Generic Menu Optimization for Multi-profile Customer Systems

Jeyhun Karimov
Computer Eng. Dept.
TOBB University of Economics and Technology
Ankara,Turkey
jkarimov@etu.edu.tr

Murat Ozbayoglu
Computer Eng. Dept.
TOBB University of Economics and Technology
Ankara,Turkey
mozbayoglu@etu.edu.tr

Bulent Tavli
Electrical & Electronics Eng. Dept.
TOBB University of Economics and Technology
Ankara,Turkey
btavli@etu.edu.tr

Erdogan Dogdu
Computer Eng. Dept.
TOBB University of Economics and Technology
Ankara,Turkey
edogdu@etu.edu.tr

***Abstract*—The use of optimal ATM menu structuring for different customer profiles is essential because of usability, efficiency, and customer satisfaction. Especially in competitive industries such as banking, having optimal user interface (UI) is a must. Determining the optimal menu structure is generally accomplished through manual adjustment of the menu elements. However, such an approach is inherently flawed due to the overwhelming size of the optimization variables' search space. Previous studies on menu optimization either are based on customer questionnaires or made for only a specific menu type using heuristic approaches (*i.e.*, not generic). In this paper, we propose an systematic optimization method for menu structuring problem through a novel Mixed Integer Programming (MIP) framework. Our optimization approach is not specific to a predetermined menu class, on the contrary, the MIP model is designed to be a generic optimization framework that can be applied to a wide range of menu optimization problems. We evaluated the performance gains on a dataset of actual ATM usage logs for a period of 18 months consisting of 40 million transactions. We validated our results with both simulation application and mining of existing data logs. The results show that the proposed optimization approach provides significant reduction in the average transaction completion time and the overall click count.**

***Keywords*-menu optimization, automated teller machine, ATM, mixed integer programming**

I. Introduction

Automated Teller Machines (ATMs) play an increasingly important role in banking services due to round the clock availability and cost reduction. However, it is challenging to design customized ATM menus for the particular needs of customers, and these menu structures are usually static, not adaptive to location or customer profiles. One of the reasons internet banking is becoming more popular over ATMs is its simplicity, user friendliness, and customer adaptiveness.

In this paper, we propose a novel solution to the optimal menu construction problem that is completely generic and not based on manual optimizations. Here, the generic solution refers to not being based on any user proposal or any questionnaire of any type. Although we used ATM real data to test and implement our solution, the implementation is not restricted to only this particular area but any type of menu optimizations is possible. We used actual ATM log data to confirm and validate our solution.

II. Related work

User interface design is becoming more important in the product development process. This phenomenon especially becomes crucial when the customers are non-expert ordinary users. Companies spend considerable amount of resources in order to come up with the most attractive user interaction experience as customer satisfaction, user-friendliness, and the level of ergonomics can be decisive factors in the users' choice for a particular product. Furthermore, the massive increase in the mobile applications and online services in the recent years resulted in the necessity for providing the best possible user interface for the companies in order to improve their marketshare.

Increasing the usage effectiveness and the service quality of ATMs for end users has been the subject of many studies from several different aspects. There have been several studies to optimize the menu structure on ATM [1] to reduce the queues and click counts. The main task is to propose a menu that is optimal or have less click count to finish the required task to all users. In [2], authors concentrated on 7 main functions of ATM and compared their usability in different bank menus using questionnaire and only one bank received the highest score on most of the usability criteria. Another research was concentrated on new hierarchical menu design of ATM menu based on seven most frequently used tasks [3]. Similarly in [4], authors show a procedure of mining ATM data set and extracting useful data from it to make adaptive ATMs. In [5], on the other hand, authors concentrated on optimizing the ATM menu structure for the needs of the student community. Another research done on reducing the ATM queues is [6], in which the authors propose a model in which we can register the amount to be withdrawn from ATM even before physically moving to any ATM centre. Yet another research that concentrates on optimizing and making more usable online banking menus is [7]. Here, authors, concentrated on making more usable online banking menu for customers of a particular bank. To increase the usability of ATMs by illiterate humans authors proposed a solution in [8] and tested on a group of six

Dutch functionally illiterate people. Apart from usability, in [9], the authors concentrated on its security while using the menu. As a result, user satisfaction was enhanced and error rate was reduced mainly due to the introduction of profile-based functions. Another research was done on the biometric verification on ATM and their usability using iris verification technology. As a result of the authors' interventions, by the implementation of this technology, from a user's perspective, significant improvement was achieved. The authors of [10], describe an evaluation of two websites with the same content but different interface styles. As a result, they conclude that the perception of information quality is affected by the interaction style implemented in the interface. Mehdi et al. concentrated on usability of mobile interfaces for novice and low-literacy users [11]. Another research looks at the attitudes towards ATMs and alternative ATM interfaces (a speech-based interface and an icon-based interface) of literate and semi-literate groups [12].

Our work differs from the previous work in several aspects. First, we built a novel MIP framework for ATM menu optimization which is generic enough to address a wide range of ATM menu optimization problems unlike the heuristic and manual based approaches in the literature which are designed for specific problems. Second, we validated the effectiveness of our approach by using a fairly large dataset of ATM usage logs, consisting of 40 million transactions for a period of 18 months. The only work in the literature that deals with ATM menu optimization using an actual and large dataset is [4] where they used a dataset of 10 million transactions which is not generic solution.

III. Problem Definition

The first step in optimizing the menu structure for a particular data set is to preprocess the huge amount of data to be able to perform a scalable optimization process. Details of the preprocessing are given in Section V. Nevertheless, at this point it is sufficient to state that to solve our problem, we are given:

- overall menu structure of a particular ATM card
- click count of each menu item

Once such data is available, the goal is to come up with a menu structure that is optimal for a particular objective function and at the same time, is easy to comprehend and also usable by non-expert users. The particular objective can be to minimize the click count to achieve a certain task, or to minimize the overall time to complete the steps, or to minimize the eye movement and visibility to reach menu items. For this problem, we define all menu structure as a tree (a hierarchical structure) for easy representation; the top level tree node represents the first menu shown on ATM screen right after entering the PIN code. A sample top level menu node is shown in Figure 1. An example of the overall tree structure for a particular ATM menu is illustrated in Figure 2. Here the focus is mainly to demonstrate the overall concept, therefore, we ommited some leaves and subtrees for brevity.

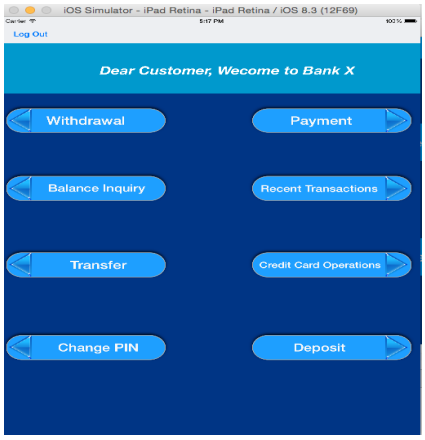

Figure 1: Sample Main Screen which comes after entering PIN

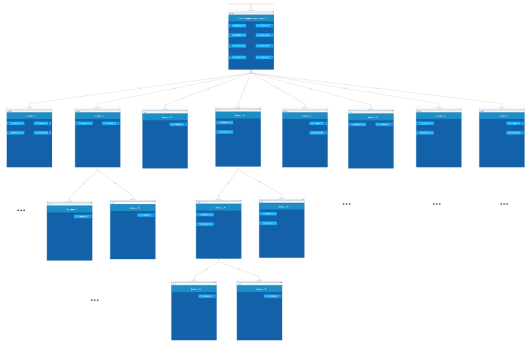

Figure 2: Sample tree menu structure of ATM.

Considering the related work on ATM menu optimization, to the best of our knowledge, there is no generic solution to this problem. It is plausible to assume that improvements can be achieved by manual modifications in small and non complex problems (*i.e.*, it is possible to improve the objective by simple visual inspection).

Consider the example menu structure in Figure 3. For this example each ATM screen can contain at most 3 menu items (a tree node can have at most 3 child nodes). The leaf nodes are represented as black boxes in the tree. Each menu has an ID from 0 (the top level) to 13 (the bottom right item) shown on the top of the menu or menu item. And the leaf nodes have the click counts attached to the left side of the box. Since we are only interested in optimizing the leaf nodes, the click counts of the non-leaf menu items are not considered. In Figure 3, it is possible to improve the objective function value by replacing the upper level low-click nodes with high-click nodes. For example, one can bring the node 9 or node 10 to Main Screen (under node 0). However, this is not the optimal solution. Moreover, when

bringing the high-click nodes to upper levels, we have to be make sure that maximum menu nodes per screen (maximum child count for a node) is not exceeded. It is clear that such simple heuristics do not always give the best possible solution and it is difficult for complex menu structures. The optimum solution for this menu is shown in Figure 4. Here we bring two new node types: Combiner (romb shape in Figure) and Optimizer (triangle shape in Figure). As we will discuss the solution in detail in Section IV, Optimizers are the non-visible menu nodes, node 6 is a direct child of the node 0 (Figure 4). Our solution can produce a number of Optimizers and Combiners at all levels.

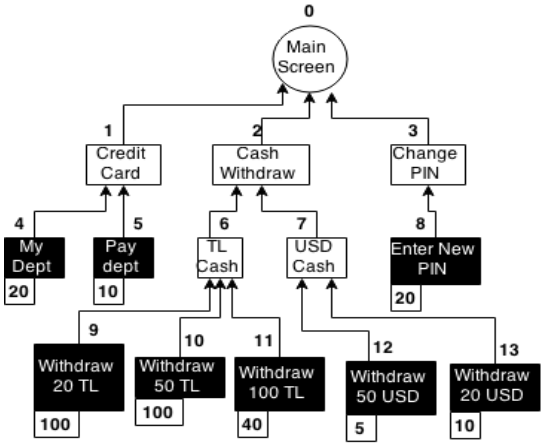

Figure 3: Before optimizing simple ATM menu.

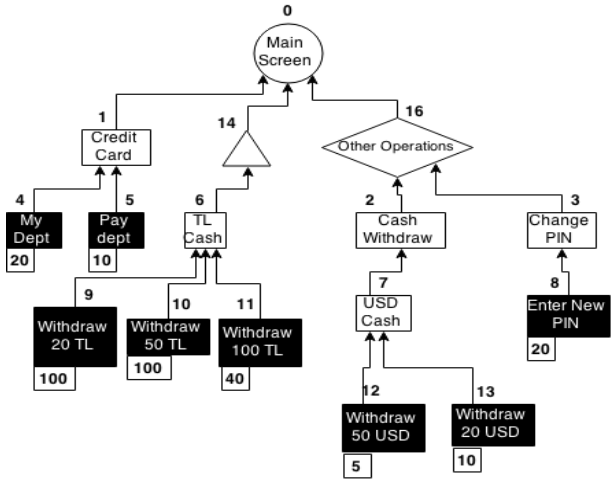

Figure 4: After optimizing ATM menu with only 1 Optimizer (triangle) and 1 Combiner (romb) in level 1.

IV. Proposed model

Before focusing on our MIP model, we show the overall concept of the solution. For example, Figure 5 is the intermediate step between Figures 3 and 4. We introduce the

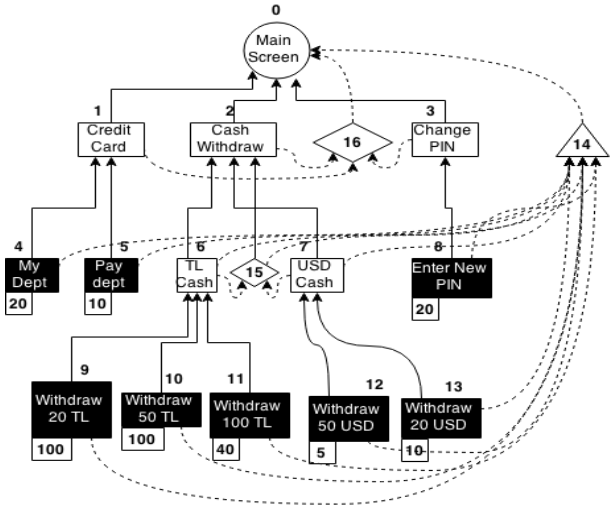

Figure 5: Overall intuition behind menu optimization.

Table I: Optimizers and Combiners

Optimizer (triangle)	**Combiner (romb)**
Used to bring lower layer node to upper layer, like shortcut	Used when max number of menu items per screen is exceeded, to combine some of them under ”Others” menu item.
Except leaf layer, any number of optimizers can be used in any layer	Except leaf layer, any number of combiners can be used in any layer
Only one outgoing arc is allowed	Only one outgoing arc is allowed
Optimizers are connected to all lower layer menu nodes in the same sub-tree, but model allows only one incoming arc, the optimum one, because one shortcut cannot keep more than one menu item.	Combiners are connected to all same layer menu items and model can select any of them in any number.

new nodes as Combiners (rombs) and Optimizers (triangles). These nodes are explained and compared in Table I.

We formulate the problem by using an MIP formulation and the definitions of constants and variables are given in Table II.

Equations describing the MIP problem are as follows:

$$a_{ij} \in \{0,1\} \ \forall i \in S, \forall j \in OUT_i \tag{1}$$

$$f_{ij} \geq 0 \ \forall i \in S, \forall j \in OUT_i \tag{2}$$

$$\sum_{j \in OUT_i} f_{ij} - \sum_{j \in IN_i} f_{ji} = \begin{cases} C_i, & \forall i \in S_E \\ -\sum_{i \in S_E} C_i, & i = S_0 \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

$$f_{ij} \leq M a_{ij}, \forall i \in S, j \in OUT_i \tag{4}$$

Table II: Terminology for MIP Formulations

Variable	Description
S (Given)	Set of all menu nodes
S_E(Given)	Set of endpoint menu nodes, leaf nodes of a tree which are black-filled
S_{OPT}(Given)	Set of Optimizer menu items (triangles in Figures 3,4)
S_0(Given)	Sink node, the root level menu item. (Node with ID=0 in Figures 3,4))
S_{COMB}(Given)	Set of Combiner menu items, the ones shown with romb in Figures 3,4)
C_i(Given)	Integer that shows click count for i^{th} node.
IN_i (Given)	Set of candidate outgoing nodes from i^{th} node.
OUT_i (Given)	Set of candidate incoming nodes to i^{th} node
L_{IN}(Given)	Constant that shows the maximum allowed number of incoming nodes to a particular node. i.
L_{OUT} (Given)	Constant that shows the maximum allowed number of outgoing nodes from a particular node. i.
f_{ij}(Variable)	Amount of flow (click in this case) between menu nodes i and j
a_{ij}(Variable)	Boolean variable which denotes if there exist a flow between the nodes i and j
d_{ij}(Given)	Integer value denoting the weights of arcs. In general, all weights are equal to one, but the ones incoming to S_{OPT} are equal to 0
M(Given)	Large constant with respect to given parameters in problem.

$$\sum_{j \in OUT_i} a_{ij} \begin{cases} \leq L_{OUT}, & \forall i \in S_{OPT} \cup S_{COMB} \\ = L_{OUT}, & \text{otherwise} \end{cases} \tag{5}$$

$$d_{ij} = \begin{cases} 0, & \text{if } j \in S_{OPT} \\ 1, & \text{otherwise} \end{cases} \forall i \in S, \forall j \in OUT_i \tag{6}$$

$$\sum_{j \in IN_i} a_{ji} \begin{cases} \leq 1, & \forall i \in S_{OPT} \\ \leq L_{IN}, & \text{otherwise} \end{cases} \tag{7}$$

$$\sum_{j \in OUT_j} a_{ij} \leq \sum_{IN_j} a_{ji}, \forall i \in S - S_E \tag{8}$$

$$\text{Minimize} : \sum_{j \in OUT_i} d_{ij} f_{ij} \forall i \in S \tag{9}$$

Equation 1 shows that a_{ij} is a boolean variable for all possible arcs defined in the system. Constraint 2 represents the non-negativity of flows in our model. Equation 3 provides the flow balance in the system. There can be 3 separate cases. If the node is a sink node, that is, the one at the top level, then all coming flow is equal to the sum of all flows generated in leaf nodes, since flows are made from clicks to leaf menu nodes. If the node is a leaf node, then the difference between outgoing and incoming nodes will be the click count on that node, as no flow is coming to the leaf node by definition. The third case is shown for the leaf nodes, that are neither sink node nor leaf nodes. In equation 4, we are converting flows to binary variables to be used in later equations. Equation 5 shows the constraint related with outer arcs from a particular node. That is, there must be exactly L_{OUT} number of arcs going out of a particular node, if the menu node is neither S_{OPT}, nor S_{COMB}. This means that, a particular node in the menu must be connected to the menu and it cannot be lost even it has no clicks. We chose $L_{OUT} = 1$, in our case, meaning, only one arc can go out of a particular menu node. But in the case of S_{OPT} and S_{COMB} nodes, they are free to be connected to the tree or not. We are not forcing them to do so. Equation 6 does the same for constraining the number of incoming arcs to a particular node. This can be thought as the maximum number of menu items under a particular menu item. We constrained S_{OPT}'s incoming nodes to maximum 1 node, as we are allocating S_{OPT} nodes itself and taking more than one incoming node would not optimize the system efficiently. Equation 6 shows the constant d_{ij}, which represents the weights of arcs. We say that every arc in a tree has an equal weight of 1, except the ones incoming to S_{OPT} menu nodes. Since we want to encourage other nodes to connect to these nodes, their weights are 0. Equation 8 shows that the incoming nodes in our model must be greater or equal than the outgoing nodes. When we consider this as a menu tree, this is the case in all models. Equation 9 shows our objective function. This is simply the minimization of flows multiplied by the arc weights. This can also be considered as the cost of a flow.

V. Experiments

We obtained initial results for only one ATM out of 2000 machines and we divided users into 3 profiles using unsupervised clustering. We are working on to get the results for all ATMs and more user profiles. Our original data size is $4 * 10^7$ ATM logs. For now, we obtained results for approximately $2 * 10^6$ of them. We are working on to get whole solution for the final paper.

We used MapReduce framework [13] to mine and process the data and CPLEX [14] to find the optimum solution using MIP [15]. Overall structure of the system design is as follows:

1) First we mine large amount of data using MapReduce and cluster the users.
2) Second, for each cluster we get the click counts of the leaf menu items using MapReduce.
3) Third, we find the optimum menu structure for each user group using the method explained in Section IV.

As the first and second steps are not the focus of this paper, we try to concentrate on the third step. However, in first step we choose k, the number of clusters to be 3 and separated users into 3 categories according to their actions in ATM, which are:

1) C1, Users that mostly took cash from ATM and look to their balance.
2) C2, Users that mostly took cash from ATM and almost never look to their balance.

3) C3, Users that do advanced operations in ATM such as money transfer, credit card operations, automatic bill payment and etc.

Our menu structure that we wanted to optimize consists of 35 menu items. The time taken to optimize this count of nodes is low (*i.e.*, it was less than a second). However we are working on heuristic methods to optimize huge number of menu structures. We tested our results with two methods. In first method (FM), we mined all logs of ATM derived a conclusion about overall click count with new menu structure. In second method (SM), we chose random person from the customers of the bank, and asked to use our app developed in iOS. SM is subset of FM because FM includes all user transactions in the past 18 month for a particular bank.

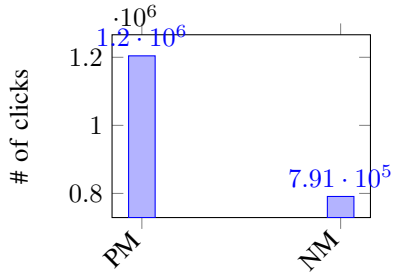

Figure 6: Overall number of clicks with new model (NM) and previous model (PM).

Figure 6 shows overall count of clicks with new (NM) and previous (PM) models. There is approximately 30% click reduction in new model.

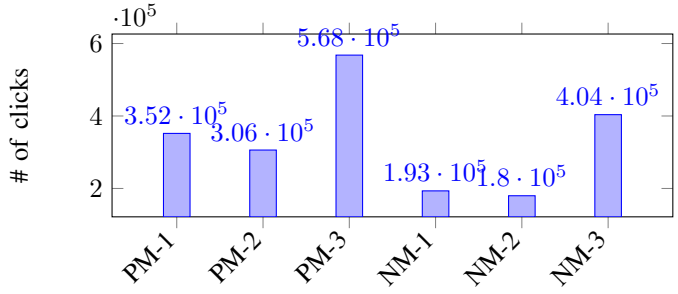

Figure 7: Profile based comparison of number of clicks.

Figure 7 shows the user profile click gains. Here NM-k shows the k^{th} profile users with new model and the PM-k shows the same with previous model. Because there was no user profiling in previous model, we did the same procedure (clustering of users) to the previous one to compare the results. The results show that, there is a significant reduction of clicks in C1 and C2. This is because of the fact that, C1 and C2 clusters are more specific profiles. C3 cluster users do all kind of operations and mostly advanced ones.

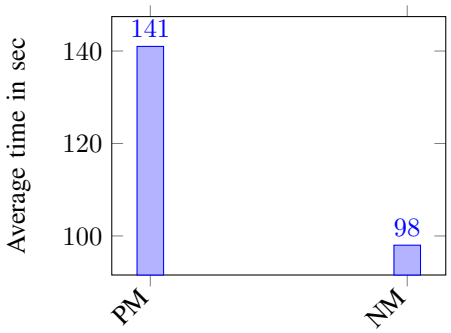

Figure 8: Average session time (in sec) on ATM with new model(NM) and previous model (PM).

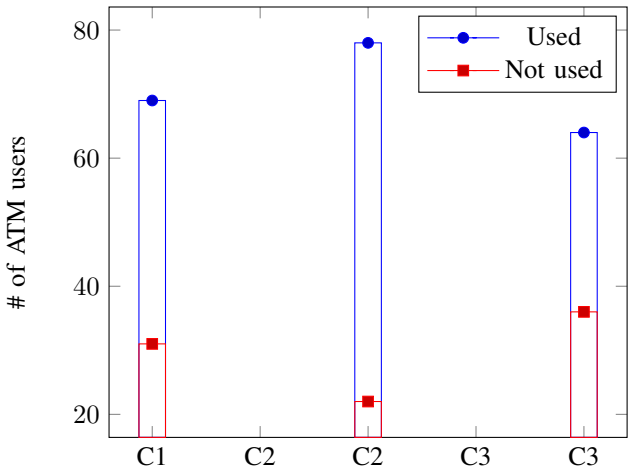

Figure 9: Percentage of users used our model's optimization menu items (S_{OPT}).

Figure 8 shows the average session time with previous and new models which showcases the gains brought by the proposed MIP model. As the users are more efficiently clustered, the reduction in completion time will higher, which is among our ongoing research efforts. Figure 9 shows the percentage of users that used our new optimizer menu items which was created by our model. As can be seen C1 and C2 users tent to use more, because they are more specific profiles.

As a second method, (SM), we used a simulator and chose 100 real customers. The logs that we are using, are hashed so that the user ID and specifications cannot be obtained. Because of this fact, while detecting a user of that bank, we cannot know in what profile he/she was while clustering. That is, we cannot obtain a model trained for that particular user's profile, because we do not know his/her ID. Therefore, we prepared a model without profiling users and programmed it in iPAD. We asked the bank's users to use it as ATM . We do not know a particular user's previous session time and click count because users' IDs are hashed. So, the only thing that we measured was, to

complete a task preferred by user in iPAD ATM simulator and look whether he/she is using new menu items created by optimization model. The results were logical as in Figure 10. Because we could not do profiling the optimal menu for all users is not efficient as the one done with user profiles. However, 46% of users accomplished their tasks with using new models optimized menu items. Nearly 30% used these keys mistakenly. This is because of fact that we cannot make unique optimal menu for all users.

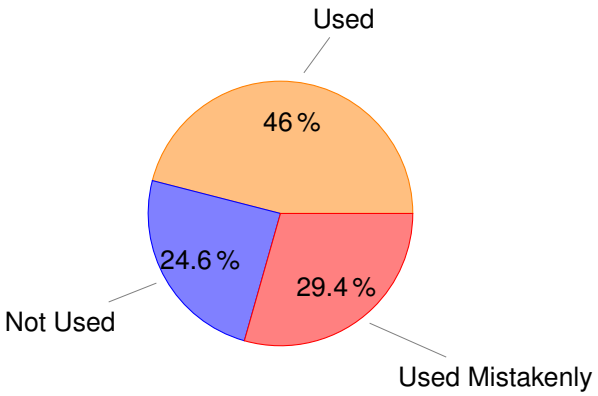

Figure 10: Percentage of users with ATM simulator used our model's optimization menu items (S_{OPT}).

VI. Conclusion

Optimization of ATM menu structures is an important problem to reduce the delay experienced by the users and for the efficiency of utilizing ATMs. It is also imperative to formulate the optimization problem in a generic fashion so that many ATM menu types can be adressed within a single optimization framework. In this study, we built a novel MIP framework to address this open question, which has never been systematically investigated in the literature before. We present the results of performance evaluations of our optimization model by using real data collected from a large transaction database. Our results reveal that significant reduction in click count, thereby, in ATM transaction completion time is possible when the menu structure is optimized by the novel MIP framework presented in this study.

Acknowledgement

This study is supported by Turkish Ministry of Science, Industry, and Tech. under grant SANTEZ-0367.2013-2.

References

[1] H. Kobayashi, "Automatic teller machine," May 6 1986, uS Patent D283,746.

[2] K. Taohai, S. Phimoltares, and N. Cooharojananone, "Usability comparisons of seven main functions for automated teller machine (atm) banking service of five banks in thailand," in *Computational Science and Its Applications (ICCSA), 2010 International Conference on*. IEEE, 2010, pp. 176–182.

[3] N. Cooharojananone, K. Taohai, and S. Phimoltares, "A new design of atm interface for banking services in thailand," in *Applications and the Internet (SAINT), 2010 10th IEEE/IPSJ International Symposium on*. IEEE, 2010, pp. 312–315.

[4] G. Mujtaba and T. Mahmood, "Adaptive automated teller machines—part i," in *Information and Communication Technologies (ICICT), 2011 International Conference on*. IEEE, 2011, pp. 1–6.

[5] G. Krishnan, S. Kumar, C. Jithin, V. V. Panicker, and R. Sridharan, "Service innovation for the user interface of an atm catering to the needs of the student community," in *Industrial Engineering and Engineering Management (IEEM), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1180–1184.

[6] R. M. Kumar, G. Varaprasad, and R. Sridharan, "An innovative proposal to increase the efficacy of the automated teller machine using mobile banking," in *Intelligent Human Computer Interaction (IHCI), 2012 4th International Conference on*. IEEE, 2012, pp. 1–6.

[7] T. G. Apari, F. Molu, N. Findik, and M. Dalci, "User experience approach in financial services," in *Technological Advances in Electrical, Electronics and Computer Engineering (TAEECE), 2013 International Conference on*. IEEE, 2013, pp. 400–403.

[8] A. H. Cremers, J. G. de Jong, and J. S. van Balken, *User-centered design with illiterate persons: the case of the ATM user interface*. Springer, 2008.

[9] A. De Luca, M. Langheinrich, and H. Hussmann, "Towards understanding atm security: a field study of real world atm use," in *Proceedings of the Sixth Symposium on Usable Privacy and Security*. ACM, 2010, p. 16.

[10] A. De Angeli, A. Sutcliffe, and J. Hartmann, "Interaction, usability and aesthetics: what influences users' preferences?" in *Proceedings of the 6th conference on Designing Interactive systems*. ACM, 2006, pp. 271–280.

[11] I. Medhi, S. Patnaik, E. Brunskill, S. Gautama, W. Thies, and K. Toyama, "Designing mobile interfaces for novice and low-literacy users," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 18, no. 1, p. 2, 2011.

[12] A. Thatcher, F. Shaik, and C. Zimmerman, "Attitudes of semi-literate and literate bank account holders to the use of automatic teller machines (atms)," *International Journal of Industrial Ergonomics*, vol. 35, no. 2, pp. 115–130, 2005.

[13] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[14] I. I. CPLEX, "V12. 1: User's manual for cplex," *International Business Machines Corporation*, vol. 46, no. 53, p. 157, 2009.

[15] L. A. Wolsey, "Mixed integer programming," *Wiley Encyclopedia of Computer Science and Engineering*, 2008.