User Response Based Recommendations: A Local Angle Approach

Kavitha Veeraruna, Salman Memon, Manjesh Hanawal, Eitan Altman, R Devanand

User Response Based Recommendations: A Local Angle Approach

VEERARUNA KAVITHA, SALMAN MEMON, MANJESH K. HANAWAL, EITAN ALTMAN[!] and DEVANAND R
IEOR, IIT Bombay, India, [!]INRIA France

***Abstract*—When a user interested in a service/item, visits an online web-portal, it provides description of its interest through initial search keywords. The system recommends items based on these keywords. The user is satisfied if it finds the item of its choice and the system benefits, otherwise the user explores an item from the list. Usually when the user explores an item, it picks an item that is nearest to its interest from the list. While the user explores an item, the system recommends new list of items. This continues till either the user finds its interest or quits. In all, the user provides ample chances and feedback for the system to learn its interest. The aim of this paper is to exploit the user-generated responses in the same session. One can further utilize the history (e.g., previous user ratings) to design good recommendation policies.**

We develop algorithms that efficiently utilize user responses to recommended items and find the item of user's interest quickly. We first derive optimal policies in the continuous Euclidean space and adapt the same to the space of discrete items. In the continuous Euclidean space, the optimal recommendations (e.g., with two recommendations) at the same time step are at 180 degrees from each other, while are at 90 degrees with respect to the ones at the previous time step. We propose the notion of local angle in the space of discrete items and develop user response-local angle (UR-LA) based recommendation policies. We compared the performance of UR-LA with widely used collaborative filtering (CF) based policies on two real datasets and showed that UR-LA performs better in majority of the test cases. We also proposed a hybrid scheme that combines the best features of both UR-LA and CF (and history) based policies, which outperforms them in most of the cases.

I. INTRODUCTION

Recommendation systems (RS) became an active research area after the research on collaborative filtering in mid-1990s, (e.g., [5]-[8]). Over the past decade recommendation engines have become ubiquitous, and used for various purposes, e.g. e-commerce, social and professional networks, such as YouTube, Pandora, Amazon etc., (e.g., [6]). Even after plenty of work done by the industry and academia to develop new methods for RS in the last decade, this research area still has lots of scope to explore practical applications ([9]).

In a RS, an user starts a session looking for an item (unknown to system) and its interest can vary drastically from one session to another. For example, a user might visit an online portal to buy a book at one time, while the same user might be interested in a mobile the next time. However, the interest of the user during the same session would be consistent and the partial information about the same is available via the user responses. The objective is to satisfy the user's requirement at the earliest using the responses of the same user to the previous recommendations in the same session.

At every step system provides a recommendation-list, and user chooses one among them to explore. It is natural for the user to explore an item, among the recommended list, which best matches the item of its interest. The exploration-recommendation continues till the user finds the item of its interest, or quits. We refer to the sequence of steps between a user entering and leaving the system as a session. This sequence of items that the user explores provides useful feedback about the user's hidden (to the system) interest. In such scenarios it is better suited to recommend 'faraway' items (from each other as well as from the current item being explored) at least initially, as opposed to the traditional methods of recommending the 'nearby' items. We derive such policies using the policies of continuous Euclidean space, where the optimal policy turns out to be recommending items at each time step that are at 180 degrees from each other (two recommendation at a time), while are at 90 degrees with respect to the ones at the previous time step. We adapt the continuous-space policies to discrete space by proposing a new 'notion of angle' for a discrete set. While translating 180 degrees, we would choose items among those which are at the same distance from a reference point, but whose sum of inter-distances is maximum. This results in recommending the 'faraway' items, however the distances reduce progressively. Basically this narrows down the system's belief of users interest geometrically faster.

We consider the similarity based distance between items obtained using the history of previous ratings of different items (e.g., [13]). Most of the works in the literature focus on finding the best one-shot recommendation and ignore the feedback generated within a session. However, we use this feedback (of the same session) to comeup with a next recommendation for the user. In [1], [2] authors consider prompting the new users for the purpose of learning them and for subsequent improved person based recommendations. They discuss the set of items which are optimal in certain information theoretic approach to learn the user, while ours uses the history as well as the user responses to learn the user of current session.

Our approach can also help cold start problems ([2], [3], [4]), in particular a new user problem, by using a non-history dependent distance measure, e.g., distance defined using the features of the items and we consider this for future reasearch.

We propose a hybrid algorithm that combines the main ideas of UR-LA and CF policies. We tested our policies on two

real datasets ([10], [11]), and they outperform the CF policy in most of the cases. The improvement is significant, more than 30% in some, and at least 15% in many cases.

II. System Dynamics

We consider a large (finite) database $\mathcal{S}$ where each item is described by a number of defining features. For example, in a music video, singer, composer, instruments etc., can be the features. For an item in an e-commerce portal like Amazon, Flipkart etc, type, make, category, price etc., are the features. The similarities/dis-similarities between the items can either be obtained by comparing these features and or by comparing the ratings provided by various users of the system. We assume there exists a distance metric $d(v_1, v_2)$ for any pair of items (v_1, v_2), which captures their similarity. It can be a well known similarity measure based distance between the two items ([13]), or can be proportional to the number of matching features between them, or can be a combination of the two.

When a user visits such a system, it is interested in one of the items referred by V_{ref}, which is unknown to the Content Provider (CP). We assume that V_{ref} is equally likely to be one of the items available with CP. The user specifies its interest by an initial search query and the system generates a set of recommendations based on this search query. User is satisfied if any one of the suggested recommendations (say a) is close to V_{ref}, i.e., if the distance $d(V_{ref}, a) \leq \underline{r}$ for some $\underline{r} \geq 0$ and then the CP derives benefit. For example, the user is satisfied if a has at least $\overline{F}$ number of matching features with V_{ref}. The satisfaction radius $\underline{r}$ can be zero, which implies the user is interested only in a particular product. The user starts a session with CP using an item X_0 (obtained after a search query) only if there is at least a slight match, i.e., only if the distance $d(X_0, V_{ref}) \leq \bar{R}$, where $\bar{R}$ is larger than $\underline{r}$. Threshold $\bar{R}$ can be the largest possible distance between any two items of the system, in which case the user always starts the session.

The CP displays a list of M recommendations (call it $\mathbf{a}_1 = (a_{1,1} \cdots, a_{1,M})$, a vector of length M) while displaying the item X_0. The user, if not satisfied with X_0, explores one of the suggestions (call it X_1) from among $\mathbf{a}_1$. CP while displaying item X_1 also suggests new recommendation list $\mathbf{a}_2$. The user, if not satisfied with X_1, chooses one among $\mathbf{a}_2$ and this continues. That is, the user navigates through the recommendations of the CP and this the user does for maximum T number of steps. We assume that at every step, the user chooses one of the items recommended by the CP which is closest to V_{ref}. That is,

$$X_k \in \arg\min_m d(V_{ref}, a_{k,m}). \tag{1}$$

In case an item recommended via initial/subsequent recommendations satisfies the user, the CP benefits and we assume the benefit is inversely proportional to the search time. The faster the user is satisfied, the more is the benefit to the CP. Our aim is to minimize the time taken to suggest an item that satisfies the user.

III. Problem formulation

We denote the space of items by $\mathcal{S}$, V_{ref} is one among $\mathcal{S}$ and a distance measure d is defined on this set. Our analysis is also applicable to the case when $\mathcal{S}$ is an (continuous) Euclidean subset. We could handle this generalization without extra effort and, more importantly, we derive recommendation policies for discrete content space by translating optimal policies of a continuous space.

A user can be interested in any item in $\mathcal{S}$. We consider that V_{ref} is selected uniformly at random from $\mathcal{S}$, i.e., the probability that V_{ref} lies in a subset $\Lambda \subset \mathcal{S}$ is given by:

$$P(V_{ref} \in \Lambda) = \frac{\mu(\Lambda)}{\mu(\mathcal{S})},$$

where μ is the Lebesgue measure (length, area respectively in one and two dimensional spaces) in the continuous case and is the cardinality (number of elements in a set) if $\mathcal{S}$ is discrete.

Distance measures: One can define distance between two items in a discrete space using a similarity measure, which in turn is obtained by the ratings provided by different users to different items. The Cosine similarity measure between two items v, u is defined as ([13]):

$$s_c(v,u) = s_{v,u} = \frac{\sum_{i \epsilon I_{v,u}} r_{v,i} r_{u,i}}{\sqrt{\sum_{i \epsilon I_{v,u}} (r_{v,i})^2} \sqrt{\sum_{i \epsilon I_{v,u}} (r_{u,i})^2}}, \tag{2}$$

where $I_{v,u}$ is the set of users that rated both the items v and u, $r_{v,i}$ is the rating of item v by user i.

In continuous Euclidean spaces, d is either Euclidean or L^1 distance. In this paper we consider L^1 distance, $d(v_1, v_2) := \sum_i |v_{1,i}, v_{2,i}|$, where $\{v_{1,i}\}_i$ are components of the item defining vector v_1. One can now define the balls around an item/point v by: $\mathcal{B}(v, \underline{r}) := \{x \in \mathcal{S} : d(x, v) < \underline{r}\}$.

We assume that any ball with satisfaction radius has same measure, i.e., $\mu(\mathcal{B}(v, \underline{r}))$ is same irrespective of the centre v. In general we can use any distance measure that is appropriate to measure similarity/dissimilarity of items in a given context and our approach works.

Time to satisfy the user

The user's interest lies in any item inside the ball $\mathcal{B}(V_{ref}, \underline{r})$ while the initial search X_0 lies inside $\mathcal{B}(V_{ref}, \overline{R})$. Recall that V_{ref} is unknown to CP, however, it knows that $V_{ref} \in \mathcal{B}(X_0, \overline{R})$. The problem is to find a strategy that optimizes the time to satisfy the user. Let τ represent the first time a recommendation is within the ball $\mathcal{B}(V_{ref}, \underline{r})$, i.e., using (1) and with $x \wedge y := \min\{x, y\}$,

$$\begin{aligned} \tau :=& \quad \inf\left\{k \geq 1 : a_{k,i} \in \mathcal{B}(V_{ref}, \underline{r}) \text{ for some } i \leq M\right\} \wedge T \\ =& \quad \inf\left\{k \geq 1 : X_k \in \mathcal{B}(V_{ref}, \underline{r})\right\} \wedge T. \end{aligned}$$

Note that we are only given T chances and are interested only if $\tau \leq T$ and hence the above definition is sufficient. Our aim is to minimize the expected value of the hitting time, $E[\tau]$, or maximize the time spent in the system, i.e. $E[T-\tau]$,

$$\arg\min_\pi E[\tau] \equiv \arg\max_\pi E[T - \tau] \tag{3}$$

where the optimization is over all possible recommendation policies π, that would be described shortly.

Recommendation Policies and Markov state

This problem is a fixed horizon problem, and the time step is represented by the time epoch at which the user has chosen a new item to explore. The recommendations depend upon the user response to the previously recommended items. Thus policy would be an M-tuple of recommendations (actions), one for each time step k as below:

$$\pi = \{(a_{1,1}, \cdots, a_{1,M}), \cdots, (a_{T-1,1}, \cdots, a_{T-1,M})\}. \quad (4)$$

Here $a_{k,i}$ represents the i-th recommendation for the k-th step and this depends upon an appropriate state of the system. The state of the system at time k is given by (X_k, V_{ref}), where X_k defined in equation (1) is the choice of user at time k and V_{ref} is the interest of the user. Since V_{ref} is unknown to CP, we consider the alternate state $Z_k = (X_k, B_k)$, inspired by Partially Observable MDP (POMDP) framework. Here belief random variable B_k is the conditional distribution of V_{ref} given history H_k (previous states, actions and current observation) defined by:

$$H_k = \{(X_0, B_0), (X_1, B_1), ...(X_{k-1}, B_{k-1}), \mathbf{a_1}, ...\mathbf{a_k}, X_k\}.$$

The belief only improves with time, in the sense for all k, B_k is concentrated on certain sets $\mathcal{A}_k \subseteq \mathcal{A}_0$, in the following manner. We provide the proof and alongside introduce the required notations. To keep notations short, we present our results for $M = 2$. These can easily be extended for $M > 2$.

***Lemma** 1:* The belief random variable B_k at any time step k depends only upon the previous belief B_{k-1}, current recommendation $\mathbf{a_k}$ and current user choice X_k. For each k, $B_k \sim \mathcal{U}(\mathcal{A}_k)$, which implies that B_k is uniformly distributed over a subset $\mathcal{A}_k \subset \mathcal{S}$. Also the subsets $\{\mathcal{A}_k\}_{k \le T}$ are nested:

$$\mathcal{S} = \mathcal{A}_0 \supset \mathcal{A}_1 \cdots \supset \mathcal{A}_{T-1} \supset \mathcal{A}_T \text{ almost surely.}$$

Proof: Fix any deterministic policy π. We prove below (with $M = 2$) that the policy is represented by 'centres'

$$\mathcal{Q}_\pi := \{a_i^k, 1 \le k \le T \text{ and } i \le 2^k\}.$$

For step $k = 1$, $a_{1,1} = a_1^1$ and $a_{1,2} = a_1^2$ represent the two recommendations provided by the system. If the user's interest V_{ref} lies in either one of the balls around the recommendations $\mathcal{B}_i^1 := \mathcal{B}(a_i^1, \underline{r})$ with $i = 1$ or 2, the user is satisfied in the first step itself. If not user chooses one among them (whichever is closer to V_{ref}), which is X_1, and we then have a partition of the area of the first belief, $\mathcal{A}_1^0 := \mathcal{A}_0 = \mathcal{S} \cap B(X_0, \bar{R})$:

$$\begin{aligned} X_1 &= a_{1,1} 1_{\{V_{ref} \in \mathcal{A}_{11}\}} + a_{1,2} 1_{\{V_{ref} \in \mathcal{A}_{12}\}}, \text{ where} \\ \mathcal{A}_{11} &= \{v \in \mathcal{A}_0 \mid d(v, a_{1,1}) < d(v, a_{1,2})\} \text{ and} \\ \mathcal{A}_{12} &= \{v \in \mathcal{A}_0 \mid d(v, a_{1,2}) < d(v, a_{1,1})\}. \end{aligned} \quad (5)$$

By Lemma 2 of Appendix A the new belief B_1 is uniformly distributed over $\mathcal{A}_1$ (i.e., $B_1 \sim \mathcal{U}(\mathcal{A}_1)$), which is either one of the above two mentioned partitions, i.e.,

$$\mathcal{A}_1 = \mathcal{A}_1^1 \text{ or } \mathcal{A}_2^1, \text{ where } \mathcal{A}_1^1 := \mathcal{A}_{11} \text{ and } \mathcal{A}_2^1 := \mathcal{A}_{12}.$$

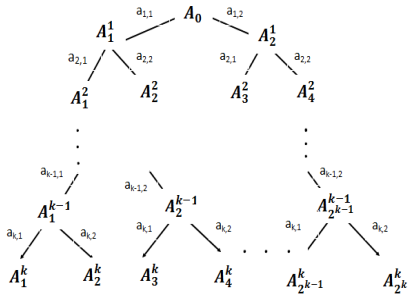

Fig. 1. Nested Notations

Given the feedback by user, X_1, the system proposes two more recommendations $a_{2,1}(X_1, B_1)$ and $a_{2,2}(X_1, B_1)$. These two recommendations depend upon the current state (X_1, B_1) and since B_1 is fixed by X_1 and $\mathcal{A}_0$ (by Lemma 2) it is actually specified only by X_1. Let a_1^2, a_2^2 be these two recommendations if user choice $X_1 = a_{1,1}$, and let a_3^2, a_4^2 be the two recommendations if $X_1 = a_{1,2}$, i.e.,

$$(a_{2,1}(X_1),\ a_{2,2}(X_1)) = \begin{cases} (a_1^2, a_2^2) & \text{if } X_1 = a_{1,1} \\ (a_3^2, a_4^2) & \text{else.} \end{cases}$$

Again the user is satisfied, if V_{ref} lies in one among the two balls, $\mathcal{B}(a_{2,1}(X_1), \underline{r})$ or $\mathcal{B}(a_{2,2}(X_1), \underline{r})$. If not, user again chooses one among the two recommended items to explore further, whichever is closer to V_{ref}. This is X_2, and it equals

$$X_2 = a_{2,1}(X_1) 1_{\{V_{ref} \in \mathcal{A}_{21}\}} + a_{2,2}(X_1) 1_{\{V_{ref} \in \mathcal{A}_{22}\}}, \text{ where}$$

$$\mathcal{A}_{21}(X_1) = \left\{v \in \mathcal{A}_1 \mid d(v, a_{2,1}(X_1)) < d(v, a_{2,2}(X_1))\right\} \text{ and}$$

$$\mathcal{A}_{22}(X_1) = \left\{v \in \mathcal{A}_1 \mid d(v, a_{2,1}(X_1)) > d(v, a_{2,2}(X_1))\right\}. \quad (6)$$

By Lemma 3 of Appendix A the new belief B_2 is concentrated on $\mathcal{A}_2$ which is either one of the 4 choices based on (X_1, X_2):

$$\begin{aligned} \mathcal{A}_1^2 &:= \mathcal{A}_{21} \text{ with } X_1 = a_1^1, \mathcal{A}_2^2 := \mathcal{A}_{22} \text{ with } X_1 = a_1^1, \\ \mathcal{A}_3^2 &:= \mathcal{A}_{21} \text{ with } X_1 = a_2^1 \text{ or } \mathcal{A}_4^2 := \mathcal{A}_{22} \text{ with } X_1 = a_2^1. \end{aligned}$$

Continuing this way, at any k we have 2^k centres given by $\mathcal{Q}_\pi$ and the corresponding 2^k nested-partition areas $\{\mathcal{A}_i^k, i \le 2^k\}$ (see Figure 1), by using Lemma 4 of Appendix A. □

To summarize, $a_{k,i}$, $i = 1, 2$ represent the two recommendations at step k. These two can be random, depend upon the recommendation history till time $k-1$, $X_1, \cdots, X_{k-1}$. At any step, the user chooses one amongst the two recommendations, which is given by X_k. Overall, the random choice X_k can be one among the 2^k values $\left\{a_i^k, i \le 2^k\right\}$. Finally the belief B_k at time step k is uniformly distributed over (random) area $\mathcal{A}_k$, i.e., $B_k \sim \mathcal{U}(\mathcal{A}_k)$, which in turn can be one among 2^k areas $\left\{\mathcal{A}_i^k, i \le 2^k\right\}$ (see Figure 1).

Further from the proof of Lemma 1, for any policy π and for any k, the sets $\{\mathcal{A}_j^k\}_{j \le 2^k}$ form a 2^k- partition of $\mathcal{S}$. And the 2^k partition is a finer division of 2^{k-1} partition, i.e.,

$$\mathcal{A}_i^{k-1} = \mathcal{A}_{2i-1}^k \cup \mathcal{A}_{2i}^k \text{ for each } i \le 2(k-1), k.$$

Objective

The sequence (X_k, B_k) is a controlled Markov Chain, controlled by sequence of state dependent actions/policy, π. The required cost $E[T - \tau]$ with any π and initial $X_0 = x$ equals

$$J(x,\pi) \;=\; E_x^{\pi}[T-\tau] = \sum_{k=0}^{T-2} P(T-\tau > k) = \sum_{k=0}^{T-2} P(\tau < T-k)$$
$$= \sum_{k=1}^{T-1} P(\tau \le k) = \sum_{k=1}^{T-1} P\Big(V_{ref} \in \cup_{l=1}^{k} \mathcal{B}(X_l, \underline{r}) \cap \mathcal{A}_0\Big), \quad (7)$$

and we are interested in maximizing the above cost.

IV. Optimal Policies

The areas $\{\mathcal{A}_i^k\}_{i,k}$ and the recommendation choices $\mathcal{Q}_\pi$ depend upon the policy π, but given a policy π one can determine all of them. Fix any policy π and define $\mathcal{B}_i^k := \mathcal{B}(a_i^k, \underline{r})$ to represent the desired ball around the recommendation $a_i^k \in \mathcal{Q}_\pi$. It is clear to see that X_l at time l would be one among a_i^l for some $i \le 2^l$, say it equals $a_{\bar{i}}^l$,. This also implies $V_{ref} \notin \mathcal{B}_i^l$ for any $i \ne \bar{i}$. In all, user is satisfied on or before time slot k, i.e., $\tau \le k$ if and only if

$$\{\tau \le k\} = \Big\{V_{ref} \in \cup_{l=1}^{k} \cup_{i=1}^{2^l} \mathcal{B}_i^l \cap \mathcal{A}_0\Big\}.$$

Hence the equation (7) modifies as below:

$$\begin{aligned} J(x,\pi) &= \sum_{k=1}^{T-1} P\Big(V_{ref} \in \cup_{l=1}^{k} \cup_{i=1}^{2^l} \mathcal{B}_i^l \cap \mathcal{A}_0\Big) \\ &\overset{b}{\le} \sum_{k=1}^{T-1} \sum_{l=1}^{k} \sum_{i=1}^{2^l} P\Big(V_{ref} \in \mathcal{B}_i^l \cap \mathcal{A}_0\Big) \\ &= \sum_{k=1}^{T-1} \sum_{l=1}^{k} \sum_{i=1}^{2^l} \frac{\mu(\mathcal{B}_i^l \cap \mathcal{A}_i^l)}{\mu(\mathcal{A}_0)}, \text{ as } \mathcal{B}_i^l \subset \mathcal{A}_i^l \subset \mathcal{A}_0, \end{aligned}$$

where the inequality is due to the union bound. Using the above we state our first result:

***Theorem** 1:* For any policy π, with $|\mathcal{B}| := \mu(\mathcal{B}(0,\underline{r}))$:

$$i)\; J(x,\pi) = E[T-\tau](\pi) \;\overset{b}{\le}\; \frac{\sum_{k=1}^{T-1}\sum_{l=1}^{k}\sum_{i=1}^{2^l}\mu(\mathcal{B}_i^l)}{\mu(\mathcal{A}_0)} \quad (8)$$
$$= \frac{\sum_{k=1}^{T-1}(2^{k+1}-2)|\mathcal{B}|}{\mu(\mathcal{A}_0)} = \frac{2\Big(2^T - T - 1\Big)|\mathcal{B}|}{\mu(\mathcal{A}_0)}.$$

$ii)$ For any π, (7) can directly be upper bounded ($P(A) \le 1$):

$$E[T-\tau](\pi) \le T.$$

$iii)$Further, $E[T-\tau](\pi) \le \min_k \left\{\frac{2\Big(2^k-k-1\Big)|\mathcal{B}|}{\mu(\mathcal{A}_0)} + T - k + 1\right\}$.

Proof: Parts (i), (ii) are immediate. Part (iii) is obtained by using upper bounds of Part (i) and Part (ii) respectively for first and second terms of the following (for any k):

$$J(x,\pi) = \sum_{t=1}^{k-1} P\Big(V_{ref} \in \cup_{l=1}^{t} \cup_{i=1}^{2^l} \mathcal{B}_i^l \cap \mathcal{A}_0\Big) + \sum_{t=k}^{T} P(\tau > t). \;\square$$

It is easy to see that the upper bound b of equation (8) can be achieved with equality if all the balls $\{\mathcal{B}_i^k\}_{k,i\le 2^k}$ are disjoint and if all these balls lie completely in $\mathcal{A}_0$. And this would provide an optimal policy as below:

Corollary 1: If there exists a policy π^* whose $(2^{T+1}-2)$-balls[1], $\{\mathcal{B}_i^{k*}\}_{k\le T-1, i\le 2^k}$, are disjoint and further if $\mathcal{B}_i^{k*} \subset \mathcal{A}_i^{k*}$ for every (k,i), then the inequality (b) in (8) is satisfied with equality and then π^* is an optimal policy because:

$$E[T-\tau](\pi^*) \ge E[T-\tau](\pi) \text{ for any policy } \pi. \qquad \square$$

On the other hand, Part (ii) upper bound is useful to obtain:

Corollary 2: If $\mathcal{A}_0 \subset \mathcal{B}(\mathbf{a}, \underline{r})$ for some $\mathbf{a}$ then π^* with $\pi^*_{k,i} = \mathbf{a}$ for all k, i is optimal achieving the upper bound T. $\square$

Similarly one can have optimal policies that achieve the upper bound of Theorem 1.(iii), via an $k \ne T$. This can be the case, for example, when $\mathcal{A}_0$ is union of L disjoint balls with $L < (2^{T+1}-2)$ (see in Appendix B). As another example, consider $\mathcal{S} = (v_1, v_2, \cdots, v_n) \subset \mathcal{R}$ with usual $d(v_1,v_2) = |v_1 - v_2|$. Say $\underline{r}$ is small such that $\mathcal{B}(v,\underline{r}) = \{v\}$ (a single element) for any $v \in \mathcal{S}$. Then the well known binary search method is an optimal policy when T is sufficiently large.

A. Optimal policies in Continuous space ($\mathcal{R}^2$) with L^1 metric

With L^1 metric on $\mathcal{R}^2$ any ball $\mathcal{B}(\mathbf{a}, r)$ is a rhombus, which on rotation becomes a square. So, user interest V_{ref} is uniformly distributed in a square of dimension, $\sqrt{2}\,\overline{R}$, with centre as the initial recommendation X_0. The user is satisfied with any item which lies inside a square of dimension $\sqrt{2}\underline{r}$ with V_{ref} as the centre. If $(2^{T+1}-2) \le \frac{\overline{R}^2}{\underline{r}^2}$, we have sufficient disjoint balls and Corollary 1 is applicable. With $R := \sqrt{2}\,\overline{R}/2 = \overline{R}/\sqrt{2}$, one can easily verify that an optimal policy π^* is given by:

$$\begin{array}{ll} a^*_{1,1} = (\frac{R}{2}, 0), & a^*_{1,2} = (-\frac{R}{2}, 0), \\ a^*_{2,1} = X_1 + (0, \frac{R}{2}), & a^*_{2,2} = X_1 + (0, -\frac{R}{2}) \\ a^*_{3,1} = X_2 + (\frac{R}{4}, 0), & a^*_{3,2} = X_2 + (-\frac{R}{4}, 0), \\ \vdots & \\ a^*_{2k-1,1} = X_{2k-1} + (0, \frac{R}{2^{2k-2}}), & \\ \qquad\qquad a^*_{2k-1,2} = X_{2k-1} + (0, -\frac{R}{2^{2k-2}}). & \\ a^*_{2k,1} = X_{2k} + (\frac{R}{2^{2k-2}}, 0), & \\ \qquad\qquad a^*_{2k,2} = X_{2k} + (-\frac{R}{2^{2k-2}}, 0) \text{ for all } k, & \end{array} \quad (9)$$

where user choice at any time step $k \ge 1$ is given by:

$$X_k \;=\; \begin{cases} a^*_{k,1} & \text{if } d(a^*_{k,1}, V_{ref}) < d(a^*_{k,2}, V_{ref}) \\ a^*_{k,2} & \text{if } d(a^*_{k,1}, V_{ref}) > d(a^*_{k,2}, V_{ref}). \end{cases} \quad (10)$$

Optimal policies for the case with $(2^{T+1}-2) > \overline{R}^2/\underline{r}^2$ are obtained using Theorem 1.(iii) in Appendix B.

L^1 metric with 4 recommendations, i.e., with $M = 4$: The above analysis can easily be extended. We will have 4^k partitions in Theorem 1 and the optimal policy for L^1 metric is ($r\angle\theta$ is representation in polar coordinates):

$$a^*_{k-1,i} = X_{k-1} + \frac{\overline{R}}{2^{k-2}}\angle((i-1)90^\circ + 45^\circ) \text{ for all } i \le 4, k.$$

V. Algorithms for continuous space

Our ultimate aim is to provide algorithms that implement good recommendation policies on any item space $\mathcal{S}$. When S is discrete one can directly derive optimal policies through exhaustive search as discussed in Appendix C. However, this could a restrictive procedure and probably works only for small T. To derive algorithm for the general spaces, we first derive optimal policies for the continuous case and then

[1]We need 2 balls at step 1, 4 at step 2 and 2^k at step k and so on up to time $k = T-1$ (Figure 1) and thus we need a total of $\sum_{k=1}^{T-1} 2^k = 2^{T+1} - 2$ disjoint balls.

adopt it to the discrete case. We do this by introducing a notion of local angle on discrete spaces. We first discuss implementation of continuous policies.

A. Implementation

We refer the optimal policy discussed in equation (9) on $\mathcal{R}^2$ by π^*_{90}; the subscript indicates that 90° plays major role as explained below.

Policy π^*_{90}: From (9) at any time step k, CP recommends two items $(a_{k,1}, a_{k,2})$, equidistant $(R/2^{k/2})$ from X_{k-1} while maintaining the following angular separations:

$$\angle\{\overrightarrow{X_{k-1}a_{k,1}}, \overrightarrow{X_{k-1}a_{k,2}}\} = 180^\circ \text{ while}$$
$$\angle\{\overrightarrow{X_{k-1}a_{k,1}}, \overrightarrow{X_{k-1}X_{k-2}}\} = 90^\circ,$$

where $\overrightarrow{xy}$ implies the line joining points $\mathbf{x}, \mathbf{y}$.

One can obtain the estimate of $E[\tau]$, using the iterative procedure described in Algorithm 1. Repeating it for many samples and averaging gives an estimate for $E[\tau]$.

Algorithm 1: Policy π^*_{90} Algorithm (for each sample)

Initialize: Generate V_{ref}, X_0 randomly (uniformly)

For steps: $k = 1, 2, \cdots$

a CP provides two recommendation $a^*_{k,i}$, $i = 1, 2$ as given by (9).

b User chooses the best recommendation according to (10) and returns X_k.

If $k \geq T$ or $X_k \in \mathcal{B}(V_{ref}, \underline{r})$ **Then** Exit with $\tau = k$

Else return to (a) with $k = k + 1$

Remark: These results in continuous space are of independent interest. For example, they can be used in robotics: in rescue robot, robot navigation systems etc.

Policy π_{180}: The policy π^*_{90} needs 90 degrees, and we will notice that the implementation of 90 degrees (after translation to discrete space) requires complicated logic. Hence we propose another policy π_{180} which only ensures 180 degree separation between the two recommendations ($a_{k,1}$ and $a_{k,2}$), but the angular separation with respect to the previous user choice X_{k-1} is chosen randomly, i.e, now

$$\angle\{\overrightarrow{X_{k-1}a_{k,1}}, \overrightarrow{X_{k-1}X_{k-2}}\} \text{ is random.}$$

Below, we study the loss of performance with this policy.

Simulation results : We compare value of $E[\tau]$ for policies π_{180} and π^*_{90}. The loss of performance with π_{180} in comparison to π^*_{90} increases with decrease in $\underline{r}$. This implies: a) in continuous space with Euclidean distance one may use less complicated π_{180} in place of π^*_{90}, when $\underline{r}$ is not very small; b) in other cases, it may be important to ensure 90 degrees separation. We leave this topic for future research.

VI. Algorithms for Discrete Database

We assume any dataset is specified by an item space $\mathcal{S}$ and a rating matrix $\mathbb{R}$. The (i, j)-th entry $r_{i,j}$ of matrix $\mathbb{R}$ is the rating given by user j to item i and it equals 0 when user has not rated it. Using this data and using the similarity measures of (2), we compute the distance matrix $\mathbb{D}$.

$\underline{r}$	$E[\tau]$	$E[\tau]$
	π^*_{90}	π_{180}
10	6.9	36.8
15	5.6	26.9
20	4.8	20.0
30	3.4	11.6
50	2.2	4.7
60	1.8	3.2
70	1.7	2.3
80	1.5	1.8

Fig. 2. $E[\tau]$ with L^1 metric, $\overline{R} = 100$

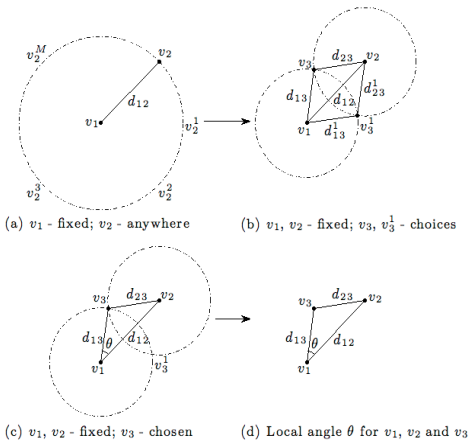

Fig. 3. Notion of 'Local' Angle.

Notion of Distance: Distance is defined using similarity measure. The similarity measure takes values between 0 and 1, as it is a normalized measure (see (2)) and is inversely proportional to the distance. Hence we can say that distance between items i, j is $d(i, j) = 1 - s_c(i, j)$ and hence is the (i, j)-th element of matrix $\mathbb{D}$. It is clear that maximizing distance based functions is equivalent to minimizing the same functions with respect to similarity measure and vice versa. The distance matrix is used to compute the neighbourhood matrix $\mathbb{N}$. Row j of matrix $\mathbb{N}$ has items arranged in the descending order of similarity with item j, i.e., $s_c(j, v_1) \geq s_c(j, v_2)$ if item v_1 is placed before item v_2 in row j.

Notion of Local Angle: We discuss the notion of 180 degrees using the distance measure described above.

Consider a center in a continuous space and points on a given circumference. There are uncountable such points. The points might be at equidistant from the center, but their inter distances can vary significantly based on their angular separations. This can be used to define the notion of 180 degrees. We begin with three items (v_1, v_2 and v_3) and let d_{12}, d_{23} and d_{13} be their inter distances. Fix the position of item v_1, then v_2 can be anywhere on the circumference of $\mathcal{B}(v_1, d_{12})$ (see Fig. 3). Fix one of these points as v_2 and now draw two circles, with respective centres as v_1 and v_2 and radii as d_{13} and d_{23}. Then item v_3 can be at one of the two intersecting points as shown (see Fig. 3). Choose any one of the two points as v_3 and form a triangle to obtain the required angle. For example, θ in Fig. 3 represents the 'local' angle between lines joining items v_1-v_2 and v_1-v_3. Note that, this angle is independent of the chosen points and depends only upon their inter distances, d_{12}, d_{23} and d_{13}.

Implementation of 180 degree: An angular separation close to 180 degrees is achieved if the inter distances between the pair of points (both at the same distance say R, from a reference point X_0) is maximized. We in fact suggest an algorithm for general M, which ensures maximum separation between the M-tuple in case their inter angular separations are at least $360/M$ degrees. We extract the row corresponding to X_0 of the matrix $\mathbb{N}$, and list consequent n ($n > M$) items starting at a distance R from X_0. Out of these n we have

to pick M items such that the sum of the inter similarity distances between the items is minimized. The following Integer Linear Programme (ILP) exactly achieves this and more details about the same is given in ([12]).

$$\begin{aligned}
\text{minimize} \quad & \sum_{i\in A}\sum_{j\in A:i>j} s_c(i,j)x_{ij} \\
\text{subject to} \quad & \sum_{i\in A}\sum_{j\in A:i>j} x_{ij} = \frac{M(M-1)}{2}, \\
& x_{ij} + 1 \geq y_i + y_j, \forall i \in A, j \in A, i > j \\
& \sum_{i\in A} y_i = M \\
& x_{ij} \in \{0,1\},\ y_i \in \{0,1\}, i,j \in A,\ A = \{1 \dots n\}.
\end{aligned} \tag{11}$$

The required set of M-items are those i, for which the solution of (11) has $y_i = 1$.

UR-LA Algorithm: We now adapt the algorithm (9) to propose a recommendation policy that can be implemented on any discrete dataset and call it as *UR-LA (User Response and Local Angle based) Algorithm*.

We assume that user is interested in a particular item V_{ref}, unknown to the recommender. The recommender starts the recommendation with a randomly selected item X_0 where $d(V_{ref}, X_0) \leq \bar{R}$. For each V_{ref} we run the algorithm for 300 different initial points picked uniformly at random. We initially tested the algorithm with $M = 2$. At every step k we choose n items using $\mathbb{N}$ starting at distance R_k from X_k (where R_k is as in (9) and $R_1 = \bar{R}/\sqrt{2}$), from previous user choice X_{k-1}, and then use program (11) to recommend two items. We set $n = 4$. The user chooses one among them, which is X_k. The algorithm terminates when a recommendation is within ball $B(V_{ref}, \underline{r})$ or after T steps.

Algorithm 2: UR-LA Algorithm - Given $\overline{R}, \underline{r}, n, M, \mathbb{R}$

Initialize:

- Generate $s_c(i,j)$ between all items i and j using (2) and store in distance matrix $\mathbb{D}$.
- Generate neighbourhood matrix $\mathbb{N}$ using $\mathbb{D}$:
- Generate V_{ref}, X_0 randomly (uniformly), such that $d(V_{ref}, X_0) \leq \overline{R}$. Set $R_1 = \overline{R}/\sqrt{2}$.

For steps $k = 1, 2, \dots$

a Choose n consequent items from a row of $\mathbb{N}$, that start at distance R_k from X_k.

b CP provides M recommendations out of n as in (11).

c User chooses the best recommendation according to (10) and returns X_k.

If $k \geq T$ or $X_k \in \mathcal{B}(V_{ref}, \underline{r})$ **Then** Exit with $\tau = k$
Else return to (a) with $k = k + 1$ and set
$R_k = R_{k-1}/2$ if $k\%2 = 0$ else $R_k = R_{k-1}$.

The UR-LA Algorithm is compared with the widely used item based Collaborative Filtering (CF) Algorithms of [13]. The CF algorithm also uses rating matrix $\mathbb{R}$ to generate the similarity based distance matrix $\mathbb{D}$. At every step k it recommends M items that are nearest to X_k. We also propose

Music	CF	UR-LA	%Imp
SA-WR	19.5	13.9	33.9
SA-WOR	16.0	13.5	17.0
SF-WR	20.6	14.2	36.5
SF-WOR	17.4	13.8	23.2

Movie	CF	UR-LA	%Imp
SA-WR	20.4	17.6	14.4
SA-WOR	18.2	17.2	5.5
SF-WR	20.8	17.8	15.7
SF-WOR	18.9	17.2	9.2

TABLE I
$E[\tau]$ FOR DIFFERENT ALGORITHMS WITH $\overline{R} = 1, \underline{r} = 0.25, n = 4$ AND $M = 2$.

Settings		CF	UR-LA	Hybrid	%Imp
Music	**WR**	18.61	13.25	12.10	42.42
Music	**WOR**	13.04	11.94	11.31	14.26
Movie	**WR**	19.70	17.94	17.40	12.39
Movie	**WOR**	16.53	16.41	15.87	4.04

TABLE II
$E[\tau]$, INCLUDING HYBRID ALGORITHM, $\overline{R} = 1, \underline{r} = 0.25, n = 4, M = 3$ AND 'START ANYWHERE'.

the following hybrid policy which combines the best features of both the UR-LA and CF algorithms:

Hybrid Recommendation (Hybrid): We found that UR-LA performs significantly better for the cases for which start radius $\bar{R}$ is large, i.e., if X_0 is often far away from V_{ref}. If X_0 is close to V_{ref} then the performance is poor. UR-LA basically recommends 'faraway' items (from say X_k), with the reasoning that user is not yet satisfied and has chosen to explore further. This is the main reason for improvement over CF. However, exactly the same reason deteriorates the performance if start-radius is small. Thus we propose an Hybrid Algorithm which uses ideas from both *CF* and *UR-LA* algorithms. One can define this algorithm for any $M \geq 3$ and we describe it here for $M = 3$. In each round, one of the recommendation is the most similar item to the previously selected item as in CF and the other two items are selected as in UR-LA. We will show that the Hybrid algorithm outperforms in most of the cases.

VII. EXPERIMENTS

In this section we test the performance of the algorithms on two real datasets. The first one ([11]) has 285 music bands rated by 1257 users. In the rating matrix, each column corresponds to an unique band and each row to an unique user. The ratings are binary– each user gives a rating 1 if he likes the band and 0 if he does not like. If no entry exists then it is considered as 0.

The second one is the MovieLens dataset [10]. We again constructed the rating matrix after appropriate conversion of the data. Each column again represents a movie and each row a user. Each entry here corresponds to the rating given by the user to a particular movie on a scale of 0 and 5. If the user has not seen/not rated the movie then the rating is set to zero. We have ratings of 1682 movies given by 943 users.

For both the datasets we set $T = 20$. Table I summarizes the simulation results of the UR-LA compared to CF on various settings. We consider the following two settings.

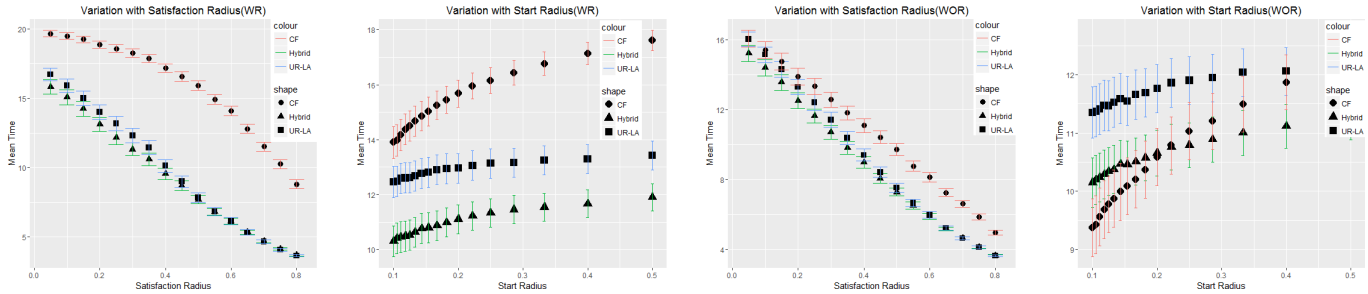

Fig. 4. Variations around the expected time to satisfy the user

	0.75	**0.65**	**0.55**	**0.45**	**0.35**	**0.25**
Hybrid	4.13	5.34	6.81	8.69	10.58	12.15
UR-LA	4.08	5.33	6.83	8.99	11.44	13.18
CF	10.24	12.79	14.91	16.56	17.85	18.58
%Imp	85.03	82.17	74.56	62.32	51.11	41.8

TABLE III
VARIANCE AROUND $E[\tau]$ VERSUS SATISFACTION RADIUS ($\underline{r}$): MUSIC DATASET, 'WR' CONDITION AND $\overline{R} = 1, n = 4, M = 3$

	0.5	**0.4**	**0.25**	**0.2**	**0.11**	**0.1**
Hybrid	11.91	11.67	11.34	11.1	10.46	10.30
UR-LA	13.42	13.3	13.13	12.96	12.6	12.46
CF	17.61	17.14	16.15	15.67	14.17	13.9
%Imp	38.63	38	35.01	34.13	30.11	29.8

TABLE IV
VARIANCE AROUND $E[\tau]$ VERSUS START RADIUS ($\overline{R}$): MUSIC DATASET, 'WR' CONDITION AND $\underline{r} = 0.25, n = 4, M = 3$

1) Start Anywhere (SA): in this setting we selected initial point uniformly at random. 2) Start Far (SF): in this setting an initial point is selected uniformly at random but at a fixed distance from the V_{ref}. In addition to the above, we also considered recommendations With Replacement (WR) and Without Replacement (WOR). In WR, an item once recommended can be recommended again, whereas in the WOR it is not recommended again. The values in the table correspond to the average steps taken by the algorithm to satisfy the user for various settings. In Table II we included the Hybrid policy, and provided a comparative study of all the three algorithms CF, UR-LA and Hybrid with $M = 3$ (for all policies).

One can observe from the tables that UR-LA algorithm has significant improvement over CF algorithm and Hybrid algorithm performs even better.

In Tables III and IV we provide the variance around $E[\tau]$ with varying satisfaction radius $\underline{r}$ and start radius $\bar{R}$, respectively. We have also plotted the corresponding graphs in Figure 4, WR setting is in sub-figures 1-2 while WOR setting is in sub-figures 3-4. As seen, UR-LA and Hybrid algorithms perform way better than the CF algorithm (even with respect to the variance) for the WR setting. With WOR setting, UR-LA and Hybrid algorithms perform better when $\overline{R}$ is large. However, when $\overline{R}$ is small, CF performs better, the performance of Hybrid algorithm is comparable to that of CF.

VIII. CONCLUSION

We presented and demonstrated novel recommendation policies, that are based on user-generated responses. Unlike the traditional recommendation schemes, our recommendations not only depend upon the history, but they also exploit the responses of the same user in the same session. We proposed a new 'notion of local angle' in the context of discrete data base, using which we translated continuous space policies to discrete space. The main difference in the new approach is that 'the inter distances between items of the same recommendation list is maximum possible' and 'the distance from the previous recommendation is initially large and decreases geometrically'. These observations readily give good policies, as via user's choices to previous recommendations (of the same session) the user has given a good hint about not only the items closer to its interest, but also about the items which are not exactly its choice.

We developed UR-LA (user response and local angle) and hybrid algorithms and tested their performance on two real datasets. We compared them with baseline CF based recommendation algorithm. Our algorithms reduce the search time significantly (upto 40%) in most of the cases.

ACKNOWLEDGMENTS

M.K. Hanawal is supported by IIT Bombay IRCC SEED grant (16IRCCSG010) and INSPIRE faculty fellowship (IFA-14/ENG-73) from DST, Govt. of India

REFERENCES

[1] A. M. Rashid, I. Albert, D. Cosley, S. K. Lam, S. McNee, J. A. Konstan, and J. Riedl. "Getting to know you: Learning new user preferences in recommender systems." In Proceedings of the 2002 International Conference on Intelligent User Interfaces, San Francisco, CA, Feb. 2002, 127-134.

[2] Rashid, Al Mamunur, George Karypis, and John Riedl. "Learning preferences of new users in recommender systems: an information theoretic approach." ACM SIGKDD Explorations Newsletter 10.2 (2008): 90-100.

[3] Blerina Lika, Kostas Kolomvatsos, and Stathes Hadjiefthymiades. "Facing the cold start problem in recommender systems." Expert Systems with Applications 41 (2014) 2065-2073.

[4] M. Stritt, K. H. L. Tso, and L. S. Thieme, Attribute aware anonymous recommender systems, in Advances in Data Analysis, R. Decker and H.-J. Lenz, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, January 15, 2014, pp. 497504.

[5] Hill, Will and Stead, Larry and Rosenstein, Mark and Furnas, George, Recommending and Evaluating Choices in a Virtual Community of Use. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '95), Denver, Colorado, USA, 194-201.

[6] Lev Grossman, "How Computers Know What We Want Before We Do." Available at: http://item.time.com/time/magazine/article/0,9171, 1992403,00.html
[7] G. Adomavicius and A. Tuzhilin. Toward the Next Generation of Recommender Systems: A Survey of the State of the Art and Possible Extensions, IEEE Trans. Knowl. Data Eng, 17 (6), 2005, pp. 734-749.
[8] J. Gaillard, M. El-Beze, E. Altman and E. Ethis. "Flash reactivity : adaptative models in recommender systems," International Conference on Data Mining (DMIN), WORLDCOMP, (2013).
[9] Adomavicius, Gediminas, and Alexander Tuzhilin. "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions." Knowledge and Data Engineering, IEEE Transactions on 17.6 (2005): 734-749.
[10] MovieLens Dataset, https://grouplens.org/datasets/movielens/.
[11] Music Dataset, https://labrosa.ee.columbia.edu/millionsong/lastfm.
[12] Technical Report available at https://www.dropbox.com/s/pww5od82q4ghe8f/TR_URLA.pdf?dl=0.
[13] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative Filtering recommendation algorithms. In Proceedings of the 10th international conference on World Wide Web, pages 285-295. ACM, 2001.

APPENDIX A: PROOFS

Lemma 2: The new belief B_1 is uniformly distributed over area $\mathcal{A}_1$ which has two choices:

$$B_1 \sim \mathcal{U}(\mathcal{A}_1) \text{ where } \mathcal{A}_1 = \mathcal{A}_{11}1_{\{X_1=a_{11}\}} + \mathcal{A}_{12}1_{\{X_1=a_{12}\}},$$

where $\mathcal{A}_{11}$, $\mathcal{A}_{12}$ are given by equation (5).

Proof of Lemma 2: By definition belief B_1 is the conditional distribution of V_{ref} given history H_1 which includes current observation X_1, the user choice. That is, $B_1 \sim V_{ref}\big|H_1$. For any subset Λ (Borel if continuous case)[2] from equation (5):

$$\begin{aligned} P(B_1 \in \Lambda) &= P\big(V_{ref} \in \Lambda \big| B_0, \mathbf{a_1}, X_1\big) \\ &= P\big(V_{ref} \in \Lambda \big| V_{ref} \in \mathcal{A}_0, \mathbf{a_1} = (a_{11}, a_{12}), X_1\big) \\ &= P\big(V_{ref} \in \Lambda \big| V_{ref} \in \mathcal{A}_0, V_{ref} \in \mathcal{A}_{11}\big)1_{\{X_1=a_{11}\}} \\ &\quad + P\big(V_{ref} \in \Lambda \big| V_{ref} \in \mathcal{A}_0, V_{ref} \in \mathcal{A}_{12}\big)1_{\{X_1=a_{12}\}} \\ &= \frac{\mu(\Lambda \cap \mathcal{A}_{11})}{\mu(\mathcal{A}_{11})}1_{\{X_1=a_{11}\}} + \frac{\mu(\Lambda \cap \mathcal{A}_{12})}{\mu(\mathcal{A}_{12})}1_{\{X_1=a_{12}\}}. \end{aligned} \tag{12}$$

Thus clearly belief B_1 is a uniform random variable either over $\mathcal{A}_{11}$ or $\mathcal{A}_{12}$, as μ is uniform. □

Lemma 3: The new belief B_2 is uniformly distributed, i.e., $B_2 \sim \mathcal{U}(\mathcal{A}_2)$, where the area $\mathcal{A}_2$ has two choices for any given X_1:

$$\mathcal{A}_2(X_1) = \mathcal{A}_{21}1_{\{X_2=a_{21}\}} + \mathcal{A}_{22}1_{\{X_2=a_{22}\}},$$

with $\mathcal{A}_{21}$, $\mathcal{A}_{22}$ defined in (6). These two choices depend further upon X_1, and hence in total $\mathcal{A}_2$ can be one among four choices.

Proof: The belief B_2 is again conditional distribution of V_{ref} given the entire history $B_0, B_1, \mathbf{a}_1, \mathbf{a}_2$ and current observation X_2. The proof goes through in exactly the same manner as in Lemma 2, except that $\mathbf{A}_2$ can now have four choices based on X_1, X_2. □

[2]This is basically belief propagation. The state $S = (X, V_{ref})$ observation $O = X$, the choice of user and then belief of the unobserved state needs to be computed.

Lemma 4: The belief B_k at time step k ($k > 1$) is uniformly distributed, i.e., $B_k \sim \mathcal{U}(\mathcal{A}_k)$, where the area $\mathcal{A}_k$ has two choices for any given X_{k-1}:

$$\mathcal{A}_k(X_{k-1}) = \mathcal{A}_{k1}1_{\{X_k=a_{k1}\}} + \mathcal{A}_{k2}1_{\{X_k=a_{k2}\}},$$

where $\mathcal{A}_{k1}$, $\mathcal{A}_{k2}$ are partitions of $\mathcal{A}_{k-1}$ as in definitions (5), (6). These two choices depend further upon $X_1, \cdots, X_{k-1}$, and hence in total $\mathcal{A}_2$ can be one among 2^k choices.

Proof: The proof goes through in exactly the same manner as in Lemmas 2-3. □

APPENDIX B: L^1 METRIC WITH $(2^{T+1} - 2) > \overline{R}^2/\underline{r}^2$

Basic idea is to obtain the optimal policy using Corollary 1 till k^* where $k^* = \arg\max_k \left\{ (2^{k+1} - 2) \le \frac{\overline{R}^2}{\underline{r}^2} \right\}$ and then using Corollary 2 for the time steps from k^*+1 till T. Basically this policy achieves the upper bound of Theorem 1.(iii), where the minimum on the right hand side is achieved using k^*.

The exact details are as below for the case when $\overline{R}/\underline{r}$ is an appropriate power of 2 such that $2^{k^*+1} - 2 = \overline{R}^2/\underline{r}^2$. One can give similar construction even otherwise. But some minor details need to be considered.

Note that we exactly have $(2^{k^*+1} - 2)$ disjoint balls and hence one can upper bound all the terms till k^* by $|\mathcal{B}|$ as in Corollary 1. Let $a^*_{k,i}$ be as defined in equation (9) for all i and for all $k < k^*$. At k^* all the remaining areas $\left\{\mathcal{A}_i^{k^*}\right\}_{i \le 2^{k^*}}$ are already of size $\underline{r}$. As in Corollary 2, define for any $k > k^*$ and i

$$a^*_{k,i} = X_{k-1} = X_{k^*}.$$

One can easily verify that $E[\tau]$ is strictly less than k^*, thus the user satisfied in an average time, less than k^*.

$$E[\tau] = k^* - \frac{(2^{k^*+1} - 2k^* - 2)|\mathcal{B}|}{\mu(\mathcal{A}_0)}.$$

APPENDIX C: OPTIMAL POLICIES IN DISCRETE SPACE

We consider binary database with F features, similarity based distance and cardinality based measure. A ball $\mathcal{B}(v, r)$ here includes all those items which match in more than $F(1-r)$ features with v, e.g., $\mathcal{B}((10), 0.5) = \{(10)\}$, $\mathcal{B}((10), 1) = \mathcal{S}$. We again use Corollary 1 to obtain optimal policies in some example scenarios. One can easily verify the following.

Case I: $F = 7$**,** $T = 2$ **and** $\underline{r} = 2/7$**:** Optimal $\mathcal{Q}_{\pi^*}$ is

$$\begin{array}{lll} a_1^1 = 1111111, & a_2^1 = 0000000, & a_1^2 = 0011111, \\ a_2^2 = 1111100, & a_3^2 = 1100000 \quad \text{and} & a_4^2 = 0000011. \end{array}$$

Case II: $F = 7$**,** $T = 2$ **and** $\underline{r} = 3/7$**:** Optimal $\mathcal{Q}_{\pi^*}$ is

$$\begin{array}{lll} a_1^1 = 1111111, & a_2^1 = 0000000, & a_1^2 = 0001111, \\ a_2^2 = 111100, & a_3^2 = 1110000 \quad \text{and} & a_4^2 = 0000111. \end{array}$$

Case III: $F = 9$**,** $T = 2$ **and** $\underline{r} = 4/9$**:** Optimal $\mathcal{Q}_{\pi^*}$ is

$$\begin{array}{lll} a_1^1 = 111111000, & a_2^1 = 000000111, & a_1^2 = 001111001, \\ a_2^2 = 110001110, & a_3^2 = 110000110 \text{ and} & a_4^2 = 001110001. \end{array}$$