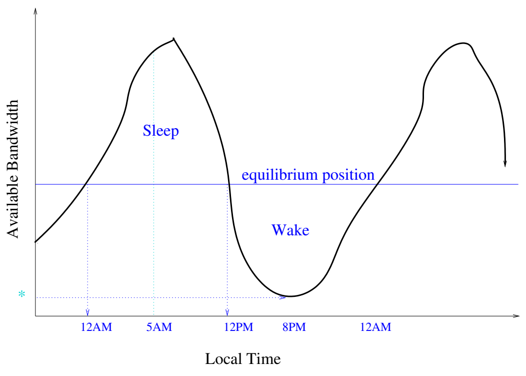


Figure 1: Available bandwidth distribution

by the end nodes as explained below.

Consider two ASs, A and B, that each have available bandwidth distribution given in Table 1. Without loss of generality, in the Table, set the base bandwidth $* = 0$ and $C = 1$. If nodes A and B are in the same time zone, then the edge (A,B) has capacity distribution shown in Table 1. If nodes A and B are in different time zones, the the edge capacity at time t cannot be greater than the minimum of A and B's availability distribution at time t. For example, suppose node A is situated in a location that is 12 hours away from node B. If t is 3:00 AM at node A, then t is 3:00 PM at node B; so the flow bandwidth along path from A to B cannot be greater than 5 (minimum of 17 and 5). Since A and B are geographically distant, there are other ASs along the path. It is not necessary to explicitly plot these nodes in the underlying flow network, but their impact should be incorporated. At time t, if the transit ASs between A and B have bandwidth that is not lower than the minimum of A and B's bandwidth, then edge (A,B) has bandwidth capacity of 5, else edge (A,B) bandwidth capacity is lower than 5. Thus, the end-to-end transmission between A and B is much lower than the actual capacity available in each of the nodes.

If an aptly selected transit node, say C, is included in the model, then more data can be transmitted between A and B by leveraging C as a store-and-forward node. Suppose C is 6 hours ahead of A and 6 hours behind B, and C also has available bandwidth distribution of Table 1. When it is 3:00 AM at A, it is 9:00 AM at C, and 12 units of flow can be transferred from A to C. After another 16 hours it is 1:00 AM at C and 7:00 AM at B, so the 12 units of flow can now be transferred from C to B within the hour. By including transit node C and ensuring that it has storage capacity, it is possible to transfer more data from A to B within the cycle time. Node C's bandwidth distribution must be sufficiently time synchronized with that of nodes A and B. Thus, the selection of transit nodes in the model is critical to the route selection and the performance of the flow transmission application.

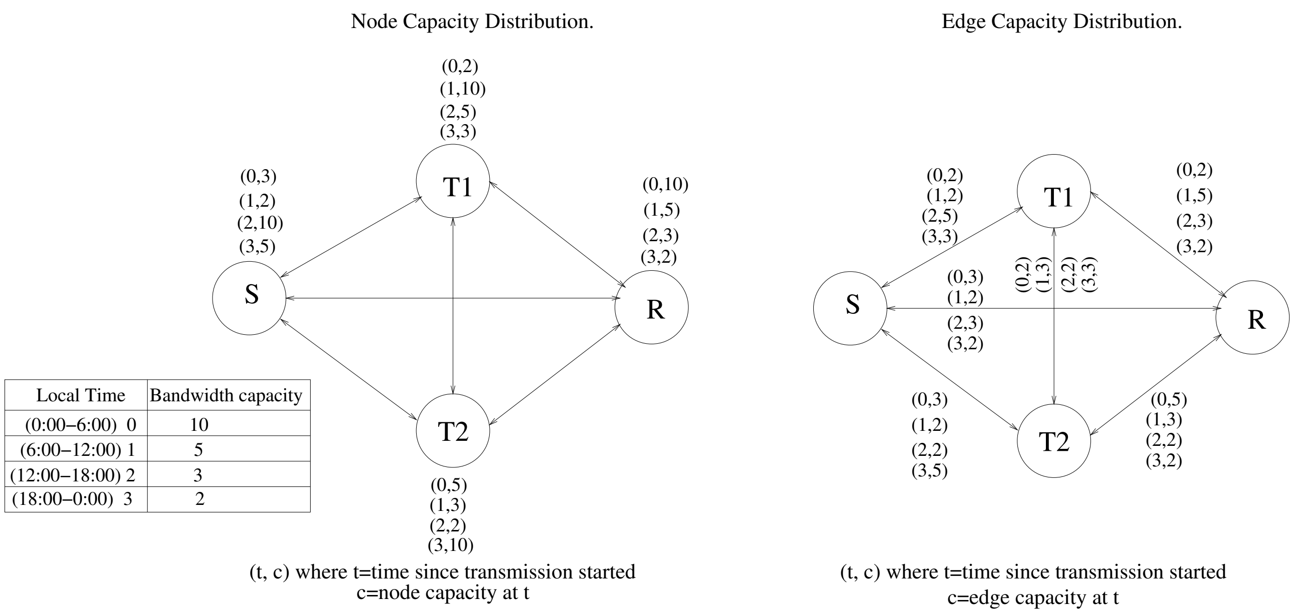
For the internet, the sleep-wake availability distribution is critical since the sender and receiver may be separated by several time zones. The selection of transit nodes reduces to the selection of location where storage is essential to synchronize the temporal misalignment of the sender and receiver's capacity distribution. This implies that it may be necessary to model a storage node in every time zone between the sender and receiver. Given the spherical shape of earth, transmission can occur either eastward or westward between the sender and receiver. Thus, the model must include a storage node in every time zone. It is necessary to model only 1 node in a time zone if the impact of multiple nodes is encapsulated in the single node.

The time zone granularity of the model can be varied. In the coarsest limit, there are no transit nodes between the sender and receiver; in this case, the edge capacity at time t is equal to the minimum of the sender and receiver capacities at time t. In some applications, the locations of the transit storage nodes are also given as inputs. For example, cloud providers have data centers around the globe, and the optimal route must be generated under the constraint that only these data centers can be used as store-and-forward hubs.

Once the transit nodes are selected, the next task of the modeler is to determine what edges must be included in the model. Note that every edge allows flow in both directions. For interconnected networks such as the internet, any two nodes are connected, either directly or indirectly via other nodes. Recalling the earlier example with the three nodes A, B, C, there are 3 edges (A,B), (A,C), (C,B); the edge capacities vary depending on the end nodes. The graph is fully connected with edges and flow permissible between any 2 nodes. A modeler could either use a fully connected underlying network, or have a graph that links edges that are closer in proximity. For example (A,C) and (C,B) are included, but not (A,B) since C geographically lies between A and B.

3 Flow network

The next step is to construct the flow network. At this modeling stage, the input parameters are the locations and the available bandwidth distributions of the sender, receiver, and transit storage nodes. There is a selection regarding the edges in the graphs, either the graph may be completely connected, or some of the edges may be specified. The routing path generated by the partially connected graph may not be optimal, but the search space would be smaller.



Local Time	Bandwidth capacity
(0:00–6:00) 0	10
(6:00–12:00) 1	5
(12:00–18:00) 2	3
(18:00–0:00) 3	2

Transmission start time when local time at S is 12:00, T1 is 18:00, T2 is 6:00, R is 0:00.

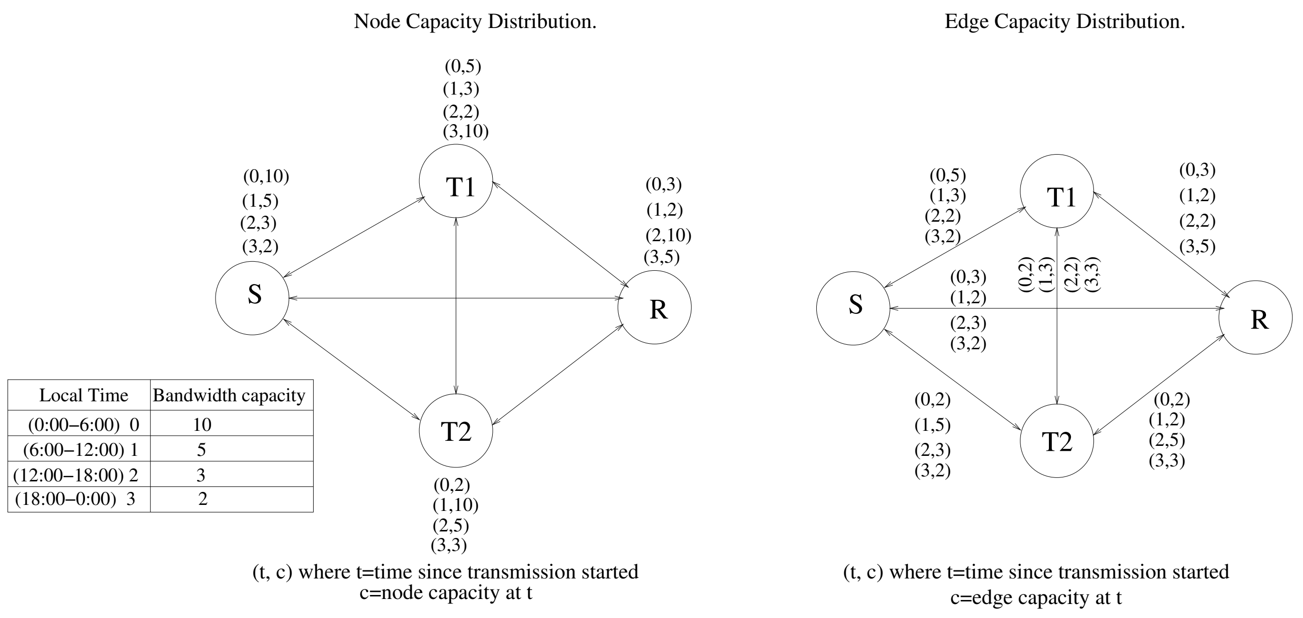
Figure 2: Time varying flow network

In order to construct the flow network model, another input is necessary, namely the start time of the transmission. A flow network measures time relative to when the flow is initiated at the sender. The time t represents time transpired since flow initiation at t=0; $t = 0, 1, 2, \cdots, T-1$, where T is the cycle time and is the maximum allowable flow time from the sender s vertex to the receiver r vertex.

The initiate time determines the edge capacities of the flow network as explained here. Reconsider the example of nodes A, B, C, (presented in the last section), and suppose the transmission is initiated from A at A's local time 3:00 AM. For the flow network clock, t=0 when it is 3:00 AM at A's local time (node capacity is 17), 3:00 PM at B's local time (node capacity is 5), and 9:00 AM at C's local time (node capacity is 12). At time t=0, the edge (A,B) has capacity 5, the edge (A,C) has capacity 12, and the edge (C,B) has capacity 5.

The unit for time can be 1 second, 5 minutes, 1 hour, or any appropriate time division. The value of T is set according to the chosen unit for t. If hour is chosen as time unit, then T=24 since the distribution of free capacity has a diurnal wave pattern. The distribution of edge capacities changes with the initiate time. In the next example, we show that the underlying flow network changes for each initiate time.

Example 1 *Consider the flow network of Figure 2, where cycle time T=4, so t=0, 1, 2, 3 and the time unit is 24/4 = 6 hours. The bandwidth distribution for every node is identical, but the nodes are located in different*



Local Time	Bandwidth capacity
(0:00–6:00) 0	10
(6:00–12:00) 1	5
(12:00–18:00) 2	3
(18:00–0:00) 3	2

Transmission start time when local time at S is 0:00, T1 is 6:00, T2 is 18:00, R is 12:00.

Figure 3: Transformed flow network of Figure 2 when start time shifted by 12 hours (2 time units).

time zones. From node S, node T1 is 6 hours ahead, node T2 is 18 hours ahead, and node R is 12 hours ahead. Suppose transmission start time is 12:00 S's local time. The first flow network shows the node capacity, while the second flow network shows the edge capacity.

Figure 3 shows the flow network when the initiate time at the sender is 0:00. The edge bandwidth distribution changes with the change in initiate time.

Changing the initiate time transforms the flow network since the distribution of edge capacity changes. If the flow network is mapped to a coordinate system with time and capacity on the axes, then shifting the initiate time is equivalent to a translation of the flow network. For each value of initiate time, t=0, 1, 2, .., T-1, a transformed flow network exists and has to be solved to find the best route with this initiate time. In order to find the best route for the best initiate time, all T flow networks must be solved.

There is another issue with the flow network model, namely, the total edge capacity of a node could exceed node capacity. For example, at time t=0, node s of Figure 2 has total capacity of 3, but the outflow from node s is 8 (2 to T1, 3 to r, 3 to T2). Standard flow routing models do not face this problem since capacities are only defined for edges, not nodes. For bulk transmission, an edge capacity cannot exceed the capacity of its end nodes, but if a node has several edges, the sum of edge capacities from the node can exceed the total capacity of the node.

If a constraint specifying that total sum of edge capacities cannot exceed node capacity is introduced, then the constructed model may not generate the optimal route. Therefore, this constraint should be enforced by the routing algorithm by checking that the flow to/from a node not exceed the total node capacity. The search space for the routing algorithm increases since different combinations of flow from a node must be evaluated to find the optimal route. Referring back to the example, the optimal route may require 2 flows along (s,T1) and 1 flow along (s, T3), but in order to find this route, all other combinations may have to be tested.

4 Beyond flow networks

A flow network is a graph theoretic model where nodes and all edges between nodes are required to describe the overlaying system. A time-varying flow network is the underlying model for generating bulk routing paths, but we have identified 2 issues with flow networks, namely,

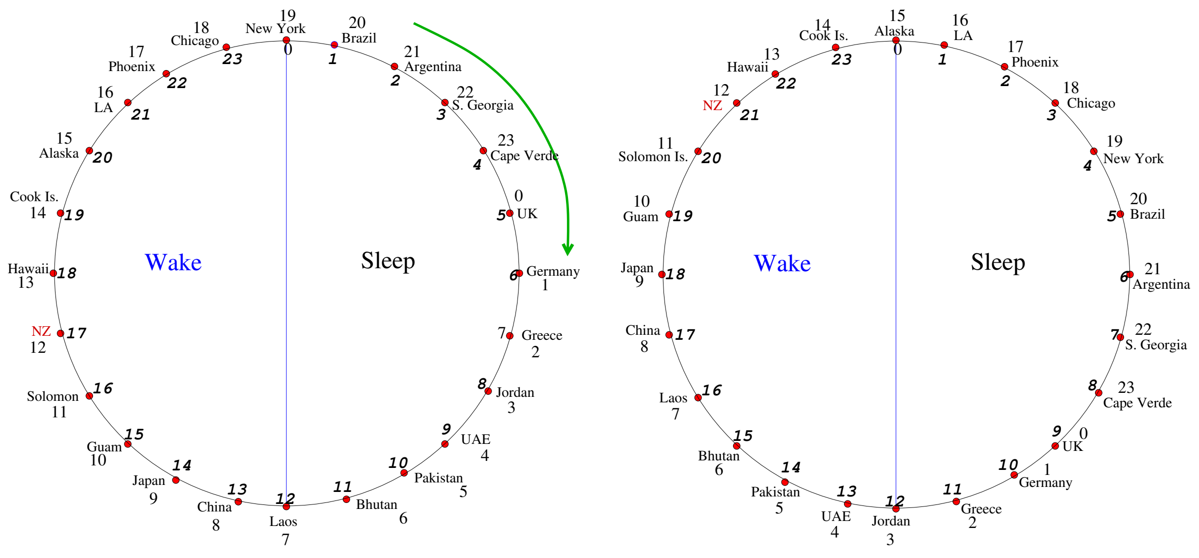
1. T flow networks may have to be constructed and solved in order to find the optimal route, and
2. the complexity of the flow algorithms.

The complexity is dependent on the number of edges, nodes, the bulk file size, and the range of flow time 0, 1, ..., T-1; in addition, the property that sum of edge flows at a node could be greater than the node's flow increases the search space.

In an earlier paper, we have formally developed the flow model for bulk transmission and proved several results. While flow networks are a good routing tool for small networks, they perform poorly as an internet flow routing tool for the following reasons:

1. a flow network does not capture the relative positioning between nodes. From the flow diagram of of Figure 2, it is not possible to judge the relative positioning of S, T1, T2, R.
2. a flow network does not model passage of time nor the relationship between local time and geographic positioning. This is the reason that a new flow network is required when the start time is changed.

The main problem is that the flow network is neither a map nor a clock, and modeling global systems requires encapsulation of both geographic positioning and time. There are three different times of relevance to the bulk transmission problem, namely, the local time at each node, the time differences between the nodes, and the transpired time since the start of transmission. The complexity of the routing algorithms arises from



1:00 AM in Brazil (zone 20) 5:00 AM in Brazil (zone 20).

Figure 4: Global clock with 24 nodes - configuration at time t: The numbers outside the circle represent the zone number (based on UTC) of node; the numbers inside the circle represent local time at the zoned node. Each zone is also referred to by the name of a country/city. The zone numbers rotate clockwise while the circle remains fixed. The zones have shifted 4 positions to the right in the second figure.

the inability of the flow network to capture the relative positioning between the nodes and the inability of the flow network to capture the relationship between local time, universal time, and transpired time at the nodes. A model for internet flow routing should incorporate a map, a clock, and a mapping between geographic positioning, time, and capacity/cost.

5 Global clock

The global clock is a tool that maps global positioning and local time. The global positioning of a point is given by GPS coordinates of latitude, longitude, and elevation. The geographic map is partitioned into time zones, where all the points located in a time zone have the same local time. By mapping GPS coordinates to time zones, a global clock is a 1-1 function that maps time zones and local time.

There are 24 standard time zones when each zone is an hour away from its adjacent time zones. Without loss of generality, let Greenwich time zone (UTC=0) be designated as zone 0 and zone numbers increase by 1 in the clockwise direction. Thus, UTC + 1 is zone 1 and UTC - 1 is zone 23. When hour is the unit of time, both local time (LTC) and zones (UTC) take discrete integer values from 0 to 23. The unit of time need not be an hour; if the unit is 30 minutes then time and zones range from 0 to 47; if the unit is 2 hours then time and zones take discrete values from 0 to 12. The necessary constraint is that the 1-1 mapping between local time and zones be maintained.

Let T represent the number of zones partitioning the clock. The local times and zones take discrete integer values from 0 to T-1. The function of the global clock is to track the local time at all zones during a 24 hour period. Let t represent clock time, that is, the transpired time since the clock starts. The initial position of a clock is the configuration at time t = 0. Figure 4 shows an example initial configuration at t=0. In the figure, the local times are the numbers inside the circle's circumference, the zones are the numbers outside. The name of a country/city in the zone is also shown in the figure. For example, it is 1:00 AM in Brazil which is in zone 20. After every 24/T hours, the zones shift 1 position to the right, and the clock's transpired time gets incremented by 1 (*i.e.,* t=t+1). The second clock in Figure 4 shows the clock configuration when t = 4; it is 5:00 AM in Brazil. The circle remains fixed while the zones rotate clockwise by 1 zone for every unit increment of t; so the local time of each zone also increases by 1 during every unit increment of t. The clock's transpired time, t, takes discrete values $0, 1, 2, \cdots, T-1$. The *cycle time* of the clock is T and is the time to execute 1 complete rotation of C. The cycle time may also be referred to as the clock's *period.* The clock functions use modulo T arithmetic; when i = T-1, i+1=0.

Definition 1 *The global clock, C, is modeled by inverse functions l and z where l gives the local time in a zone at t, and z gives the zone number with the local time at t. The time $t = 0, 1, \cdots, T-1$ represents time transpired from the instant C starts rotating.*
The configuration of C at t is given by l(t) and z(t) where $l(i,t) = j \iff z(j,t) = i;\ \ i, j \in \{0, 1, 2, \cdots, T-1\}$.
The initial configuration of C is given by l(t=0), z(t=0). The time unit of C is 24/T hour(s), so $t = t+1$ every 24/T hour(s).
The rotation of C during time $0 \le t < T$ is defined by: $l(i,t) = j \Longrightarrow l(i, t+1) = j+1$.
The relative positioning of time zones is defined by: $l(i,t) = j \Longrightarrow l(i+1, t) = j+1$.

In the first clock of Figure 4, it is 1:00 AM in Brazil, so $l(20,t) = 1, z(1,t) = 20$. In the second clock of Figure 4, t = t+4 and it is 5:00 AM in Brazil, so $l(20, t+4) = 5, z(1, t+4) = 16$. By construction, for $\theta > 0$,

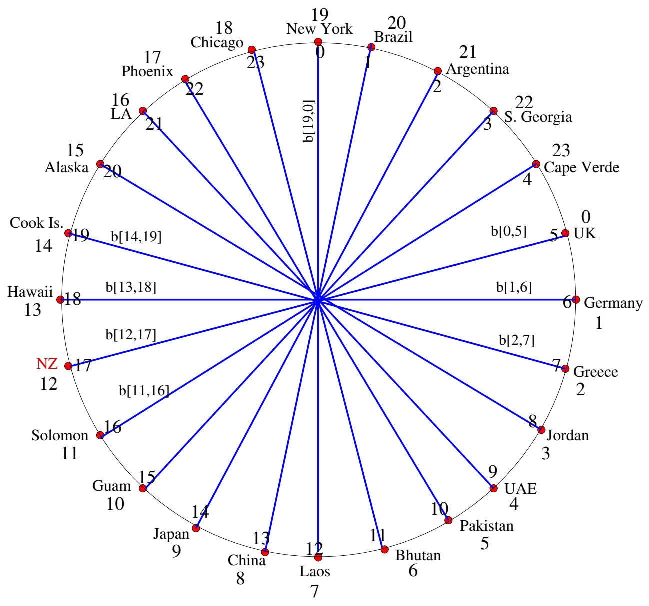


Figure 5: Clock net with cycle time T=24. The capacity in zone i at local time j is b[i,j].

$$l(i, t+\theta) = l(i,t) + \theta; \quad z(i, t+\theta) = z(i,t) - \theta. \tag{1}$$

in mod T arithmetic.

6 Clock net

The global clock is not only useful in tracking time but also in tracking relative positioning of zones. We take advantage of both these properties to develop a global time-zoned flow network.

Definition 2 *The clock net N = (C, b, d) is modeled by global clock C, a 2-dimensional array b[i,j] representing the bandwidth capacity of zone i at local time j, and array d[i] representing the storage capacity of zone i. The period of N (and C) is T and* $i, j = 0, 1, \cdots, T-1$. *The configuration of N at t is given by C(t), b[i, l(i,t)], d[i];* $i = 0, 1, \cdots, T-1$.

The bandwidth capacity of each zone is a function of local time, while storage capacity in a zone is fixed. Global systems like the internet and roadways display link availability distribution based on time-of-day, but storage is assumed to be ample and is typically not dependent on local time. Note that $b[i, l(i,t)]$ can be written as $b[z(j,t), j]$ since the l and z functions are invertible.

A clock net allows one to map distributed systems that span the globe, such as the internet. The clock network naturally encapsulates the effect of time on parameters such as available bandwidth that changes with time of day. In addition, by capturing geographic layout of the globe, a clock net naturally shows the paths between different zones. Figure 5 shows a clock net overlaid on the global clock of Figure 4. Two zones that are not adjacent to each other are connected indirectly via the intermediate zones that lie between the zones. There are two paths from a sender zone to a receiver zone, one in the clockwise direction and another in the anti-clockwise direction. For example, the clockwise path from NY to Germany is via Brazil, Argentina, Cape Verde, and UK; the anti-clockwise path is via Chicago, Phoenix, ..., Jordan, and Greece.

We now define flow, f(i,j,t), between zone i and zone j in the clock net, where flow is the movement of data from zone i to zone j during time t. Without loss of generality, let $i < j$. There exists 2 disjoint paths between zones i and j, one path is in the clockwise direction from i to j and the other path is in the anti-clockwise direction from i to j. The clockwise path is $\langle i, i+1, i+2, \cdots, j-1, j\rangle$ and the anti-clockwise path is $\langle i, i-1, \cdots, j+1, j\rangle$. Flow can be directed along either path. The clockwise flow from i to j at time t is given by:

f(i,j,t) $\leq$ MIN { b[i,l(i,t)], b[i+1,l(i+1,t)], $\cdots$, b[j,l(j,t)] }

The anti-clockwise flow from i to j is given by

f(i,j,t) $\leq$ MIN { b[i,l(i,t)], b[i-1,l(i-1,t)], $\cdots$, b[j,l(j,t)] }

Let the local times in zones i and j be τ_i and τ_j. That is, $l(i,t) = \tau_i$ and $l(j,t) = \tau_j$. From Equation 1, the flow equations can be written as:

f(i,j,t) $\leq$ MIN { b[i,τ_i], b[i+1,τ_i+1], $\cdots$, b[j,τ_j] }

The anti-clockwise flow from i to j is given by

f(i,j,t) $\leq$ MIN { b[i,τ_i], b[i-1,τ_i-1], $\cdots$, b[j,τ_j] }

The total outflow (inflow) at a node cannot exceed the bandwidth capacity of the node. This gives,

$$\sum_{\forall j \neq i} f(i,j,t) \leq b(i,\tau_i), \text{ and } \sum_{\forall j \neq i} f(j,i,t) \leq b(i,\tau_i)$$

The flow stored in zone i at time t, f(i, t), equals the total inflow into zone i minus the total outflow out of zone i by time t. By the definition of f(i,j,t), it follows that

$$f(i,t) = \sum_{\forall j \neq i; \theta \leq t} f(j,i,\theta) - \sum_{\forall j \neq i; \theta \leq t} f(i,j,\theta)$$

At start of clock net rotation, only the sender has stored flow. That is, at $t = 0^-$, $f(s, 0^-) > 0, f(i, 0^-) = 0, \forall i \neq s$. At the end of clock net rotation, $f(r,T)$ is the value of flows sent from s to r within the cycle time T.

We now present the global flow routing problem in the mathematical framework of clock nets.

Definition 3 *Global flow routing from sender in zone s to receiver in zone r is modeled by a clock net N = (C,b,d) where flow is permissible during $t = 0, 1, 2, \cdots, T-1$. The time t is the transpired time since transmission initiation at t = 0.*

Let λ specify a set of paths from sender to receiver, and f(λ, t) be the total flow with solution λ within the time limit $t < T$. That is, λ is the sender to receiver transmission routes generated by a routing algorithm. Then,

$$f(\lambda, t) = \sum_{i \neq r, \theta \leq t} f(i, r, \theta) \tag{2}$$

and

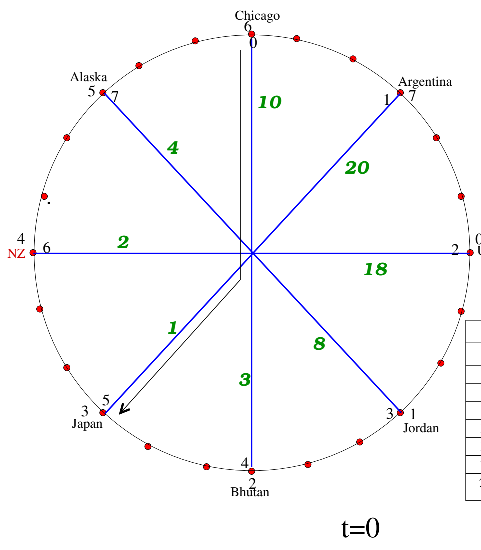
$$f(\lambda, t) = \sum_{i \neq s, \theta \leq t} f(s, i, \theta) - \sum_{i \neq s, r} f(x, t) \tag{3}$$

It follows that $f(\lambda, T)$ is the value of flows sent from s to r within the time limit T using routing λ. For bulk transmission, $f(s, 0^-)$ is the size of the bulk data set at the sender node just before transmission at time t=0, and $f(\lambda, T)$ is the size of the data set transmitted from s to r using routing paths λ within time T.

Bulk transmission routing algorithms address the following optimization problems:

1. maximize the size of the bulk data set that can be transmitted from sender to receiver within cycle period; and
2. minimize the time to transmit the input bulk data set from sender to receiver.

In the domain of clock nets, the problems translate to the following objective functions.



Local Time	Bandwidth/3 hours
0:00–3:00 (0)	10
3:00–6:00 (1)	20
6:00–9:00 (2)	18
9:00–12:00 (3)	8
12:00–15:00 (4)	3
15:00–18:00 (5)	1
18:00–21:00 (6)	2
21:00–23:00 (7)	4

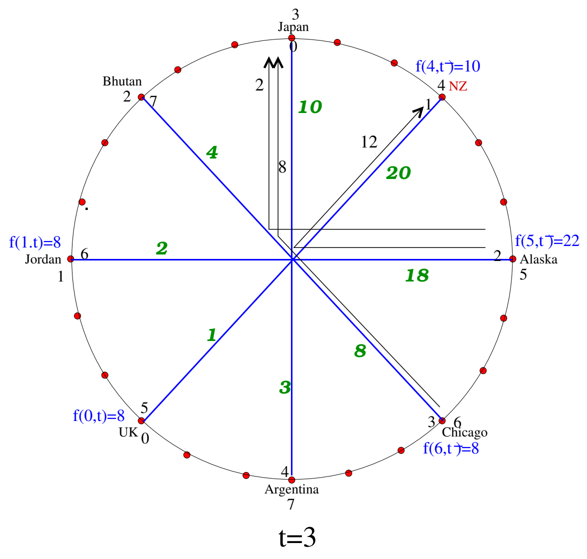


Figure 6: Clock net for Chicago to Japan transmission

Objective function 1 *Generate λ^{max} such that $f(\lambda^{max}, T) \geq f(\lambda, T), \forall\lambda$ in N.*

Objective function 2 *For a given $f(s, 0^-)$, generate λ^*, where $f(s, 0^-) = f(\lambda^*, \tau) > f(\lambda, t)\ \forall\lambda$ in N when $t < \tau$.*

There are other objective functions of interest, such as latest initiate time and latest arrival time, latest initiate and earliest arrival, earliest initiate and earliest arrival, earliest initiate and latest arrival. Instead of time, one could focus on the objective of minimizing cost. If objective functions 1 and 2 are solved, then so can the others. The above objective functions are common to flow optimization problems regardless of bulk size, global scale and diurnal time varying nature of capacity. An objective function of particular interest to big data sizes and global scale is the following: compute all the above objective functions with the additional constraint that all transmissions are constrained to be along edges whose corresponding nodes are in the sleep phase of their bandwidth availability. In the next section, we develop an algorithm to generate routing for maximal flow when all transmission occurs during sleep phase.

7 Algorithm

We first demonstrate clock net powered dynamic flow with an example. The objective is maximal flow from Chicago to Japan using the clock net of Figure 6 with the constraint that transmissions are restricted to nodes

Table 2: Solution by clock net based dynamic flow algorithm

t	S flow f(i,t^-)	S – edge flow –> R f(i,j,t)	R flow f(j,t)	Sleep Region (z(0,t), z(1,t), z(2,t), z(3,t))
0	56	Chicago – 2 –> UK	2	Chicago, Argentina, UK, Jordan
		Chicago – 8 –> Jordan	8	
1	46	Chicago – 10 –> Alaska	10	Alaska, Chicago, Argentina, UK
		Chicago – 4 –> Argentina	4	
		Chicago – 6 –> UK	8	
2	26	Chicago – 10 –> NZ	10	NZ, Alaska, Chicago, Argentina
		Chicago – 8 –> Alaska	18	
	4	Argentina – 4 –> Alaska	22	
3	8	Chicago – 8 –> Japan	8	Japan, NZ, Alaska, Chicago
	22	Alaska – 12 –> NZ	22	
	10	Alaska – 2 –> Japan	10	
4	8	Alaska – 8 –> Japan	18	Bhutan, Japan, NZ, Alaska
	22	NZ – 12 –> Japan	30	
	10	NZ – 2 –> Bhutan	2	
5	8	NZ – 8 –> Japan	38	Jordan, Bhutan, Japan, NZ
	2	Bhutan – 2 –> Japan	40	
	8	Jordan – 8 –> Japan	48	
6	8	UK – 8 –> Japan	56	UK, Jordan, Bhutan, Japan

in the sleep phase. All transmission paths must lie in the sleep phase. This clock net has 8 time zones, so t = 0, 1, .., 7; and the unit of time is 3 hours. For pedagogical simplicity, let all the zones have the same bandwidth distribution given in the table. Since transmission is limited to the sleep phase, a maximum of 56 units of flow can be transmitted. The initial configuration is as shown in the first clock net of Figure 6 with Chicago's local time 0:00. UK is arbitrarily assigned zone number 0, so Japan, which is 9 hours from UK is zone 3, and Chicago is zone 6.

A maximal flow solution is presented in Table 2. The first column gives the transpired time t, the third column gives the transmissions fired at t, the second and fourth column gives the stored flow in the sending and receiving nodes, respectively, corresponding to each transmission, and the last column lists the nodes that are in the sleep region. Using the notation of clock nets, the columns in order are, t, f(i,t^-), f(i,j,t), f(j,t), and (z(0,t), z(1,t), z(2,t), z(3,t)). The second clock net of Figure 6 presents a pictorial representation of the solution at t=3.

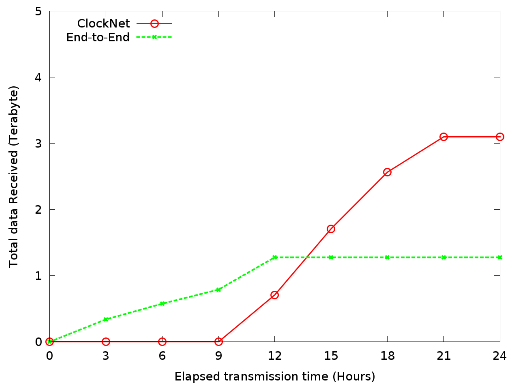
Before explaining the details of Algorithm 1 (presented in the Appendix), we present the underlying principle of the flow logic, namely, nodes at the start of the sleep phase have available bandwidth capacity until the end of the sleep phase; nodes nearing the end of the sleep phase have to get past the wake phase before bandwidth is available for max flow. The logic of the max flow algorithm is as follows: At each t, the sender pushes flow to all nodes with positive residual capacity (*i.e.,* flow that can be end-to-end transmitted to r), starting with nodes at the end of the sleep phase. Once these nodes are exhausted, the sender pushes its capacity flow to nodes with zero or negative residual capacity that are at the start of their sleep phase. The nodes with negative residual capacity must transmit their negative flow to nodes that have positive residual capacity or are earlier in the sleep phase. The receiver pulls flow, starting from nodes at the end of their sleep phase. At the end of each iteration, the clock rotates by 1 time unit.

Briefly, the details of Algorithm 1 are as follows: since all transmissions are limited to the sleep region, set b(i,j)=0 $\forall i$ when j=4,5,6,7. In addition to clock net's parameters, the algorithm has 2 variables, namely, arrays e2e[0..T-1] and res[0..T-1]. The sender is node s, receiver is node r, and t represents the transpired time since the clock starts rotating (*i.e.,* transmission begins). The array e2e[i]$\geq$ 0 represents end-to-end flow that is possible between nodes i and r during the remaining flow period t+1, ..., T-1. The array res[i] $\leq$ e2e[i] represents residual capacity, and equals the remaining flow that can be stored in node i during the flow period t+1, ..., T-1. When res[i] $>$ 0, the value represents the additional capacity of i for end-to-end transmission to r during t+1, .., T-1, and when res[i] $<$ 0, the absolute value represents the additional flow currently stored in i that cannot be end-to-end transmitted to r during t+1, .., T-1; if this excess flow is not eventually transmitted to a node with positive residual capacity, then these flow units will not reach r during the cycle time.

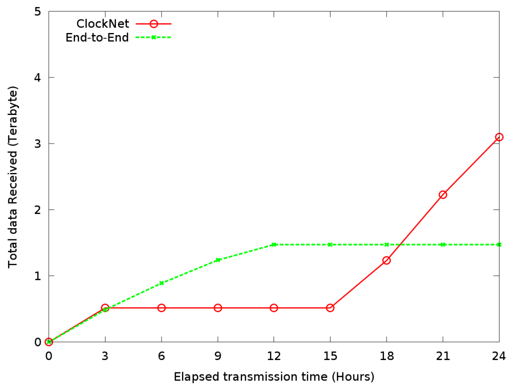
Let x = r - i. The initialization is: $e2e[i] = \sum_{\forall 0 \leq \theta < T/2}$ MIN$\{b(r,\theta), b(i,\theta+x)\}$

Only the end nodes are considered in the end-to-end flow because the paths are constrained to the sleep phase and the assumption of wave pattern of availability distribution ensures that the transit nodes are not the bottleneck. If this assumption is not true, then all nodes in the path should be considered. The end-to-end flow array e2e[i] gets decremented on each clock tick (t++) by MIN$\{b(r,l(r,t)), b(i,l(i,t))\}$.

On each clock tick (t++), set $res[i] = e2e[i] - f(i)$. When flow enters (or departs) i, update res[i] = res[i] - f(j,i,t) (or res[i] = res[i] + f(i,j,t)). If $res[i] < 0$, then this negative flow must be transmitted to a node j which has $res[j] > 0$. The flow to be transmitted from s, $f(s,0^{-1}) = \sum_{\forall 0 \leq \theta < T/2} b(s,\theta)$. Initially, the sender's residual capacity is res[s]=$-f(s,0^{-1}) + e2e[s]$. The initial configuration of the clock places s at local time 0, and the clock configuration is updated each time the clock rotates (t++). Note that all operations are in mod T arithmetic.

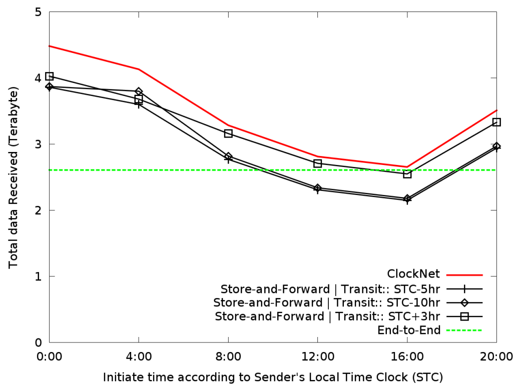


Chicago to Japan (r is 9 hours behind s)

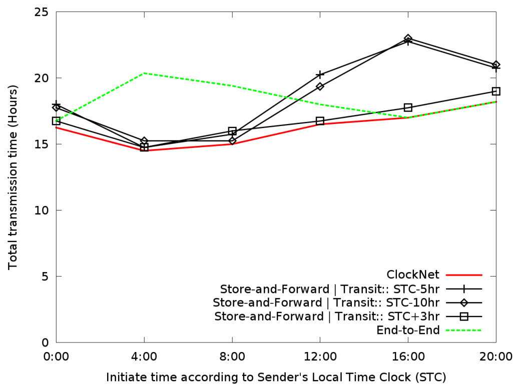


Chicago to Jordan (r is 9 hours ahead of s)

Figure 7: Nodes constrained to transmitting in sleep phase



Bulk data transmitted during 24 hours



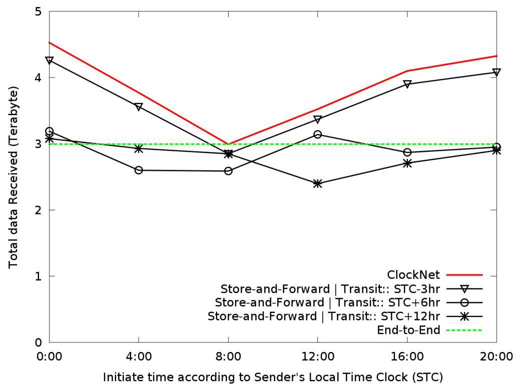
Time to transmit 2TB file

Figure 8: Transmission from LA to Germany along various routes.

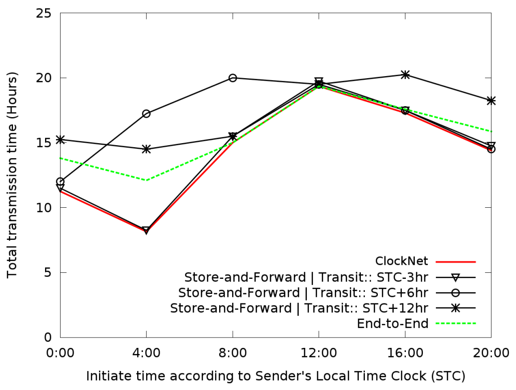
8 Experiments

This section experimentally evaluates the impact of initiate time and route selection on the performance of bulk transmission. We use OPNET, a commercial simulator capable of simulating a wide variety of network components and workloads [8]. The background traffic emulates network usage based on DE-CIX traffic statistics [1] which follows the sleep-wake cycle. A parameter varied in our experiments is the locations of the sender, receiver and transit nodes relative to each other. The location displacement is represented by the time difference of the receiver and transit nodes with respect to the sender's local time. For example, if the sender is in Chicago and the receiver is in Jordan, then the receiver is 9 hours ahead of the sender.

Figure 7 plots the file size when all nodes are constrained to be in the sleep phase while transmitting. The



Bulk data transmitted during 24 hours



Time to transmit 2TB file

Figure 9: Transmission from Chicago to NZ along various routes.

dashed line plots end-to-end transmission from sender to receiver where a sender transmits for 12 hours of its sleep phase and the receiver receives during these 12 hours. The solid line plots store-and-forward transmission along a path generated by clock net where all transmission is limited to nodes in the sleep phase. In the next 2 figures, the clock net transmission is the plain solid red line, the end-to-end transmission is the dashed line, and other lines represent store-and-forward transmissions where the transit nodes are randomly chosen. The sleep phase restriction is dropped, so transmission can also occur during the wake phase for a total of 24 hours. Figure 8 sends data from LA to Germany via routes with transit nodes in various time zones. The first graph plots data transmitted over a period of 24 hours, the second graph plots the total time to transmit a 2 TB file from sender to receiver while the transmission start time from sender is varied. Figure 9 shows similar graphs with receiver 6 hours behind sender.

As one can see, the total data transmitted depends on the initiate time and relative displacement between sender, receiver, and transit nodes. The choice of transit nodes (routes) has a big impact on performance. In fact, the graphs show that rather than choosing random routes, it is often better to transmit directly from sender to receiver. Similarly, the transmission start time determines the duration of transmission.

9 Conclusions

Bulk transmission via the internet is a representative application of time-dependent flow routing. Prior papers have developed bulk routing algorithms using flow networks; these papers started with an already constructed flow network model. This paper, on the other hand, takes a step back and constructs the flow model. We show that the classic graph theoretic time-varying flow network is a poor underlying model for internet flow routing. The paper proposes clock net as an alternative to flow networks for modeling time-dependent internet bulk flow. We then develop a polynomial time complexity algorithm for bulk transmission. Algorithm 1 underscores the naturalness of clock net for internet flow routing. The power is in the modeling; the algorithm simplicity is a consequence of the modeling. The main contribution of this paper is the performance modeling of internet max flow. It should be possible to improve on the algorithm presented here.

The clock net is a natural tool to model the internet with its inter-connectivity and end-to-end flow transmissions. If the transmission start time is changed, then the initial configuration of the clock is changed by setting the local time of the sender to the new start time; the clock net model and algorithm are unchanged. In a flow network model, each change in transmission start time requires the construction of a new model by transformation.

We plan to study possible variations/extensions of the basic clock net presented here. The relation between a clock net and a flow network's parameters has to be understood. The complexity of the clock net algorithm is $O(T^2)$, but T, the period of flow, is interpreted differently in a clock net and in a flow network. In a clock net, the parameters representing time, nodes, and edges are all combined. How does one model a network with 6 nodes where 4 of the nodes are in the US, spaced an hour apart, the fifth node is in Australia, and the sixth node is in Germany. In this case, one would have to model 24 zones, each an hour apart (T=24), but restrict storage capacity to the 6 actual nodes. The search is limited to the 6 nodes (n=6), so the complexity would be $O(Tn)$, (n $\leq$ T). If there are 2 data centers within the same time zone, their bandwidth capacities are added and the 2 nodes are modeled as 1 zone, not impacting the complexity of the algorithm.

References

[1] De-cix where networks meet; statistics, http://www.de-cix.net/about/statistics/, 2013.

[2] Brosh, E., Baset, S. A., Rubenstein, D., and Schulzrinne, H. The delay-friendliness of tcp. In *2008 ACM SIGMETRICS* (2008), ACM, pp. 49–60.

[3] Chhabra, P., Erramilli, V., Laoutaris, N., Sundaram, R., and Rodriguez, P. Algorithms for constrained bulk-transfer of delay-tolerant data. In *ICC'10* (2010), pp. 1–5.

[4] LAKHINA, A., PAPAGIANNAKI, K., CROVELLA, M., DIOT, C., KOLACZYK, E. D., AND TAFT, N. Structural analysis of network traffic flows. In *SIGMETRICS'04* (2004), pp. 61–72.

[5] LAOUTARIS, N., SIRIVIANOS, M., YANG, X., AND RODRIGUEZ, P. Inter-datacenter bulk transfers with netstitcher. In *Proceedings of the ACM SIGCOMM 2011*.

[6] LAOUTARIS, N., SMARAGDAKIS, G., RODRIGUEZ, P., AND SUNDARAM, R. Delay tolerant bulk data transfers on the internet. In *SIGMETRICS* (2009), pp. 229–238.

[7] LIMONCELLI, T. A. Openflow: A radical new idea in networking. *Queue 10*, 6 (June 2012), 40:40–40:46.

[8] LUCIO, G. F., PAREDES-FARRERA, M., JAMMEH, E., FLEURY, M., AND REED, M. J. Opnet modeler and ns-2. In *ICOSMO* (2003), pp. 700–707.

[9] SHI, C., AMMAR, M. H., AND ZEGURA, E. W. idtt: Delay tolerant data transfer for p2p file sharing systems. In *GLOBECOM'11* (2011), pp. 1–5.

[10] TEITELBAUM, B., HARES, S., DUNN, L., SYSTEMS, C., NARAYAN, V., AND NEILSON, R. Internet2 qbone - building a testbed for differentiated services. In *IEEE Network Magazine, Special Issue on Integrated and Differentiated Services for the Internet* (September 1999).

[11] VENKATARAMANI, A., KOKKU, R., AND DAHLIN, M. Tcp nice: A mechanism for background transfers. In *OSDI'02* (2002), pp. –1–1.

[12] XIAOQIANG CAI, DAN SHA, C. K. W. *Time-Varying Network Optimization (International Series in Operations Research & Management Science)*. Springer, 2007.

Algorithm 1: MAX FLOW DURING SLEEP

```
initialize clock
// Sender starts transmission at local time 0.
l(s,0)=0;
initialize e2e[i]; i=0,..,T-1
for t=0 to T-1 do
    update e2e[i]; i=0,..,T-1;
    update res[0..T-1];
    // push from s to nodes with positive residual, starting with r
    while $b(s, l(s, t)) > 0$ do
        if b(r,l(r,t)>0) then
            f(r,t) = f(s,r,t)+f(r,t);
            f(s,t) = f(s,t)-f(s,r,t);
            update b(s,l(s,t)), b(r,l(r,t)), res[s], res[r];
        for j=z(T/2-1,t) to z(0,t); $j \neq s$ do
            if $res[j] > 0$ then
                f(j,t) = f(s,j,t)+f(j,t);
                f(s,t) = f(s,t)-f(s,j,t);
                update b(s,l(s,t)), b(j,l(j,t)), res[s], res[j];
    // push from s to nodes with negative residual
    while $b(s, l(s, t)) > 0$ do
        for j=z(0,t) to z(T/2-1,t) do
            f(j,t) = f(s,j,t)+f(j,t);
            f(s,t) = f(s,t)-f(s,j,t);
            update b(s,l(s,t)), b(j,l(j,t)), res[s], res[j];
    // pull to r from nodes with capacity and flow.
    while $b(r, l(r, t)) > 0$ do
        for j=z(T/2-1,t) to z(0,t), j$\neq$ r do
            if $b(j, l(j, t)) > 0$ then
                f(r,t) = f(j,r,t)+f(r,t);
                f(j,t) = f(j,t)-f(j,r,t);
                update b(r,l(r,t)), b(j,l(j,t)), res[j], res[r];
    // Nodes other than s, r
    // push from nodes with negative residual to nodes earlier in the sleep phase.
    k=z(0,t);
    for j=z(T/2-1,t) to z(0,t) do
        if j = s or r then
            continue
        if $res[j] < 0$ and $b(j, l(j, t)) > 0$ then
            // scan nodes from k until node j.
            // break out of loop when k and j cross over.
            if $k > j$ then
                break
            f(k,t) = f(j,k,t)+f(k,t);
            f(j,t) = f(j,t)-f(j,k,t);
            update b(j,l(j,t)), b(k,l(k,t)), res[j], res[k];
        if b(k,l(k,t)) == 0 then
            k=k+1;
    // rotate clock net by 1 time unit.
    l(s,t+1)= l(s,t)++;
```