Latency Minimization Through Optimal User Matchmaking in Multi-Party Online Applications

Ning Wang and Jie Wu
Center for Networked Computing, Temple University, USA
Email: {ning.wang, jiewu}@temple.edu

***Abstract*—To improve the user experience in multi-party online applications, i.e., low gaming lag in online gaming, the maximum latency between any pair of users should be minimized. Considering the multiple preferences of users, i.e., which group a user can join, we address the User Latency Minimization (ULM) problem by performing an optimal matchmaking. This paper proves that the ULM problem is NP-hard if users have more than two preferences. We prove that the ULM problem has the sub-modular property and we apply the classic greedy algorithm with an approximation ratio of $1 + \ln n$, where n is the number of users with multiple preferences. Furthermore, we observe that a matchmaking priority for users in different locations exists and thus we propose a revised greedy algorithm with an approximation bound and discuss its performance in the tree structure and the general structure. Specifically, the revised greedy algorithm achieves an approximation ratio of $m/2$ with lower complexity in the tree structure, where m is the number of preferences in the general structure. Finally, we develop a distributed greedy approach which converges quickly. Extensive trace-driven experiments from Internet measurements demonstrates that our schemes achieve good performances.**

***Index Terms*—Geometric optimization, interactive application, network combinatorial optimization.**

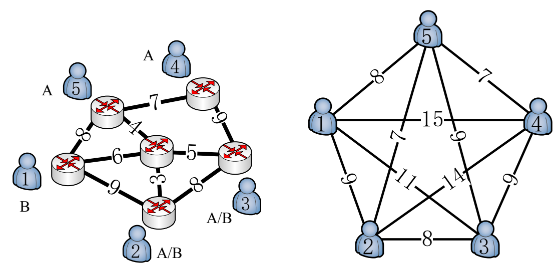
I. Introduction

Nowadays, multi-party interactive application, such as online gaming and video/audio conferencing, is very popular due to the popularity of laptops, tablets, and smart phones, and recent advances in cellular networks. The interactive application means that all users interact with each other and thus, there is communication between any pair of users. Among the interactive applications, online games remain the most popular application category in both the iOS and Android ecosystems [1] in terms of downloads, usage, and revenue earned. In particular, multi-user games are becoming increasingly popular to both game users and developers. Users find that the unpredictability that arises from playing against human opponents keeps them engaged for much longer periods while game developers find that more engaged users generate a lot more ads and in-game sales revenues. In 2016, the Supercell (a game company) reported an annual revenue of around 2.11 billion Euro [2]. Another game, called King of Glory, has 160 million monthly active users [3].

One fundamental challenge in interactive applications is that clients are geographically distributed, and each client needs to receive a stream from all the other clients in real time. Appealing to increasingly discriminating game users requires careful attention to their chief concerns known as lag, the perceived time between an action and its effect [4]. In this paper, we focus on the Peer-to-Peer (P2P) network model, which is common in online games. In the P2P environment, the primary contributor to the lag is the direct communication between participants game machines over the Internet. To reduce lag and hence improve the game experience, it is critical to design efficient schemes from hardware to software, e.g., increasing bandwidth and designing streaming techniques.

TABLE I
Comparison of different matchmaking strategies.

	Matchmaking strategy		
	First	Second	Third
Group A	2, 4, 5	4, 5	3, 4, 5
Group B	1, 3	1, 2, 3	1, 2
Max. latency	14	11	9



(a) Internet network (b) Latency mapping

Fig. 1. The illustration of different matching strategies.

The objective of this paper is to minimize the maximum pair latency of users in every group, called the User Latency Minimization (ULM) problem. Instead of trying to improve the hardware or software, this paper tries to reduce the latency from another angle based on the observation that each user might have *multiple preferences*, i.e., potential groups to join, and different matchmaking assignments lead to different maximum latencies. Note that there is no special player number requirement in each group except the maximum number, which is the new trend in battle royale game [5]. Therefore, we propose an optimal matchmaking in multi-party online applications to reduce the latency. The ULM problem is challenging for the following two reasons: (1) each assignment will have an influence on all users in that group; (2) the matchmaking assignment a user may correlate with future other users' assignments and thus it is hard to evaluate its correctness.

A motivational example is shown in Fig. 1, where there are five users. Fig. 1(a) shows the network topology, and the corresponding link weight is the communication latency. Fig. 1(b) shows the minimum latency between them by searching all shortest-paths. Note that among the five users, there are two users who can be assigned to the group A or B. Table. I shows three matchmaking strategies. The first one has a maximum delay of 14, and the second one has a maximum delay of 11. The reason is due to grouping of large latency users, i.e., users 2 and 5 have a large latency. However, even users are grouped based on distances, like in matchmaking strategy 2. It is still not optimal. Another matchmaking strategy is that users 1 and 2 are grouped, and the remaining users are grouped into the other group. In this case, the maximum latency is 9.

In this paper, we first prove that the proposed ULM problem can be solved optimally if each user has at most 2 preferences. If a user has more than 2 preferences, the ULM problem is NP-hard. Then, we prove that ULM has the sub-modular property and thus, the classic greedy algorithm with an approximation ratio of $1+\ln n$ can be applied to the ULM problem, where n is the number of users with multiple preferences. Furthermore, we observe the unique properties in the ULM problem and thus propose a revised greedy algorithm. The discussion of the revised greedy algorithm begins from the tree network structure. We observe that some users' matchmakings are dominated by other users' matchmaking results; therefore, different users have different assignment priorities based on their locations and thus the matchmaking assignment is more likely to be good, and the running time can be reduced at the same time. The revised greedy algorithm has an approximation ratio of $m/2$, where m is the number of groups. In general topology, we prove from theorems and experiments that the revised greedy algorithm can improve the performance of the classic greedy algorithm. In addition, we propose a distributed greedy algorithm which can converge to a close-to-optimal solution quickly.

The contributions of this paper are summarized as follows:

- To our best knowledge, we are the first to consider this optimal matchmaking to reduce the maximum latency in an online multi-party interactive application.
- We first prove that the proposed problem has the sub-modular property and thus, the classic greedy algorithm with an approximation bound can be applied.
- We observe the unique properties in the ULM problem and propose a revised greedy approach, which further improves the performance regarding latency minimization.
- We propose a distributed greedy algorithm which can converge to a close-to-optimal solution quickly.

The remainder of the paper is organized as follows. The related works are in Section II. The problem statement is introduced in Section III. The sub-modular proof and the corresponding classic greedy algorithm are provided in Section IV. The new observations of the ULM problem and the revised greedy algorithm are presented in Section V. The distributed implementation algorithm is presented in Section VI. The experimental results from real Internet traces are shown in Section VII. We conclude the paper in Section VIII.

II. Related Works

With the popularity of multi-party interactive applications, the delay minimization problem has drawn much attention from researchers [6–11]. The network environment can be roughly grouped into two categories: (1) the client/server model [7, 9, 12] and (2) the P2P model [10, 13–15]. The P2P model is common in the real-time strategy genre due to its suitability for games with large numbers of tokens. This paper falls within the second category.

Client/server environment: Hu et al. [7] considered how to optimally place servers so that the end-to-end delay of clients is minimized. The geometric property is considered and the problem is formulated as a Euclidean k-median problem. Given servers' locations, Hajiesmaili et al. [9] discussed the optimal server selection problem. They argued that selecting the closest server may not be the optimal due to the character of the interactive application. That is, the delay is not determined locally but by pairs of users. They proposed a greedy approach to update the configuration iteratively. In [12], the authors discussed the optimal streaming due to the multicast communication character in the online gaming. Previous works only considered how to optimize users' distances, but bandwidth consumption was not considered. Therefore, they proposed a bandwidth minimization algorithm to discuss where and how to split the data streaming paths.

Peer-to-peer environment: Chen et al. [10] designed a scheme to deliver videos with low end-to-end delays. The bandwidth of each link is considered to optimize the QoS over arbitrary network topologies where bottlenecks can be anywhere in the network. In [13], authors proposed a bandwidth-fair N-Tree algorithm to balance the bandwidth consumption in each link. In [14], they argued the delay among users were dynamically changed. Therefore, their approach identified and ranked potential detour paths between any two users and dynamically selects the most suitable one based on network and client conditions. However, according to the real Internet trace measurement [15], the Internet has triangle inequality violations, and thus detour design is challenging.

To our best knowledge, the existing works do not discuss the matchmaking due to users' multiple preferences. This paper is the first to theoretically discuss the optimal user matching problem in multi-party online gaming.

III. Problem Statement

The network model, the proposed problem, and the corresponding hardness are provided in this section.

A. Problem Formulation

This paper discusses a multi-party online applications for geo-distributed users through the Internet. Each user has a coordination in the 2-D network. To simplify the illustration, we assume the network is discretized into grids. The distance between the users is calculated based on grids, and the calculation error is bound by the discretization level, i.e., the grid size. For each user, there is a gaming profile/preference profile, which indicates the potential game groups that a user would like to join. Note that there may be *multiple preferences*

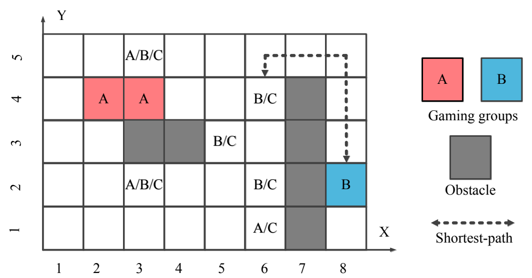


Fig. 2. The network model

for a user. Let us denote that there are a total of m different groups in the network, $G = \{g_1, g_2, \cdots, g_m\}$, where a user has at least one potential game group. m is usually much smaller than n. Note that we use capital letters to represent different groups in the figures, e.g., $g_1 = A$. The number of users with multiple potential game groups is n. Without loss of generality, let us denote users as $\mathbb{U} = \{u_1, u_2, \cdots, u_n\}$ and their corresponding gaming profiles as $\mathbb{V} = \{v_1, v_2, \cdots, v_n\}$, e.g., $v_1 = \{A, B\}$. There is a matchmaking assignment vector $H = \{h_{11}, h_{12}, \cdots, h_{nm}\}$, where $h_{ik} = 1$ ($h_{ik} = 0$) means that the user i is (not) matched into group k, g_k. In addition, we use a decision vector D to denote all the matchmaking decisions that we have made, i.e., $D = \{H | h_{ik} = 1\}$.

To ensure a good gaming experience, the lag, the perceived time between an action and its effect [4], should be minimized or controlled within a threshold. The latency of a pair of users is determined by their communication latency in a session during a multi-party online game, which can be approximately estimated through their Internet latency. The Internet latency can be further approximately estimated by their geo-distance metrics based on [16, 17], e.g., the shortest path, the absolute value of their coordination differences, and the area of the coordination differences between them. The shortest-path can be calculated using existing approaches, as in [18]. Note that in the remainder of this paper, the communication latency of a pair of users u_i and u_j, denoted as l_{ij}, is approximated by the shortest-path between the grids that the pair of users u_i and u_j belong to. The other latency estimation metrics are similar and we evaluate them in the experiments.

A network illustration is shown in Fig. 2, where there are some users located in the network waiting for a matchmaking in a specific time slot. The capital letters in the grids indicate the groups that they can join. For example, a user located at $(8, 2)$ should be matched into group B. Another user who is located at $(3, 5)$ can be matched into groups A or B or C. The shortest-path between a pair of users is denoted by a dashed arrow in Fig. 2.

Different matchmaking assignments for users who have multiple potential game preferences has a huge influence on the latency minimization. Therefore, we propose the User Latency Minimization (ULM) problem which tries to minimize the maximum latency in the network based on the proposed network model. It is mathematically formulated as follows:

$$
\begin{aligned}
\min \quad & \max x_{ij}^k l_{ij} \\
\text{s.t.} \quad & \sum h_{ik} = 1, \ h_{ik} = \{0, 1\}, \\
& x_{ij}^k \geq h_{ik} + h_{jk} - 1, \ x_{ij}^k \leq h_{ik}, \\
& x_{ij}^k \leq h_{jk}, \ x_{ij}^k = \{0, 1\}, \ \forall i, j, k.
\end{aligned} \tag{1}
$$

The objective means that the maximum latency of any pair of users in the same group should be minimized. The first constraint ensures that each user is only assigned into one group. The second and third constraints jointly ensure that $x_{ij}^k = 1$ only if users u_i and u_j are grouped into the same group, g_k, i.e., $h_{ik} = 1$ and $h_{jk} = 1$.

B. NP-hardness Proof

The proposed ULM problem is a combinatorial optimization. We have proved the hardness of the ULM problem. If there exists a case where a user has more than 2 potential selections, the ULM problem is NP-hard.

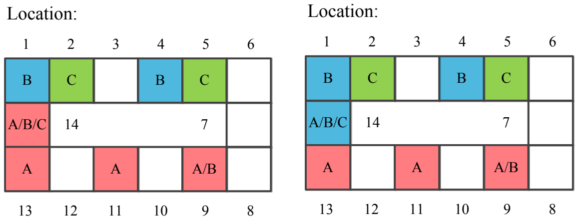
Theorem 1. *The ULM problem is NP-hard when users have more than* 2 *preferences.*

Proof. Firstly, the ULM problem belongs to NP class because for a given matchmaking, H, we can verify if all constraints are satisfied simultaneously in polynomial time. Now, we show it is NP-hard by a reduction of the 3SAT problem.

The 3SAT problem [19] is as follows: Given a set of clauses $C_1, C_2, ..., C_m$ in a 3 Conjunctive Normal Form (CNF) from variables $y_1, y_2, ..., y_n$, we must check if all the clauses are simultaneously satisfiable.

We relax the equality constraint of the ULM problem into an inequality, e.g., $\sum h_{ik} \geq 1$. Since the solution space of the ULM problem is within the relaxed ULM problem, if we can solve the ULM problem in polynomial time, the relaxed ULM is also solved in polynomial time. Therefore, the ULM problem is at least as hard as the relaxed ULM problem. In the following proof, we prove the relaxed ULM problem is NP-hard. For explanation simplicity, we use a new variable z to denote integer variables h and x in Eq. 1.

For each clause in the 3SAT instance, we reduce each clause into an inequality constraint of the relaxed ULM problem as follows: (1) If the literal in the clause is in negation form of a variable, say $\bar{x}_i$, then add $(1 - z_i)$ into the inequality. (2) If the literal, say y_i, is not in the negation form, then simply add z_i into the inequality. For example, if we have a clause $\{y_1 \vee \bar{y}_2 \vee y_3\}$ in the 3SAT instance, we have a constraint $z_1 + (1 - z_2) + z_3 \geq 1$ in the ULM problem, which is a special instance of the relaxed ULM problem. On the one hand, in any SAT solution, a true literal corresponds to 1 in the relaxed ULM program, since the clause is satisfied. Therefore, the sum in each clause inequality is no smaller than 1. On the other hand, any relaxed ULM solution gives a 3SAT solution, for any solution to this ULM instance, at least one variable value in each inequality will make the corresponding literal true. So, it is a legal assignment, which must also satisfy all the clauses. Therefore, the relaxed ULM problem is NP-hard. And thus the ULM problem is NP-hard. $\square$



(a) min-max algorithm (b) optimal

Fig. 3. An illustration of the min-max algorithm.

Note that the 2SAT problem is not NP-hard, which means that when the number of groups is 2, i.e., $m = 2$, the ULM can be solved optimally.

IV. General Property and Approach

In this section, we first prove that the ULM problem owns the sub-modular property and thus, we can apply the well-known approximation algorithm to the ULM problem.

A. Sub-modular Property

There is a high complexity in trying every combination and backtracking if a matchmaking combination leads to a bad result. Instead, we try to gradually "expand" the matchmaking assignment. However, during the expanding procedure, the error might increase. We prove that the error can be bounded, which is the insight of the sub-modular property.

Given a non-empty finite set, $\mathbb{S}$, and a function, f, defined on the power set $2^{\mathbb{S}}$ of $\mathbb{S}$, $2^{\mathbb{S}} \rightarrow \mathbb{R}$. The definitions of them are as follows: *Nonnegative:* f is called nonnegative if $f(S) \geq 0$ for all $S \subseteq \mathbb{S}$. *Monotone:* f is called monotone if $f(S) \leq f(S')$ for all $S \subseteq S' \subseteq \mathbb{S}$. *Sub-modular:* f is called sub-modular if $f(S \cup \{s_1\}) + f(S \cup \{s_2\}) \geq f(S \cup \{s_1, s_2\}) + f(S)$ for every set S, where $s_1, s_2 \in \mathbb{S} \backslash S$. Then, we prove the ULM problem has the nonnegative, monotone and sub-modular properties.

Theorem 2. *The objective function $f(D)$ of the ULM problem is nonnegative, monotone, and sub-modular.*

Proof. In the ULM problem $f(\cdot)$ is the maximum latency in all groups. According to the definition of latency, proportional to the shortest path between a pair users, the latency has a minimal value 0 and it cannot be negative. Therefore, $f(D)$ is nonnegative. If there exists $D', D'', D' \subseteq D'' \subseteq D$, the corresponding assigned user sets are U', U'' and $U' \subseteq U'' \subseteq U$, respectively. Let us denote the user sets in g_k as U'_k, and U''_k for the corresponding of D'_k and D''_k, respectively. Clearly, $D'_k \subseteq D''_k$ and $U'_k \subseteq U''_k$, otherwise, there is a contradiction of $D' \subseteq D''$. Then, let us denote $f_k(\cdot)$ as the objective value in g_k. Based on the inclusion relation, $f(D'_k) = \max_{u_i, u_j \in U'} l_{ij} \leq f_k(D''_k) = \max_{u_i, u_j \in U''_k} l_{ij}$ and $f(D) = \max_{k \in [1,m]} f_k(D_k)$. Therefore, $f(D') \leq f(D'')$ and $f(D)$ is monotone.

Based on the calculation of $f_k(D_k)$, for any group,

$$f_k(D_k \cup h_{i'j'}) = \max\{f_k(D_k), \max_{u_i, u_j \in \{U_k \cup u_{i'}\}} l_{ij}\}. \quad (2)$$

Let us denote $h^1_{i'j'}$ and $h^2_{i''j''}$ as two new matchmaking assignments. $u_{i'}$ and $u_{i''}$ are corresponding users and $u_{i'}$ and $u_{i''} \in \mathbb{U} \backslash U$. If the following inequation is true,

$$f(D \cup \{h^1_{i'j'}\}) + f(D \cup \{h^2_{i''j''}\}) \geq f(D \cup \{h^1_{ij}, h^2_{i''j''}\}) + f(D), \quad (3)$$

the ULM is a sub-modular. When $u_{i'}$ and $u_{i''}$ belong to different groups, the two sides of above function are equal since $u_{i'}$ and $u_{i''}$ have no influence to each other. Therefore, we focus on the following condition, where $u_{i'}$ and $u_{i''}$ belong to the same group, g_k, that is,

$$\begin{aligned} &f_k(D_k \cup \{h^1_{ij}\}) + f_k(D_k \cup \{h^2_{i'j'}\}) \\ &\geq f_k(D_k \cup \{h^1_{ij}, h^2_{i'j'}\}) + f_k(D_k), \end{aligned} \quad (4)$$

Based on Eq. 2, we prove that the InEq. 4 is true in all cases. (1) if $\max_{u_i, u_j \in \{U_k \cup u_{i'}\}} l_{ij} \leq f_k(D_k)$ and $\max_{u_i, u_j \in \{U_k \cup u''_i\}} l_{ij} \leq f_k(D_k)$. In this case, the two sides of InEq. 4 are the same and equal to $2f_k(D_k)$. Therefore, InEq. 3 is true. (2) if $\max_{u_i, u_j \in \{U_k \cup u_{i'}\}} l_{ij} \geq f_k(D_k)$ and $\max_{u_i, u_j \in \{U_k \cup u_{i''}\}} l_{ij} \geq f_k(D_k)$, the left side of InEq. 3 is $\max_{u_i, u_j \in \{U_k \cup u_{i'}\}} l_{ij} + \max_{u_i, u_j \in \{U_k \cup u_{i''}\}} l_{ij} \geq \max_{u_i, u_j \in \{U_k \cup \{u_{i'}, u_{i''}\}} l_{ij} + f_k(D_k)$ and InEq. 3 is true. (3) if $\max_{u_i, u_j \in \{U_k \cup u_{i'}\}} l_{ij} \geq f_k(D_k)$ or $\max_{u_i, u_j \in \{U_k \cup u_{i''}\}} l_{ij} \geq f_k(D_k)$, two sides of InEq. 4 equal to $\max_{u_i, u_j \in \{U_k \cup \{u_{i'}, u_{i''}\}} l_{ij} + f_k(D_k)$ and InEq. 4 is true. InEq. 4 is always true and $f(D)$ is sub-modular. □

Algorithm 1 Min-Max (MM) Algorithm

Input: The location and gaming profile of users.
Output: The matchmaking decision vector D.

```
D = ∅ and U = ∅
while |U| < |𝕌| do
    I = ∞, idx = {−1, −1}
    for i from 1 to n do
        if u_i ∈ 𝕌\U then
            for j from 1 to m do
                if g_j ∈ v_i and f_j(D ∪ {h_ij}) − f_j(D) < I then
                    I = f_j(D ∪ {h_ij}) − f_j(D), idx = {i, j}
    Set h_ij = 1 , D = {D, h_ij} and U = {U, u_i}.
```

According to the results in [20], we have a $1 + \ln n$ approximation algorithm, as shown in Algorithm 1, which starts with an empty set. In each iteration, we check all the unassigned users and add the matchmaking which minimizes the marginal gain of the objective function, i.e.,

$$h_{ij} \leftarrow \arg \min_{u_i \in \mathbb{U} \backslash U} \{f(D \cup h_{ij}) - f(D)\} \quad (5)$$

An illustration of the Algorithm 1 is shown in Fig. 3, where users in locations 7 and 14 needs to be matched and the current maximum latency is 4. If the user in location 7 is matched to group A, the maximum latency increase is the least, i.e., 1. If the user in location 14 is matched to group A, the maximum latency increases the least, which is 0. Therefore, the user in location 14 is assigned to group A first. Then, we re-calculate the minimum cost increase for the user in location 7. Since it is still better to match this user to group A than to group B. The min-max algorithm will assign user in location 7 to group A in the second round. Finally, the maximum latency is 6. The optimal matchmaking has a maximum latency of 5 as shown in Fig. 3(b).

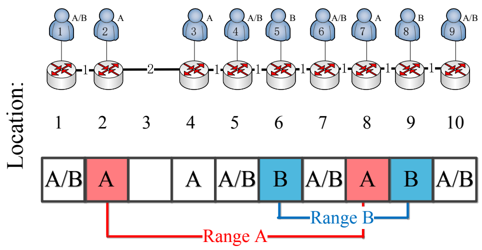


Fig. 4. The network model in the line topology

V. Revised Greedy Algorithm

In this section, we will discuss observations for the ULM problem and then propose a revised greedy algorithm. The performance of the revised greedy algorithm is analyzed under (1) the tree topology and (2) the general topology with loops.

A. Tree topology

The tree topology assumption justification is based on the previous researches [21, 22] that the Internet topology can be considered as a tree or can be approximated into the tree structure with a bounded performance loss.

Definition 1. *A user that is dominated by a group if his/her matchmaking does not have an influence on the maximum latency in that group.*

We begin with a special tree-topology, i.e., the line topology, to present the property in tree topology and the idea of the revised greedy algorithm, followed by the general topology. An illustration of the line network is shown in Fig. 4.

Observation 1. *A user is dominated if there are users in the same group on his/her two sides in the line topology.*

In the line topology, the maximum latency of a particular group is determined by the *boundary users*, e.g., the right-most user and the left-most user in that group. The reason is that any user's distance with the boundary users is always smaller than the distance between the boundary users. From the Observation 1, all users within a pair of boundary users in a group, called the group range, are dominated. The domination property can be used to simplify and guide the matchmaking, i.e., we can conduct matchmaking for dominated users without losing optimality.

Fig. 4 shows an illustration of the Observation 1, where boundary users and the group range are marked with corresponding colors. For example, whatever a user in location 7 is matched to groups A or B, users with the same group are on both sides. Therefore, this user is dominated by groups A and B. The user in location 5 is dominated by group A but not by group B. Therefore, if this user is matched to group B, the maximum latency of group B will increase. The solution space can be reduced without losing optimality based on this observation, i.e., assigning the user in location 7 to group A/B and assigning the user in location 5 to group A.

Definition 2. *A boundary location for a group is a location where the matchmaking for the user in that location can increase the maximum latency most.*

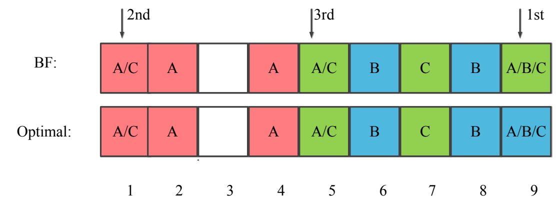


Fig. 5. The analysis of the boundary-first algorithm

Algorithm 2 Boundary-First (BF) Algorithm

Input: The location and gaming profile of each user.
Output: The matchmaking decision vector D.

$D = \emptyset$ and $U = \emptyset$
while $|U| < |\mathbb{U}|$ **do**
  $I = 0$, $idx = -1$.
  // **if** the network is a tree topology
  // Conduct matchmaking for dominated users.
  // Check boundary nodes from boundary locations.
  for i from 1 to n **do**
    if $u_i \in \mathbb{U} \backslash U$ **then**
      for j from 1 to m **do**
        if $g_j \in v_i$ and $f_j(D \cup \{h_{ij}\}) - f_j(D) > I$ **then**
          $I = f_j(D \cup \{h_{ij}\}) - f_j(D)$, $idx = i$.
  $j \leftarrow \arg\min_{g_j \in v_i} \{f_j(D \cup \{h_{ij}\}) - f_j(D)\}$
  Set $h_{ij} = 1$, $D = \{D, h_{ij}\}$ and $U = \{U, u_i\}$.

Observation 2. *A user in the boundary location of a group has a higher priority in the matchmaking procedure.*

The reason of the Observation 2 is that an inner user has a high probability of being dominated by a user in the boundary location. In Fig. 4, location 1 is the boundary location for group B and location 10 is the boundary location for group A in the current matchmaking assignment. Note that the boundary location is the right-most/left-most location of a group in the line topology.

Based on the Observations 1 and 2, we propose the boundary-first algorithm as follows: in each round, we first conduct matchmaking to all dominated users. If there still exists unmatched users, the boundary-first algorithm iteratively finds users in the boundary location of each group. Among these users in the boundary locations, we find the user whose improper assignment can lead to the largest latency increase and conduct matchmaking to him/her so that his/her matchmaking increases the maximum latency least. The boundary-first algorithm is shown in Algorithm 2.

A running example of the Algorithm 2 is shown in Fig. 5. Before matching any users with multiple preferences, the maximum latency is 3. In the first round, the user in location 9 is selected to matchmaking, since his/her worst assignment, i.e., matchmaking to the group A, will lead to a maximum latency increase of 5. Through checking, if he/she is assigned to the group C, the maximum latency increase is 0, which is the minimum. Similarly, in the second round, the user in location 1 should be assigned, since the improper assignment to this user will lead to a maximum latency increase of 6, i.e., matchmaking to group C. The best matchmaking for him/her

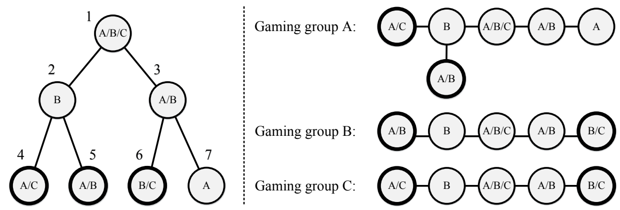


Fig. 6. An illustration of the boundary-first algorithm in the tree topology

is group A, and the maximum latency increases 1. In the third round, the user in location 5 should be assigned to the group A or C, which does not matter in terms of the result. We use the group C as an example. The maximum latency is 5 in the end. The optimal matchmaking has a result of 4.

Theorem 3. *In the line topology, the boundary-first algorithm has an approximation ratio of $m/2$ with identical link latency.*

Proof. Since the maximum latency of a group is proportional to the group range in the line topology, we use the group range to represent the maximum latency of a group in this proof. Assume the length of the network is L. (1) If the initial maximum group range is no smaller than $L/2$, the boundary-first algorithm will not match users into the group with the maximum range due to its greedy matchmaking criterion. Therefore, the maximum range is unchanged and the boundary-first algorithm is optimal. (2) If the initial maximum group range is no larger than $L/2$, the boundary-first algorithm will balance the increase of each group range in a greedy manner, i.e., always increasing the group with a small range. Therefore, in the end, the maximum group range should be no larger than $L/2$ except in the case that the minimum increase leads to a maximum group range whose length is larger than $L/2$. In the latter case, it returns to the case (1). The group range of the optimal solution, called OPT, must satisfy the condition that $OPT \geq L/m$. The reason is that if $OPT < L/m$, it is impossible to matchmake all users. Therefore, the approximation bound of the boundary-first algorithm is $m/2$ in the line topology. □

The insight of the Theorem 3 is that because the Algorithm 2 is a round-by-round optimization and thus, the cooperation between multiple rounds is ignored. The opportunity of cooperation increases as the group number also increases.

The boundary-first algorithm in the line topology can be applied into a general tree topology with some modifications. For each unmatched leaf node, boundary-first algorithm checks whether it has an influence on the maximum latency for each group. If not, the checking passes to its parent node. Otherwise, there must be a line topology with the maximum length that needs to be matched. After finding the line topology, the remaining steps are the same as the line topology. An example of the explanation is shown in Fig. 6, where link latencies between nodes are the same, so that the hop-count represents the latency. User 4 is involved into the maximum latency of groups A and C. User 5 is involved into the maximum latency of groups A and B. User 6 is involved into the maximum latency of groups B and C. After finding these boundary nodes, we transfer the tree topology to m lines as shown on the right of Fig. 6. The tree-topology optimization can be considered as jointly optimizing multiple co-related lines. In each round, we select the line whose improper matchmaking can lead to the maximum latency increase to optimize. Since the maximum latency of tree is determined by one of the lines, Theorem 3 holds.

B. General topology with loops

In this subsection, we do not have any assumption about the network topology and loops may exist in the network. The matchmaking becomes even more challenging, since we cannot find users in boundary locations easily. We can still use the boundary-first algorithm. However, in each round, we need to check all unassigned users to find whose improper assignment can increase the latency most, the same time complexity as the min-max algorithm. Then, we perform the best assignment for the selected user, i.e., increase the latency the least, until all the unassigned users are matched.

Theorem 4. *The boundary-first algorithm always achieves at least the same performance as the min-max algorithm.*

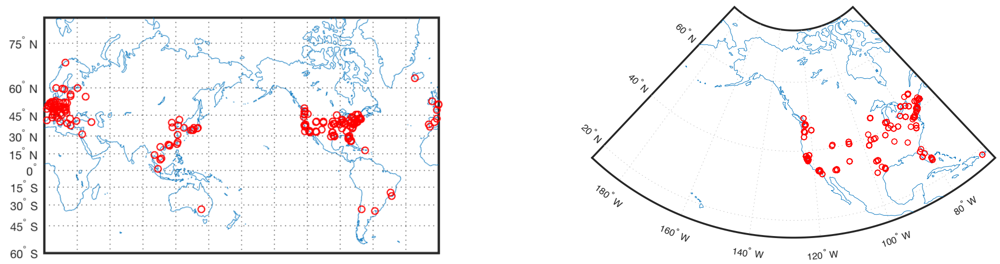
Proof. Theorem 4 is proven through contradiction. If the boundary-first algorithm achieves a larger maximum latency than the min-max algorithm, there are a pair of users, u_i and u_j, which belongs to the same group, leading to a maximum latency but the matchmaking for these two users are different in the min-max algorithm. However, we can modify the matchmaking of users u_i and u_j to the same matchmaking as the min-max algorithm to reduce latency. It is because these two algorithms follow the same greedy metric for a selected user, i.e., matching the user into a group which can increase the maximum latency least. If so, this is a contradiction of the greedy metric. If not, it means that the matchmaking result is worse after the modification. Without loss of generality, we assume that modifying user u_j's matching will lead to a worse pair, e.g., users u_j and u_k. Clearly, u_j and u_k belong to different groups in the matching result of the min-max algorithm, or there is a contradiction. Then, we reach into the same situation for users u_j and u_k as users u_i and u_j in the initial assignment. Therefore, we can prove by repeating the aforementioned procedure until we find a contradiction. □

VI. Distributed Greedy Approach

Many P2P systems, the common environment that online applications are implemented in a distribution manner. However, it is hard for a distributed solution to guarantee the performance. In this section, we adapt the Markov approximation approach in [23] and develop a distributed solution.

A. Markov Approximation Framework

The idea of the Markov approximation framework is that each matchmaking for n users can be considered as a state in a Markov chain. Therefore, we build a time-reversible Markov chains between all states. A well-designed Markov chain ensures that it can converge to a close-to-optimal state.



(a) World (b) United States

Fig. 7. An illustration of nodes geo-locations in the PlantLab trace.

Let $\mathbb{D}$ be a set of matchmaking result set and D is one of the feasible matchmaking. Then if p_D is the percentage of time that the matchmaking result, i.e., $f(D)$ stays in a Markov chain. The objective function of Eq. 1 has the same optimal value as the following equation,

$$\min \sum_{D\in\mathbb{D}} p_d f(D) \qquad \text{s.t.} \quad \sum p_D = 1. \tag{6}$$

Eq. 6 is NP-hard, which cannot be solved directly. However, it can be approximated by adding a second-order characterization. In this paper, we approximate it by using the log-sum-approximation. That is, the Eq. 6 is approximated to,

$$\min \sum_{D\in\mathbb{D}} p_D f(D) + \frac{1}{\beta}\sum_{D\in\mathbb{D}} p_D \log p_D, \text{ s.t.} \sum p_D = 1 \tag{7}$$

The Theorem in [23] shows that the optimal value between Eqs. 6 and 7 have the following relationship,

$$\min f^\star - \frac{1}{\beta}\log|\mathbb{D}| \le \hat{f}^\star \le f^\star \tag{8}$$

where $f^\star$ is the optimal value of Eq. 6 and $\hat{f}^\star$ is the optimal value of Eq. 7. Therefore, the β value and cardinality of $\mathbb{H}$ bound the approximation accuracy. The optimal solution of Eq. 7 is a convex optimization problem and thus can be solved by using Karush-Kuhn-Tucker (KKT) conditions [24]. As a result

$$p_D^\star = \frac{\exp(-\beta f(D))}{\sum_{D\in\mathbb{D}}\exp(-\beta f'(D))}, \tag{9}$$

and therefore the optimal solution is

$$\hat{f}^\star = -\frac{1}{\beta}\log(\sum_{H\in\mathbb{H}}\exp(-\beta f(D))) \tag{10}$$

B. Distributed Greedy Algorithm Design

The basic design idea to find the optimal value, $\hat{f}^\star$ is to simulate a time-reversible Markov chain over time. The initial state can be any feasible matchmaking. However, it can transfer to other feasible states and will stay in the optimal state most of the time. To build the ergodic Markov chain whose stationary distribution is $p_D^\star$, the following two conditions should be satisfied:

(1) any two states are reachable from each other;
(2) the transfer probability is symmetric for all pairs of states, i.e., $p_{D_1}^\star q_{D_1,D_2} = p_{D_2}^\star q_{D_2,D_1}, \forall D_1, D_2 \in \mathbb{D}$, where q is the transition probability.

Based on these two conditions, we build the Markov chain and the transition probability between states as follows: (1) setting the transfer probability between any two states to be zero, i.e., cutting off the direct transition between them, given that they are still reachable from any other states. This setting can reduce the migration overhead of the P2P system; (2) the transition probability of any two states is

$$q_{D_1,D_2} = \alpha\exp(\beta(f(D_1) - f(D_2))) \tag{11}$$

where there is a trade-off in the α value selection, i.e., larger α reduces the convergence but it may impose the overhead of frequent assignment. The distributed greedy algorithm is shown in Algorithm 3. Note that the transition probability can be implemented locally. Algorithm 3 converges to a stationary state with provable convergence time [23].

Algorithm 3 Distributed Greedy Algorithm

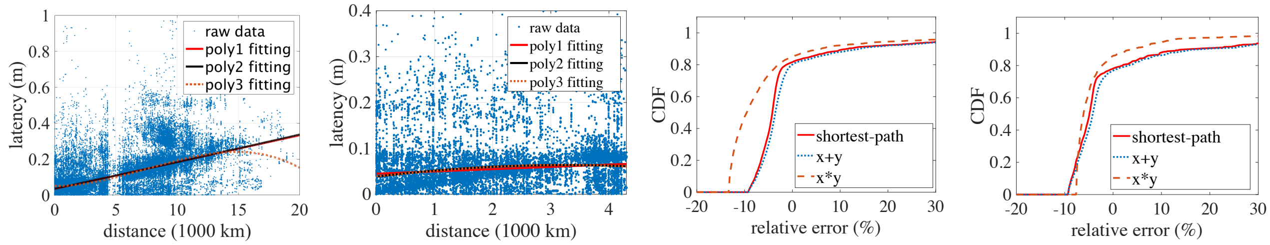
Procedure: WAIT
  Create a random countdown time number with mean $1/\alpha$.
  while the timer is still larger than 0 **do**
    if Receive a Suspend message **then** Pause.
    else Resume and Invoke WAKE.
Procedure: WAKE
  Broadcast a Suspend message to other users.
  Find all feasible solutions with only one different decision.
  Change the matchmaking assignment with probability proportional to the Eq. 11.
  Broadcast an Continue message to other users.
  Invoke WAIT.

VII. Performance Evaluation

In this section, we demonstrate the effectiveness of proposed algorithms by using the real Internet trace.

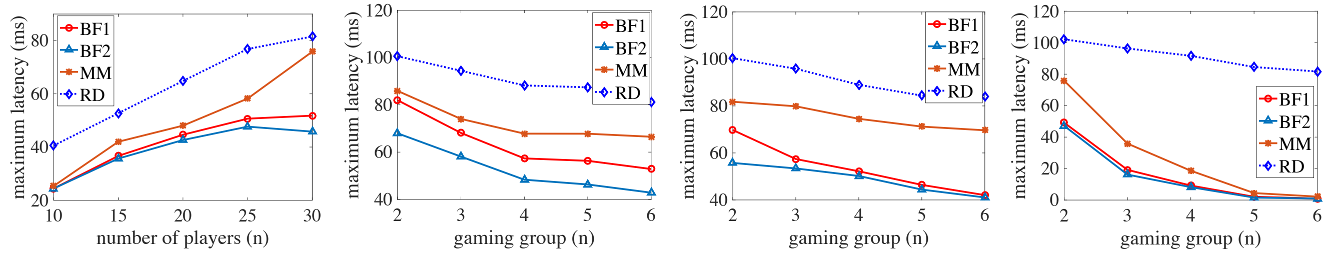
A. Trace Introduction

In this paper, we use the PlantLab trace [25] generated from the PlantLab testbed. It contains a set of geo-distributed hosts worldwide. In this trace, the medians of all latencies, i.e., RTT, between nodes are measured through the King method between the 325 PlantLab nodes and 400 other most-popular websites in the world are measured. An illustration of the nodes' geo-locations is shown in Fig. 7. In PlantLab trace, the domain of each node is provided. Each node's geometric location is retrieved through the domain-to-IP database and the IP-to-Geo database, provided by and [26] and [27], respectively. Some domains are no longer in service. In the end, there are 689 nodes. It is reported that the mapping error is within 5mi and can be ignored in our experiments.



(a) worldwide (b) the United States (c) worldwide (d) the United States

Fig. 8. An illustration of the distance-latency relationship in PlantLab trace.



(a) 6 groups (b) 30 users (c) 30 users (d) 30 users

Fig. 9. Performance comparison in the line topology.

B. Experiment Setting

We conduct experiments on two scales, i.e., the world and the United States in the experiments. For the United States scale, we use the nodes on the west coast to simulate the line-topology. The number of the groups, m, is changed from 2 to 6. The number of users is changed from 10 to 60. There are at most 2^m different types of users in the network. We conduct two main different preferences settings in the experiments. (1) the number of users with different preference cardinalities and each type of users are the uniform distribution (2) the number of users with different preference cardinalities is an exponential distribution with a parameter 1. We use the exponential distribution to simulate different flexibilities in the matching assignment.

C. Algorithm Comparison

We compare the performance of the proposed algorithm under different topology settings.

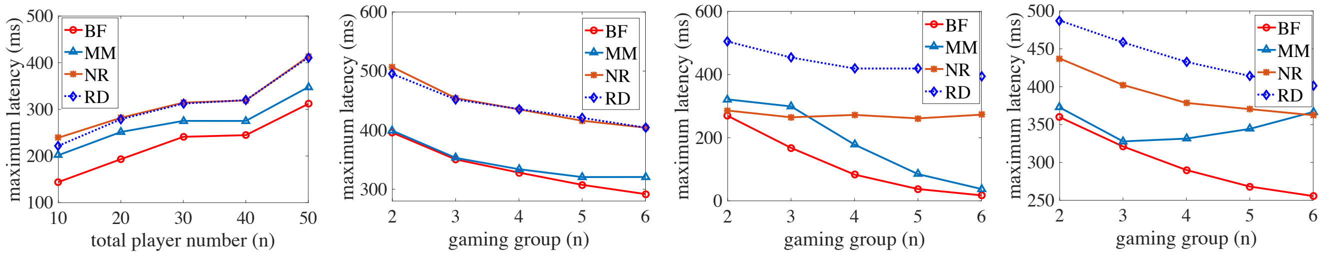
- Min-Max (MM) algorithm: explained in Section IV, which is widely used in combination optimization [20].
- Boundary-First (BF) algorithm, which is proposed in this paper and well explained in Section V.
- Nearest-Representative (NR) algorithm, which tries to assign each user to its nearest group, where the distance is determined by the distance between the current node and the group representative [28].
- Random algorithm (RD) algorithm, which randomly selects a matching for a user. RD algorithm is a baseline approach, and it is also the comparison in [28].

D. Results Analysis

Fig. 8 presents the trace analysis results from two scales, i.e., worldwide and the United States. Specifically, Figs. 8(a) and 8(b) show the distance-latency relationship results, where the coordinates of raw data show a distance-latency measurement in the PlantLab trace. We try to use the first-order, the second-order, and the third-order polynomial functions to fit the distance-latency relationship. The results show that the first-order polynomial fitting is enough since higher-order fittings have almost the same fitting result. Therefore, in the following experiments, we use first-order mapping to calculate the fitting error in Figs. 8(c) and 8(d). The results show the cumulative distribution functions of three distance measurement, i.e., the shortest path, the sum of latitude and longitude distance, and the area between two nodes in terms of latency estimation. Fig. 8(c) shows that using the area has a relative large estimation error. In the following, we will use the shortest path to estimate the latency, whose error rate is within 10% for 80% of nodes.

Fig. 9 gives the performance results where the nodes are picked randomly from the west coast of the United States as shown in Fig. 7(b). These nodes are connected through the main cable to each other, and thus, the network structure is an approximation line. The BF2 algorithm is the same as the BF1 algorithm except that it further uses the observation 1 to eliminate unnecessary errors. The BF2 algorithm always achieves the best performance, followed by the BF1, MM, and RD algorithms. Note that in Fig. 9(a), along with the increase of user number, the error of the MM algorithm increases significantly. However, the errors of BF1 and BF2 algorithms gradually converge. Fig. 9(b) shows that along with the increasing number of the group, the BF2 further reduces the maximum latency 40% maximum latency more than the MM algorithm. In Figs. 9(c) and 9(d), the amount of users with different preferences, i.e., 2 to 6, follows exponential distribution. Compared with these two figures, if there are many matchmaking flexibilities, the overall latency can be reduced. In Fig. 10(d), RD has the worst performance which demonstrates the necessity of matchmaking.

Fig. 10 shows the performance results of 4 algorithms in the general topology. Particularly, in Figs. 10(a) and 10(b),



(a) 6 groups (b) 60 users (c) 60 users (d) 60 users

Fig. 10. Performance comparison in the general topology.

each user has a uniform probability of having a preference in a group. The performance shows that BF has the best performance, followed by the MM algorithm. The NR and RD have similar performances in such an experimental setting. It is clear that when the total number of users is small, or the total number of group is large, the BF algorithm outperforms the MM algorithm. The reason is that in this case, a wrong matchmaking will increase the maximum latency significantly. In Figs. 10(c) and 10(d), the amount of users with different preferences, i.e., 2 to 6, follows the exponential distribution. The result shows that when there are many matchmaking flexibilities or there are only a few groups, the MM algorithm achieves similar performance with the BF algorithm.

VIII. Conclusion

This paper addresses optimal matchmaking in multi-party online applications, considering user's multiple preferences, to minimize the maximum latency between any pair of users. We prove that the proposed problem is NP-hard in the general topology. Then, we first prove that the proposed problem has a sub-modular property. In addition, we observe that the assignment priority exists and thus, we propose a revised greedy algorithm with an approximation bound and discuss its performance in the tree topology and the general topology. Finally, we discuss a distributed greedy algorithm which converges quickly. Extensive real trace-driven experiments demonstrate the effectiveness of proposed schemes.

References

[1] "Flurry five-year report: Its an app world. the web just lives in it." http://flurrymobile.tumblr.com/

[2] K. Jia, M. Kenney, T. Seppälä *et al.*, "Supercell joins a mobile internet giant: Fitting into tencents overall business strategy," The Research Institute of the Finnish Economy, Tech. Rep., 2017.

[3] https://en.wikipedia.org/wiki/King_of_Glory

[4] S. Agarwal and J. R. Lorch, "Matchmaking for online games and other latency-sensitive p2p systems," in *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 4, 2009, pp. 315–326.

[5] https://en.wikipedia.org/wiki/Battle_royale_game

[6] S. Wang, Y. Liu, and S. Dey, "Wireless network aware cloud scheduler for scalable cloud mobile gaming," in *IEEE ICC*, 2012.

[7] Y. Hu, D. Niu, and Z. Li, "A geometric approach to server selection for interactive video streaming," *IEEE Transactions on Multimedia*, vol. 18, no. 5, pp. 840–851, 2016.

[8] Y. Feng, B. Li, and B. Li, "Airlift: Video conferencing as a cloud service using inter-datacenter networks," in *Proceedings of the IEEE ICNP*, 2012.

[9] M. H. Hajiesmaili, L. T. Mak, Z. Wang, C. Wu, M. Chen, and A. Khonsari, "Cost-effective low-delay design for multi-party cloud video conferencing," *IEEE Transactions on Multimedia*, 2017.

[10] X. Chen, M. Chen, B. Li, Y. Zhao, Y. Wu, and J. Li, "Celerity: a low-delay multi-party conferencing solution," in *ACM ICME*, 2011.

[11] Z. Deng, S. Todorovic, and L. J. Latecki, "Unsupervised object region proposals for rgb-d indoor scenes," *Computer Vision and Image Understanding*, vol. 154, pp. 127–136, 2017.

[12] J. He, Y. Zhang, G. Huang, Y. Shi, and J. Cao, "Distributed data possession checking for securing multiple replicas in geographically-dispersed clouds," *Journal of Computer and System Sciences*, vol. 78, no. 5, pp. 1345–1358, 2012.

[13] B. Lim, E. K. Karrupiah, E. S. Lin, T. K. Phan, N. Thoai, E. Muramoto, and P. Tan, "Bandwidth fair application layer multicast for multi-party video conference application," in *IEEE CCNC*, 2009.

[14] C. Ly, C.-H. Hsu, and M. Hefeeda, "Improving online gaming quality using detour paths," in *ACM ICME*, 2010.

[15] C. Lumezanu, R. Baden, N. Spring, and B. Bhattacharjee, "Triangle inequality variations in the internet," in *ACM IMC*, 2009.

[16] K. Obraczka and F. Silva, "Network latency metrics for server proximity," in *IEEE GLOBECOM*, 2000.

[17] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: A decentralized network coordinate system," in *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4, 2004, pp. 15–26.

[18] T. H. Cormen, *Introduction to algorithms*. MIT press, 2009.

[19] J. Gu, P. W. Purdom, J. Franco, and B. W. Wah, "Algorithms for the satisfiability (sat) problem," in *Handbook of Combinatorial Optimization*. Springer, 1999, pp. 379–572.

[20] A. Krause and D. Golovin, "Submodular function maximization." 2014.

[21] K. L. Calvert, M. B. Doar, and E. W. Zegura, "Modeling internet topology," *IEEE Communications magazine*, vol. 35, no. 6, pp. 160–163, 1997.

[22] V. Ramasubramanian, D. Malkhi, F. Kuhn, M. Balakrishnan, A. Gupta, and A. Akella, "On the treeness of internet latency and bandwidth," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 37, no. 1, 2009, pp. 61–72.

[23] M. Chen, S. C. Liew, Z. Shao, and C. Kai, "Markov approximation for combinatorial network optimization," *IEEE transactions on information theory*, vol. 59, no. 10, pp. 6301–6327, 2013.

[24] H. Hindi, "A tutorial on convex optimization ii: duality and interior point methods," in *American Control Conference, 2006*. IEEE, 2006, pp. 11–pp.

[25] C. Lumezanu and N. Spring, "Measurement manipulation and space selection in network coordinates," in *IEEE ICDCS*, 2008.

[26] [Online]. Available: https://www.infobyip.com/ipbulklookup.php

[27] [Online]. Available: https://www.maxmind.com

[28] Y. Aikawa, Y. Hei, T. Ogishi, S. Niida, and T. Hasegawa, "Topology-aware clustering to achieve latency comparable to one-frame in multiplayer online games," in *IEEE ICC*, 2013.