Per User Profile Replication in Mobile Environments

Shivakumar, Jannink and Widom

Stanford University

Jim Miani

CIS 642

Outline

- Paper Abstract
- Introduction
- Objectives
- Max-Flow Min-cost Algorithm
- Empirical Analysis
- Conclusions and Future Work
- Criticism/Discussion

Abstract

Problem: To locate mobile users in a Personal Communication Service system.

Proposal: Use a minimum-cost maximum flow algorithm to compute the set of sites at which a user profile should be replicated.

Introduction

- Personal Communications Service (PCS)
- Cells - bounded geographical areas
- User placing call contacts *base station* in same cell via wireless medium.
- Base station then contacts receiving PCS device.
- User may cross cell boundaries with PCS device.

Introduction continued …

- *Location lookup problem:* to locate users moving from cell to cell within reasonable time constraints
- *Zones* are cells or group of cells
- Each zone contains a database of user profiles
- Profile structure: <PID, ZID>
- PID - unique ID for PCS device
- ZID - zone ID for PID's current zone

More intro ...

- Each PID maps to a *home zone* (a.k.a. *Home Location Register*) which always maintains up-to-date copy of user profile
- Suppose user A calls user B. In *pure HLR* scheme, the algorithm to locate B must perform a (possibly) remote lookup to HLR of B.
- To defray this cost, maintain *VLRs.* This is useful when a user tends to receive calls from the currently occupied zone.
- VLRs are a simple, limited *replication scheme*

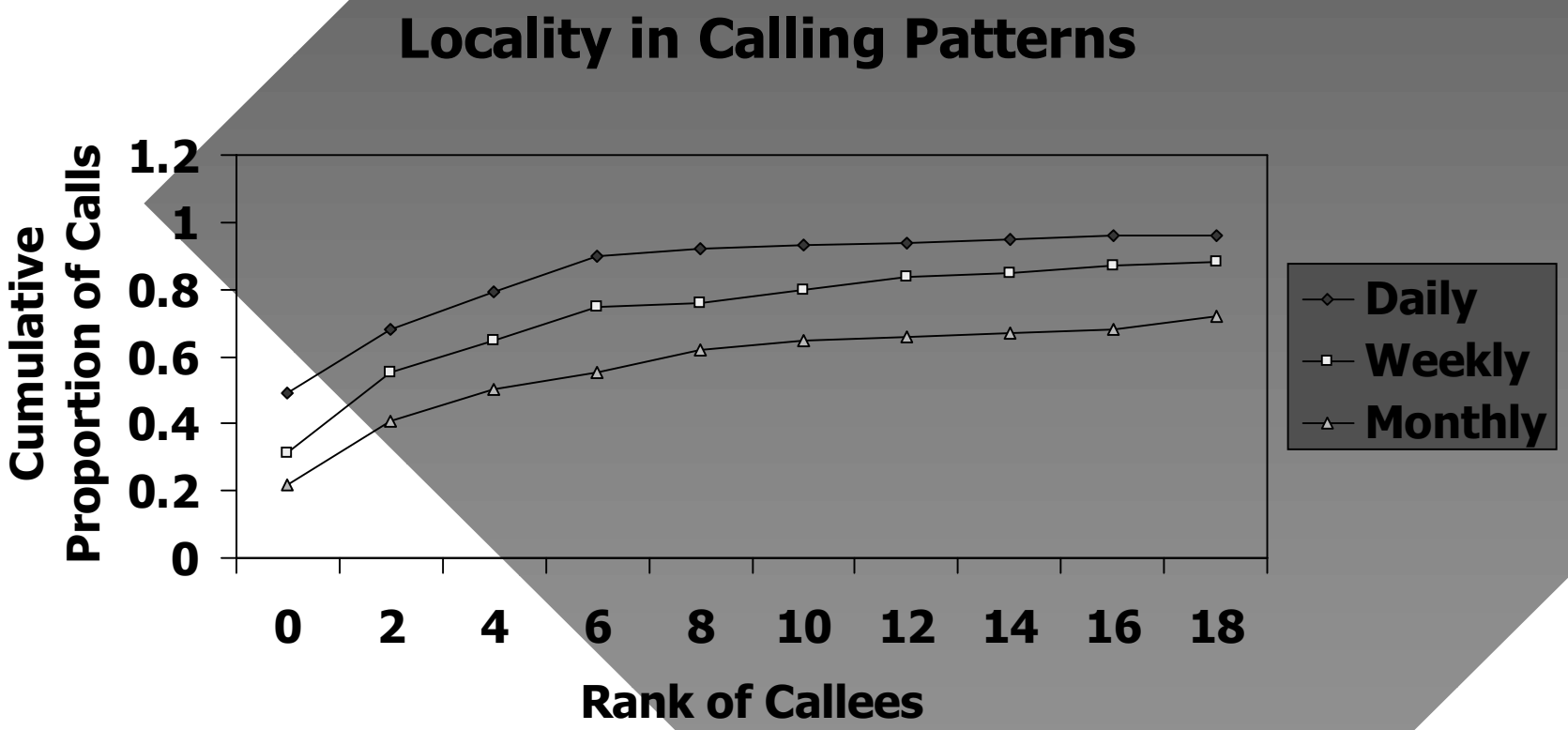
Last bit of intro …

- Ideally, replicate user profiles in all zones.
- Naturally the ideal scenario won't work.
- Basic algorithm works essentially like VLR:
    1. Query database in caller's zone.
    2. If callee's profile not found, search callee's HLR.
- Assume the home location keeps track of sites possessing copies of a user's profile. When zones are crossed, replication is initiated.
- Algorithm for computing additional sites at which the profile is replicated is max-flow min-cut.

Objectives

1. Select the best zones for replication user profiles based on calling and mobility patterns.
2. Adapt to changes in access patterns.
3. Choose sites for replication based on cost-benefit analysis.

Supporting Data



Note that more than 70% of the calls in a week are made to the top 5 callees.

Issues In Replication

- Cost of maintaining consistent replicas across distributed databases every time a user moves.
- Use "loose" definition of replica consistency and faster method of replication.
- Store temporary forwarding pointers at old location to handle calls from uninformed calling zones.
- HLR must initiate updates every time user moves, and network must carry this traffic.
- Limit number of zones at which profile is replicated.
- Limit total number of replicas stored in a given zone to ensure fast lookup and updates.

Algorithm Parameters

- M is the number of zones; Z_j is the jth zone where j = 1,2,…,M.
- p_j is the maximum number of profiles serviceable by database of zone Z_j.
- N is the number of PCS users and P_i is the ith PCS user for i = 1,2, … N.
- $C_{i,j}$ is the expected number of calls from zone Z_j to user P_i over a set time period T.
- U_i is the number of moves made by P_i over T.

Algorithm Parameters continued

- r_i is the maximum number of sites at which P_i's profile can be replicated.
- α is the savings achieved when a local lookup succeeds rather than a remote lookup.
- β is the cost of updating a profile replica.

A replication is considered judicious if the cost savings due to replication exceeds the cost incurred.

Thus, it is judicious to replicate P_i at Z_j if $\alpha * C_{i,j} \geq \beta * U_i$

- $R(P_i)$ is the *replication set* of user P_i, the zones at which P_i's profile is replicated according to specified algorithm

Computing the Replication Plan

Construct a flow network F = (V,E) where V and E are vertices and edges in the network, respectively. Each edge has two associated attributes: (cost, capacity).

1. V <- Ø, E <- Ø

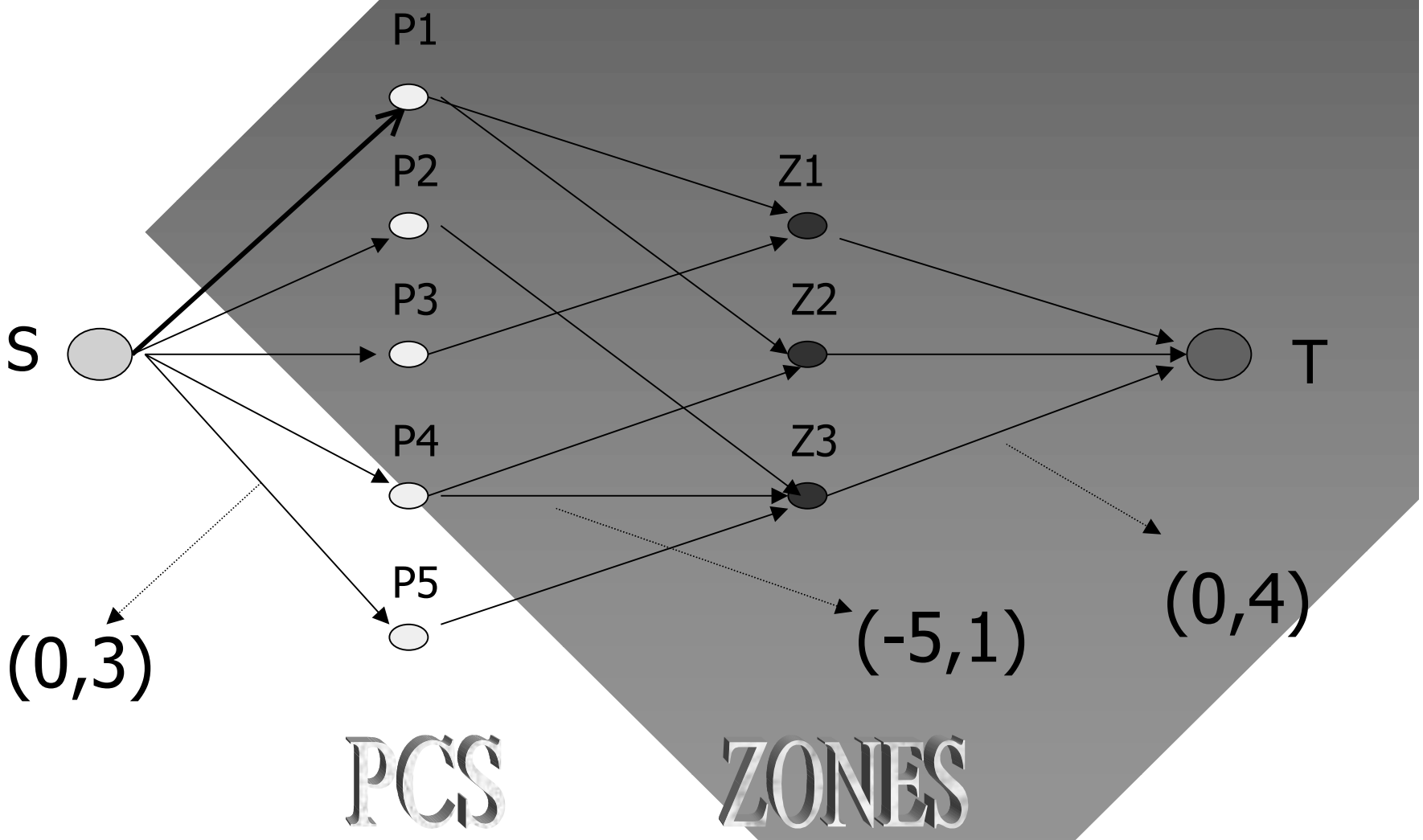
2. Add source s and sink t to V.

3. Add all P_i and Z_j to V for i = 1,2, … N and j = 1,2, … M.

4. Add to E directed edges from s to all P_i with (cost, capacity) = (0, r_i) and from all Z_j to t with (cost, capacity) = (0, p_j).

5. For every < P_i, Z_j > pair, if $\alpha * C_{i,j} \geq \beta * U_i$, then add an edge from P_i to Z_j with cost, capacity = ($\beta * U_i - \alpha * C_{i,j}$, 1)

Sample Flow Network



Computing Min-Cost Max-Flow

Objective: To find an assignment of profiles to databases such that the number of useful replicas is maximized and the system cost is minimized.

Think of an edge (u,v) with capacity k as k *virtual* edges each of capacity one. An *edge reversal* means that one of the virtual edges from u to v is reversed so it is now (v, u).

Recall that an *augmenting path* is a directed path along virtual edges from source to sink.

Computing Min-Cost Max-Flow

Algorithm 1:

Repeat the following until no more augmenting paths can be found:

1. Find the least-cost augmenting path from source s to sink t.

2. Edge-reverse each *virtual* edge in this path.

When complete, P_i's profile is replicated at Z_j if there is a directed virtual edge from Z_j to P_i. Formally, $R(P_i) = \{Z_j \mid (Z_j, P_i) \in E\}$

This Algorithm Guarantees:

1. **The number of replicated profiles at zones does not exceed maximum serviceable capacity of their databases.**

2. **The profile of a user is not replicated at more than the specified maximum number of replication sites.**

3. **The system savings is maximized.**

Cumulative Cost of Replication Plan:

$$\sum_{I=1}^{\mathbb{N}} \sum_{j=1, Zj \in R(Pi)}^{\mathbb{M}} \beta * U_i - \alpha * C_{i,j}$$

Computing Algorithm Parameters

- How can we determine what the (cost, capacity) pair is for a given PCS and zone?

Let $\sigma_{i,k}$ = E(calls from P_i to P_k)

Let $\lambda_{i,j}$ be the expected amount of time P_i spends in zone Z_j (locational distribution).

Assume $\sigma_{i,k}$ and $\lambda_{i,j}$ are independent. Estimate number of calls for P_i from zone Z_j to be: $C_{i,j} = \sum_{k=1}^{N} \sigma_{k,i} * \lambda_{k,j}$

This is the *LCMR*, local call to mobility ratio

Dynamically Altering the Replication Strategy

- The prior algorithm is guaranteed to determine the best replication plan *given fixed calling and mobility patterns.*
- How can we incrementally adjust the replication plan while avoiding wholesale re-computation of the replication plan?

Incremental Max-Flow

Incremental Max-Flow

- **Let $F(\tau_{new})$ denote the flow network for traffic pattern τ_{new} ; likewise for τ_{old}**
- **How to incrementally compute max-flow for τ_{new} given max-flow min-cost for τ_{old} ?**
- **When changing the flow network, you will either be adding or deleting edges from the network.**
- **Max-flow is easily maintained on insertions: add the new edge, find any augmenting path, and perform edge reversals as done before!**
- **Deleting an edge is somewhat more difficult:**

Deleting an edge

- **Consider 3 cases:**

  1. The deleted edge is a forward edge from P_i to Z_j.

  2. The delete edge is an edge from Z_j to P_i. Consider two sub-cases:

  a) Satisfiable "Vacant Slot": We can find an augmenting path from P_i to Z_j . Reverse the edges on the augmenting path, thus compensating for the loss of one unit of flow through the deleted edge. We maintain max-flow by pushing one unit of flow from P_i to Z_j .

  b) Unsatisfiable "Vacant Slot": Suppose we cannot find an augmenting path. If we cannot find an augmenting path, then this is already a max-flow network. To maintain correct levels of flow all the way through the network, drop a unit of flow from source to PCS and from zone to sink.

Now make it min-cost!

- Adapt the cycle-canceling algorithm to find the min-cost, given max-flow.
- Cycle canceling algorithm:

  1) Compute max-flow of network (done).

  2) Repeat until no more negative cycles are found:

  Find negative cycles through the sink and perform edge reversals of edges in the cycle.

Additional Improvements

- Factor cost of moving profiles about into computation of max-cost min-flow network.
- Method 1: Tempered Max-Cost Min-Flow
- Method 2: Evolution with Mean Cycles

Empirical Results

- Various experimental conditions:

  - 5 day period

  - Progressively degrade accuracy of LCMR prediction

  - schemes are HLR/VLR, pure HLR, another caching scheme and optimal replication

  - Compare latencies, database and network loads

Empirical Results

- Optimally, compute replication plan 2x a day (morning and evening rush hour)
- Optimal replication provides low lookup latency (converts up to 81% of remote queries in HLR and HLR/VLR to local lookups)
- Replication requires 15-25% less network bandwidth than HLR or HLR/VLR schemes

Criticism/Discussion

- Precise and thorough
- Well-structured