InfoPuzzle: Exploring Group Decision Making in Mobile Peer-to-Peer Databases

Aaron J. Elmore[‡] Sudipto Das[§*] Divyakant Agrawal[‡] Amr El Abbadi[‡]

[‡]University of California, Santa Barbara
Santa Barbara, CA, USA
{aelmore, agrawal, amr}@cs.ucsb.edu

[§]Microsoft Research
Redmond, WA, USA
sudipto.das@microsoft.com

ABSTRACT

As Internet-based services and mobile computing devices, such as smartphones and tablets, become ubiquitous, society's reliance on them to accomplish critical and time-sensitive tasks, such as information dissemination and collaborative decision making, also increases. Dependence on these media magnifies the damage caused by their disruption, whether malicious or natural. For instance, a natural disaster disrupting cellular and Internet infrastructures impedes information spread, which in turn leads to chaos, both among the victims as well as the aid providers. Decentralized and ad-hoc mechanisms for information dissemination and decision making are paramount to help restore order.

We demonstrate InfoPuzzle, a mobile peer-to-peer database that utilizes direct device communication to enable group decision making, or consensus, without reliance on centralized communication services. InfoPuzzle minimizes the system's resource consumption, to prolong the lifetime of the power constrained devices by minimizing communication overhead, computational complexity, and persistent storage size. Due to user mobility and the limited range of point-to-point communication, knowing the exact number of participants is impossible, and therefore traditional consensus or quorum protocols cannot be used. We rely of distinct counting techniques, probabilistic thresholds, and bounded time based approaches to reach agreement. In this demo, we will explore various challenges and heuristics in estimating group participation to aid users in reconciling consensus without centralized services.

1. INTRODUCTION

From Tahrir Square to Wall Street, new technologies, such as social networks and mobile computing devices, are enabling people to quickly organize in a decentralized manner. Social networks are unintentionally serving as groupware to synchronize and facilitate human interactions [4]. The phenomenon of information diffusion and influence in social networks has been the interest of recent research and modeling. Conceptually, social networks enable a user to express an idea and subsequently propagate the idea through a network of peers. Simplicity of the user interface, and the ease to diffuse information has enabled people to organize in a lightweight manner, which is essential to facilitate large scale group interaction. However, this approach is not entirely decentralized, as these tools rely on Internet services to act as a centralized coordinator for user messages within a network[1]. In the event of a natural disaster or an administration turning adversarial to halt a movement, Internet access may become limited or unavailable. Several recent events demonstrate this scenario. With the 2010 earthquake in Haiti, rescue workers relied on text messaging (SMS) to coordinate efforts, as cell phone networks strained under failure and overloading [5,6]. Planned protests at the Bay Area Rapid Transit (BART) in San Francisco, USA resulted in cellular service being cut in order to stave off protests [2]. Lastly, and perhaps most infamously, Hosni Mubarak's government shut down Internet access to Egypt in an attempt to thwart not only social network coordination, but also privacy filters, such as Tor, that bypass censorship firewalls [3].

Despite the availability of a centralized service, certain actions still require coordination of a large group, whether it be a protest of conditions or organizing humanitarian efforts. Without a single point of communication, planned actions can become disjoint and unclear, and result in a less unified effort. Within this environment, InfoPuzzle explores how an information system disseminates an idea, or proposal, amongst a network of peers in order to ascertain users' intention and determine an expected outcome. As knowledge and context of the proposal is required to determine success, a proposer should specify a **tipping point** (*quorum value*), which is the number of users that are needed to reach an agreement. Due to the lack of a central authority and mobility of the users, the number of users that will observe a given proposal is unknown. As a result, classical notions of quorums, group size estimation, or consensus in a static distributed system [7, 11] cannot be applied directly in this problem setting. In this context, we use consensus as the consent of a specified group size for a proposed value, and not the stringent definition of consensus where all non-faulty processes agree on a single value. Since a consistent view of the number of users that will observe a proposal is not known, an application-specific quorum value threshold is needed to approximate consensus. A quorum value is application specific since the number of people required to organize a meeting at a database conference is very different compared to the number required to effectively organize a city protest.

The group decision making problem framed in a disconnected environment is technically challenging. Users that are mobile are likely to have different views on the state of the ballot due to the lack of a single point of truth and observing intentions of disjoint peers. Divergent views need to be reconciled between users, so an

*Work done while at UCSB.

Proceedings of the VLDB Endowment, Vol. 5, No. 12

[1]While services are built on distributed components, access is centralized through a point of authority such as a DNS server.

approximate view can be consistently determined in order to derive a proposal's outcome. The state of the proposal should be persisted beyond volatile memory. The reconciliation of disconnected replicas, efficient persistent storage, managing data replication in a chaotic decentralized environment, and an understanding of consensus call for a database system solution to this timely problem.

The reliance on a centralized cellular or Internet access does not prevent modern mobile devices from communicating with their peers at a large scale. Many smartphones have capabilities to directly communicate with other mobile devices within a limited range, including IEEE 802.11, Bluetooth, or Ultra Wide Band. These communication media provide the ability to discover peers within a few hundred feet allowing for the construction of a mobile peer-to-peer (**P2P**) network to exchange information [11]. Leveraging these networks and motivated by the need for decentralized organizational tools, we introduce InfoPuzzle: a mobile P2P database that enables group decision making without relying on centralized services. While mobile P2P databases for disaster situations or military applications have been proposed before, specifics were not provided. Moreover, we focus on group decision making and not only communication overlays [10]. The name InfoPuzzle symbolizes our goal to create an aggregate view of information composed of smaller (puzzle) pieces of information that can be difficult to piece together and individually cannot represent the global picture.

This demonstration will highlight the need for InfoPuzzle and the challenges associated with enabling group decision making in a P2P mobile environment. A demonstration involving mobile devices will provide users an interactive experience of how this system enables proposals and how peers interact with the proposal. The demonstration will also involve a simulator to provide a visualization of proposal dissemination and acceptance. The demonstration will highlight challenges in user-driven data management for mobile environments.

2. PRELIMINARIES

InfoPuzzle is a database system running on mobile devices that communicates in a P2P fashion with other mobile devices in its vicinity. In InfoPuzzle, a user can propose a question (the **proposal**), accompanied by a suggested answer (the **value**). The proposal is broadcast to all users within the proposer's direct communication range. Since the communication range is limited and the users are mobile, a proposal might eventually be relayed to areas where the original user initiating the proposal is not present. We use the term **proposer** for the user that introduces a proposal to a set of peers who have not received the proposal earlier. A proposer can therefore be the original initiator or a relay node. The mobile agents, or devices, forward this proposal, allowing users to agree with the proposal, suggest a new value, or reject the proposal. New values are only allowed if none of the proposed valued have reached the tipping point. The users that respond non-negatively to a proposal are **subscribers** to the proposal. Every proposal has an associated **expiration time** after which it is no longer valid; a proposal is spread until its expiration time is reached. The proposal is encapsulated within a **ballot** that contains the proposal, the suggested value, the expiration time, the tipping point, and the set of users who have accepted each of the ballot's potential values. All subscribers are notified of the proposal's **outcome** either when a value reaches distinct votes above tipping point or when the proposal expired. Due to the mobile and disconnected nature of the underlying infrastructure, data might be incomplete and votes can have associated confidence levels.

Let us consider a hypothetical scenario at the upcoming VLDB Conference. Some attendees intend to plan a meeting time and location to initiate a motion to support video conference PC meetings for the next VLDB. The conference Wi-Fi is overloaded and is barely working. Since there are a large numbers of international attendees, most of the participants do not have cellular connectivity. Additionally, since the motion is against the current VLDB Executive Committee's will, the organizers have been banned posting related messages on the conference bulletin boards. Point-to-point verbal communication also does not work due to the scale of the people involved, the short duration to reach consensus, and the challenge in tracking divergent views as the information is propagated amongst the conference attendees. This is a scenario where InfoPuzzle will help organize the motion, gather votes for proposals, enable new proposals, as well as help reconcile divergent motions in a completely decentralized setting. However, it goes without saying that InfoPuzzle's real applications will be in disaster recovery scenarios or to help gather more meaningful and larger scale motions; this hypothetical scenario relates us to this demonstration setting.

```
ballot{
 proposal: "Gather to demand video conference
           PC meetings for VLDB",
 expiration: "2012-08-27 10:00:00",
 suggestedValue: "Byzantine Room 8/27 11:00",
 quorumValue: 20,
 acceptsByUserId: {
   "Byzantine Room 8/27 11:00" : [31083,
      13091, 38919, 900941, 109381],
   "Paxos Hall 8/27 14:00" : [13134]
 }
 key: 107074168843,
 proposerId: 31480
}
```

Figure 1: A sample ballot to demand conference review changes.

Figure 1 describes a sample ballot for the hypothetical proposal. The ballot is expressed in JavaScript Object Notation (JSON) format for the ease of exposition; InfoPuzzle stores and transmits a ballot in a compact compressed binary format. In this example, the proposer suggests that at least twenty participants are required to achieve the tipping point. The snapshot of Figure 1 captures a scenario where an alternative value has also been proposed. The ballot lists both the proposed values: the initial proposal of `Byzantine Room at 11 am` that has six acceptors, and a later proposal `Paxos Hall 5/24 14:00`, that has a single vote. For brevity, the error values and divergence in versions are not shown, but InfoPuzzle tracks them internally. The ballot is broadcast to all peers within communication range. Each peer, or active InfoPuzzle instance, will notify its user of a new ballot. A user can take action on the ballot by either *accepting* the ballot, *relaying* it, *proposing* a new value, or *ignoring* the ballot. The proposer for this ballot is notified of this user's intention. An *accept* signifies that the peer agrees with the proposal, and will relay the proposal to all future peers. A *relay* signifies that the peer will not commit to the proposed value but will subscribe to the ballot and rebroadcast to future peers. If a peer disagrees with the suggested value of the ballot, an alternate value can be proposed. *Ignore* states that the user rejects the ballot, and that this peer should ignore all future messages regarding this ballot. The proposer adds all accepted users and proposed values to the ballot, makes the ballot locally persistent in InfoPuzzle, and rebroadcasts the updated ballot to all users in range. On receipt of a message, a peer determines if the ballot contains new information or should be ignored. Additionally, a peer does not rebroadcast a

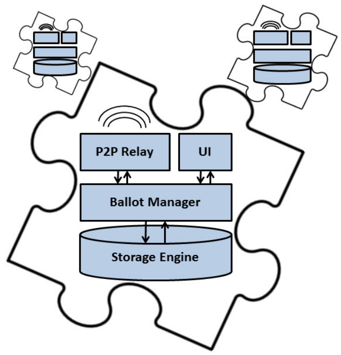

Figure 2: Internal InfoPuzzle components.

ballot with an identical state (i.e. no new acceptors or values) to peers already aware of the ballot.

3. SYSTEM DESIGN

Each InfoPuzzle instance is composed of four major components shown in Figure 2. The *storage engine* provides persistence for the observed ballots and the actions taken by the user. The *ballot manager* coordinates all components, decides what information to relay between components, ignore and broadcast ballot information to new peers, and inform the user about the status and outcome of a ballot. The *P2P relay* notifies the ballot manager of new peers observed, incoming ballots, and ballot proposal responses. The relay also manages proposal broadcasts and responses from the ballot manager. The *UI* module interfaces the ballot manager with the user to acquire responses about proposals and notify with updates on the subscribed ballots.

Several challenges arise in designing an efficient ballot manager. Limited battery life, network bandwidth, and storage capacities require minimizing InfoPuzzle's footprint and resource consumption. Each InfoPuzzle instance stores detailed information about a ballot so that the ballot manager can appropriately orchestrate coordination between the components. However, since network communication is expensive, peers should exchange a compressed ballot header to determine if additional information is needed to synchronize the ballot views of two nodes.

Ballot transmission frequency must also be optimized. While a great deal of literature exists on gossip protocols and the properties of epidemic communication in mobile environments [11], InfoPuzzle can leverage additional context in communication protocols. This context can include expiration time, agents' mobility patterns, and popularity of a proposal. Mobility patterns will be especially important when a small subset of the peers are mobile, while the majority remain static and are less likely to interact with new peers.

Due to the decentralized nature of InfoPuzzle, timestamps alone cannot accurately determine if peers have a consistent view of a ballot. As users move and are disconnected, their versions of the ballot might diverge. As a result, InfoPuzzle must address important research challenges in the reconciliation of divergent ballots, and their respective counts, eliminating duplicates from the counts, as well as tracking the lineage of the ballot versions as they diverge. For instance, a user U_j receives the ballot from another user U_i. After accepting the proposal, U_j moves to the vicinity of U_k and passes the ballot. Now, if U_k moves into the vicinity of U_i, U_i must be able to eliminate the duplicate counts through the lineage of divergent versions. Storing and exchanging the full accepted user set to track lineage is expensive in terms of bandwidth and computation. A naïve optimization involves comparing a hash of an ordered set of users (such as bloom filters); however, this approach incurs a high cost to merge large sets and is also an approximate set membership. Additionally, storing and exchanging large sorted sets can be too costly for storage and computation on a device where resources are finite.

We are exploring combinations of probabilistic approaches to minimize communication and computation costs when merging sets for reconciliation and determining if a proposal has been accepted. When deciding to reconcile nodes, a counting Bloom filter can approximate the number of differences between sets [1]. If the relative set difference is small compared to the cost of merging sets, an accurate merge may be considered too costly. Alternatively, upon discovering a large set of new neighbors, InfoPuzzle can broadcast a set of accepted votes, and leverage distinct counting techniques, such as sampling or sketch based techniques, to approximate unique votes [8]. If the quorum value or accepted value set is large, a hybrid approach may be required for determining outcomes. Wu et al. propose a combination of linked counting Bloom filters and random gossiping to find global icebergs (items with counts above a threshold) in a distributed setting [12]. Approaches for reconciliation and outcome will be evaluated on accuracy, storage requirements, communication overhead, and computation costs. The user demo will highlight these challenges.

Most importantly, InfoPuzzle should also be able to reason about the confidence of consensus beyond ballots reaching a designated threshold of votes. However, due to the uncertain nature of the network topology and communication, true consensus cannot be achieved due to the impossibility of agreement and validity. Often, traditional distributed systems notions, like consistency, assume a static number of nodes. Many of these assumptions were reexamined with the rise in popularity of dynamic systems, where the number of nodes varies over time. Research in *Group Size Estimation* and dynamic system modeling will guide efforts in building robust models of consensus for a mobile environment [7, 9]. Since the mobile agents differ from dynamic systems by having spatial and temporal patterns, InfoPuzzle considers empirical observations about the mobility of a user and the churn and mobility of peers for estimating outcome confidence. Finally, a feedback mechanism on the outcome is requested from the proposer, for reinforcing models that accurately stated which proposed value achieved consensus.

In addition to the above mentioned challenges, InfoPuzzle must also address multiple issues, such as privacy of the users, potentially intermittent centralized sources of truth, malicious behavior, and trustworthiness. Detailed analysis and discussion of all these challenges and their solutions are beyond the scope of this demonstration proposal and will appear as a future research paper.

4. DEMONSTRATION DETAILS

The InfoPuzzle demonstration will consist of three parts. First, a small slide show to motivate the need for InfoPuzzle by highlighting recent grassroots mass organizations and how these group decisions can be disrupted by a lack of a centralized Internet service. Challenges in achieving consensus and reconciliation in a heavily partitioned environment will be emphasized. The slides will include an explanation of how InfoPuzzle's protocol works, system architecture, and the optimizations being undertaken for both system performance and modeling consensus.

The second—and most important—component of the demonstration will include InfoPuzzle running on several smartphones to provide a user experience of how the system will enable users to coordinate easily at a large scale. InfoPuzzle will be demonstrated as a web application accessible through a small Wi-Fi hotspot. Even

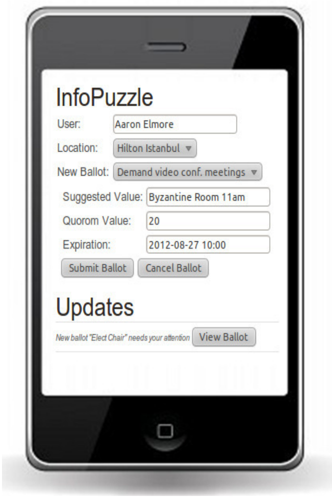

Figure 3: New ballot screen.

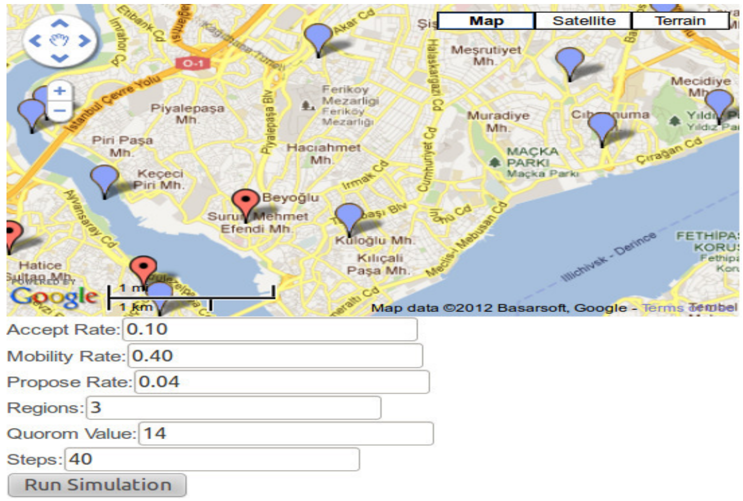

Figure 4: A screenshot of the simulator.

though the InfoPuzzle system is designed for decentralized operation, we choose a centralized infrastructure for the demonstration for pragmatic reasons. A web-based interface allows us to scale-up the system by augmenting participation from the demo audience by using their smartphones as InfoPuzzle nodes. A web-based application deployed on a local wireless router, obviates the need to install a mobile applications or reliance on conference network access.

Users will interact with InfoPuzzle through any web browser. To simulate local broadcasts and mobility, users can specify a 'current' virtual geographic area from defined options with a drop-down selection. Requests will be broadcast to users currently in the same simulated geographic area. When a user moves (simulated by updating the active selected location), new peers will be discovered and the ballot propagated and reconciled. Users will be able to propose ballots, participate in ballots, and observe notifications and outcomes of ballots. A server console for InfoPuzzle will enable the injection of virtual users into geographic locations. The virtual users will have configurable probabilities to change locations, or to accept or reject a ballot proposed by a demo user. Having the optional virtual users, will increase demo interaction for small groups, but we anticipate the demo to be more interesting with a higher number of concurrent users. Figure 3 shows a sample screenshot.

In addition to user interaction with the mobile application, we will display an InfoPuzzle application log at the server that explains the interactions and reconciliations resulting from the mobile interactions. For example, if a user switches virtual locations, the log will explain the peer discovery and the process peers undertake to reconcile their ballots.

The third component of the demo, a ballot simulator, will provide a visualization on how a proposal may spread and become accepted between locations. The simulator will be a separate web-based application, built on a spatial database, with a fixed number of virtual regions set around the conference grounds. The simulation will allow for adjustment of various parameters including expiration time, agent mobility, acceptance likelihood, new proposal likelihood, and quorum values. The simulator will run in a series of steps that updates the map to show simulated peers as they accept, reject, propose new values, or spread proposals to new areas. Figure 4 provides a screenshot of the simulator. The simulator will then demonstrate if a consensus was achieved and aggregate views of user's views on the consensus.

This demonstration will interactively highlight the research challenges in providing a group decision engine based on a mobile peer-to-peer infrastructure, the optimizations for the database components, the intuitive nature of the application, and the timely need for InfoPuzzle.

Acknowledgments

This work is partly funded by NSF grants IIS 1018637, CNS 1053594, and NEC Labs America.

5. REFERENCES

[1] S. Agarwal and A. Trachtenberg. Approximating the number of differences between remote sets. In *IEEE Information Theory Workshop*, pages 217–221, 2006.

[2] Washington Post: BART San Francisco cut cell services to avert protest. `http://wapo.st/tMomtM`, 2011.

[3] The Telegraph : How Egypt shut down the internet. `http://tgr.ph/vidllq`, 2011.

[4] C. A. Ellis, S. J. Gibbs, and G. L. Rein. Groupware: Some issues and experiences. *Commun. ACM*, 34(1):39–58, 1991.

[5] IEEE Spectrum : Why Haiti's Cellphone Networks Failed. `http://bit.ly/tnud7v`, 2010.

[6] Reuters : Cell phones and radios help save lives after Haiti earthquake. `http://reut.rs/vU1gMh`, 2010.

[7] D. Kostoulas, D. Psaltoulis, I. Gupta, K. P. Birman, and A. J. Demers. Active and passive techniques for group size estimation in large-scale and dynamic distributed systems. *J. Syst. Softw.*, pages 1639–1658, 2007.

[8] A. Metwally, D. Agrawal, and A. El Abbadi. Why go logarithmic if we can go linear?: Towards effective distinct counting of search traffic. In *EDBT*, pages 618–629, 2008.

[9] A. Mostéfaoui, M. Raynal, C. Travers, S. Patterson, D. Agrawal, and A. El Abbadi. From static distributed systems to dynamic systems. In *SRDS*, pages 109–118, 2005.

[10] S. Shekhar and H. Xiong, editors. *Encyclopedia of GIS*. Springer, 2008.

[11] O. Wolfson, B. Xu, H. Yin, and H. Cao. Search-and-discover in mobile p2p network databases. In *ICDCS*, 2006.

[12] K. Wu, Y. Xiao, J. Li, and B. Sun. A distributed algorithm for finding global icebergs with linked counting bloom filters. In *ICC*, pages 2757–2761, 2008.