

Network Coordinates for Oblivious Clients with Saccades

Kevin P. Shanahan
Michael J. Freedman
New York University

Introduction

Internet Systems rely on replicated content

- Performance dependent upon server selection
- Optimal server selection often based on round trip time (RTT) to a client

Current Systems

Commercial CDNs build network maps to redirect clients

- Akamai, ...

Requires:

- Extensive network knowledge
- Expensive aggregation
- Large centralized infrastructure
- System distribution control

Prospective Systems

Emerging peer-to-peer applications:

- Decentralized static / dynamic CDNs
 - ◆ CoralCDN, OpenEdge, ...
- Distributed hash storage systems
 - ◆ OpenDHT, ...
- New Internet naming systems
 - ◆ SFR, ...

Inherent Limitations:

- Decentralized infrastructure
- Cannot produce / maintain accurate network maps
- Uncontrolled node deployment

Proposal

Decentralized, accurate mapping system

- Accurately predicts locality within a system
- Suitable for current and emerging P2P systems
 - ◆ Light-weight
 - ◆ Scalable
- Functional for unmodified clients
 - ◆ Must run on top of existing protocols (DNS, HTTP, ...)

Underlying Schema

Seeks to efficiently determine locality information

- Every node continuously ping every other node?
 - ◆ Severe network congestion
 - ◆ Not scalable
- Every node ping every client?
 - ◆ Leads to client failure, abuse complaints, etc.

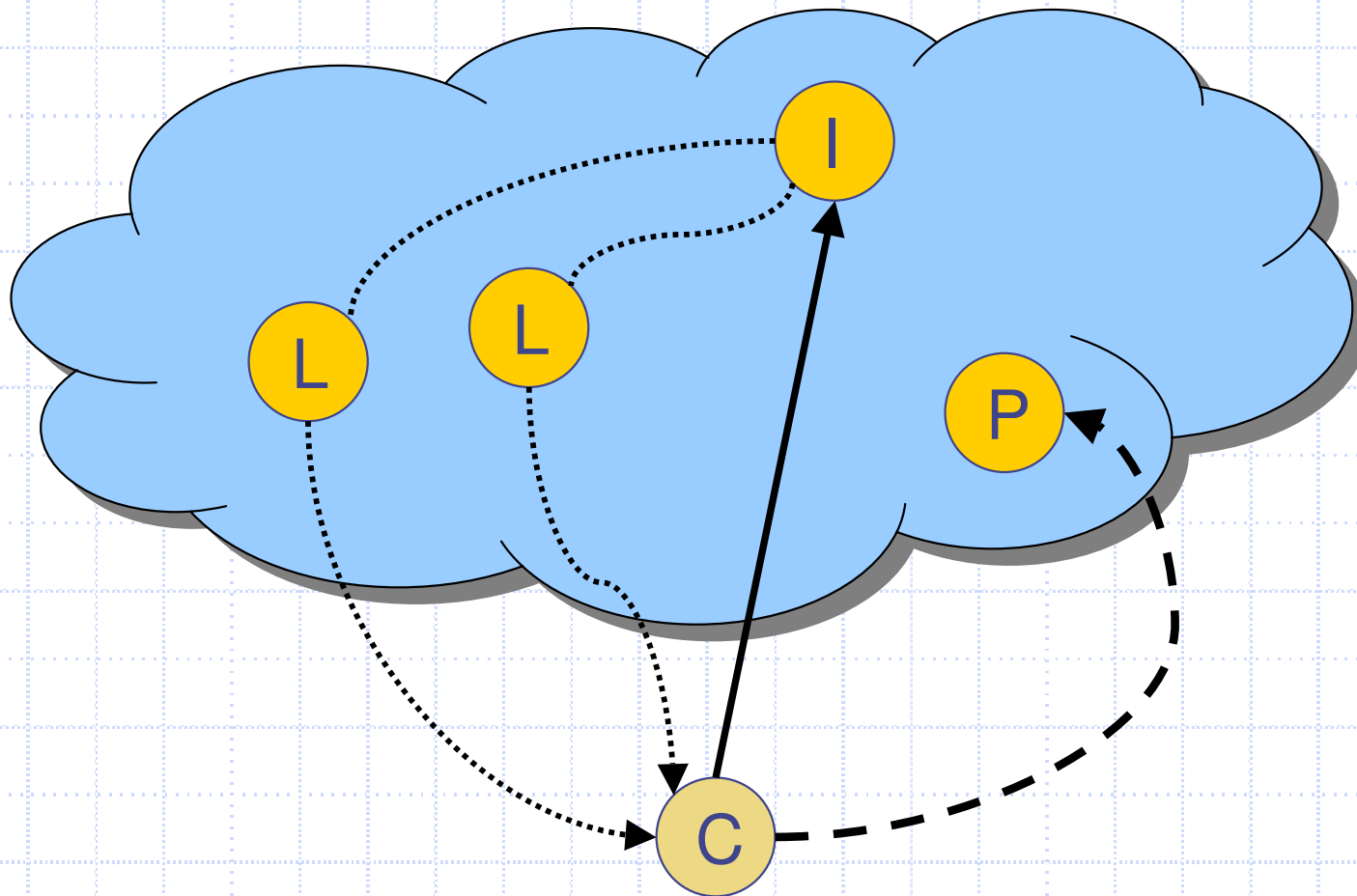
Utilize Network Coordinates

- Coordinate space defines a distance metric
 - ◆ Distance in the coordinate space predicts RTT
- Several proposed systems (Vivaldi, GNP, PIC, ...)

Methodology: Overview

1. Unmodified clients connect into the system:
 - Clients are unable to run system code
 - Clients are unable to calculate their own coordinates
2. System predicts coordinates of client
3. Client is then redirected internally based on predicted coordinates

Client Coordinate Assignment



Methodology: Saccades

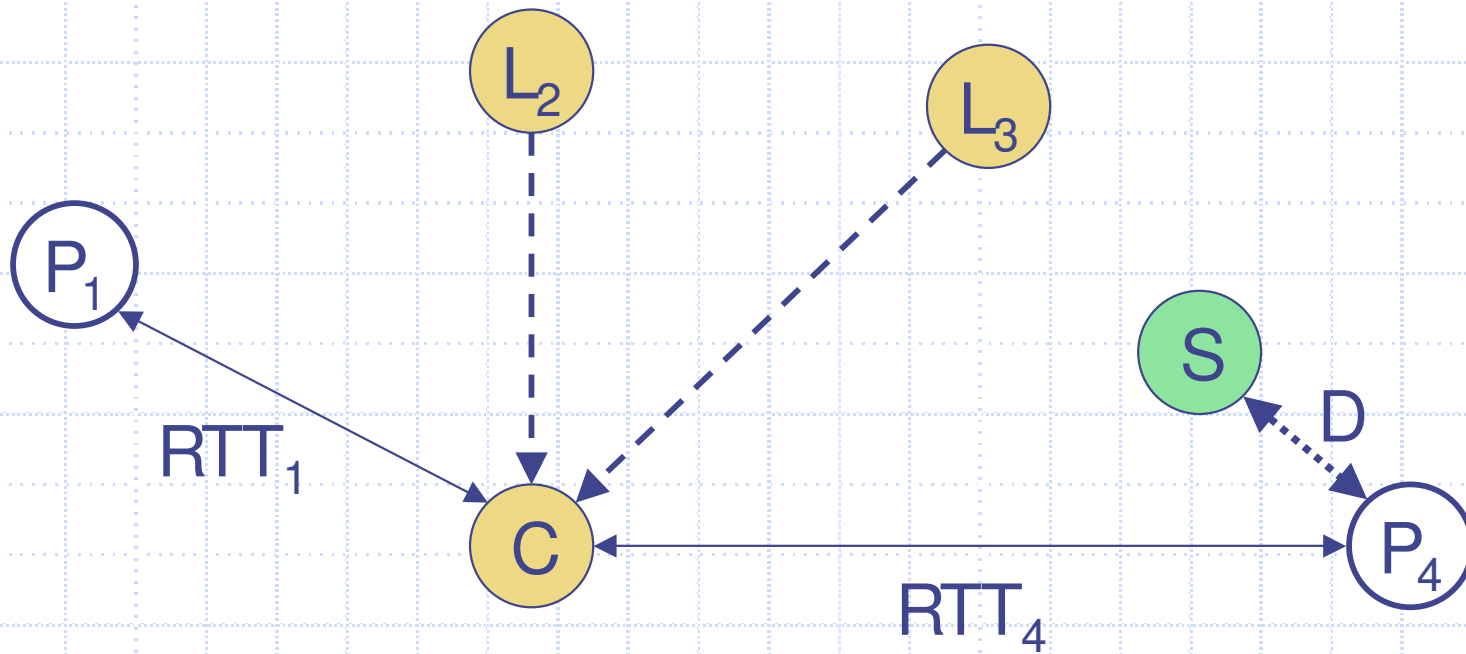
Saccades is a lightweight, scalable infrastructure

- Uses Vivaldi Network Coordinate System
 - ◆ Decentralized
 - ◆ Maintains coordinates by piggybacking atop normal msgs
- Returns nearby internal nodes
 - ◆ Minimum of system probing

Coordinate Prediction

- ◆ Unknown client contacts *ingress* node
- ◆ *Ingress* node contacts k internal system *landmarks*
 - In some systems not all nodes are considered viable landmarks
- ◆ Each selected *landmark* probes the client and measures RTT
 - Multiple probes can reduce error of transient network conditions
- ◆ Each system node returns RTT(s) with its current coordinates to *ingress* node
- ◆ *Ingress* node simulates network coordinate algorithm on data set to compute most accurate client coordinates (triangulation)
- ◆ Client is redirected to the internal system node with coordinates closest to the predicted client coordinates

Accuracy Metric



$$\text{peer assignment accuracy} = RTT_4 - RTT_1$$

Experimental

Tests were run on the PlanetLab test-bed

- ~ 140 nodes running Vivaldi
- Control node requested ~ 400 pings
 - ◆ All pings were to PlanetLab nodes
 - ◆ Each node pings 3 times
 - ◆ 25ms ping interval
- 25ms interval between probe initiation

Accuracy Results

