OPaPi: Optimized Parts Pick-up routing for efficient manufacturing

Chidansh Bhatt
FXPAL
Palo Alto, CA
bhatt@fxpal.com

Jian Zhao
FXPAL
Palo Alto, CA
zhao@fxpal.com

Hideto Oda
Fuji Xerox Co., Ltd.
Yokohama, Japan
hideto.oda@fujixerox.co.jp

Francine Chen
FXPAL
Palo Alto, CA
chen@fxpal.com

Matthew Lee
FXPAL
Palo Alto, CA
mattlee@fxpal.com

ABSTRACT

Manufacturing environments require changes in work procedures and settings based on changes in product demand affecting the types of products for production. Resource re-organization and time needed for worker adaptation to such frequent changes can be expensive. For example, for each change, managers in a factory may be required to manually create a list of inventory items to be picked up by workers. Uncertainty in predicting the appropriate pick-up time due to differences in worker-determined routes may make it difficult for managers to generate a fixed schedule for delivery to the assembly line. To address these problems, we propose *OPaPi*, a human-centric system that improves the efficiency of manufacturing by optimizing parts pick-up routes and scheduling. *OPaPi* leverages frequent pattern mining and the traveling salesman problem solver to suggest rack placement for more efficient routes. The system further employs interactive visualization to incorporate an expert's domain knowledge and different manufacturing constraints for real-time adaptive decision making.

CCS CONCEPTS

• **Information systems → Data analytics**; • **Human-centered computing → Visual analytics**;

HILDA'19, July 5, 2019, Amsterdam, Netherlands
© 2019 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-6791-2/19/07…$15.00
https://doi.org/10.1145/3328519.3329135

KEYWORDS

Frequent pattern mining, Traveling Salesman Problem (TSP), interactive visualization, dynamic scheduling, manufacturing, inventory management, re-routing, itemset mining

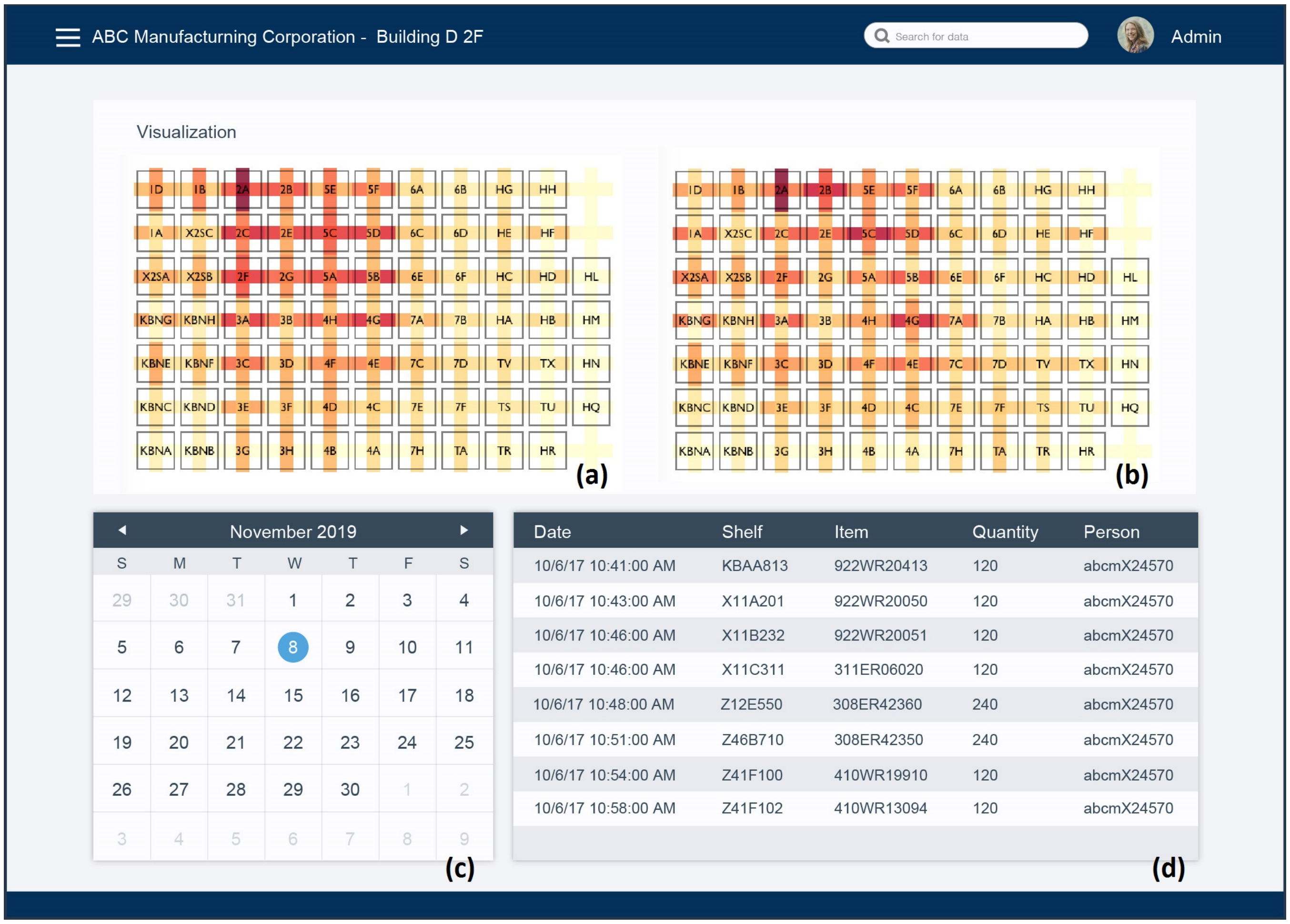
1 INTRODUCTION

Production scheduling in a factory generally assumes that all situations are known in advance and can be resolved without human supervision, leading to automatic scheduling without human input. Such automatically-generated schedules are useful for planning the production activities. Unfortunately, manufacturing systems operate in dynamic environments subject to various real-time events. An optimal automatic schedule may become either infeasible or non-optimal due to unforeseen events, such as accidents. While such systems enable high-level scheduling (e.g., warehouse spaces can be prepared as early as possible for early orders [2]) they are unable to provide granular level decision making.

For example, in the event of an accident on the inventory floor between pick-up carts, the specific route may become temporarily inaccessible to workers. Or the managers in a factory may be required to manually create a new list of inventory items to be picked up to meet the production deadline. Inventory planning and scheduling can drastically impact the complete supply chain. Delays at the beginning of the manufacturing supply chain can propagate to the end of the chain, leading to delays that can incur major financial (e.g., brand name) loss.

Robustness and transparency are key factors to preserving the stability of manufacturing systems in the presence of uncertainties. An important problem in the manufacturing environment is to decide where to keep parts on shelves to minimize the pick-up time from the parts inventory area and maximize throughput to assembling area. Another consideration is to minimize the labor cost as well as extra costs incurred due to delayed production for manufacturing. Managers need to be enabled to make informed decisions such as efficiently allocating pick-up tasks to the workers without disrupting the production.

We use the term *lot* to describe the list of the items required for a specific product. For example, a *lot* for producing one quantity of


ABC Manufacturing Corporation - Building D 2F
Search for data
Admin
Visualization
ID IB 2A 2B 5E 5F 6A 6B HG HH
IA X2SC 2C 2E 5C 5D 6C 6D HE HF
X2SA X2SB 2F 2G 5A 5B 6E 6F HC HD HL
KBNG KBNH 3A 3B 4H 4G 7A 7B HA HB HM
KBNE KBNF 3C 3D 4F 4E 7C 7D TV TX HN
KBNC KBND 3E 3F 4D 4C 7E 7F TS TU HQ
KBNA KBNB 3G 3H 4B 4A 7H TA TR HR
(a)
ID IB 2A 2B 5E 5F 6A 6B HG HH
IA X2SC 2C 2E 5C 5D 6C 6D HE HF
X2SA X2SB 2F 2G 5A 5B 6E 6F HC HD HL
KBNG KBNH 3A 3B 4H 4G 7A 7B HA HB HM
KBNE KBNF 3C 3D 4F 4E 7C 7D TV TX HN
KBNC KBND 3E 3F 4D 4C 7E 7F TS TU HQ
KBNA KBNB 3G 3H 4B 4A 7H TA TR HR
(b)
November 2019
S M T W T F S
29 30 31 1 2 3 4
5 6 7 8 9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30 1 2
3 4 5 6 7 8 9
(c)
(d)

Date	Shelf	Item	Quantity	Person
10/6/17 10:41:00 AM	KBAA813	922WR20413	120	abcmX24570
10/6/17 10:43:00 AM	X11A201	922WR20050	120	abcmX24570
10/6/17 10:46:00 AM	X11B232	922WR20051	120	abcmX24570
10/6/17 10:46:00 AM	X11C311	311ER06020	120	abcmX24570
10/6/17 10:48:00 AM	Z12E550	308ER42360	240	abcmX24570
10/6/17 10:51:00 AM	Z46B710	308ER42350	240	abcmX24570
10/6/17 10:54:00 AM	Z41F100	410WR19910	120	abcmX24570
10/6/17 10:58:00 AM	Z41F102	410WR13094	120	abcmX24570

Figure 1: An example screen-shot of the *OPaPi* system interface: (a) a real-time view of the aggregated trajectories of parts picking by workers, (b) a view of the aggregated trajectories with an alternative scheduling for comparison, (c) a calendar view for querying data within certain time ranges, and (d) a detailed view of schedules.

product X require $item1, item2, \ldots, itemZ$. Currently, item pick-up lists for a *lot* are generated without optimally incorporating the existing inventory layout context. Thus, workers end up picking items with a non-fixed route, leading to going back and forth several times to the same aisle or racks or sections for picking the items. This uncertainty in routes leads to a non-fixed schedule that is difficult to predict the exact pick-up times as different employees will pick the items differently without a fixed route. Uncertainty in predicting the appropriate pick-up time and workers behaviour may make it difficult for managers to generate a fixed schedule for delivery to the assembly line. It may cause a complete halt of production on the assembly line in case the items are not delivered on time.

Recently, human-centric approaches renewed its importance in smart manufacturing especially for assembly applications where human dexterity and informal knowledge are dispensable at present and in the near future [13]. Furthermore, introducing novel technologies makes manufacturing system much more complicated. Therefore, highly skilled, well-trained people like factory managers are very important to make systems sustainable and resilient [9]. We worked with factory managers and found out that along with an automatic system, they wanted the capability to explore and investigate the results by themselves for making data-driven decisions.

To this end, in this paper, we propose *OPaPi* (Figure 1), a human-centric scheduling system that incorporates the real-time context about the items, layout, workers and other environmental constraints to help a factory manager generate real-time optimal routes and robust schedules for each worker to complete delivery tasks on-time. Further, *OPaPi* allows for interactive human intervention of route planning and schedule generation through visualization, facilitating a factory manager with making informed decisions. To develop the *OPaPi* system, we worked closely with factory managers to identify their needs and design effective solutions. In the end, the *OPaPi* system produces optimal routes and schedules dynamically based on the current inventory using frequent itemset mining and the traveling salesman problem (TSP) solver, and provides an interactive visualization for factory managers to compare the impact of applying different routes on the total distance travelled by workers to pick up items when recovering from unexpected delays.

2 RELATED WORK

Manufacturing environments are dynamic in nature and are subject to various disruptions, referred to as real-time events, which can change system status and affect its performance. Literature on dynamic scheduling has considered a significant number of real-time events as (a) resource related: machine breakdowns, defective parts or (b) job related: change in job priority or processing time. These real-time events are considered in the context of categorization of manufacturing systems as single machine systems, parallel machine systems, flow shops, job shops, and flexible manufacturing systems (FMS) [7]. Dynamic approaches are categorized as completely reactive, predictive-reactive, robust predictive-reactive and pro-active scheduling. *OPaPi* incorporates environment-related events e.g., changes in inventory layouts due to accidental events, along with resource-related and job-related real-time events for flexible manufacturing systems (FMS). *OPaPi* combines frequent pattern mining and the TSP solver with an interactive user inputs to provide a novel solution that is explainable with statistical data mining, optimization algorithm and an intuitive visualization.

Priore et al. provided a detailed review on machine learning based dynamic scheduling of manufacturing systems [10]. They discussed pros and cons of inductive learning, Neural Networks (NN), case-based reasoning, support vector machines, reinforcement learning and other mixed approaches. Tasks of such machine learning-based dynamic scheduling algorithms were examined in terms of determining the optimal number of training examples, selection of the monitoring period, selection of control attributes, refinement of the knowledge base etc. Qu et al. proposed a distributed reinforcement learning method to overcome parameter estimation of traditional queuing models and hyper-parameter tuning of multi-agent-based reinforcement learning [11]. Overall, such systems lack the intuitive human interaction and easy-to-understand statistical analysis that is an essential part of dynamic scheduling. We overcome the complexity associated with some of the Neural Network (NN) based methods by proposing an explainable statistical machine learning-based method that adapts to the human expert in the loop for decision making.

For example, Hirayama et al.'s study proposed a system using daily work performance data to issue appropriate work instruction by understanding work-site improvements and environment changes in the warehouse to achieve an 8% reduction in work time [5]. There are no specific machine learning or visualization techniques mentioned in their work. The system is described as *H* system that calculates the relation between Key Performance Indicators (KPIs) and explanatory variables related to KPIs. Another hybrid technique used search and optimization algorithm e.g., simulated annealing and Dijikstra's algorithm to solve the routing problem for automated guided vehicle in manufacturing [12]. In contrast to such methods, we are using a statistical machine learning technique and optimization algorithm integrated tightly with human interaction for tackling the real-time event-based changes that needs a human's domain expertise.

Frequent itemset mining is traditionally applied to market basket analysis problems. It needs to determine which products should be placed next to each other. It is a similar problem to the proposed manufacturing problem in some aspects. Market basket analysis applications in [1, 6] are focused on maximizing a supermarket's profit by placing items from different categories together, increasing the travel distance of a customer and their unplanned impulsive purchases. Also, frequent itemset mining was not combined with TSP solvers in the past. On the other hand, *OpaPi* is focused on minimizing the travel distance as well as avoiding congestion / collision among workers using a combination of frequent itemset mining and TSP solver.

Most of the previous works focused on the design of smart manufacturing systems favoring a technocentric approach which gave priority to the definition and allocation of tasks with automated system and computing resources, only taking human operators into consideration at the end of the design process [8]. This kind of design approach assumes human as "magic" who must behave perfectly when unexpected situations happened [14]. However, as the technological accidents are caused primarily by human errors (63%) [8], putting a human in the control loop of smart manufacturing system requires proper human-centered design (HCD) to cope with more and more complex problems. Inspired by Industry 4.0 design [3], we placed a human in the center of proposed *OpaPi* system.

Overall, the novelty of *OpaPi* lies in the incorporation of a domain expert's knowledge to address new constraints on the fly using interactive visualization; this is tightly coupled with data mining algorithms and optimization techniques to provide real-time, informed decision-making capabilities.

3 PROPOSED OPaPi SYSTEM

We worked with factory managers during the development process, conducting a series of interviews with them and designing the *OPaPi* system iteratively. One of their main goals is to optimize the routes for parts picking to reduce labor cost. They demanded an automatic system to provide the optimization, but at the same time wanted the capabilities to explore the results by themselves for making data-driven decisions. Further, they required real-time investigation of the parts-picking routes and schedules in order to fulfill special constraints of the product line and conduct on-demand planning. Based on their needs, as shown in Figure 2, we propose the *OPaPi* system that consists of data mining, optimization and interactive visualization to help a manager make informed decisions to re-organize the inventory layout, select optimal routes for parts pick-up workers according to real-time constraints for the production.

3.1 Routes/schedule optimization

OPaPi's analysis consists of three-stages. In the first stage, we mine frequent maximal itemsets by applying the FPMax Algorithm [4] on a parts delivery database of manufacturing to find: (a) frequent closed itemset: *a frequent itemset that is not included in a proper*

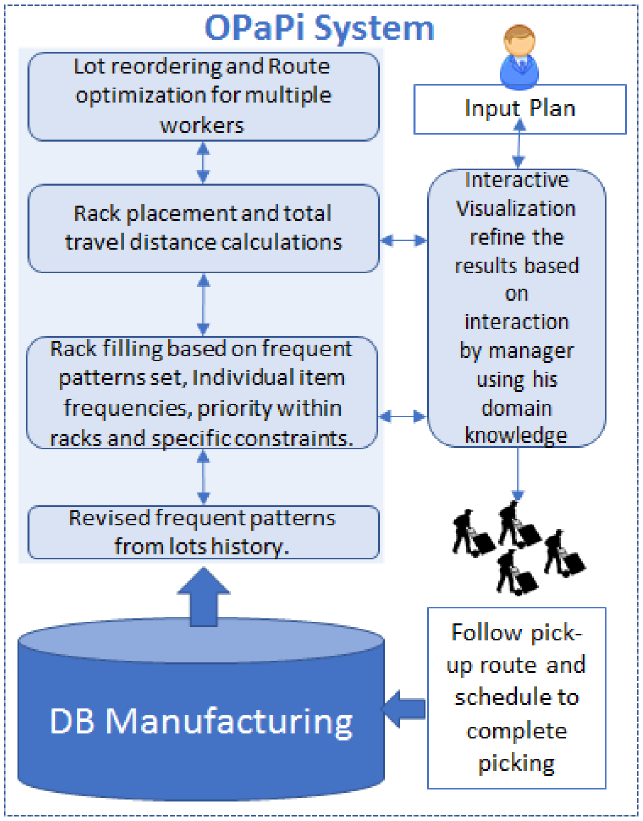

OPaPi System
Lot reordering and Route optimization for multiple workers
Input Plan
Rack placement and total travel distance calculations
Interactive Visualization refine the results based on interaction by manager using his domain knowledge
Rack filling based on frequent patterns set, Individual item frequencies, priority within racks and specific constraints.
Revised frequent patterns from lots history.
DB Manufacturing
Follow pick-up route and schedule to complete picking

Figure 2: The architecture overview of *OPaPi*.

superset having the same support and (b) frequent maximal itemset: *a frequent itemset that is not included in a proper superset.* In other words, we try to minimize the large number of redundant frequent patterns and determine the greatest number of items that are frequently used together across diverse production scenarios that can be placed together.

Once we discover such frequent maximal itemsets, we pick the itemset with the highest length. All the items in the picked itemset are placed in the rack. Then, we revise remaining itemsets by removing already placed items in the rack. Iteration will continue until we placed all the items in the racks. Our proposed method incorporates item-level constraints while placing them in the rack along with their strongly associated items based on the individual items frequency. For example, one of the item-level constraints is that the most frequent item within the itemset is utilized much often than other items in the itemset. The location of such high priority items within the racks can be selected based on an environmental constraint. For example, one of the environmental constraints is set to pick up high priority items with ease from the middle section of the rack compared to the lower or upper sections of the rack.

In the second stage, the system identifies rack locations to minimize the overall distance travelled by the workers to pick-up the parts irrespective of the different productions. Additionally, we incorporate the constraint that multiple workers picking-up the inventory items at the same time are not congesting the routes. Such optimization ensure that the most frequent item racks are not all placed together. Also, the racks are distributed such that it allows multiple workers to work efficiently without congesting the aisles, racks or sections for picking-up the items.

The system leverages the input floor plan to generate a hybrid distance. To compute the hybrid distance first we compute the Manhattan distance matrix for all the $N \times N$ racks. In case source to destination racks are in a non-adjacent aisle we incorporate an extra distance value (provided by factory manager) to include in the hybrid distance computation. For computing the total distance travelled by all the workers, hybrid distances are converted to real distances from the actual floor plan of the manufacturing facility. The manager will input the known sizes of racks and aisles from the actual inventory floor map to the system for such distance calculations. Thus, the proposed system can incorporate different inventory layouts to generate optimized resource placements.

For the third-stage, the system leverages the computed distances between the racks to generate optimal routes with an ordered list of pick-up items for each *lot*. The travelling Salesman Problem (TSP) solution is utilized for finding the optimized routes.

The system leverages a python implementation of a TSP solver [1] to incorporate constraints like having a unique pick-up list for each worker to avoid collisions or congestion while picking up similar items. In turn, we produced fixed item pick-up routes and approximately fixed schedules for item pick-ups at a granular level of one rack to the next rack. Now, this schedule can be dynamically changed by the manager using the interactive visualization. Mostly in the scenarios where the manager observes any real-time event that will required change in current schedule, he will utilize the interactive analytics and visualization to decide on a feasible solution for the overall operation for collecting the items from inventory and sending it to the assembly line. Also, *OPaPi* is flexible in terms of enabling managers to skip the first two stages for optimal resource placement and directly execute the third stage for generating the optimal route and schedules for workers.

3.2 Interactive visualization

Based on the optimized inventory routes and schedule, *OPaPi* enables a manager to incorporate their domain expertise as well as real-time constraints through interactive visualization. It provides updated analytical results as managers interact with the visual system.

For example, Figure 1 is a screenshot of the *OPaPi* system. Based on the input from the manager, the detailed schedules, as shown in Figure 1(d), have been generated for the selected day, as shown in Figure 1(c). Figure 1(a) shows a real-time visualization of the aggregated trajectories of all the workers who pick up the parts, where red regions indicate more congestion of the trajectories on the floor plan. By interacting with the system, a manager can quickly browse through other potential schedules and select a new schedule to compare. Figure 1(b) shows a less congested aggregated set of trajectories with the new schedule. This interactive feature of *OPaPi* can be used to avoid heavy congestion that could cause

[1]https://github.com/dmishin/tsp-solver

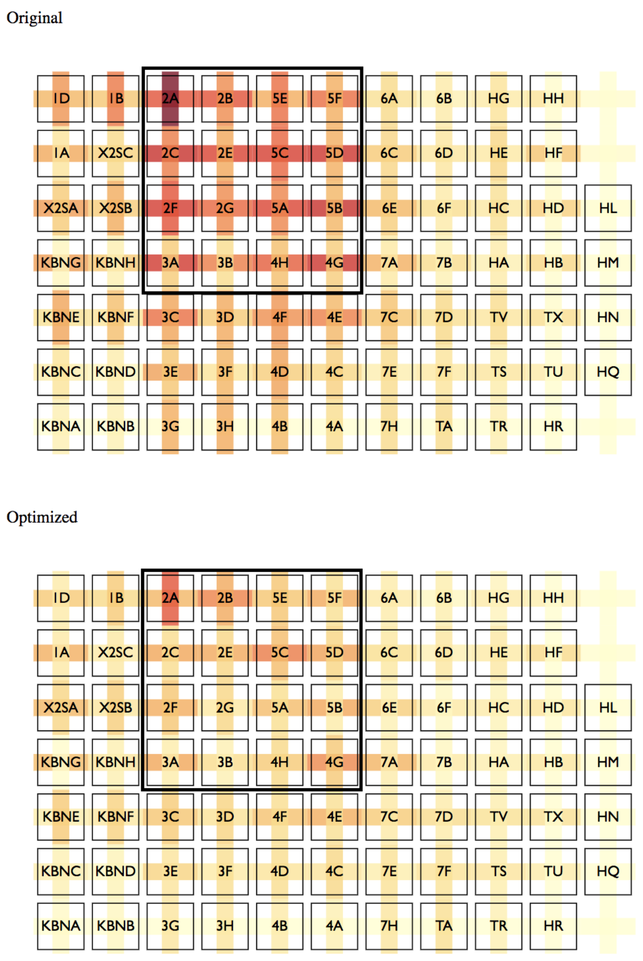


Figure 3: Aggregated view of trajectories (top) without optimized routes and (bottom) with optimized routes without changing inventory items and racks layout.

an accident and the complete halt of manufacturing, supporting a dynamic decision-making process.

In addition, a manager can obtain an aggregated view of all the optimized and non-optimized trajectories within a selected time range.

For example, from the top of Figure 3, a manager can investigate which parts of the factory are frequently navigated by workers before applying the *OPaPi* based optimized routes, thus seeing a bigger picture about the behaviors and productivity of the workers. As shown in bottom of Figure 3, a manager can visualize whether aggregate trajectories using optimized routes will improve the behavior and/or productivity of the workers. For example, the boundary box covering (2A-5F-3A-4G) rack area is very dense (top), indicating high-risk congestion or collision areas that could impact worker productivity. After applying *OPaPi* based optimized routes (bottom) there are no high-risk congestion or collision-prone areas.

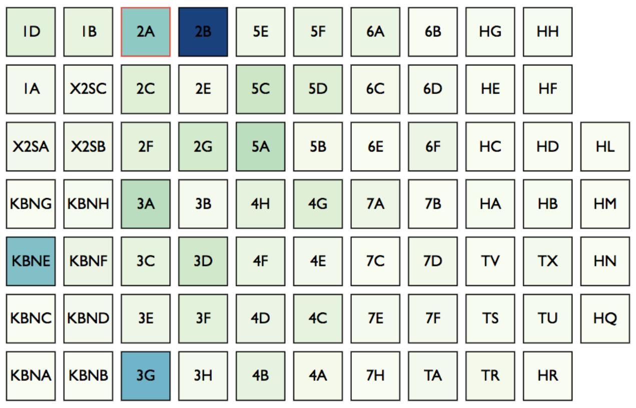


Figure 4: Rack association map of the selected rack 2A.

On top of the aggregated visualization of the trajectories, another type of information that can help a manager assess different rack placements is the association of the racks based on the parts picked-up by the workers. As shown in Figure 4, when a manager selects rack 2A (red boundary), all the associated lots are color-coded in a dark to light blue color-scale, where dark blue means more items are picked up together with those in the selected rack. Racks that are not strongly associated with the selected one are colored in a light green color. The association maps can be viewed and compared in the main view of the *OPaPi* interface, similar to the trajectory views in Figure 1.

Based on this association map, a manager can obtain a more in-depth view of the placement of the lots, thus helping him further optimize the solution based on the algorithm outputs. For example, he may need to move rack 2A due to, for example, it being in a potentially high congestion point, having caused an accident or having some hazardous items that needs to be moved. Using the rack association map in Figure 4, a manager can infer that rack 2B is strongly associated with rack 2A. Moving racks 2A and 2B together towards racks KBNE and 3G can be a better choice in terms of reducing total travel distance for workers (in turn saving time and money). Based on a manager's domain knowledge, he can interactively rearrange the racks using drag-and drop. *OPaPi* will reflect the total travel distance with new changes compared to the previous arrangements. Different placements can be saved and compared using the system, allowing for interactive trials of different plans by assessing their advantages and disadvantages. Managers can make informed decision based on the situation and plan better for smooth production operations.

4 EVALUATION

We received the inventory parts pick-up data from the manufacturing facility[2] for their operation over 10 months. The database is composed of item id, quantity, rack layout, pickup time, item location and *lot* number which indicates an item group delivered

[2]*Name of the manufacturing factory is not disclose to preserve their confidentiality.

together. Each *lot* needs to be delivered to the assembly line. A manufacturing database with 3,124 *lots* is utilized for the experiment and evaluation.

In the first-stage of *OPaPi* using FPmax pattern mining, we discovered a total of 105 patterns with 351 distinct items. The average number of items per pattern is 6.5 ($std = 6.6, var = 44$). The results indicate that the frequent closed maximal itemsets have the potential to discover the top 18% of frequently used items. Organizing only these 18% items appropriately can make a significant difference in overall manufacturing efficiency.

OPaPi utilized the existing placement of racks and items at the manufacturing facility to identify the baseline improvement even without re-organizing inventory. We leveraged the existing layout context alone with the travelling salesman problem (TSP) solution for finding the efficient routes without utilizing the benefits from re-organized inventory. The maximum number of rack locations are 20 for the given dataset. We applied the brute force TSP algorithm to obtain an optimized path when the number of locations is less than 7. In other cases, we applied the greedy TSP algorithm to obtain a quasi-optimized path. Using this baseline approach *OPaPi* achieved a 12% reduction in total travelling distance. For the manufacturing dataset consisting of 10 months of operations, the original total travelling distance for workers is 202,352 meters and after applying the baseline method, and the total travelling distance was reduced to 177,610 meters. *OPaPi* can provide more efficient solutions for manufacturing in terms of reduction in total travelling distance and in turn reduce the labor cost with more accurate routes and scheduling time for item pick-up. These results indicate that *OPaPi's* automatic scheduling can be useful for enhancing the efficiency of the factory with their existing racks and item set-up. With the visualizations in Figure 3 and Figure 4, a manager can incorporate their domain expertise to further optimize the schedules.

5 SCOPE AND LIMITATION

OPaPi was designed and implemented based on requirements which emerged out of the interviews with manufacturing managers at the factory. *OPaPi* derived the optimized item placement and generated optimized schedule for parts pick-up based on ten months of past data from the factory. The scope of *OPaPi* is to leverage both item placement and optimized pick-up parts with human-centric interactive visualization. An existing limitation is in terms of actually deploying *OPaPi* in a manufacturing facility to understand it true scope and limitations. In case, the factory managers decide to re-organized their inventory based on the optimized item placement and start using the interactive visualization to deploy their knowledge in real-time situations to leverage the optimal paths for smooth manufacturing. It is possible that we can learn more about the actual pros and cons of the system in future. At present our assumption about the use of visualization as well as the combined gain using item placement and part-pickup optimization is limited. We suspect that once managers start using *OPaPi*, we may find more utility and feedback to refine the system. For example, heat maps showing aggregate trajectories are depicting the path utilization that may sometimes be different depending on whether the actual workers are on time or running late or faster than their schedule. Also, the time-window for visualizing such aggregate trajectories could provide different insights to actual events. In case managers decide to re-organize inventory based on the proposed item-placement, then the aggregate trajectory over all the past (larger time window selected as start and end date from the calendar) data with new item-placement information might provide better insights to managers for organizing the rack layout appropriately to avoid the congestion. On the other hand, when a manager is choosing a smaller time window for visualizing the aggregate trajectories, e.g., for the day, the present batch or even every 5 minutes it will provide more granular insight to real-time incidents scenarios for incorporating dynamic decision by manager in to reschedule the parts pick-up in real-time for smooth manufacturing.

6 CONCLUSION

OPaPi presents a novel way to incorporate the domain expert's knowledge and domain constraints on the fly into data mining algorithms to provide real-time informed decision-making using interactive visualization. It generates optimal parts pick-up routes with granular-level dynamic scheduling using real-time analytics that can help the manager to re-organize inventory items and racks and also to allocate a sufficient workforce to meet the dynamic production requirements in places such as a factory, warehouse, or supermarket. An immediate future task for *OPaPi* will be to further enhance the efficiency of manufacturing by leveraging results from optimal item placement and rack arrangements. In the future, we plan to extend the *OPaPi* system with consideration of physiological, individualized human models and profiles to better adjust to different manufacturing applications.

ACKNOWLEDGMENTS

The authors would like to thank Mr. Takahiro Okayasu, Intellectual Property Manager at FXPAL, for providing valuable support and legal advice during the collaboration with the manufacturing facility in Japan. We also like to thank Mr. Takashi Watanabe, Team Manager of Product Logistics and Inventory, for providing the requirements during the interviews. We like to thank Xiaojing Zhang, Designer at Fuji Xerox, for providing feedback to the visual design.

REFERENCES

[1] G. Aloysius and D. Binu. 2012. An approach to products placement in supermarkets using PrefixSpan algorithm. *Journal of King Saud University Computer and Information Sciences*.

[2] W. Du, W. Zhong, Y. Tang, W. Du, and Y. Jin. 2018. High-Dimensional Robust Multi-Objective Optimization for Order Scheduling: A Decision Variable Classification Approach. *IEEE Transactions on Industrial Informatics*.

[3] Paola Fantini, Marta Pinzone, and Marco Taisch. 2018. Placing the operator at the centre of Industry 4.0 design: Modelling and assessing human activities within cyber-physical systemss. *Comput. Ind. Eng.*

[4] GÃűsta Grahne and Jianfei Zhu. 2003. High performance mining of maximal frequent itemsets. *International Workshop on High Performance Data Mining*.

[5] J. Hirayama, T. Akitomi, F. Kudo, A. Miyamoto, and R. Mine. 2016. Use of AI in the Logistic Sector: Case study of Improving Productivity in Warehouse Work. In *Hitachi Review*.

[6] M. Nafari and J. Shahrabi. 2010. A temporal data mining approach for shelf-space allocation with consideration of product price. In *Expert Systems with Applications*.

[7] Djamila Ouelhadj and Sanja Petrovic. 2009. A Survey of Dynamic Scheduling in Manufacturing Systems. *Journal of Scheduling*.

[8] Marie-Pierre Pacaux-Lemoine, Damien Trentesaux, Gabriel Zambrano Rey, and Patrick Millot. 2017. Designing Intelligent Manufacturing Systems Through Human-Machine Cooperation Principles. *Comput. Ind. Eng.* 111, C.

[9] Sabine Pfeiffer. 2016. Robots, Industry 4.0 and Humans, or Why Assembly Work Is More than Routine Work. *Societies*.

[10] Paolo Priore, Alberto Gomez, Raúl Pino, and Rafael Rosillo. 2014. Dynamic scheduling of manufacturing systems using machine learning: An updated review. *AI EDAM* 28, 83–97.

[11] Shuhui Qu, Jie Wang, and Juergen Jasperneite. 2018. Dynamic scheduling in large-scale stochastic processing networks for demand-driven manufacturing using distributed reinforcement learning. In *IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA)*.

[12] Cansu Soyleyici and Sinem Bozkurt Keser. 2016. A Hybrid Algorithm for Automated Guided Vehicle Routing Problem. *International Journal of Intelligent Systems and Applications in Engineering*.

[13] Qingmeng Tan, Yifei Tong, Shaofeng Wu, and Dongbo Li. 2019. Anthropocentric Approach for Smart Assembly: Integration and Collaboration. *Journal of Robotics*.

[14] Damien Trentesaux and Patrick Millot. 2015. A Human-Centred Design to Break the Myth of the "Magic Human" in Intelligent Manufacturing Systems. In *Service Orientation in Holonic and Multi-Agent Manufacturing (Studies in Computational Intelligence)*, Vol. 640. Springer, 103–113.

Appendices

A ALGORITHMS

We provide a high-level algorithm to facilitate quick insight into the architecture overview of *OPaPi* as shown in Figure 2. There are four main modules for *OPaPi* and each module is tightly integrated with human interaction. Manufacturing factory manager with domain knowledge is a main user to interact with the *OPaPi*. Also, the workers are able to view their assigned routes and schedules using the system as well as make an update every time they pick-up an item. In the following subsections, we provide a high-level algorithm of main modules like optimized item placement in the subsection A.1, optimized rack placement in subsection A.2, route and schedule cost computation in the subsection A.3, and optimized routes and schedule in the subsection A.4. Data, results, and human-interaction are commonly described in the subsection A.1 and all the modules are described as the function in the following subsections.

A.1 Optimized item placement

```
Data: DB_manufacturing, inventory_layout, rack_details,
      rack_placement, set_of_constraints,
      new_orders_items_pickup_list
Result: make informed decisions to re-organize the inventory
        layout, select optimal routes for parts pick-up
        schedule choices and interactive visualization for
        decision making
/* Human-Interaction: aggregate trajectory
   visualization with selected time window,
   real-time investigation of parts-picking routes
   and selection of the schedule from the list of
   schedule options with the associated cost (e.g.,
   travel distance and corresponding trajectory
   visualization), select/drag/drop racks in the
   layout, selection of next visualization from
   the ranked lists etc.                            */
/* Initialization                                   */
batch_DB = extract_transaction_DB (DB_manufacturing);
item_set = find_unique_itemset(batch_DB);
/* Optimized item placement function                */
Function Optimized_item_placement(DB_manufacturing,
  inventory_layout, rack_details, rack_placement,
  set_of_constraints)
:
    freq_max_itemsets = apply_FPMax(batch_DB);
    sorted_freq_max_itemset =
      sort_by_itemset_length(freq_max_itemsets);
    racks_available = True;
    while item_set != NULL and racks_available == True do
        top_max_itemset = get_top_pattern
          (sorted_freq_max_itemset);
        rack_no, items_placed = placement_in_rack
          (top_max_itemset, set_of_constraints);
        item_set = delete_from_itemset(item_set,
          top_req_max_itemset);
        if item_set != NULL or available_racks != 0 then
            sorted_freq_max_itemset =
              revised_frq_max_itemset (
              sorted_freq_max_itemset, top_max_itemset );
        else
    end
    return rack_details;
```

Algorithm 1: *OPaPi* Optimized item placement algorithm

A.2 Optimized rack placement

```
/* Optimized rack placement function */
Function Optimized_rack_placement(inventory_layout,
  rack_details, time_window, set_of_constraints)
:
    racks_placement =
      racks_placement_generated(inventory_layout,
      rack_details);
    /* visualizations are generated for new
       arrangement based on past transaction to
       calculate the total cost. */
    generate_visualization_ranked_ascending_cost(rack_placement,
      time_window);
    while user_select_rack_placement != True do
        user_analyze(aggregate_trajectory_with_cost_vis,
          time_window);
        if constraints_cost_satisfactory == False then
            rack_placement =
              user_manually_move_racks(rack_placement);
            user_analyze(aggregate_trajectory_with_cost_vis,
              time_window)
        else if next_visualization_available == True then
            user_selects_next_vis_from_ranked_list();
            repeat_the_process_until_user_satisfied();
        else
            user_select_rack_placement = True ;
            rack_placement = selected_rack_placement() ;
        end
    end
    return rack_placement
```

Algorithm 2: *OPaPi* Optimized rack placement algorithm

A.3 Route and schedule cost computation

```
Function
  ranked_list_route_schedule_cost_computation(new_orders
  _items_pickup_list,selected_distance_matrix,inventory_layout,
  rack_placement, rack_details, time_window,set_of_constraints)
:
    hybrid_rack_to_rack_distance =
      compute_hybrid_dist(selected_distance_matrix,
      rack_placement, rack_details, set_of_constraints);
    rack_to_rack_dist = (inventory_layout,
      hybrid_rack_to_rack_distance);
    ranked_list_item_routes_schedule_total_cost =
      get_TSP_route_schedule_cost_list
      (new_orders_items_pickup_list, rack_to_rack_dist,
      set_of_constraints);
    return ranked_list_item_routes_schedule_total_cost
```

Algorithm 3: *OPaPi* generating ranked list of routes and schedule algorithm

A.4 Optimal routes and schedule

```
/* Optimized routes and schedule function */
Function
  Optimized_routes_schedule(new_orders_items_pick-
  up_list, selected_distance_matrix,inventory_layout,
  rack_placement, rack_details, time_window,
  set_of_constraints)
:
    ranked_list_item_routes_schedule_total_cost =
      ranked_list_route_schedule_cost_computation
      (selected_distance_matrix, rack_placement, rack_details,
      set_of_constraints, inventory_layout);
    /* interactive visualization will allow the
       manager to select the optimal schedule from
       the ranked list. */
    while user_converge_to_schedule != True do
        user_analyze(aggregate_trajectory_with_cost_vis,
          time_window);
        if any_realtime_events_observed() == True then
            item_routes_schedule, total_cost =
              user_manually_change_route_schedule ();
            remaining_items_list =
              item_list(new_orders_items_pickup_list,
              items_already_delivered);
            new_orders_items_pickup_list =
              update_details(item_routes_schedule, total_cost,
              remaining_items_list);
            /* manager (user) can provide the
               time_window to look at the aggregate
               trajectories or individual worker
               trajectory for remaining pickup items
               from the present time to the expected
               assembly line delivery time */
            ranked_list_route_schedule_cost_computation
              (new_orders_items_pickup_list,
              selected_distance_matrix, rack_placement,
              rack_details, set_of_constraints,
              inventory_layout);
            user_analyze(aggregate_trajectory_with_cost_vis,
              time_window)
        else if user_not_satisfied() == True then
            continue_with_next_option_and_interactive_changes();
        else
            routes_schedule= opti-
              mal_schedule_and_routes_selected_broadcast();
              manager_converge_to_schedule = True;
        end
    end
    return routes_schedule
```

Algorithm 4: *OPaPi* Optimized routes and schedule algorithm