LOM: 領隊導向多人連線遊戲自動匹配演算法

宋管翊 江振瑞
國立中央大學 資訊工程學系
jrjiang@csie.ncu.edu.tw

摘要

在多人連線遊戲(Multiplayer Online Games)中，自動匹配(matchmaking)的定義為將多位玩者(player)匹配在一起共同進行遊戲的程序。目前自動匹配的準則可以分為兩大類: Connection Based 與 Skill Based。前者將一群彼此網路互相連線速度較快的玩者匹配在一起;而後者則將一群彼此技能評分(Skill Rating)接近的玩者匹配在一起。本論文首先提出一項稱為 Association Based 的玩者匹配準則。在此準則下，關聯度(Association)較高的玩者被匹配在一起進行遊戲。並提出一個新的自動匹配演算法 LOM (Leader Oriented Matchmaking)來匹配系統中的玩者。LOM 使用到了 Minimum Cost Maximum Flow 演算法的概念，以最佳化的方式來匹配玩者。我們針對 LOM 與基本的貪婪式(greedy)玩者匹配演算法及隨機(randomized)玩者匹配演算法進行模擬與分析，以評估其效能。我們發現雖然 LOM 執行時間較長，但可產生最佳的平均玩者關聯度。

關鍵詞：多人連線遊戲、自動匹配、最小成本最大流量演算法、技能評分。

Abstract

In multiplayer online games (MOGs), matchmaking is a process that arranges players for online game sessions. Currently, there are two types of matchmaking criteria: connection-based and skill-based. In the connection-based criterion, players with higher mutual network connection speed are arranged together. In the skill-based criterion, players with close skill ratings are arranged together. In this paper, we propose a new criterion, association-based criterion, by which players with high association are arranged together. According to the proposed criterion, we also propose a new matchmaking algorithm, called LOM (Leader Oriented Matchmaking), using the concept of the minimum-cost maximum-flow algorithm to arrange players in an optimized way. We perform simulations and analyses for LOM, and compare it with two basic algorithms, the greedy and the randomized matchmaking algorithms. Although LOM has the longest execution time, it produces the best average association.

Keywords: Multiplayer online games, matchmaking, minimum cost maximum flow algorithm, skill rating.

1. 前言

多人連線遊戲(Multiplayer Online Games)是讓多位玩者透過網際網路(Internet)、區域網路(LAN)或是隨意網路(Ad-Hoc)連線在一起共同遊玩的一種遊戲模式。遊玩的平台可以是個人電腦(Personal Computer)、手持行動裝置(Handheld Device)或是遊戲主機(Video Game Console)。

自動匹配(Matchmaking)在多人連線遊戲的定義上，是將多位玩者(players)匹配在一起共同玩遊戲的一個過程。其興起和電視遊樂器開始支援網路連線功能有著密不可分的關係，與傳統 PC game 從伺服器瀏覽器中點選自己想進入的伺服器相比，直接點選一顆按鈕即能找到其他玩者的自動匹配明顯花費較少的力氣進行遊玩，也因此適用在電視遊樂器中較不方便的視窗瀏覽操作上。故自動匹配有取代伺服器瀏覽器的趨勢。現在已經有許多 PC game 也開始支援自動匹配的功能，並且也有許多學術研究開始探討自動匹配議題[1][3][6][7][8]。

自動匹配依照其匹配玩者的準則可以分為兩大類: Connection Based 與 Skill Based。前者將一群連線速度相近的玩者們匹配在一起遊玩，故其匹配玩者的準則為玩者彼此間互相連線的速度。後者付予每位玩者一個用來代表該玩者實力強弱的數值，稱為技能評分(Skill Rating)[2][5][9]。在匹配玩者時，自動匹配將技能評分相近的玩者匹配在一起比賽，故其匹配玩者的準則為玩者彼此間技能評分的差異。

本論文提出一個新的玩者匹配準則，並依此準則提出一項新的自動匹配演算法來匹配系統中所有的玩者。新提出的玩者匹配準則稱為 Association Based。在此準則下，關聯度(Association)較高的玩者被匹配在一起進行遊戲，而關聯度值得大小則代表兩位玩者間的關聯程度高低。新提出的自動匹配演算法稱為 LOM (Leader Oriented Matchmaking)，利用最小成本為大流量(Minimum Cost Maximum Flow)演算法[4]的概念，以最佳化的方式，將關聯度最高的玩者匹配在一起。值得一提的是，Connection Based、Skill Based 與 Association Based 準則，皆可套用於此演算法中去匹配玩者，並皆可依各自的需求得到最佳的輸出結果。套用在 Association Based 準則，它可以幫系統中的每一位玩者匹配出最佳的團隊出來。套用在 Connection Based 準則，它可以最佳化系統中每一位玩者的網路連線速度。套用在 Skill Based 準則，它可以讓系

統中同場遊戲玩者間的技能評分差距最小。

本論文其他部份的內容安排如下: 我們在第二節中介紹相關的研究，第三節中描述問題的定義並在的第四節中提出解決的方法。第五節則是呈現相關方法的模擬結果，而第六節則總結全篇論文。

2. 相關研究

我們在本節中回顧一下過去有關於多人連線遊戲之 Matchmaking 的相關研究。

在 2009 年，Microsoft Research 的 Sharad Agarwal 與 Jacob R. Lorch 於 SIGCOMM '09 發表了一篇論文[1]，它的主題集中在 Connection based 的匹配玩者準則。提出了一個名為 Htrae 的系統，它為一基於地理位置資訊的網路延遲預測系統 (Geographical Based Network Latency Pre-diction System)，用來為預估玩者間的網路延遲大小。

在 2011 年，Justin Manweiler, Sharad Agarwaly, Ming Zhangy, Romit Roy Choudhury 和 Paramvir Bahly 於 MobiSys11 發表了 Switchboard 系統 [3]，它提出了一個適用於手持行動裝置的 Matchmaking 系統架構。並做了許多實驗探討了不同基地台間之手持行動裝置在進行 P2P 遊戲時的網路延遲大小。

在 2012 年，Olivier Delalleau, Emile Contal, Eric Thibodeau-Laufer, Raul Chandias Ferrari, Yoshua Bengio 與 Frank Zhang 針對 Skill based 匹配玩者準則加以改進[3]，對於網路連線速度方面他是假設所有玩者都可以連線到延遲大小對他而言較低的伺服器，故此論文不考慮玩者的網路連線速度，只考慮玩者間實力差距的平衡。

3. 問題定義

假設系統中有 n 個玩者，分成若干隊，每一隊人數皆為 i 人，$1 < i \le \lfloor n/2 \rfloor$。我們先選出 k 位玩者當隊長，並令隊長(Leader)所成的集合為$\mathcal{L}$，$\mathcal{L} = \{\ell_1, \ell_2, \ell_3, \dots, \ell_k\}$。再取系統中的隊伍數為 k 個，並令隊伍所成之集合為 G，$G=\{\mathcal{g}_1, \mathcal{g}_2, \mathcal{g}_3, \dots, \mathcal{g}_k\}$，其中 $k=\lfloor n/i \rfloor$。剩下的$n-k$位玩者稱為隊員(Member)，令隊員所成的集合為$\mathcal{M}$，$\mathcal{M} = \{m_1, m_2, m_3, \dots, m_{n-k}\}$，每位隊長各需在其中選出 i-1 位玩者來當自己的隊員。這也代表會有 n mode i 位屬於$\mathcal{M}$集合的玩者被排除在外，不屬於任一隊。在此定義隊員和隊長間存在一個權重值，代表他們倆間關聯度之高低、稱為 Association Factor，簡稱$AF_{\ell_\alpha m_\alpha}$，值愈低代表關聯度愈高。而在 Skill Based 準則中，此權重值代表他們倆間技能評分之差值、在 Connection Based 準則中，它代表他們倆間網路連線延遲大小，由於本論文提出此演算法之最主要目的是去解如何以 Association Based 準則來匹配玩者之議題，故接下來我們統一使用$AF_{\ell_\alpha m_\alpha}$來稱呼此權重值，$\forall \ell_\alpha \in \mathcal{L}$，$\forall m_\alpha \in \mathcal{M}$。假設目前系統中存在一個隊伍$\mathcal{g}_A \in G$，此隊有 1 個隊長$\ell_A \in \mathcal{L}$及 j 個隊員$\{m_{A1}, m_{A2}, m_{A3}, \dots, m_{Aj}\} \subseteq \mathcal{M}$，則我們在此定義 $\mathcal{g}_A$ 隊的綜合權重值為 $AF_{\mathcal{g}_A}$，其中$AF_{\mathcal{g}_A}=\sum_{k=1}^{j} AF_{\ell_A m_k}$。此問題定義之圖解如圖 1 所示。為了能讓系統中的每位隊長都找到最適合他的隊員，我們的目標為：1.$\forall \mathcal{g}_\alpha \in G$，$|\mathcal{g}_\alpha|=i$。2.最小化 $\sum_{\mathcal{g}_\alpha \in \mathcal{T}} AF_{\mathcal{g}_\alpha} \Big/ k*(i-1)$，$\forall k$=系統中的隊伍數，i=每一隊需要的人數，之所以要除 k*(i-1)是為了去突顯 Minimum Cost Maximum Flow 定義中 minimum cost “per” flow 的概念。

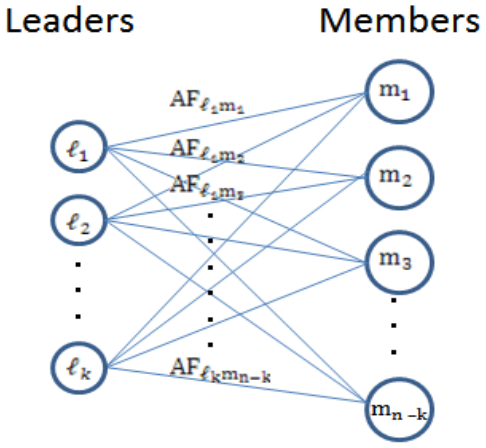

圖 1 問題定義示意

4. 提出方法

我們將上段導出的模型轉換成適合套用 Minimum Cost Maximum Flow 演算法的型式。首先新增兩個虛擬結點 S 和 T，並將 S 與$\mathcal{L}$集合中所有結點相連，T 與$\mathcal{M}$集合中所有結點相連。再來付予圖中的點α一個權重值 d(α)，並重新定義邊上的權重為二維向量$(CP_{\alpha\beta}, AF_{\alpha\beta})$。我們可以將$CP_{\alpha\beta}$看成邊$\overrightarrow{\alpha\beta}$上的容量、$AF_{\alpha\beta}$看成流過邊$\overrightarrow{\alpha\beta}$的成本，$\forall \alpha, \beta \in S \cup \mathcal{L} \cup \mathcal{M} \cup T$。其中$CP_{S\ell}$=i-1，$AF_{S\ell}$=0，$CP_{\ell m}$=1。$CP_{mT}$=1，$AF_{mT}=0$。在演算法的輸出結果中若$\ell$和 m 之間有流量，則代表隊長$\ell$和隊員 m 被匹配到同一隊伍中，若沒有，則代表他們不在同一隊，$\forall \ell \in \mathcal{L}$，$\forall m \in \mathcal{M}$。如圖 2 所示，我們稱此圖為$\mathcal{N}$。

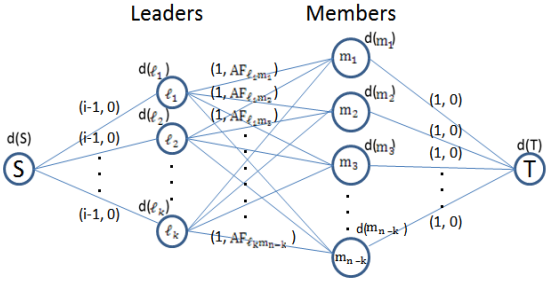

圖 2 $\mathcal{N}$的圖解

有了$\mathcal{N}$之後，我們可以將它視為 Minimum Cost Maximum Flow 問題。此問題有各種不同的演算法可以解。在本論文中，我們採用 Algorithm Design : Foundations, Analysis and Internet Examples[4]一書中 Chapter 8.4 提出之 Successive Shortest Path Algorithm。詳細流程如下：

Algorithm : MinCostFlow

***Input*:** Weighted flow network $\mathcal{N}$

***Output*:** A maximum flow with maximum cost f_{ST} for $\mathcal{N}$

```
1: Initialize flows of all edges to 0
2: Initialize weights of all vertex to 0
3: Repeat
4:    Compute the weighted residual network Rf
5:     for each edge uv ∈Rf do
6:       AF_uv ← AF_uv+ d(u) - d(v)
7:     end for
8:     Run Dijkstra's algorithm on Rf using the
weights AF'_uv
9:     for each vertex α ∈ N do
10:       d(α)←Distance ofαfrom S in Rf
11:    end for
12:    if d(t) < +∞ then
13:       Δ← ∞
14:       for each edge αβ ∈ π do
15:          if Δ_f(αβ) <Δ then
16:             Δ← Δ_f(αβ)
17:          end if
18:       end for
19:       for each edge αβ ∈ π do
20:          if αβis a forward edge then
21:             f_αβ ← f_αβ+Δ
22:          else if αβ is a backward edge
23:             f_αβ ← f_αβ-Δ
24:          end if
25:       end for
26:    else
27:       stop← true
28:    end if
29: until stop
```

在解釋該演算法如何運作前，我們先說明何謂剩餘網路(Residual Network)。首先定義何謂剩餘容量(Residual Capacity)。假設$\mathcal{N}$為一網路圖，$\forall\overrightarrow{\alpha\beta} \in \mathcal{N}$，則我們定義$\overrightarrow{\alpha\beta}$之剩餘容量$\Delta_f(\overrightarrow{\alpha\beta})=CP_{\alpha\beta}-f_{\alpha\beta}$，$\overrightarrow{\beta\alpha}$之剩餘容量$\Delta_f(\overrightarrow{\beta\alpha})=f_{\alpha\beta}$，其中$CP_{\alpha\beta}$為邊$\overrightarrow{\alpha\beta}$的容量，$f_{\alpha\beta}$為邊$\overrightarrow{\alpha\beta}$的網路流量。假設現在有一空圖 R$f$，若$\exists\overrightarrow{\alpha\beta} \in \mathcal{N}$使得$\Delta_f(\overrightarrow{\alpha\beta})\neq 0$，且$\overrightarrow{\alpha\beta} \notin$ Rf，則我們在圖 Rf 中加上邊$\overrightarrow{\alpha\beta}$與邊$\overrightarrow{\beta\alpha}$，並將其權重值設為$\Delta_f(\overrightarrow{\alpha\beta})$與$\Delta_f(\overrightarrow{\beta\alpha})$，直到$\mathcal{N}$中所有剩餘容量不為 0 的邊都存在於 R$f$ 為止。最後的 Rf 即為$\mathcal{N}$之剩餘網路，它顯示了$\mathcal{N}$中每條邊可用的流量剩多少。

Successive Shortest Path Algorithm 大致的作法為，在每一回合中，從剩餘圖中找出一條從 S 到 T，邊上權重值總和最小的路徑，並延著它送出一條最大並且可容納於此邊之容量的網路流。再來更新剩餘網路上的資訊，更新的方法為，假設邊$\overrightarrow{\alpha\beta}$有一條網路流流過，$\forall\overrightarrow{\alpha\beta} \subseteq \mathcal{N}$，$\overrightarrow{\alpha\beta}$之成本為$AF_{\alpha\beta}$、容量為$CP_{\alpha\beta}$，則我們在剩餘網路中將$CP_{\beta\alpha}$加一、$CP_{\alpha\beta}$減一，並將$AF_{\beta\alpha}$設為負$AF_{\alpha\beta}$。在每一回合中重複作相同的事，直到無法從剩餘網路中找到任何可以從 S 通到 T 的路徑為止。而根據在剩餘網路中尋找最短路徑的找法，此演算法可分為兩種版本，運算時間複雜度 $O(|f_{ST}|m\log n)$的 Dijkstra 版本與運算時間複雜度 $O(|f_{ST}|mn)$的 Bellman-Ford 版本，其中$|f_{ST}|$=網路流量數，m=邊數，n=點數。Bellman-Ford 版本可以處理權重值為負的邊，而 Dijkstra 版本沒辦法。但是我們可以去調整邊上的權重值使之為正。調整方法為，先為剩餘圖上每一點α賦予一權重值 d(α)，它代表在剩餘圖中從 S 到α節點需經過的邊數，並在每一回合演算法去更新剩餘圖中的資訊時，將每邊的成本AF_{uv}調整為AF'_{uv}，其中$AF'_{uv} = AF_{uv} + d(u) - d(v)$，這樣就能保證每邊的成本為正。由於時間複雜度較低之關係，我們選用 Dijkstra 版本並去調整邊上的權重值使之為正。

5. 模擬

在本章中，我們以 Association Based 準則為例，讓我們的自動匹配演算法 LOM (Leader-Oriented Matchmaking)，去和其它自動匹配演算法進行比較。比較的方式為執行結果的好壞與執行時間之快慢。比較的對象分別為 M2L Greedy、L2M Greedy 和 Random。M2L Greedy 即為第一章提到的 Greedy 法，每一位隊員會去輪流檢查每一位隊長，並選出關聯度最佳的與之相連。L2M 為 M2L 的變形，作法為每一位隊長輪流去檢查每一位隊員，再選出關聯度最佳的與之相連。Random 法，假設系統中有 k 個隊長，每一隊需要 i 人。我們隨機選出 k*(i-1)位隊員，並將之切成 k 個區間，每個區間的玩者匹配給屬於各自區間的隊長。至於在 Association Factor 的定義方面，在一般情況中，系統為了求出此值，通常會先去取出每個玩者的個人資料(Profile)與記錄檔(Log)，並依此計算出玩者間的差異性。由於較小的差異性代通常表著較接近的關係，故我們在此定義 Association Factor 值愈小代表關聯度愈佳。

我們使用 C++來實作自動匹配演算法的模擬程式，並選定每一隊人數為 10 人，系統中等待遊玩的玩者數量設從 100 人到 500 人。每一次固定選前 k(同上一章所描述的 k)名玩者當隊長，剩下沒被選中的玩者當隊員。再來在所有隊長和隊員間隨機產生介於 1 到 100 間的關聯度。再來讓本論文提出的演算法 LOM 與其它三種演算法分別下去執行，比較輸出結果。在圖 3 中，橫軸的部份代表系統中

玩者的數量，縱軸的部份代表系統每位玩者的平均關聯度大小。

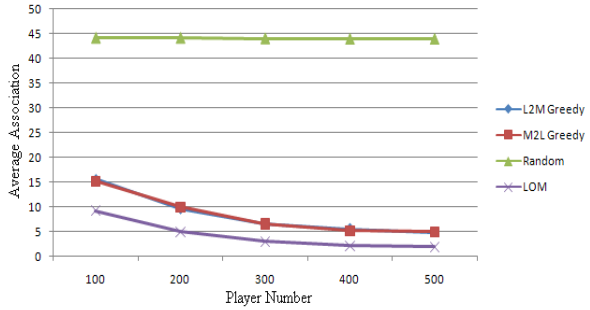

圖 3 各種自動匹配演算法輸出結果比較，其中橫軸的部份代表系統中玩者的數量，縱軸的部份代表系統每位玩者的平均關聯度大小

由圖 3 可以看出當使用 L2M Greedy、M2L Greedy 與 LOM 來匹配玩者時，它們的曲線有開口朝上，緩慢下滑的趨勢。原因在於，當系統中的玩者數量增多時，隊長與隊員之間的邊數隨之增多，進而代表權重值(關聯度)特別高和特別低的邊，其出現次數變多。又由於這三種演算法會專門去挑選權重值特別低的邊。導致系統中玩者愈多時，被挑中之低權重值的邊也愈多，造成整體系統的權重值之平均下降。

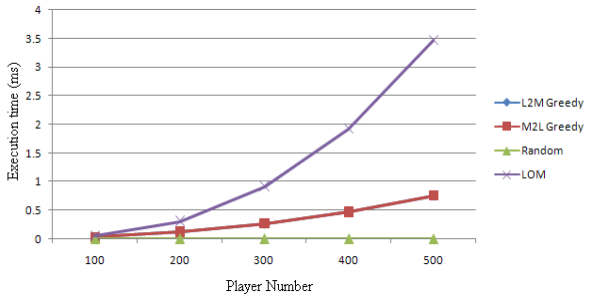

圖 4 各演算法的執行時間模擬結果

圖 4 為各演算法的執行時間模擬結果，橫軸為系統中玩者數量，縱軸為執行時間，單位為毫秒(millisecond, MS)。執行本模擬之電腦的規格為：AMD Phenom(tm) 9650 Processor 2.41 GHZ、2.00 GB Memory、Windows XP32 bits。雖然在執行時間之快慢的比較上，本論文提出的自動匹配演算法是需要較多時間的，但是因為採用多項式執行時間複雜度(polynomial time complexity)的 Minimum Cost Maximum Flow 演算法的概念解決問題，因此執行時間仍然在可以接受的範圍內，而且隨著電腦硬體性能快速的提升，以及近幾年非常熱門的雲端運算技術，相信此問題可以在這些因素的幫助下被克服。

6. 結論

目前自動匹配有兩大匹配玩者的準則，Connection Based 與 Skill Based 準則，前者將一群網路互相連線速度快的玩者匹配在一起進行遊戲，後者將一群彼此技能評分接近的玩者匹配在一起進行遊戲。但是這兩種準則都是以找出適合當自己對手的玩者為目標，而忽略了在多人連線遊戲中玩者的另一種互動模式，合作。故本論文額外提出了一種新的匹配玩者準則，稱為 Association Based。在此準則下，一群彼此關聯度的玩者被匹配在一起進行遊戲，可以大大的增加玩者間團隊合作的樂趣。

為了去解如何以 Association Based 準則來匹配玩者之議題，本論文提出了一個自動匹配演算法，稱為 LOM (Leader Oriented Matchmaking)。它使用到了 Minimum Cost Maximum Flow 演算法，以最佳化的方式去匹配系統中的玩者。值得一提的是目前現存的兩大匹配玩者準則，Connection Based 與 Skill Based，亦可套用於 LOM 中去匹配玩者，並皆可依各自的需求得到最佳的輸出結果。套用在 Association Based 準則，它可以幫系統中的每一位玩者匹配出最佳的團隊出來。套用在 Connection Based 準則，它可以最佳化系統中每一位玩者的網路連線速度。套用在 Skill Based 準則，它可以讓系統中同場遊戲玩者間的技能評分差距最小。故本論文提出的自動匹配演算法可以針對不同的匹配玩者準則，做到最佳的輸出結果。進而最佳化系統中每一位玩者在進行多人連線遊戲時的體驗。

在模擬中，我們以 Association Based 準則為例，讓我們的自動匹配演算法 LOM，去和其它自動匹配演算法進行比較。比較的方式為執行結果的好壞與執行時間之快慢，在執行結果好壞的比較中，我們驗證了 LOM 的執行結果優於其它演算法。雖然在執行時間之快慢的比較上，LOM 是最慢的，但是隨者電腦硬體性能快速的提升，以及近幾年非常熱門的雲端運算技術，相信此問題可以在這些因素的幫助下被克服。

未來，我們希望能探討在 Association Based 匹配玩者的準則下，應該如何更精確的利用玩者的個人資料(Profile)與記錄檔(Log)上的資料，來計算隊長和隊員間的關聯度。以求能更真實的反映出他們之間的團隊默契。

參考文獻

[1] S. Agarwal, and J. R. Lorch, “Matchmaking for online games and other latency-sensitive P2P systems,” in Proc. of ACM SIGCOMM Conference on Data Communication, pp. 315-326, 2009.

[2] R. Coulom, “Whole-history rating: A Bayesian rating system for players of time-varying strength, “ in Proc. of the 6th International Conference on Computers and Games (CG ’08), pp. 113–124, 2008.

[3] O. Delalleau, E. Contal, E. Thibodeau-Laufer, R. C. Ferrari, Y. Bengio, and F. Zhang, “Beyond skill rating: advanced matchmaking in ghost recon online,” IEEE Trans. on Computational Intelligence and AI in Games, Vol. 4, Issue. 3, pp. 167-177, Sep. 2012.

[4] M. T. Goodrich and R. Tamassia, Algorithm Design: Foundations, Analysis, and Internet Examples, Baker & Taylor Books, 2001.

[5] R. Herbrich, T. Minka, and T. Graepel, “TrueSkill: a bayesian skill rating system,” Advances in Neural Information Processing Systems, pp. 569–576, 2006.

[6] J. Jimenez-Rodriguez, G. Jimenez-Diaz, and B. Diaz-Agudo, “Matchmaking and case-based recommendations,” in Proc. of Workshop on Case-Based Reasoning for Computer Games, the 19th Int. Conf. on Case Based Reasoning, pp. 53–62, 2011.

[7] J. Manweiler, S. Agarwaly, M. Zhangy, R. R. Choudhury, and P. Bahlym, “Switchboard: a matchmaking system for multiplayer mobile games,” in Proc. of the 9th International Conference on Mobile Systems, Applications, and Services (MobiSys '11), pp. 71-84, 2011.

[8] J. Riegelsberger, S. Counts, S. Farnham, and B. C. Philips, “Personality matters: incorporating detailed user attributes and preferences into the matchmaking process,” in Proc. of the 40th Annu. Hawaii Int. Conf. on System Sciences, pp. 87-87, 2007.

[9] R. C. Weng, and C.-J. Lin, “A Bayesian approximation method for online ranking,” Journal of Machine Learning Research, pp. 267–300, 2011.