

# An Interactive System for Capturing Users' Qualitative Preferences in Recommender Systems

Kushal Dave and Malek Mouhoub<sup>a</sup>

*Department of Computer Science, University of Regina, Regina, SK, Canada*

**Keywords:** Recommender System, CP-net, Membership Query, Preference Elicitation, Dictionary.

**Abstract:** We propose a new interactive system for eliciting and learning users' qualitative preferences. These preferences are modelled as a conditional preference network (CP-net). The CP-net is a known graphical model representing qualitative and conditional preferences in a compact form. User's preferences are first captured through a learning method based on membership queries. These preferences are then compiled into a list of conditional preference statements. The CP-net is finally generated from this list. We are also incorporating a collaborative technique so that when a CP-net of a given user is generated, the latter will receive suggestions based on similarities with other users.

## 1 INTRODUCTION

A recommender engine is a tool that collects a large amount of data related to users' satisfaction and desires (Ikemoto and Kuwabara, 2019; Bobadilla et al., 2013; Karimi et al., 2018). These data are then used to provide suggestions and recommendations to other users. For instance, in social media such as Facebook and LinkedIn users are given recommendations on friends and connections respectively. On the other hand, streaming services such as Netflix recommend movies meeting users interest, online shopping systems like Amazon and eBay suggest products of similar interest, and tinder provides personalized recommendation for partners. The recommender system gives users suggestions of things with a similar or relative interest that they might like. All these recommender systems work primarily with quantitative preferences (expressed as numerical ratings) using Collaborative, Content-based or Hybrid Filtering (Balabanovic and Shoham, 1997; Mohammed et al., 2013; Adomavicius and Tuzhilin, 2005). There are many scenarios however where qualitative preferences are preferred. The latter express ordinal preferences which might be more natural than quantitative preferences. Indeed, it is often more appropriate to ask which option is more preferred rather than providing specific scores for each option. This motivated us to explore recommendation systems in the

context of qualitative preferences. In this regard, we propose a new interactive GUI for eliciting and learning users' qualitative preferences through membership queries. The learned preferences are then represented in a compact manner user the CP-net graphical model.

The remaining of the paper is structured as follows. The next section reports on background materials related to recommender systems, preference reasoning and preference learning. Section 3 describes our proposed approach and presents the proposed GUI. Finally concluding remarks and ideas for future works are listed in Section 5.

## 2 BACKGROUND

### 2.1 Preferences and Recommender Systems

In this fast-growing world, you will get alternatives and options for everything. As many alternatives and options are there, it is often difficult to choose from. Qualitative preferences over a set of available options order these options such that a more desired option comes before a less desirable option. The items in this collection of options will be referred to as the outcome.(Brafman and Domshlak, 2009). Preferences are now part of the routine. Knowingly or unknowingly, everyone has preferences in their day-

<sup>a</sup>  <https://orcid.org/0000-0001-7381-1064>

to-day life. For example, one prefers tea over coffee. Preferences direct our actions and everyday decisions. In addition, preferences help to make practical reasoning. Preferences have been studied in Logic, economics, operations research, Psychology, philosophy, game theory, decision theory, and Artificial Intelligence (Doyle, 2004; Thomason et al., 2018; Thomason, 2014). Preferences have an essential role in knowledge representation (Boutilier et al., 2004), multi-agent systems (Shoham and Leyton-Brown, 2009), recommender systems (Kostkova et al., 2014; Adomavicius and Tuzhilin, 2005; Karpus et al., 2016; Carvalho and Belo, 2016), and many more (Thomason et al., 2018). In this context, a common goal is to manage preferences when making automated decisions as well as when making users' recommendations (Alanazi et al., 2019). The latter rely on quantitative ratings elicited from users. Recommendations are then suggested to other users according to the following models: conditional filtering, collaborative filtering, content-based filtering, and hybrid filtering techniques (Balabanovic and Shoham, 1997; Mohammed et al., 2013; Adomavicius and Tuzhilin, 2005). More precisely recommendations are based on collected scores and discovering similarities between items in an item-based model, between users in a user-neighbourhood model, or between users and items in a user-item based model (Deshpande and Karypis, 2004; Adomavicius and Tuzhilin, 2005).

## 2.2 CP-net

In (Boutilier et al., 2004), Boutilier et.al. proposed a qualitative graphical representation of conditional qualitative preferences, under a ceteris paribus (all else being equal) interpretation. The model is called the Conditional Preference Network (CP-net) and is capable of representing qualitative preferences in a compact, intuitive and structural graphical manner. Here, preferences are with or without conditions. Preferences are defined with the succeeds " $\succ$ " or precedes " $\prec$ " symbol. If  $x$  is preferred over  $y$  with depends on  $z$  then that will be denoted with the following preference statement,  $z : y \succ x$  or  $z : x \prec y$

**Example 2.1.** "I like coke when pizza is chosen, and gingerale when burritos is selected" is a conditional preference as the selection of coke and gingerale, depends on the chosen main dish. This example will be denoted with the following preference statements.  
 Burritos: gingerale  $\succ$  coke  
 Pizza: coke  $\succ$  gingerale

**Definition 1.** A CP-net over a set of variables  $V = X_1, \dots, X_n$  is a directed graph  $G$  whose nodes (vari-

ables) are annotated with conditional preference tables  $CPT(X_i)$  for each  $X_i \in V$ . Each conditional preference table  $CPT(X_i)$  associates a total order  $\succ_u^i$  with each instantiation  $u$  of  $X_i$ 's parents  $Pa(X_i) = U$  (Boutilier et al., 2004).

CP-net is a compact representation of conditional preference statements. The CP-net can be expanded to an induced graph, where nodes represent all possible outcomes while arcs denote the dominance relation between them.

**Example 2.2.** Consider the CP-net in Figure 1 that expresses preferences over meal configurations. This network consists of two variables  $F$  and  $D$ , standing for the food and drink, respectively. Here, burritos ( $F_b$ ) is unconditionally preferred to Pizza ( $F_p$ ), while the preference between coke ( $D_c$ ) and gingerale ( $D_g$ ) is conditional on the value assigned to  $F$ .

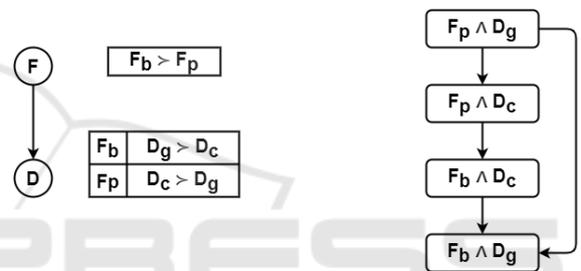


Figure 1: A CP-net (left) and its corresponding induced preference graph (right).

**Example 2.3.** Consider the CP-net in Figure 2 that expresses preference for dressing configurations. This network consists of three variables  $T$ ,  $B$ , and  $S$ , standing for the top, bottom, and shoes, respectively. Light blue top ( $T_b$ ) is strictly preferred to white top ( $T_w$ ), while the preference between Khaki ( $B_k$ ) and the Navy ( $B_n$ ) bottom is conditional on the selection for tops. Navy bottoms is preferred with light blue tops while khaki bottoms is preferred with white tops. The preference for black ( $S_l$ ) and burgundy ( $S_u$ ) shoes is conditional on the selection of bottom; black shoes are preferred with navy bottom and burgundy shoes are preferred with khaki bottom. Below is the representation of CP-net and induced graph.

The graphs representing the above CP-nets are acyclic. In cyclic CP-nets, preferences are preferred over one another in a cycle. Cyclic CP-nets can be inconsistent (leading to a cycle in the corresponding induced graph). Figure 3 (Domshlak, 2002) shows two CP-nets represented with the same graph depicted in Figure 3 (e). The first one with the CP-table in Figure 3 (a) is consistent as its induced graph shown in Figure 3 (b) is acyclic. However, the CP-net with the

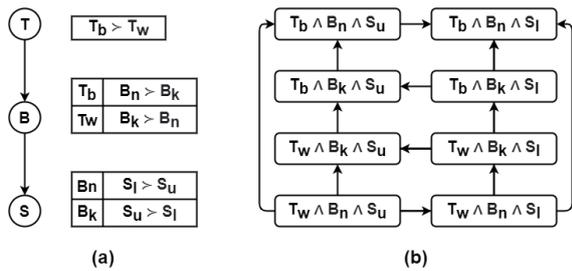


Figure 2: A CP-net representing the preferences for Top, Bottom & Shoes (left) and its induced preference graph (right).

CP-table in Figure 3 (c) is inconsistent as its induced graph in Figure 3 (d) is cyclic.

While cyclic CP-nets come with a challenge of checking for their consistency, they might be useful to express preferences in a natural way, in some applications (Boutilier et al., 2004; Domshlak, 2002).

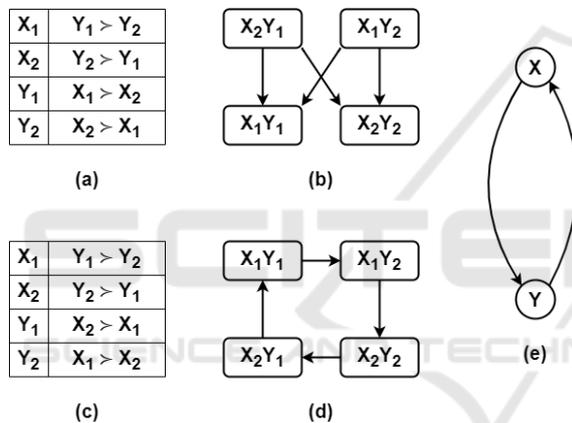


Figure 3: Consistent and inconsistent cyclic CP-nets.

Another type of CP-net are separable CP-nets. A separable CP-net is one in which the dependency graph has no edges. Essentially, this means that preferences over the domain of every attribute are independent of preferences over the domain of any other attribute.

### 2.3 CP-net Learning

There are mainly two ways to learn CP-nets (Fürnkranz and Hüllermeier, 2011): passive learning and active learning. Passive learning corresponds to learning from historical data. Basically, the learner tries to concentrate on a small amount of data that was previously supplied by the user to construct the CP-net. Active learning, on the other hand, is typically employed in real-time with the user when asking questions about their preferences and constructing a CP-net based on their responses (Chevalyere et al.,

2011).

In (Chevalyere et al., 2011), the authors addressed a wide range of challenges pertaining to CP-net learning. At first, the idea is to develop a CP-net such that the user’s preference relation corresponds to its induced preference relation, or is a closer approximation to the user’s complete preference relation.

In (Koriche and Zanuttini, 2010), the authors introduced two algorithms to learn CP-nets with equivalence and membership queries. It was stated that CP-nets are not learnable with equivalence query alone, whereas acyclic CP-nets are attribute-learnable when both equivalence and membership queries are available. Furthermore, the authors introduced two separate algorithms to learn acyclic CP-nets and tree CP-nets using equivalence and membership queries. Basically, the main idea is to start with an equivalence query and then search for parents accordingly using membership queries. This was the first attempt using active learning for graphical preference languages.

In (Guerin et al., 2013), the authors introduced the process of learning CP-net in two separate phases. The first phase establishes a separable CP-net base by asking the user to specify a default choice for each accessible variable and therefore creates the initial impression of the CP-net. In the second step, the CP-net is refined by eliciting user preferences for pairs of outcomes and establishing conditional relationships.

In (Alanazi et al., 2019), the authors report on the first study that exactly calculates the sample complexity for learning qualitative preferences, through acyclic CP-nets. The bounds of the VC dimension, the teaching dimension and the recursive teaching dimension have been defined.

In (Allen, 2015; Allen et al., 2017) the authors describe a local search algorithm for tree-shaped CP-nets. They also proved that the algorithm also works in the presence of noise. The proposed algorithm follows passive learning and uses pairwise comparison of data to generate a target CP-net. The authors introduced encoding for acyclic CP-nets with constraints on indegree and multivalued domains, and a separate encoding that is specific to tree structured CP-nets with binary domains and complete tables. While previous research from (Alanazi et al., 2019) also introduced a learning algorithm for tree structure CP-net, the method in (Allen, 2015; Allen et al., 2017) is capable of handling noisy comparison sets robustly.

In (Mohammed et al., 2013), the authors proposed a new interactive system that enables users to express their needs and desires. Managing constraints and preferences is one of the major components of the proposed system. The user is allowed to select preferences as qualitative or quantitative or both. Data min-

ing association rules technique is used for preference suggestions based on learning from other clients.

In (Labernia et al., 2017), the authors suggest an online learning technique of acyclic CP-nets where observations arrive as a stream and cannot be memorized as in a recommender system whereas user's clicks could generate them. Noise can be caused by many factors: distracted user, corrupted data, unobserved variables, etc. The author introduced an algorithm to observe CP-net comparisons online. The author analyzed the proposed algorithm by showing its usefulness using synthetic and real-world datasets.

In (Labernia et al., 2018), the authors introduced a query-based learning CP-net approach, where an inspiration from (Labernia et al., 2017) was used to decompose the procedure into general learning phase and parent search phase, based on equivalence and membership queries.

In (Liu et al., 2018), the authors introduced dependent degrees to calculate the dependency relationship among attributes. In this method, the authors use passive learning for generating CP-nets. Additionally, three algorithms were proposed. The first algorithm filters the preference database, while the second algorithm finds the degree of dependence of pairwise attributes which uses a filtered preference database from the previous algorithm. The third algorithm task is to generate the CP-net. This algorithm uses the previous tasks to calculate the degree of attribute dependence for each pair. Then this method is applied on a database with 18 attributes. The results for generating CP-nets for those data where reported.

In (Ali, 2019), the author propose an algorithm to aggregate a set of CP-nets into a single summary CP-net. Generated CP-nets most of the time are cyclic in nature as each user has their own preferences. Along with generating CP-nets, the algorithm also calculates relative swap disagreements (disagreements over preference selection by each user on the basis of total swap disagreements). Generated CP-net is more focused to get an idea of overlapping or disagreements of user preferences. And also these algorithms are applicable on a set of CP-nets only.

## 2.4 Membership Query with Swap Examples

As explained in (Chevaleyre et al., 2011), in active preferences learning, the user has preferences in mind among the options but does not know how to represent that structure in CP-net. In this case, the user helps the learner to answer preferences through Membership Queries (MQs). More precisely, the learner asks the user MQs such as: “do you prefer ‘x’ over

‘y’?” Where ‘x’ and ‘y’ are outcomes chosen by the learner. The width of MQ (x,y) needs to be set. This width is the number of entries (attribute values) on which the outcomes x and y differ. A MQ of width 1 is called a swap membership query. The primary issue with MQs is that they are rarely comparable with a high width value; otherwise, the learner is forced to ask polynomial queries. The following is an example explaining membership queries with swap examples. Let us recall the dressing example where the user is picking Top, Bottom & Shoes from different colors. Assume we choose light blue top( $T_b$ ), navy bottom( $B_n$ ), black shoes ( $S_l$ ) over white top( $T_w$ ), navy bottom( $B_n$ ), black shoes ( $S_l$ ). Here, among all selections only Top light blue differs from white and all other characteristics are the same. Similarly, if we choose light blue top( $T_b$ ), navy bottom( $B_n$ ), black shoes ( $S_l$ ) over light blue top( $T_b$ ), navy bottom( $B_n$ ), burgundy( $S_u$ ) then in that case black shoes ( $S_l$ ) is different from burgundy ( $S_u$ ) and the remaining characteristics are the same. During the MQs learning process, the Users will be asked all possible combinations to get precise results of the target CP net. In most cases, limiting the characteristics helps to reduce membership queries and also helps users by giving them limited preferences instead of polynomial queries.

## 2.5 Dictionary

A dictionary is a data structure used to store a collection of items. This data structure allows the use of a wide variety of data types including strings and tuples as the index. These indices are known as keys, and are used to refer to a particular value in a collection. This means that the “key” and “value” of each entry in a dictionary are paired up and stored as a series of key-value pairs. This is a container for a collection of objects structured in key-value pairs that can be utilized for a variety of purposes <sup>1</sup>.

Many operations are often supported by dictionaries, such as retrieve a value (based on the programming language, attempting to retrieve a missing key may provide a default value or throw an exception), inserting or updating a value (typically, if the key does not exist in the dictionary, the key-value pair is inserted; if the key already exists, its corresponding value is overwritten with the new one), remove or delete a key-value pair, and test or verify for existence of a key. Most programming languages with dictionaries support iteration over the keys or values in a dictionary. Dictionary entries are written in curly brackets (). A colon (:) separates the

<sup>1</sup><https://infx511.github.io/dictionaries.html>

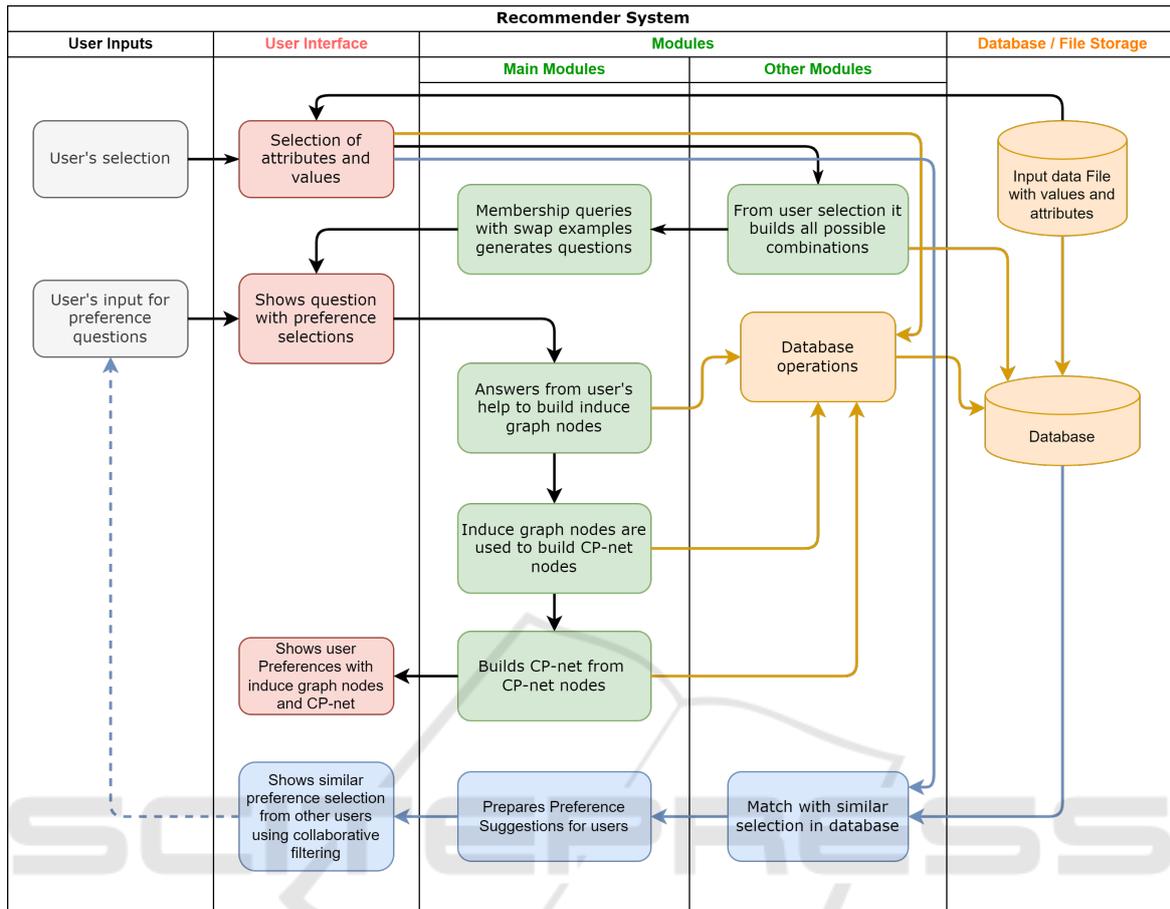


Figure 4: System Architecture.

key and value in each key-value pair, and a comma (,) separates each element (pair) in the dictionary. Below is an example of a Dictionary.

```
{ 'f1': 'apple',
  'f2': 'orange',
  'f3': 'banana' }
```

Here  $f_1, f_2, f_3$  are keys to apple, orange and banana respectively.

Dictionary in python supports different operations including easy access to key and values, adding or changing dictionary element, removing element from dictionary, combining two dictionaries, and also provides in-built functions of length, sort, compare, copy, etc.

The following example highlights one of the comprehension features of python using dictionaries and range function.

```
Qubes = { x: x*x*x for x in range (5) }
print(Qubes)
```

```
#Output
{0: 0, 1: 1, 2: 8, 3: 27, 4: 64}
```

This above code can be written as

```
Qubes = {}
for x in range(5):
    Qubes[x] = x*x*x
print(Qubes)
```

```
#Output
{0: 0, 1: 1, 2: 8, 3: 27, 4: 64}
```

A Dictionary can keep several values in an integrative manner, which is the reason we adopted it to store input, output, and in between values in our proposed system. In the following sections, we will show how these dictionaries store data and pass them to functions for manipulation and creating desired output.

### 3 PROPOSED SYSTEM

The architecture of our proposed system is depicted in Figure 4. The aim of the system is to generate a CP-net by eliciting user's preferences, and using dictionaries and MQs with swap examples. The first step is for the user to select the list of attributes and values

they are interested in. The user will then receive a set of recommendations from similar users. The preferences of the user will then be elicited through a set of MQs. The induced graph will then be incrementally build while receiving user's answers. Finally the CP-net together with the induced graph will be presented to the user, and added to the database for future recommendations.

Let us illustrate the overall process using a variant of Example 2.3. Assume we want to learn user's preferences for an outfit. When given the list of possible items, we assume the user has selected Top (T), Bottom (B) and Shoes(S). The user will then be asked to select the possible colors for each. This is done through the related GUI of our system, depicted in Figure 5. Here, the user can either select the colors to consider (for each item) or choose not to select any color. In the latter case, all selected colors will be considered.

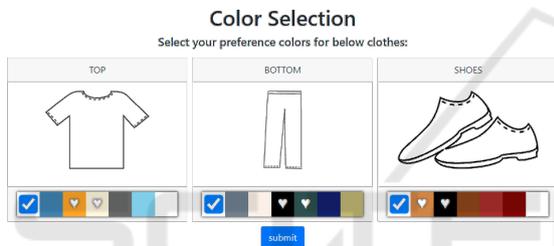


Figure 5: Color Selection.



Figure 6: Recommender System - User Selection.

Given the user's selection for items and colors, the system will display these choices as shown in Figure 6. The user will then be presented with a list of MQs and will provide the answers accordingly, as depicted in Figure 7.

Note that in the worst case scenario, the number of queries to consider is equal to the total number of swaps,  $n \times d^{n-1}$ , where  $n$  is the number of attributes and  $d$  their domain size. In the case where we are dealing with binary attributes values then the total number is  $n \times 2^{n-1}$ . This number is justified as follows. If we have  $n$  variables then, if we decide to flip one variable value, we will have  $2^{n-1}$  possibilities (corresponding to all possible combinations for the  $n - 1$  variable values). When considering each of the  $n$  variables, the total number will be  $n \times 2^{n-1}$ .

The system will then compile the MQs answers and summarize them in an induced graph, as listed in Figure 8. Here, each of the outcomes on the left



Figure 7: Recommender System - Preference Selection.

of each row will dominate all the outcomes on the right. For instance, the outcome on the left of the first row, (Top = Blue, Bottom = Linen, Shoes = Black), dominates the three outcomes on the right of the same row.

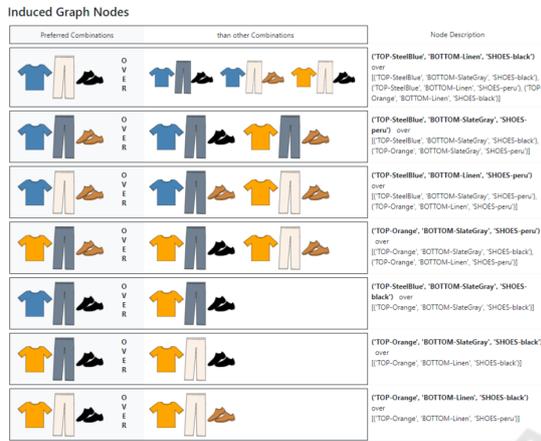


Figure 8: Preferences - Induced Graph Nodes.

Using the induced graph depicted in Figure 8, we will extract the (conditional) preference statements for each of the three items. The result is listed in Figure 9.

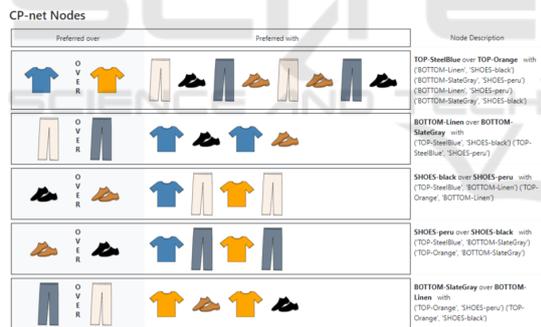


Figure 9: Extracted preference statements.

Finally, from the preference statement in Figure 9, we will build the CP-net depicted in Figure 10.

#### 4 CONCLUSION AND FUTURE WORK

We have proposed a new system for learning user’s qualitative preferences through a friendly GUI. The target model to learn is a CP-net and the learning process is conducted through membership queries. The system can be useful for preference elicitation, decision making systems, reasoning tasks, including e-commerce, combinatorial optimization, multi-agent

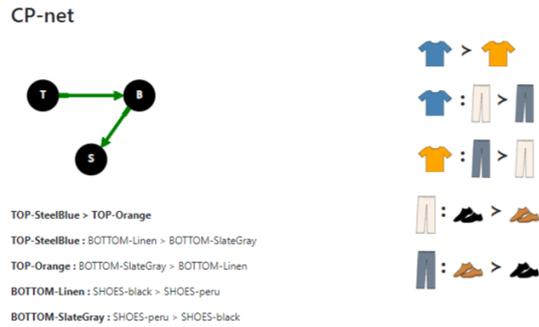


Figure 10: Target CP-net to learn.

planning and agreement, etc. This is the first attempt using features like dictionary, user interface, SVG, membership query and membership queries to produce accurate CP-net on fly. The current version of the system is focused on specific data but this can be expanded, customized and moulded to accommodate different scenarios. Another future direction is to look for new techniques to reduce the exponential number of MQs needed to learn the CP-net. One way is to elicit constraints from the user, in addition to preferences. This will help eliminating those MQs that are inconsistent with the learned constraints. The latter process can be effective when using the Constraint Satisfaction Problem (CSP) framework with constraint propagation and variable ordering heuristics (Dechter et al., 2003; Mouhoub et al., 2021). Following on the work in (Alkhiri and Mouhoub, 2022), we also plan to define new similarity measures between CP-nets as this will help provide user’s suggestions based on the their target CP-net. In this regard, we will consider aggregating CP-nets methods (Ali, 2019) that are needed every time we need to add a new elicited CP-net. Aggregating CP-nets can be represented using probabilistic reasoning as done in (Ahmed and Mouhoub, 2017). Finally we will rely on a new CP-net variant (Ahmed and Mouhoub, 2020) to consider both habitual behavior (represented as conditional preferences) and genuine decisions.

#### REFERENCES

Adomavicius, G. and Tuzhilin, A. (2005). 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*, 17:734–749.

Ahmed, S. and Mouhoub, M. (2017). Probabilistic tcp-net. In Mouhoub, M. and Langlais, P., editors, *Advances in Artificial Intelligence - 30th Canadian Conference on Artificial Intelligence, Canadian AI 2017, Edmonton, AB, Canada, May 16-19, 2017, Proceedings*, volume

- 10233 of *Lecture Notes in Computer Science*, pages 293–304.
- Ahmed, S. and Mouhoub, M. (2020). Conditional preference networks with user's genuine decisions. *Comput. Intell.*, 36(3):1414–1442.
- Alanazi, E., Mouhoub, M., and Zilles, S. (2019). The complexity of exact learning of acyclic conditional preference networks from swap examples. *Artificial Intelligence*, 278:103182.
- Ali, A. M. H. (2019). Summarizing conditional preference networks.
- Alkhiri, H. and Mouhoub, M. (2022). Constrained cp-nets similarity. In Rocha, A. P., Steels, L., and van den Herik, H. J., editors, *Proceedings of the 14th International Conference on Agents and Artificial Intelligence, ICAART 2022, Volume 3, Online Streaming, February 3-5, 2022*, pages 226–233. SCITEPRESS.
- Allen, T. (2015). Cp-nets: From theory to practice. In *International Conference on Algorithmic Decision Theory*, pages 555–560.
- Allen, T., Siler, C., and Goldsmith, J. (2017). Learning tree-structured cp-nets with local search. In *FLAIRS 2017*.
- Balabanovic, M. and Shoham, Y. (1997). Fab: Content-based, collaborative recommendation. *Communications of the ACM*, 40:66–72.
- Bobadilla, J., Ortega, F., Hernando, A., and Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-based systems*, 46:109–132.
- Boutilier, C., Brafman, R., Domshlak, C., Hoos, H., and Poole, D. (2004). Cp-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. *J. Artif. Intell. Res. (JAIR)*, 21:135–191.
- Brafman, R. and Domshlak, C. (2009). Preference handling - an introductory tutorial. *AI Magazine*, 30:58–86.
- Carvalho, M. and Belo, O. (2016). Enriching what-if scenarios with olap usage preferences. In *Proceedings of the 8th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management - Volume 1: KDIR, (IC3K 2016)*, pages 213–220. INSTICC, SciTePress.
- Chevalyere, Y., Koriche, F., Mengin, J., and Zanuttini, B. (2011). Learning ordinal preferences on multiattribute domains: the case of cp-nets. *Preference Learning*.
- Dechter, R., Cohen, D., et al. (2003). *Constraint processing*. Morgan Kaufmann.
- Deshpande, M. and Karypis, G. (2004). Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems - TOIS*, 22:143–177.
- Domshlak, C. (2002). Modeling and reasoning about preferences with cp-nets.
- Doyle, J. (2004). Prospects for preferences. *Computational Intelligence*, 20:111–136.
- Fürnkranz, J. and Hüllermeier, E. (2011). Preference learning: An introduction. In *Preference Learning*. Springer-Verlag.
- Guerin, J., Allen, T., and Goldsmith, J. (2013). Learning cp-net preferences online from user queries. volume 8176.
- Ikemoto, Y. and Kuwabara, K. (2019). On-the-spot knowledge refinement for an interactive recommender system. In *ICAART (2)*, pages 817–823.
- Karimi, M., Jannach, D., and Jugovac, M. (2018). News recommender systems—survey and roads ahead. *Information Processing & Management*, 54(6):1203–1227.
- Karpus, A., Noia, T. D., Tomeo, P., and Goczyla, K. (2016). Rating prediction with contextual conditional preferences. In *Proceedings of the 8th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management - Volume 1: KDIR, (IC3K 2016)*, pages 419–424. INSTICC, SciTePress.
- Koriche, F. and Zanuttini, B. (2010). Learning conditional preference networks. *Artificial Intelligence*, 174:685–703.
- Kostkova, P., Jawaheer, G., and Weller, P. (2014). Modeling user preferences in recommender systems. *ACM Transactions on Interactive Intelligent Systems*, 4:1–26.
- Labernia, F., Yger, F., Mayag, B., and Atif, J. (2018). Query-based learning of acyclic conditional preference networks from noisy data.
- Labernia, F., Zanuttini, B., Mayag, B., Yger, F., and Atif, J. (2017). Online learning of acyclic conditional preference networks from noisy data. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 247–256.
- Liu, Z., Zhong, Z., Li, K., and Zhang, C. (2018). Structure learning of conditional preference networks based on dependent degree of attributes from preference database. *IEEE Access*, PP:1–1.
- Mohammed, B., Mouhoub, M., Alanazi, E., and Sadaoui, S. (2013). Data mining techniques and preference learning in recommender systems. *Computer and Information Science*, 6(4):88.
- Mouhoub, M., Marri, H. A., and Alanazi, E. (2021). Exact learning of qualitative constraint networks from membership queries. *CoRR*, abs/2109.11668.
- Shoham, Y. and Leyton-Brown, K. (2009). *Multiagent Systems*.
- Thomason, R. H. (2014). The formalization of practical reasoning: Problems and prospects. *FLAP*, 1(2):47–76.
- Thomason, R. H., Gabbay, D. M., and Guentner, F. (2018). The formalization of practical reasoning: Problems and prospects. In *Handbook of Philosophical Logic*, pages 105–132. Springer.