A BOUND FOR UNIVERSAL ROUTING ON TREES

REAL

Abstract. Universal routing is a model of ad-hoc distributed routing where routing relies on two objects: A universal routing function that is independent of the network structure, and an information function that corresponds to the network structure. We show that for universal routing in a tree shaped network of n participants, every participant has to remember at least $\frac{n-2}{n}\log n$ bits of information. We derive another lower bound for stronger routing requirements on connected graphs.

1. Intro

Consider n computers on a network with some formation. Some computers have a direct connection (A cable) between them. We want to be able to pass messages between any two computers on the network. Assuming that a message originates from some computer, at each stage the current computer passes the message to one of his neighbours. Eventually the message should arrive at its destination computer. We also assume that every computer on the network has only limited amount of memory, and hence can not comprehend the full structure of the network.

We can think of the network structure as a simple graph, where vertices represent computers in the network. Two vertices in the graph have an edge between them if the two corresponding computers are directly connected. We consider a specific model for routing in graphs called **Universal Routing**. In this model we will reason about the amount of memory needed for every computer in the network in order to allow routing.

The Universal Routing model assumes the existence of one universal routing function that is used to route messages in any network structure of n computers. For every network structure, every computer is "magically" given a small piece of information. Using this small piece of information and the knowledge of the universal routing function, every computer knows how to pass a message to the next neighboring computer so that the message will eventually arrive at its destination.

2. Universal Routing Schemes

Given that the network contains n computers, we assign every computer a name $i \in 1, \ldots, n$. We denote that information every computer can remember about network routing as a symbol out of the set $\{1, \ldots, k\}$. In other words, we assume that every computer in the network has $\log(k)$ bits of memory available for routing. More formally:

Definition 2.1. An (n, k) **information function** is a function $h : [n] \to [k]$

Date: 11.12.2016.

2010 *Mathematics Subject Classification.* Primary 05C30.

Freedomlayer research facility.

We denote by $\mathcal{I}_{(n,k)}$ the set of information functions on a network with n computers and $\log(k)$ bits of memory per computer.

Definition 2.2. An (n,k) **routing function** is a function $r : [n] \times [k] \times [n] \to [n]$

Routing functions work together with information functions to allow routing in a network of computers. We formalise this idea with the following definition:

Definition 2.3. Given a (simple) graph G over the set of vertices $[n]$, an (n,k) routing information function h and (n,k) routing function r, we say that r, h are a **routing solution** for G if for every two distinct vertices $v, u \in G$, The series: $a_0 = v$, $a_{n+1} = r(a_n, h(a_n), u)$ satisfies:

(1) $\{a_n\}_n$ converges to u. In other words, for some $T \geq 0$, $a_i = u$ for all $i \geq T$.
(2) For every $n \geq 0$, if $a_n \neq a_{n+1}$ then $\{a_n, a_{n-1}\}$ is an edge in the graph G.

It could be useful to find some routing function r that can route in many graphs at the same time, given the right information function h. Having such a function allows us to tell about the function r to all the computers in the network ahead of time. Then for any change of the network topology, we will just have to change the information function for all the participating computers.

Definition 2.4. A (n,k) **Universal Routing Scheme** over a subset $\mathcal{U}$ of graphs over n vertices consist of:

(1) A universal routing function $r : [n] \times [k] \times [n] \to [n]$
(2) A function $m : \mathcal{U} \to \mathcal{I}_{(n,k)}$ such that r and $m(G)$ route G for every $G \in \mathcal{U}$.

In the above definition, the function m is some magic oracle that generates an information function out of the network topology. We assume that there is some distributed algorithm that allows all the computers in the network to compute $m(G)$ together, such that in the end of the algorithm execution every node v ends up with his piece of information $(m(G))(v)$ that can be used for routing.

This kind of algorithm might not always exist, but we assume here that it does, so that we will be able to bound the amount of information $\log(k)$ that every computer must keep in order to allow routing in the network.

3. Universal Routing on trees

Assume that a set of computers are connected in a tree formation: There is exactly one path between every two computers in the network (There are no cycles and the network is connected).

How much memory should every computer hold in order to allow routing of messages between any two computers in the network? We will try to answer this question for the Universal Routing model.

Theorem 3.1 (Lower bound for universal routing in trees)**.** *Let $\mathcal{T}_n$ be the set of all trees on n vertices. Let (r,m) be an (n,k) universal routing scheme over $\mathcal{T}_n$. Then $k \geq n^{\frac{n-2}{n}}$.*

Proof. Suppose that (r,m) is an (n,k) universal routing scheme over $\mathcal{T}_n$. We first show that m is a bijection.

Let $T_1, T_2 \in \mathcal{T}_n$ be two distinct trees. This means that there are some v, u such that without loss of generality, $\{v,u\} \in T_1$ and $\{v,u\} \notin T_2$. Denote $h_1 = m(T_1), h_2 = m(T_2)$. Then $r(v, h_1(v), u) = u$ because v, u is the only path from

v to u in T_1. In addition, $r(v, h_2(v), u) \neq u$, because $\{v, u\} \notin T_2$. Therefore $h_1(v) \neq h_2(v)$, which means that $h_1 \neq h_2$. Hence $m : \mathcal{T}_n \to \mathcal{I}_{(n,k)}$ is a bijection.

Therefore $\left|\mathcal{I}_{(n,k)}\right| \geq |\mathcal{T}_n|$. By Cayley's formula for the amount of trees on n labeled vertices, $|\mathcal{T}_n| = n^{n-2}$.

Hence $\left|\mathcal{I}_{(n,k)}\right| = k^n \geq |\mathcal{T}_n| = n^{n-2}$. So $k^n \geq n^{n-2}$, and $k \geq n^{\frac{n-2}{n}}$.

□

This means that approximately at least $\log(n)$ bits of memory are required for each computer in the network to allow universal routing for any tree graph.

4. Optimal Universal Routing on connected graphs

We increase our requirements from the routing scheme, and then try to find out what are the new corresponding requirement on the amount of routing memory for every computer in the network.

Definition 4.1. An (n, k) universal routing scheme (r, m) over a subset $\mathcal{U}$ of graphs on n vertices is called **optimal universal routing scheme** if for every $G \in \mathcal{U}$, for every two distinct vertices $u, v \in G$ the series $a_0 = v$, $a_{n+1} = r(a_n, (m(G))(a_n), u)$ is a shortest path between u and v on the graph G.

Theorem 4.2 (Lower bound for optimal universal routing in connected graphs)**.** *Let $\mathcal{C}_n$ be the set of all (simple) connected graphs on n vertices. Let (r, m) be an (n, k) optimal universal routing scheme over $\mathcal{T}_n$. Then $k \geq \sqrt[n]{c(n)}$, where $c(n)$ is the amount of connected graphs on n labeled vertices.*

Proof. Suppose that (r, m) is an (n, k) optimal universal routing scheme over $\mathcal{C}_n$. We first show that m is a bijection.

Let $C_1, C_2 \in \mathcal{C}_n$ be two distinct connected graphs. This means that there are some v, u such that without loss of generality, $\{v, u\} \in C_1$ and $\{v, u\} \notin C_2$. Denote $h_1 = m(C_1), h_2 = m(C_2)$. Then $r(v, h_1(v), u) = u$ because v, u is a shortest path from v to u in C_1. In addition, $r(v, h_2(v), u) \neq u$, because $\{v, u\} \notin C_2$. Therefore $h_1(v) \neq h_2(v)$, which means that $h_1 \neq h_2$. Hence $m : \mathcal{C}_n \to \mathcal{I}_{(n,k)}$ is a bijection.

Therefore $\left|\mathcal{I}_{(n,k)}\right| \geq |\mathcal{T}_n|$. Denote the number of connected graphs on n labeled vertices to be $c(n)$.[1]

Hence $\left|\mathcal{I}_{(n,k)}\right| = k^n \geq |\mathcal{T}_n| = c(n)$. So $k \geq \sqrt[n]{c(n)}$

□

References

1. F.R.K. Chung, R.L. Graham, D. Coppersmith *On Trees Containing All Small Trees* `http://www.math.ucsd.edu/~ronspubs/81_04_small_trees.pdf`

E-mail address, real: `real@freedomlayer.org`

[1] A closed formula for $c(n)$ is not known at this time. See https://oeis.org/A001187