Determining the new product option that maximizes total cover ratio on users

Keming Li
Computer science and technology   Supervisor: Bo Tang

Abstraction:

Decision making problem is popular among these days. In rank aware processing, a user will choose the option than rank top for himself. Concretely, user preferences are usually represented as weight vectors. Each attribute of weight vector means how important is that attribute to the user. The score of an option respecting to a user is the dot product between the user's preference weight vector and the option. Only the options with top-k scores can attract the user. An option covers a user if and only if it can rank top-k for that user. Usually, a company has many products (options), each of which covers some of the users. A user covered by a company means at least one of its products covers this user. The company has to develop new product that satisfy a constrain and make its all products including this new product cover as more users as possible. In this problem we study how to determining which newly added option can maximize the cover ratio of the company. This problem is essential in developing new product, market decision, advertising, etc. We refer this problem as k-Cover Ratio Maximization ($kCMR$). In the report, we begin from top-k problem's computational geometric nature using tree to represented option spaces, and then from the relationship among constrain, options and user preference weight vectors to more efficiently solve this problem.

Key words: Cover ratio, new option, top-k query

1. Introduction

Take smartphone market as an example, there is different models of smartphone $(D = \{r_1, r_2, \dots, r_n\})$. Each model has different price, pixel, battery capacity, cooling capacity and etc. $(r_i = [r_i^1, r_i^2, \dots, r_i^d])$. Now company P has m models $(P = \{p_1, p_2, \dots, p_m\} \subset D)$, and have user data set $W = \{w_1, w_2, \dots, w_x\}$. Different users prefer in different aspect, for example some of them may prefer smartphone that with large battery capacity but some prefer better quality of screen, so each $w_i = [w_i^1, w_i^2, \dots, w_i^d]$ present the preference weight vector of a single user. The score of a model r_i respecting to a user w_i is the dot product $r_i \cdot w_i$. Usually a user will only make choose between his own top-k, so a product ranks top-k for the users is quite important. A product covers a user when the score of it ranks top k among all existing products (D). With developing of company and market, company has to develop new product. But with the limitation of technology, money and other factors, one can not develop a perfect product to cover all users. It can only develop a product that covers as more user as possible. For a company, some of its products have covered some users, so what it wants to do is how to make this new product cover more users that uncovered before.

In real life, k-Cover Ratio Maximization $(kCRM)$ can only solve the problems that how to decide the next generation product for companies. In advertising industry, it can help merchants how to cover specific group such as students, pregnant women and children. Besides it can tell advertiser where to set up new advertising board and if do so it can cover which group of people. Data analysts can use it to discover which group of people is ignored by the market.

Generally speaking, it is not a product covers a user but a group of products covers a user. And for different users, there are different groups of products. Among these groups of products, there are uncertain number of identity products. In our problem, we are aim to find this new product in continuous product space, which means there are infinite candidate products, so it is difficult to tell which product covers the most users.

In this report, we will from computational geometric nature of $kCRM$ to explain how to find the exact optimal options (products) with the data structure $CellTree$ mentioned in $kSPR$[1]. Besides, from some observations of this problem, we propose advance method that ignore irrelevant users and candidate products to save time.

2. problem definition

Given user dataset $D = \{r_1, r_2, \dots, r_m\}$, $r_i = [r_i^1, r_i^2, \dots, r_i^d]$, and a user preference weight vector set $W = \{w_1, w_2, \dots, w_x\}, w_i = [w_i^1, w_i^2, \dots, w_i^d]$

$Definition\ 1$: $A\ product\ \boldsymbol{p}'s$ score respecting to a user $\boldsymbol{w}$ is the dot product $\boldsymbol{p} \cdot \boldsymbol{w}$

$Definition\ 2$: $A\ product\ \boldsymbol{p}\ covers\ a\ user\ \boldsymbol{w}\ when\ the\ score\ respecting\ to\ \boldsymbol{w}\ ranks\ top\ \boldsymbol{k}\ among\ \boldsymbol{D}$

$Problem\ 1$:

$Given\ dataset\ D = \{r_1, r_2, \dots, r_m\}, r = [r_i^1, r_i^2, \dots, r_i^d], which\ r_i^j \in [0,1]$

$another\ product\ dataset\ P = \{p_1, p_2, \dots, p_n\} \subset D$

$product\ creation\ budget\ B,$

$a\ positive\ integer\ k,$

$a\ creation\ cost\ function\ C(p_i) = \Sigma c^j\ p_i^j,$

$a\ user\ preference\ tuple\ set\ W = \{w_1, w_2, \dots, w_x\}, w_i = [w_i^1, w_i^2, \dots, w_i^d],$

$$which\ w_i^j \in [0,1]\ and\ \sum_{j=1}^{d} w_i^j = 1$$

$The\ score\ of\ a\ model\ r_i\ respecting\ to\ a\ user\ w_i\ is\ the\ dot\ product\ r_i \cdot w_i$

$A\ model\ covers\ a\ user\ when\ the\ score\ of\ it\ ranks\ top\ k\ among\ D.$

$Find\ a\ new\ product\ p\ such\ that\ satisfies\ the\ constrain\ C(p) \le B$

$and\ maximize\ P \cup \{p\}'s\ cover\ ratio$

$$cp(p, P, k) = \frac{\left|\{w \mid \forall w \in W, \{P \cup \{p\}\} \cap TopK(w) \neq \emptyset\}\right|}{|W|}$$

3. geometric computation nature

Because the dataset D is with limited number of products, for a user w, it is easy to calculate the top-k score of w among D. Suppose S_k is the score that rank top-k for w, then if a new product p wants to rank top-k for this user, in other words, covers this user, p should follow $p \cdot w \ge S_k$. If we draw it when dimensionality $d = 2$, it would be shown as figure 1.

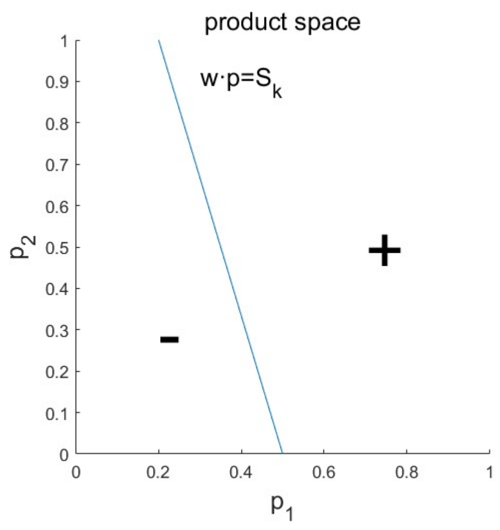


Figure 1

From figure 1, we can see that $w \cdot p = S_k$ divides the candidate product space into 2 half-space, one is marked as "+", where represents $w \cdot p > S_k$; the other is marked as "-", where represents $w \cdot p < S_k$. If and only if the product in the region "+" can cover w.

4. $CellTree$ representation

4.1 from 1 user to many users

We have seen the example when there is 1 user, which $w \cdot p = S_k$ will divide the space into 2 half-space. When it comes to multi users, such as 2 users (w_1, w_2), firstly $w_1 \cdot p = S_{k1}$ divides product space into 2 half-space as shown in figure 2; on the basis of figure 2, $w_2 \cdot p = S_{k2}$ divides h_1^+ and h_1^- into 2 half-space respectively as shown in figure 3. Region $h_1^+ \cap h_2^+$ covers w_1, w_2; Region $h_1^+ \cap h_2^-$ could only cover w_1; Region $h_1^- \cap h_2^+$ could only cover w_2; Region $h_1^- \cap h_2^-$ couldn't cover any user.

$Definition$ 3: $cover\ count\ of\ a\ product\ space\ region$

$The\ cover\ count\ of\ a\ product\ space\ region\ is\ the\ number\ of\ users\ that\ any\ product\ in\ this\ region\ could\ cover.$

For example, Region $h_1^+ \cap h_2^+$ covers w_1, w_2, so the cover count of $h_1^+ \cap h_2^+$ is 2. Problem 1 actually is finding the region which with the maximal cover count.

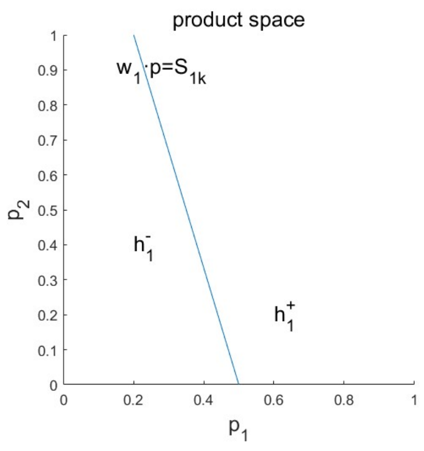


Figure 2

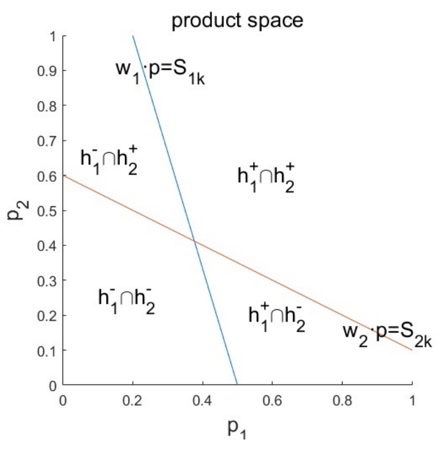
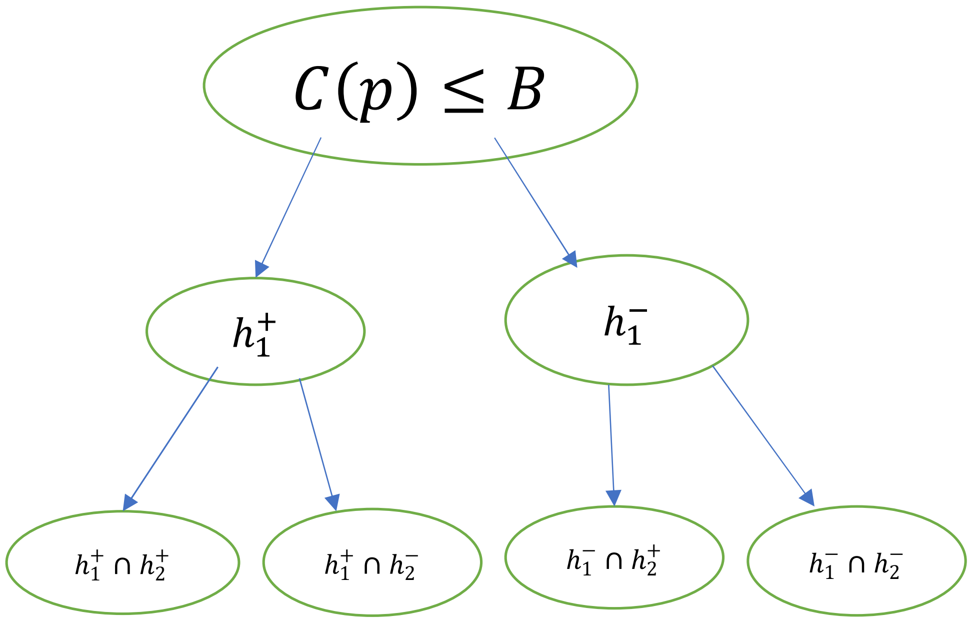


Figure 3

4.2 $CellTree$ representation and $CellTree$ insertion

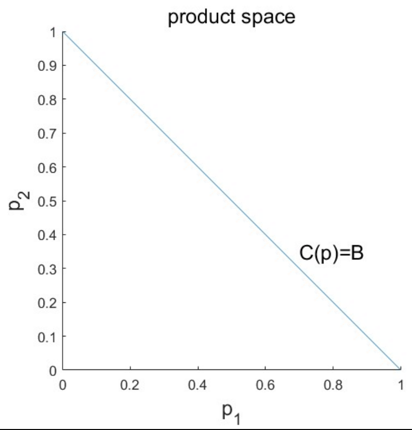
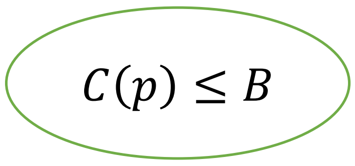
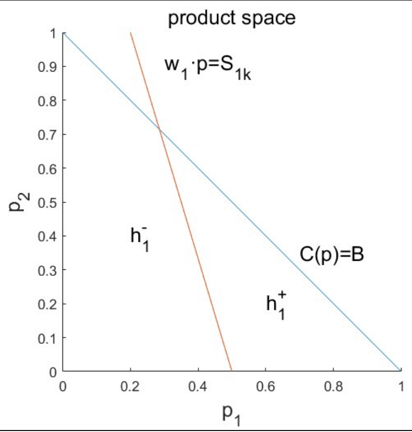
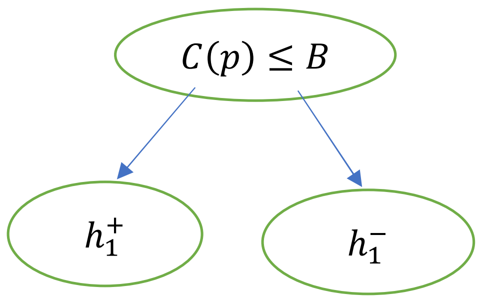
$CellTree$ is a binary tree. We built a $CellTree$ based on the example mentioned in 5.1

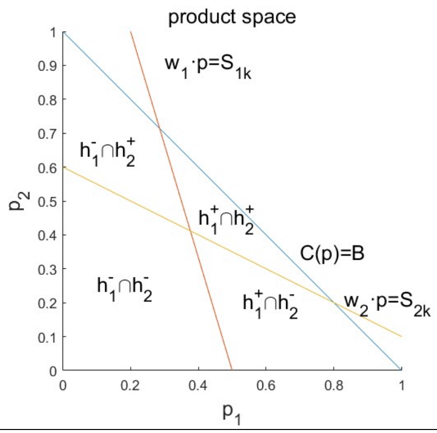
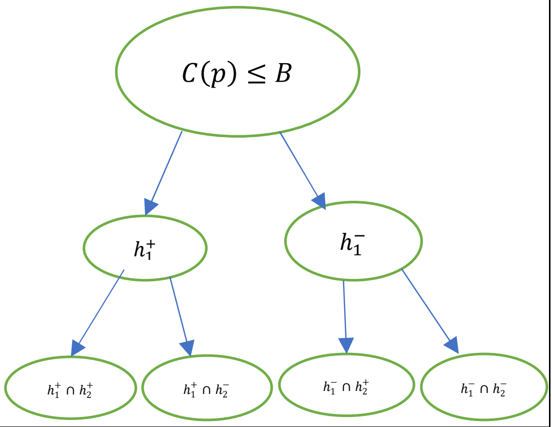


The root of the tree is the candidate option space which is limited by the constrain $C(p) \leq B$. Node $h_1^+ \cap h_2^+$ represents the region $h_1^+ \cap h_2^+$. In practice, node $h_1^+ \cap h_2^+$ will only use h_2^+ to present to save memory.

5. Baseline solution

5.1 baseline solution example

1. Deal with the constrain $C(p) \leq B$	
	
2. to cover user w_1, the space is divided into h_1^+ and h_1^-. In CellTree, node $C(p) \leq B$ generates two children, one is h_1^+ and the other is h_1^-.	
	
3. to cover user w_2, the space h_1^+ is divided into $h_1^+ \cap h_2^+$ and $h_1^+ \cap h_2^-$; the space h_1^- is divided into $h_1^- \cap h_2^+$ and $h_1^- \cap h_2^-$. In $CellTree$, node h_1^+ generates two children, one is $h_1^+ \cap h_2^+$ and the other is $h_1^+ \cap h_2^-$; node h_1^- generates two children, one is $h_1^- \cap h_2^+$ and the other is $h_1^- \cap h_2^-$.	

	 $C(p) \leq B$ h_1^+ h_1^- $h_1^+ \cap h_2^+$ $h_1^+ \cap h_2^-$ $h_1^- \cap h_2^+$ $h_1^- \cap h_2^-$
4. return region $h_1^+ \cap h_2^+$	

5.2 Baseline solution algorithm

1. For each w_i in W, based on $w_i \cdot p \leq S_{ik}$ using cell tree divide model space $C(p) \leq B$, where S_{ik} is the score to rank top k respecting to w_i
2. count each divided region's user cover number
3. Return the region that satisfies $C(p) \leq B$ and with the greatest cover count

6. observations

6.1 observation 1:

The constrain is $C(p) \leq B$, we can only consider $C(p) = B$. This means whatever option found in $C(p) < B$ there will always be a better option in $C(p) = B$.

6.2 observation 2:

There are users that we don't need to consider because we may never cover them under the constrain or we can always cover them if we choose the option on the constrain. Figure 4 shows the example.

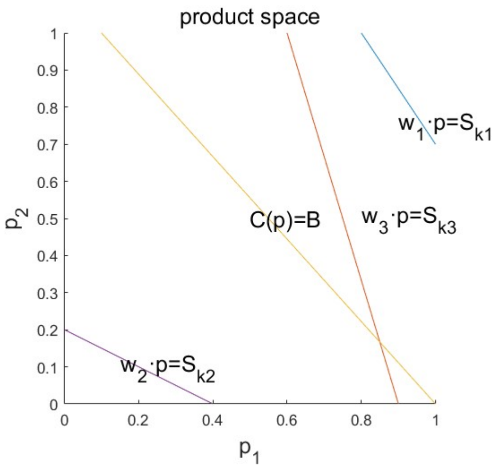


Figure 4

If we decide to choose the new product from $C(p) = B$, we can see that $w_1 \cdot p = S_{k1}$ also divides the product space into 2 half-spaces, but all the products on $C(p) = B$ is in the "-" half-space, which means all of them can not cover w_1. Different from w_1, the products that on $C(p) = B$ can always cover w_2. From this observation, we can move out all the users that $w_i \cdot p = S_{ki}$ doesn't intersect with $C(p) = B$.

6.3 observation 3:

As we mentioned before, a product will cover some users and dataset P will also do so, so we can ignore the products that already covered by P.

6.4 observation 4:

lemma 1:

Suppose CH is the convex hull defined by original point O, point set $\mathrm{P} = \{\mathrm{p}_1, \mathrm{p}_2, \dots, \mathrm{p}_n\}$ and the projection points from points of P to coordinate plane, then if a product C is in convex hull CH and C could cover weight vector w, then one of P can also cover w.

Here we list an example for lemma 1. Suppose $P = \{p_1, p_2, p_3\}$, $p_1(0.4, 1)$, $p_2(0.6, 0.5), p_3(0.5, 0.6)$, original point O (0, 0), points of P projection point of coordinate plane is (0.4, 0), (0, 1), (0.6, 0), (0, 0.5), (0.5, 0), (0, 0.6). These points form the convex hull CH: $O(0,0) - (0,1) - p_1(0.4, 1) - p_2(0.6, 0.5) - (0.6, 0)$ as shown in figure 5.

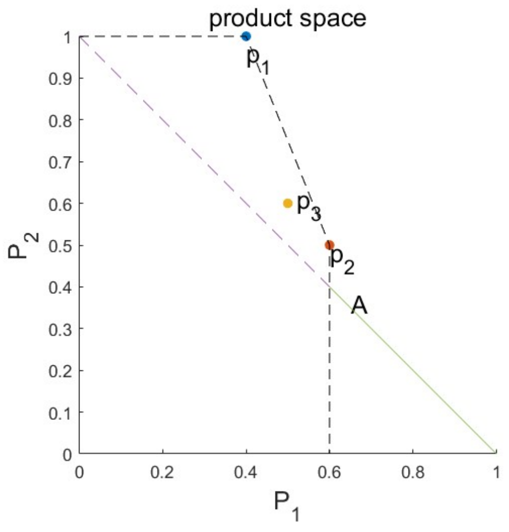


Figure 5

We can see from figure 5 such that some part of the space $C(p) = B$ is in CH (the dashed part).

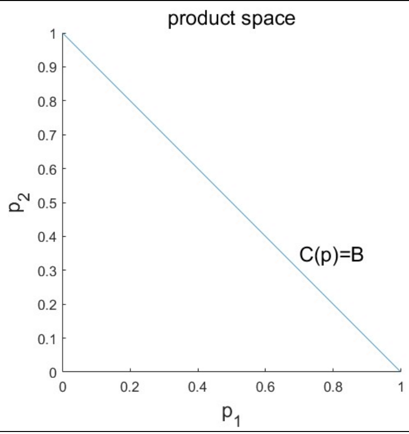
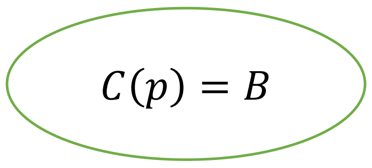
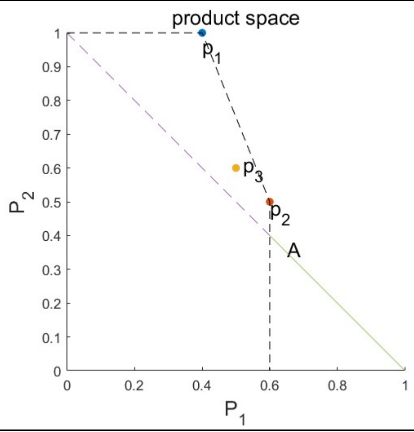
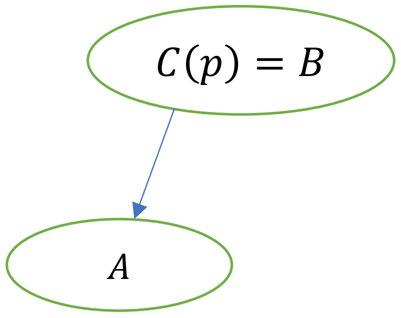
Lemma1 says that for any product p in CH, if p covers a user w, then one of $\{p_1, p_2, p_3\}$ (here more exactly is $\{p_1, p_2\}$) will also cover w.
Because we already have the product dataset P for $kCRM$ and the points in P will form a convex hull CH and CH covers some part of $C(p) = B$, we can ignore the part that covered by CH, which means when we built cell tree, because of the decreasing numbers the candidate options, the root of cell tree would become smaller.

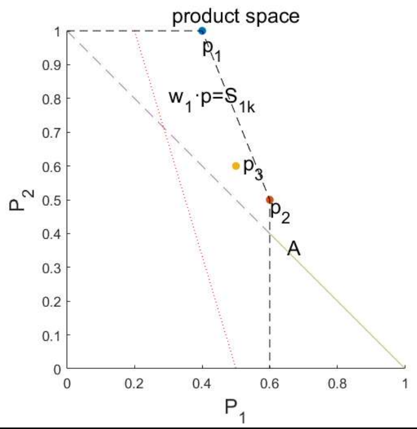
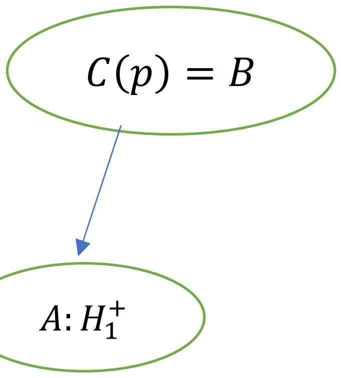
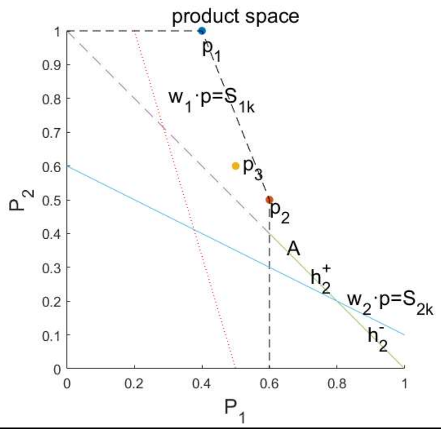
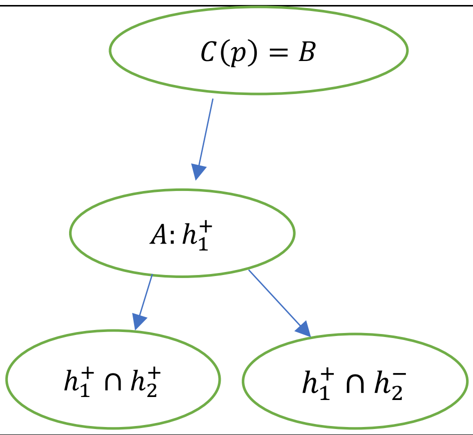
Lemma 1 is inspire from $k - hit\ query$'s lemma 4. The proof of lemma 1 can be found in appendix proof 1.

7. advance solution

The different between base solution and advance solution is the 4 observations mentioned in part 7.

7.1 advance solution example

1. discard all those w that covered by P	
2. only consider the w such tha $w \cdot p = S_k$ intersects with constrain $C(p)$=B	
(suppose after step 2, we only left w_1, w_2)	
3. only consider $C(p) = B$	
	
4. pruning for $C(p)$=B with lemma 1, which means we discard the dashed part, only left part A as shown in figure below.	
	

5. consider w_1, the intersect part is in convex full, directly mark region A as H_1^+	
	
6. insert $w_2 \cdot p = S_{k2}$	
	
7. return $h_1^+ \cap h_2^+$	

7.2 advance solution

1. Preprocessing I, discard all user that P already covers:
$$W = W - \{w \mid \forall w \in W, P \cap TopK(w) \neq \emptyset\}$$
2. we only consider $C(p) = B$
3. Find the biggest convex hull of point set P (name it as B_CH)
4. Preprocessing II, on $C(p) = B$, discard the part which is in convex hull (***see proof 1***)
$$A = \{p \mid C(p) = B\}$$
$$A = A - B\_CH$$
5. Preprocessing III, discard all w such that $w \cdot p = S_{ik}$ doesn't intersect with $C(p) = B$
$$W = W - \{w \mid \forall w \in W, \{p \mid w \cdot p = S_k\} \cap \{p \mid C(p) = B\} = \emptyset\}$$
6. For each w_i in W, based on $w_i \cdot p \leq S_{ik}$ using cell tree divide model space A, where S_{ik} is the score to rank top k respecting to w_i
7. count each divided region's user cover number
8. Return the region that with the greatest cover count

8. conclusion

For the final project we have done all of the theory part including formal problem definition, baseline solution from geometric computation nature, advance solution with observations and proof of observations. The next step is to complete the experiment part.

Reference

1. Bo Tang, Kyriakos Mouratidis, Man Lung Yiu. Determining the Impact Regions of Competing Options in Preference Space. Proceedings of the ACM Conference on Management of Data (SIGMOD), Chicago, Illinois, USA, May 2017.
2. Peng P, Wong RC-W (2015) k-hit query: top-k query with probabilistic utility function. In: Proceedings of the 2015 ACM SIGMOD international conference on management of data (SIGMOD). ACM, pp 577–592

Appendix

Proof 1:

lemma 1:

Suppose CH is the convex hull defined by original point O, point set $\mathrm{P} = \{\mathrm{p}_1, \mathrm{p}_2, \dots, \mathrm{p}_\mathrm{n}\}$ and the projection points from points of P to coordinate plane, then if a product C is in convex hull CH and C could cover weight vector w, then one of P can also cover w.

Proof:

Suppose point O is the original point; point C is inside convex hull; D is the crossover point between OC and convex hull; on convex hull, D is on the hyperplane $H_1H_2 \dots H_d$. Then

$$\overrightarrow{OD} = \lambda_1\overrightarrow{OH_1} + \lambda_2\overrightarrow{OH_2} + \cdots + \lambda_{d-1}\overrightarrow{OH_{d-1}} + \lambda_d\overrightarrow{OH_d}, \quad \sum_{i=1}^{d} \lambda_i = 1;\ \forall i \in [1,d], \lambda_i \geq 0$$

Next,

$$w \cdot \overrightarrow{OD} = w \cdot (\lambda_1\overrightarrow{OH_1} + \lambda_2\overrightarrow{OH_2} + \cdots + \lambda_{d-1}\overrightarrow{OH_{d-1}} + \lambda_d\overrightarrow{OH_d})$$

,

Without loss of generality, suppose $\forall i \in [1,d], w \cdot \overrightarrow{OH_1} \geq w \cdot \overrightarrow{OH_\iota}$

Then

$$\begin{aligned}
& w \cdot \overrightarrow{OC} \\
& < w \cdot \overrightarrow{OD} \\
& = w \cdot \left(\lambda_1\overrightarrow{OH_1} + \lambda_2\overrightarrow{OH_2} + \cdots + \lambda_{d-1}\overrightarrow{OH_{d-1}} + \lambda_d\overrightarrow{OH_d}\right) \\
& \leq w \cdot (\lambda_1\overrightarrow{OH_1} + \lambda_2\overrightarrow{OH_1} + \cdots + \lambda_{d-1}\overrightarrow{OH_1} + \lambda_d\overrightarrow{OH_1}) \\
& = w \cdot \overrightarrow{OH_1}
\end{aligned}$$

That is

$$w \cdot \overrightarrow{OC} < w \cdot \overrightarrow{OH_1}$$

Proof ends.