Modeling time-synchronized dynamic flow in internet

Elizabeth Varki
University of New Hampshire
email: varki@cs.unh.edu
Phone: (603) 862-2319

Abstract

Time synchronized dynamic flow refers to max flow across networks where link capacity is dependent on the time-of-day. Transportation networks and the internet are examples of networks where traffic is dependent on time-of-day, thereby impacting capacity available for max flow applications. Time-varying flow networks are the underlying model for dynamic flow algorithms. Unlike static flows, dynamic flow algorithms have high complexity due to the number of parameters to be considered, which include the number of nodes, the number of links, the flow duration, and the maximum link capacity. In part, the complexity of the algorithm arises from the non-suitability of the flow network as the underlying model for dynamic flow. Therefore, instead of tackling dynamic flow by focusing on the algorithm, this paper takes another direction by focusing on the model. A new model, clock net, is developed for time-dependent dynamic flow; subsequently, the paper presents a new flow algorithm with complexity $O(T^2)$ where T is the flow duration. The complexity does not depend on the number of nodes, edges, or edge capacity because clock net is not a graph theoretic model. The data structures underlying clock net are a clock and a map. The contribution of this paper is the development of a new model for dynamic time-dependent flow along global, densely interconnected networks such as the internet. The simplicity of the algorithm is a consequence of the suitability of the clock net as a modeling tool for dynamic flow.

1 Introduction

This paper presents a new routing model for max flow across global networks where the flow capacity is dependent on time-of-day. The dependence of flow capacity on time-of-day - available capacity decreases during high traffic times - is prevalent in networks such as the internet and in transportation grids such as the roadways. Max flow across time-varying capacity networks is a fundamental logistical problem for several applications. This paper addresses max flow from the domain of bulk transmission across the internet - how to efficiently transmit a very large data set from a sender's network to a receiver's network [1, 3, 6]. Max flow involves finding the route that allows the maximum amount of data transmission between sender and receiver. The data set can be divided into several subsets that are routed independently; the data can be temporarily stored in-transit, while waiting for bandwidth capacity to become available in outgoing links. This problem falls in the category of dynamic flows which was introduced by Ford and Fulkerson [4] and first studied by Gale [2]. The underlying model for max flow is a time-varying flow network where the nodes have storage capacity and the edges have capacity that vary with time [8, 11].

The fastest algorithm for universal maximal dynamic flow has a complexity of O(UneT^2), where U is the maximum edge capacity, n is the number of nodes, e is the number of edges, and T is the flow duration [10]. Universal maximal flow refers to a flow which is maximum for each flow duration [9]. We have developed an algorithm for dynamic max flow with complexity $O(T^2)$. While our algorithm has lower complexity (O(T^2) versus O(UneT^2)), the point to note is that our algorithm's complexity is not dependent on the number of nodes and edges in the underlying graph. We have developed a faster algorithm by developing a new underlying routing model whose basis is not classic graph theory.

Routing algorithms are constructed from graph theoretic models. What has remained constant in max flow research is the underlying flow network model; new algorithms are developed but the model has remained largely unchanged. The early max flow algorithms assume constant edge capacity and zero node capacity; algorithm complexity depends on the number of nodes and edges in the graph. Time-varying edge capacity is a later add-on, and so is storage capacity at the nodes. The complexity of a time-varying routing algorithm depends on nodes and edges that have to be searched during each time increment of flow duration. While flow networks are a good tool for static flows, they perform poorly for dynamic flows since flow networks do not capture the relationship between time, geographic positioning, and capacity.

There are three different times of relevance to dynamic time-of-day flows, 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 max flow algorithms arises from 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 dynamic flows in global systems should incorporate a map, a clock, and a mapping between geographic positioning, time, and capacity. By construction, such a routing model would automatically weed out inefficient paths, thereby pruning the search space for the routing algorithm. With a reduction in search space, the complexity of the flow routing algorithm would be lower.

This paper develops a new model, clock net, for dynamic flow when capacity is dependent on time-of-day. A clock net is not a traditional graph theoretic model. The underlying data structures for a clock net are a clock and a map. A clock net views dynamic flow as a time synchronization problem, and moves flow units to locations that are time synchronized with the sender and receiver. The **contributions** of this paper are the formulation of a new routing model, clock net and the development of a new dynamic flow algorithm with complexity $O(T^2)$.

The rest of this paper is organized as follows: the next section explains the issues and challenges of modeling dynamic flow in a network as dense and vast as the internet. Sections 3 and 4 formulate our new model, clock net. Section 5 develops a dynamic flow algorithm from clock net.

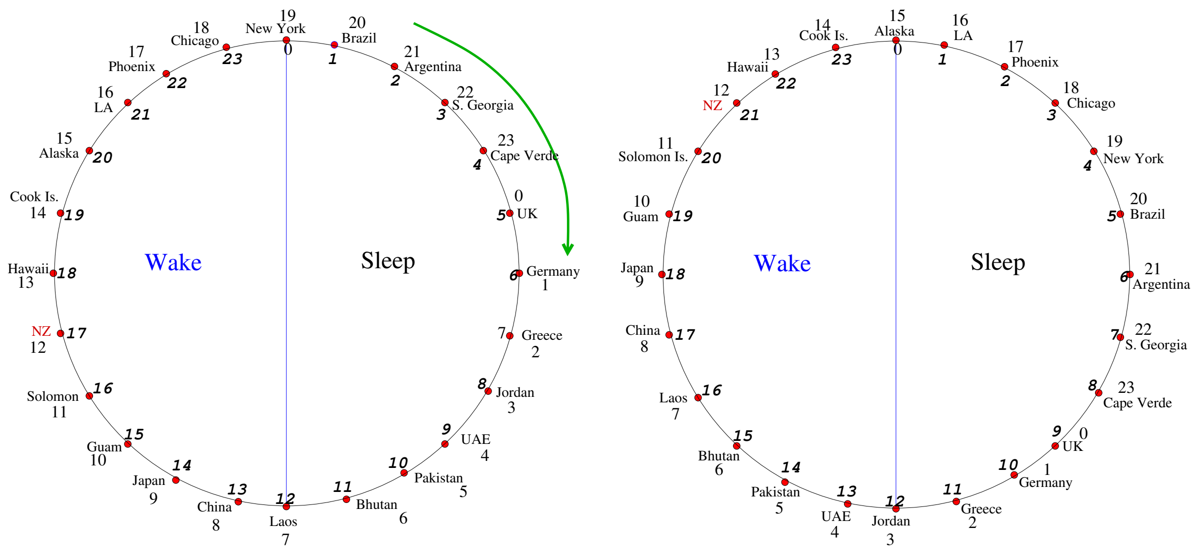
2 Internet flow model

Internet flow is the transmission of flow units from a sender's network to a receiver's network via one or more intermediate transit networks. The sender, receiver, and transit networks are represented by nodes. Each node has a fixed capacity, representing the physical bandwidth of the network. Every network has one or more incoming/outgoing links to the rest of the internet. An edge connecting two networks represents the inflow/outflow bandwidth between the two networks. The edge capacity determines end-to-end flow between the networks and is equal to the minimum capacity of the two nodes. For example, suppose the sender, s, and receiver, r, are campus networks where s has a 10 Gb/s link to the internet and r has a 5 Gb/s link to the internet; the edge connecting s and r has capacity 5 Gb/s (assuming that intermediate transit links are all greater than 5 Gb/s). Thus, a node represents a network in its time zone, while an edge represents inflow/outflow between two nodes and could cross several time zones.

The available capacity of a node represents the free capacity remaining after bandwidth usage by the network's users, and this is the capacity available for max flow applications. Bandwidth usage has a diurnal wave pattern that mimics the sleep-wake cycle of internet's users [5, 7]. 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. The available capacity distribution along an edge is determined by the end nodes as explained here: consider two networks, A and B with identical physical capacity and identical available bandwidth distribution; if nodes A and B are in the same time zone then edge (A,B) has identical distribution as A and B. If nodes A and B are in different time zones, then edge (A,B) 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; A is in the sleep phase and has plenty of bandwidth, but B is in the wake phase and has limited available bandwidth; the edge (A,B) bandwidth at t would be equal to B's bandwidth. Thus, regardless of the physical capacities of the networks, the end-to-end transmission between A and B may be much lower than the available capacity in each of the nodes due to temporal misalignment.

The edge capacity in the internet flow model represents end-to-end flow capacity which is a function of end node capacities. Note that end-to-end flow between nodes in different time zones must cross the intermediate time zones. When A and B are geographically distant, it is not necessary to explicitly plot transit nodes in the underlying flow network, but their impact should be incorporated. At time t, if the transit networks 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 equal to minimum A, B capacity, else edge (A,B) bandwidth capacity is lower than minimum A, B capacity at t.

The internet is a densely interconnected global network. Typically, there are several paths between any two nodes. The number of nodes and links in the internet keep increasing with time. A flow network is a graph theoretic model where nodes and all edges between nodes are required to describe the overlaying system. Thus, a flow network of the internet is a hodgepodge of nodes and links. The flow network model reflects the dense confusion of the internet without providing any order or clarity; the scale and inter-connectivity of the internet and the diurnal bandwidth availability cycle results in having to search for a solution from potentially exponentially many solutions. However, the edges in the flow model are superfluous since the edge capacity is a function of bandwidth distribution of nodes along the path. The internet flow model can be vastly simplified when viewed from the perspective of nodes without edges. In the following sections, we present such an alternative to the network flow model.



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

Figure 1: 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.

3 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. The first clock in Figure 1 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 1 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

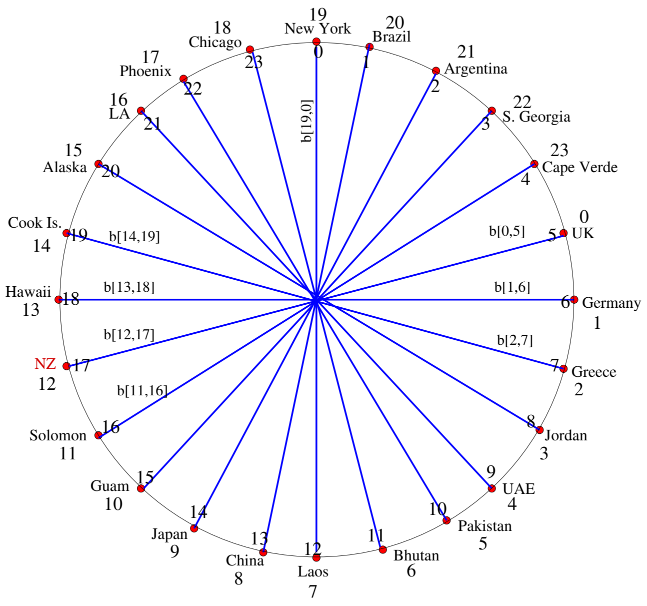


Figure 2: Clock net with cycle time T=24. The capacity in zone i at local time j is b[i,j].

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 $0 \leq t < T$ is defined by $l(i,t) = j \implies l(i, t+1) = j+1$.
The relative positioning of time zones is defined by $l(i,t) = j \implies l(i+1, t) = j+1$.

In the first clock of Figure 1, it is 1:00 AM in Brazil, so $l(20, t) = 1, z(1, t) = 20$. In the second clock, 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 $\theta > 0$,

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

in mod T arithmetic.

4 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]; $i = 0, 1, \cdots, T-1$.

If the storage capacity of a node is dependent on time, then array d[i] would be represented by a 2-dimensional array d[i,j] (similar to b[i,j]). For now, it is assumed that storage capacity of a node is fixed and plentiful (*i.e.,* storage capacity is not the bottleneck in internet transmissions); this is a reasonable assumption given the ample disk capacity in data centers. 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 model routing in networks 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 2 shows a clock net overlaid on the global clock of Figure 1. 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 an end node at t cannot exceed the bandwidth capacity of the node at t. This gives,

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

Note that the above constraint on total flow at t only applies to end nodes in the end-to-end flow, and does not apply to the intermediate nodes that lie along the path of the flow. This models the inter-connectivity of the internet - several (concurrent) links between two end networks - so each end-to-end flow at t between nodes i and j can be routed along different links. The only invariant nodes in the end-to-end flow are the end nodes i and j, so the total flow at time t for nodes i, j cannot exceed their bandwidth capacity at time t. If this assumption is not valid for the application being modeled, then the definition should add similar constraints on total flow at t for all nodes along the end-to-end flow path.

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.

The clock net is a better tool for dynamic flow routing than the time-varying flow network. For example, flow units between New York and Germany should be routed either eastward or westward, but a flow network routing algorithm checks out all paths, including paths going to Chicago, Texas, and then zigging back to Florida (or New York), London, Germany. The selection between an eastward or westward path is determined by time. For example, flow from New York to Germany when start time is 5:00 AM EST, should be transmitted westward with possible storage hops in western US, Asia, and Europe rather than eastward since time zones in UK would be entering the peak bandwidth usage times. A flow network routing algorithm would have to evaluate all routing paths since the sleep-wake bandwidth availability distribution is not captured by flow networks.

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

Definition 3 *Global dynamic flow 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

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

The notation $f(\lambda, T)$ refers to 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.

Dynamic flow algorithms address the following optimization problems:

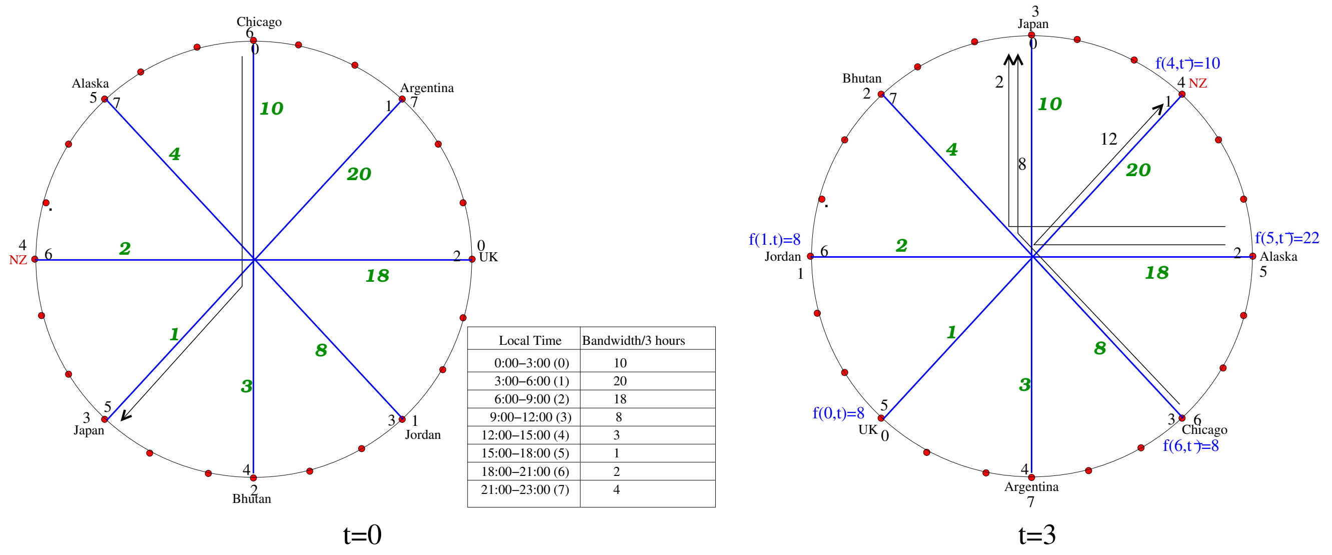
1. maximize the flow units that can be transmitted from sender to receiver within cycle period; and

2. minimize the time to transmit a given set of flow units from sender to receiver.

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

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 $t < \tau$.*



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 3: Clock net for Chicago to Japan transmission

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. An objective function of particular interest while transmitting big data sets is the following: compute all the above objective functions with the additional constraint that all transmissions are constrained to be along nodes in the sleep phase of their bandwidth availability. That is, all transmissions are constrained to low traffic times when critical applications are not using the network. In the next section, we develop a maximal flow algorithm where all transmissions are constrained to nodes in the sleep phase.

5 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 3 with the constraint that transmissions are restricted to nodes 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 3 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 1. 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 3 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 band-

Table 1: 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

width 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:

$$e2e[i] = \sum_{\forall 0 \leq \theta < T/2} \text{MIN}\{b(r,\theta), b(i,\theta + x)\}$$

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.

6 Conclusions

The paper proposes clock net as an alternative to flow networks for modeling time-dependent dynamic flow. This paper does not answer the question of whether a clock net is as powerful as a flow network for modeling time-varying dynamic flow. In order to do so, one would have to prove the equivalence between a clock net and a flow network. Is it possible to map every time-varying flow network to a clock net and vice versa? As future work, we plan to evaluate the mapping between a clock net and a flow network.

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.

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.

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. It should be possible to improve on the algorithm presented here. The paper has not evaluated the algorithm for max flow or universal max flow. Intuitively, the algorithm generates universal max flow, but a formal proof is required.

References

[1] Parminder Chhabra, Vijay Erramilli, Nikolaos Laoutaris, Ravi Sundaram, and Pablo Rodriguez. Algorithms for constrained bulk-transfer of delay-tolerant data. In *ICC'10*, pages 1–5, 2010.

[2] D. Gale. Transient flows in networks. *Michigan Mathematical Journal*, 6:59–63, 1959.

[3] Sushant Jain, Kevin Fall, and Rabin Patra. Routing in a delay tolerant network. *SIGCOMM Comput. Commun. Rev.*, 34(4):145–158, August 2004.

[4] L.R. Ford Jr. and D.R. Fulkerson. Constructing maximal dynamic flows from static flows. *Operations Research*, 6:419–433, 1958.

[5] Anukool Lakhina, Konstantina Papagiannaki, Mark Crovella, Christophe Diot, Eric D. Kolaczyk, and Nina Taft. Structural analysis of network traffic flows. In *SIGMETRICS'04*, pages 61–72, 2004.

[6] Nikolaos Laoutaris, Michael Sirivianos, Xiaoyuan Yang, and Pablo Rodriguez. Inter-datacenter bulk transfers with netstitcher. In *Proceedings of the ACM SIGCOMM 2011*.

[7] Nikolaos Laoutaris, Georgios Smaragdakis, Pablo Rodriguez, and Ravi Sundaram. Delay tolerant bulk data transfers on the internet. In *SIGMETRICS*, pages 229–238, 2009.

[8] A. Orda and R. Rom. On continuous network flows. *Operations Research Letters*, 17:27–36, 1995.

[9] W.L. Wilkinson. An algorithm for universal maximal dynamic flows in a network. *Operations Research*, 19:1602–1612, 1971.

[10] C. K. Wong Xiaoqiang Cai, Dan Sha. *Time-Varying Network Optimization (International Series in Operations Research & Management Science)*. Springer, 2007.

[11] G. Xu, S. Sun, and J.B. Rosen. Fast data transmission and maximal dynamic flow. *Information Processing Letters*, 66:127–132, 1998.

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)++;
```