Optimization of Circular Conveyor Belt Systems with Multi-Commodity Network Flows

Antonin Novak[1], Matous Pikous[1,2] and Zdenek Hanzalek[1]
[1]*Czech Institute of Informatics, Robotics and Cybernetics, Czech Technical University in Prague, Czech Republic*
[2]*Faculty of Electrical Engineering, Czech Technical University in Prague, Czech Republic*

Keywords: Circular Conveyor Belts, Manufacturing, Multi-Commodity Network Flow, Mixed-Integer Linear Programming.

Abstract: Modern industrial production with alternative process plans and the use of complex machine equipment increases requirements for its intralogistics operations in terms of efficiency, resilience, and flexibility. One of the most common solutions for transporting workpieces between the manufacturing stations is a system of conveyor belts where each conveyor rotates in a fixed direction at a constant speed. The movement of the individual workpieces can be controlled only indirectly via a set of gates connecting different carousels. In this paper, we aim to increase the flexibility of conveyor belt systems by carefully scheduling the gates to route the workpieces efficiently along the production line according to their process plans. The key component of our solution is the discretization of both the time and positions on the belts to represent the system by a directed graph with circular components. To find the routing of workpieces that minimizes the total flow time, we have reduced the problem to the integer multi-commodity flow on the time-expanded network with an extension for the vertex precedences. Despite the simplicity of the formulation, the results suggest that off-the-shelf solvers can find optimized routing for instances with tens of workpieces and more than hundreds of belt positions within a few minutes.

1 INTRODUCTION

One of the core concepts in Industry 4.0 is a highly flexible and customized production. To keep up with the rising demand for many variants of products, deployment of more complex machine equipment such as reconfigurable manufacturing systems (RMS) (Fatemi-Anaraki et al., 2022) and intelligent internal logistics systems are required. Due to the fact that products are highly customized, they no longer follow the identical production process plan, but rather different product variants need to visit different manufacturing stations. Therefore, the stations are interconnected with a transport system that moves workpieces over the shop floor.

The transport and routing can be realized with, e.g., autonomous ground vehicles (AGV) (Qiu et al., 2002) or monorail systems such as Montrac, which use an individual transport platform to handle the movement of each workpiece separately. Although these systems are very flexible, they share certain disadvantages, such as low throughput and high cost.

Another option is conveyor belt systems, such as the one shown in Figure 1. Here the difference is that the system consists of several individual circular conveyor belts interconnected with controllable gates. Each individual belt has its independent asynchronous electric motor that rotates the belt at a constant speed. Therefore, all pieces sharing the same belt are moved simultaneously in the direction defined by the drive movement. If a piece needs to be transported to a machine located at a different belt, the piece needs to stay on the belt before reaching a specific gate which is switched at the right moment to transfer the piece to a different belt. The advantage of conveyor belt systems is that they are cheaper to operate and of-

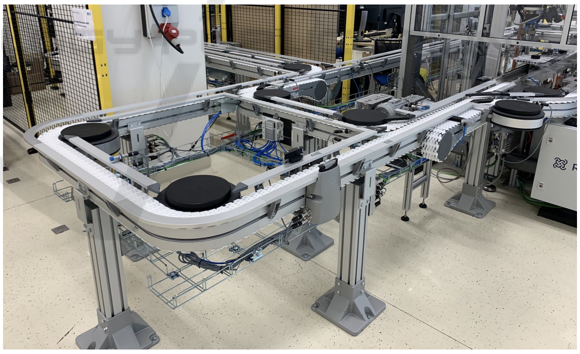


Figure 1: Conveyor belt in Testbed for Industry 4.0 at CIIRC CTU.

Novak, A., Pikous, M. and Hanzalek, Z.
Optimization of Circular Conveyor Belt Systems with Multi-Commodity Network Flows.
DOI: 10.5220/0011784800003396
In *Proceedings of the 12th International Conference on Operations Research and Enterprise Systems (ICORES 2023)*, pages 203-210
ISBN: 978-989-758-627-9; ISSN: 2184-4372

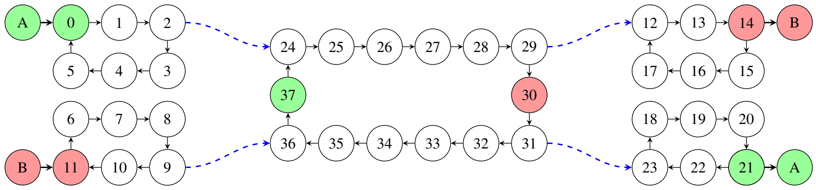


Figure 2: System model with five clockwise-rotating carousels, four gates, and two workpieces, each with three stations to visit.

fer higher capacity. However, they are less flexible as the individual piece movement is controlled only indirectly by the gate switches. Moreover, the individual conveyor belts are shared resources in the system; thus, one needs to carefully schedule access to them, e.g., to prevent collisions of pieces when switching the belts.

In this work, we aim to improve the flexibility of conveyor belt systems by careful scheduling of operations. The inspiration was taken from the Testbed for Industry 4.0 located at the Czech Technical University in Prague (Novák and Vyskočil, 2022), where a set of machine tools and robots is interconnected with a system of conveyor belts with controllable gates (see Figure 1). Our main idea is to discretize time and positions on the belts and model the system with a graph consisting of circular components, as depicted in Figure 2. Then, we employ an extended integer multi-commodity flow problem formulation for a time-expanded system graph to find optimal routing for the set of workpieces while visiting all requested stations without collisions. The proposed formalism easily allows for the minimization of different criteria expressed as a function of the completion times of pieces, such as the makespan or the total flow time. The experiments demonstrate that even though the resulting optimization models are quite large, the underlying network flow structure of the problem allows mixed-integer linear programming solvers to retain impressive scaling capability. Specifically, the main contributions of this paper are:

(i) proposing a discretized model of a circular conveyor belt system and the optimization problem of workpiece routing with a sequence of stations to visit,

(ii) a formulation of the problem via integer multi-commodity flow problem on a time-expanded network with the extension for vertex precedences,

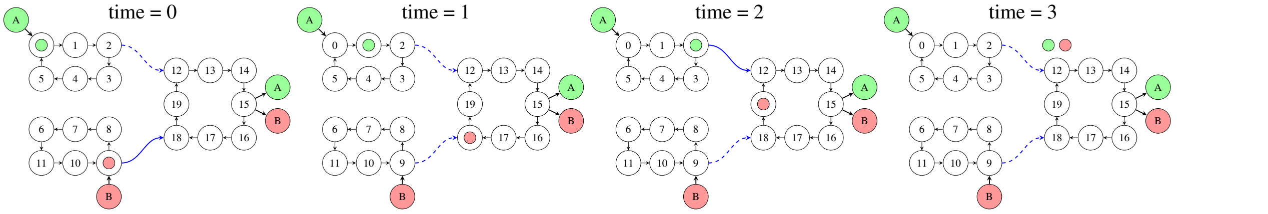
(iii) the experimental evaluation of the proposed mixed-integer linear programming model.

2 RELATED WORK

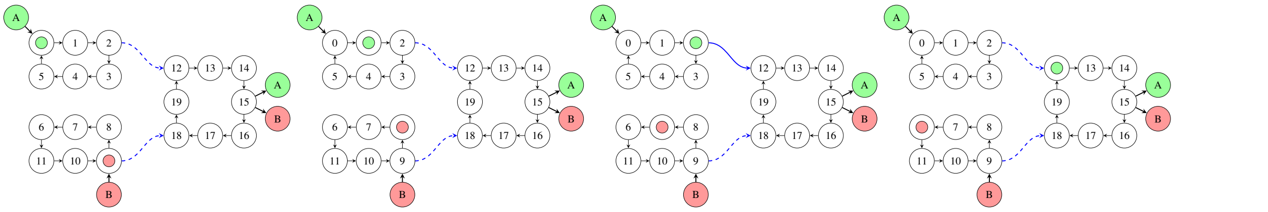
One of the most frequently appearing applications of the conveyor belt systems can be seen in various package sorting tasks, e.g., in fulfillment centers. For example, in (Chen et al., 2021), a simulation-optimization approach is proposed to improve the processing capacity of a circular conveyor belt in a parcel sorting system by designing skip connections to improve the processing capacity. However, the conveyor belt layout is often fixed and is not subject to optimization. In these cases, the scheduling of operations can be applied to improve the utilization of the system. For example, (Bock and Bruhn, 2021) study the problem of mould injection for product casting with a circular conveyor belt. In their problem, they also consider a circular conveyor belt with several robotic stations which perform activities on the workpieces traveling on the belt.

From the perspective of the underlying optimization problem, our problem is closely related to the multi-agent path-finding problem (MAPF) (Barták et al., 2018). The MAPF is among the classical problems in the literature, being studied under many different settings (Stern et al., 2019). The classical version of MAPF assumes that a set of agents need to find paths on an undirected graph from their sources to the destinations such that they avoid conflicts at all vertices. However, such a setting does not apply to our problem since our agents (i.e., workpieces) operate on a directed graph, they cannot wait at an arbitrary vertex to avoid collisions, they need to visit multiple locations in a specific order, and they do disappear at target (Stern et al., 2019).

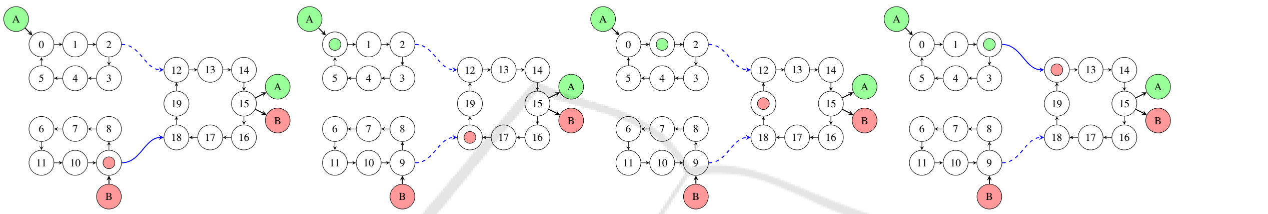
Another related, but more general, optimization problem is the resource-constrained shortest path problem (RCSPP) (Pugliese and Guerriero, 2013). RCSPP at its most general setting specifies a set of resources and so-called resource extension functions (REFs) which adjust the values of resources along the found *s*-*t* path in the given graph. However, the majority of the existing efficient algorithms for RCSPP consider specific subsets of the problem, such as non-decreasing REFs, rather than the general case. What



(a) Infeasible solution: conflict in vertex 12 at time 3.



(b) Feasible solution: conflict is avoided by postponing the transfer of *B* to the other belt by one rotation.



(c) Feasible solution: conflict avoided by a delayed release of A to the belt.

Figure 3: Examples of different solutions.

is more, when k workpieces are present, we need to find k vertex-disjoint paths, which further complicates the modeling as an RCSPP.

Although the studied problem displays common characteristics with the existing problems, such as MAPF or RCSPP, we are not aware of any existing problem or an algorithm that would efficiently encapsulate the problem addressed in this work.

3 SYSTEM MODEL

3.1 Model Description

In this section, we describe the assumptions behind the used system model, and we formally define the problem statement. Similarly as (Bock and Bruhn, 2021), we assume that positions on the belt and time instants are discretized, meaning that each belt can transport at any moment only a finite number of pieces. Indeed, this is a reasonable assumption because the pieces on the belt are spaced out by sufficient margins to avoid problems. Thus, we assume that each belt has a fixed sense of rotation with the speed of one position per time unit.

A conveyor belt system consists of several independent carousels interconnected by controllable gates that can, at the specified moment, transfer a workpiece from one belt to another. The workpiece appears at its inbound location at its release time and is offloaded to the belt at the time defined by the schedule. After the workpiece is offloaded to the belt, it needs to visit a sequence of the required positions and is removed from the system as soon as it reaches its outbound location. For the specific example, see Figure 2. The displayed graph represents the formalization of a system with two workpieces and five carousels interconnected by four gates (depicted in blue). The workpiece A has to visit stations $0 \rightarrow 37 \rightarrow 21$ while workpiece B visits $11 \rightarrow 30 \rightarrow 14$. The inbound and outbound locations for A are connected to locations 0 and to 21, respectively. For B, the inbound location is connected to 11, and the outbound to 14.

3.2 Problem Statement

The input to the problem consists of a directed graph $G = (V, E)$, which describes the system of conveyor belts. It is assumed that G consists of a finite number of strongly connected components, where a component is the cycle graph C_n representing a carousel with n positions. Next, we are given a set of gates

$S \subset E$, which represents controllable gates in the system transporting workpieces between carousels.

The workload is represented by a set of k workpieces $M = \{1,\ldots,k\}$. Each workpiece $m \in M$ is associated with the release time $r^{(m)} \in \mathbb{N}_0$, denoting its earliest possible time when the workpiece can be loaded on the belt. Before the workpiece is loaded on the belt, it stays at its inbound location $I^{(m)} \in V$. Furthermore, the workpiece $m \in M$ specifies the sequence of vertices (stations) $\pi^{(m)} = \left(\pi_1^{(m)},\ldots,\pi_{n_m}^{(m)}\right)$, $\pi_i^{(m)} \in V$ which have to be visited by m. After position $\pi_{n_m}^{(m)}$ is reached by m, it is moved to its outbound location $O^{(m)} \in V$ and is effectively removed from the system since it no longer occupies any belt position.

The solution to the problem is represented by a schedule $y_t^{(m)} \in V$, which for each workpiece m specifies the position it occupies at time t. Furthermore, it defines for each gate $(u,v) \in S$ a binary value $z_t(u,v) \in \{0,1\}$ which is set to 1 if and only if the gate (u,v) is activated at time t.

We say that the schedule is feasible if workpieces satisfy their release times and for any $m_a, m_b \in M$, $m_a \neq m_b$: $\forall t : y_t^{(m_a)} \neq y_t^{(m_b)}$, i.e., at any moment, no two workpieces occupy the same position. Furthermore, a workpiece m which at time t occupies $y_t^{(m)}$ subsequently occupies its neighboring position of $y_t^{(m)}$ at time $t+1$. The neighboring position of vertex $v \in V$ at time t is either $u \in V$ such that $(v,u) \in E$ (i.e., u is the successor position of the v on the same carousel) or $w \in V$ if $(v,w) \in S$ and $z_t(v,w) = 1$, i.e., the workpiece is transported to a different carousel with (v,w) gate. The objective is to minimize the total flow time, i.e., the sum of differences between the time reaching the outbound location and the release time over every workpiece.

3.3 Example

To demonstrate the defined quantities and constraints of the problem, please see an example in Figure 3. There, we depict three different solutions for the problem with two workpieces A and B. We assume that the release times of both workpieces are equal to zero, i.e., $r^{(A)} = r^{(B)} = 0$ and the sequences of stations to visit are $\pi^{(A)} = (0,15)$ and $\pi^{(B)} = (9,15)$.

In Figure 3a, both workpieces are loaded to the carousel at time $t = 0$, thus $y_0^{(A)} = 0$ and $y_0^{(B)} = 9$. At the time $t = 1$, workpieces are moved to their neighboring location. In this solution, the gate $(9,18)$ was activated at time $t = 0$, i.e., $z_0(9,18) = 1$. In subsequent time instant, both workpieces are again moved to their neighboring positions; therefore, $y_1^{(A)} = 1$ and $y_1^{(B)} = 18$. After one additional move, the gate $(2,12)$ is activated at time 2 (i.e., $z_2(2,12) = 1$), thus, we have $y_3^{(A)} = 12$. However, this violates the feasibility condition since $y_3^{(B)} = 12$ as well, and the solution (a) is infeasible.

A different solution is shown in Figure 3b. In this case, the gate $(9,18)$ is not activated at time $t = 0$. Therefore, the workpiece B is forced to perform one additional rotation with its initial carousel before it is (eventually) transferred. Thus, the conflict at position 12 is avoided at the expense of increased flow time of B by 6 time units.

Finally, the third solution is displayed in Figure 3c. Here, the workpiece B is unloaded onto the carousel immediately at time $t = 0$, whereas workpiece A waits at its inbound location until time $t = 1$. Since A is released 1 time unit later than in solution (a), it also avoids the conflict at position 12, but the total flow time is smaller than in solution (b).

The above examples show some important considerations to be made when solving the problem—both the timings of the gates as well the times when the workpieces are loaded to the carousels affect the quality of the solution. Thus, they need to be considered simultaneously, which represents an interesting optimization problem.

4 TIME-EXPANDED INTEGER MULTI-COMMODITY FLOW

First, we explain the concept of a time-expanded network for the conveyor belt scheduling problem. Then, we give a mixed-integer linear programming (MILP) formulation of the problem, which resembles an ordinary integer multi-commodity network flow problem with one additional constraint.

4.1 Time-Expanded Network

The main difficulty of using network flow formalism for problems with time-related constraints (e.g., release times and vertex ordering) is that a flow in the network does not capture the notion of time. One of the possible options how to accommodate these constraints is the so-called *time expansion* of the network, which is used, e.g., for dynamic network flow problems (Ahuja et al., 1988).

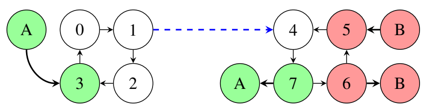


Figure 4: An example of a system G.

The core idea of the time expansion is to construct copies of the original graph, where each copy represents the network at a specific time instant. The copies of the original network are connected in a way that represents possible state transitions between the past and future time instants. In our case, the time-expanded network $\overline{G} = (\overline{V}, \overline{E})$ of the conveyor belt system $G = (V, E)$ is constructed as follows. First, the value of time horizon $H \in \mathbb{N}$ needs to be chosen such that all workpieces $m \in M$ can visit their stations $\pi^{(m)}$ and reach the outbound location $O^{(m)}$ within H time steps.

Next, for each time instant within the horizon H, one copy of the network, except for inbound and outbound locations, is created as a so-called layer. At each layer, all edges that exist in the original graph G, i.e., the representation of belt rotations and the positions of the gates, are removed. Instead, an edge between vertex v in layer t and vertex u in layer $t+1$ is introduced, if and only if u is *possible* neighboring vertex of v at time t in G, i.e., either u is a successor position on the conveyor or $(v, u) \in S$ is a gate. Finally, for each inbound $I^{(m)}$ and outbound location $O^{(m)}$ a corresponding vertex is added to $\overline{V}$.

Note that the edges $\overline{E}$ in time-expanded network $\overline{G}$ can be uniquely associated with a specific time instant. Indeed, each outgoing edge from any vertex $i \in \overline{V}$ can be assigned to a time instant t corresponding to in which layer t the vertex i is contained. Additionally, an edge leaving inbound location $I^{(m)}$ can be associated with time instant $t-1$ if it enters a vertex v in layer t. In this way, every edge $e \in \overline{E}$ in a time-expanded network can be described with a triplet $(t, i, j) \in \overline{E}$. To model the release time constraint for a workpiece $m \in M$, we introduce a single vertex representing inbound location $I^{(m)}$ and connect it with the position $\pi_1^{(m)}$ in every layer $t \geq r^{(m)}$. Similarly, $\pi_{n_m}^{(m)}$ location at every layer $t \geq r^{(m)}$ is connected to a single outbound location $O^{(m)}$.

To demonstrate the structure of the time-expanded network, let us consider a system described by the graph G in Figure 4 with the parameters of workpieces given in Table 1. The resulting time-expanded network $\overline{G}$ with horizon $H = 4$ (i.e., in total five layers including the time instant $t = 0$) can be seen in Figure 5.

Table 1: Example parameters of workpieces.

workpiece m	stations $\pi^{(m)}$	release time $r^{(m)}$
A	$3 \rightarrow 7$	1
B	$5 \rightarrow 6$	2

4.2 Integer Multi-Commodity Flows with Vertex Precedences

Having the time expansion of the network, the movements of the workpieces can be modeled as flows transported from their inbound to the outbound locations. To consider the individual identities of the flows, they need to be modeled as different commodities to prevent that, e.g., the workpiece A would reach outbound location $O^{(B)}$ instead of $O^{(A)}$. The integer multi-commodity flow problem specifies the set of k commodities to be transported over a directed network $\overline{G}$ and a k-dimensional balance vector $b(v)$ for each vertex $v \in \overline{G}$. In our case, we set $b^{(m)}(I^{(m)}) = 1$, $b^{(m)}(O^{(m)}) = -1, \forall m \in M$ and for all other vertices v we set $b^{(m)}(v) = 0, \forall m \in M$. The objective function is the sum of costs for all edges times the amount of flow transported over the edge. The cost of all edges $e \in \overline{E}$ is equal to zero except the edges $(v, O^{(m)})$ that have cost $t - r^{(m)}$ if v is a vertex in layer t. In this way, we model minimizing the total flow time of workpieces $m \in M$.

As a next constraint, we need to ensure that, at most, one unit of a flow can enter any vertex to avoid situations such as the one depicted in Figure 3a at time $t = 3$. This constraint can be easily accommodated into multi-commodity network flow formalism by the vertex expansion (Ahuja et al., 1988). Furthermore, the solution needs to enforce that if for some workpiece $m \in M$ and its station sequence $\pi^{(m)} = \left(\pi_1^{(m)}, \ldots, \pi_i^{(m)}, \ldots, \pi_j^{(m)}, \ldots, \pi_{n_m}^{(m)}\right)$ the $\pi_i^{(m)}$ is visited at time t_i, and $\pi_j^{(m)}$ is visited at time t_j, then $t_i < t_j$. However, from the perspective of a solution, it is important to distinguish whether the workpiece $m \in M$ visits the station $\pi_i^{(m)}$ in a sense as it is given by the problem statement (i.e., operating on the workpiece), or whether it transits through the position to reach a different destination. Although this seems obvious, it introduces surprising difficulties when modeling the problem with an ordinary multi-commodity network flow. Therefore, we introduce an extension of unit-capacity integer multi-commodity flows that we call vertex precedences. A vertex precedence for commodity $m \in M$ is given in the form of $i \rightarrow j$, where $i, j \in V$ are vertices in the original graph G. The constraint requires that the resulting flow of the commodity m in the time-expanded network $\overline{G}$ transits through the vertex i in layer t_i and the vertex j in layer t_j, such that $t_i < t_j$.

An inconvenient property of this extension is that it disqualifies the use of the existing algorithms for the multi-commodity network flow problem. However, we note that the original integer multi-commodity

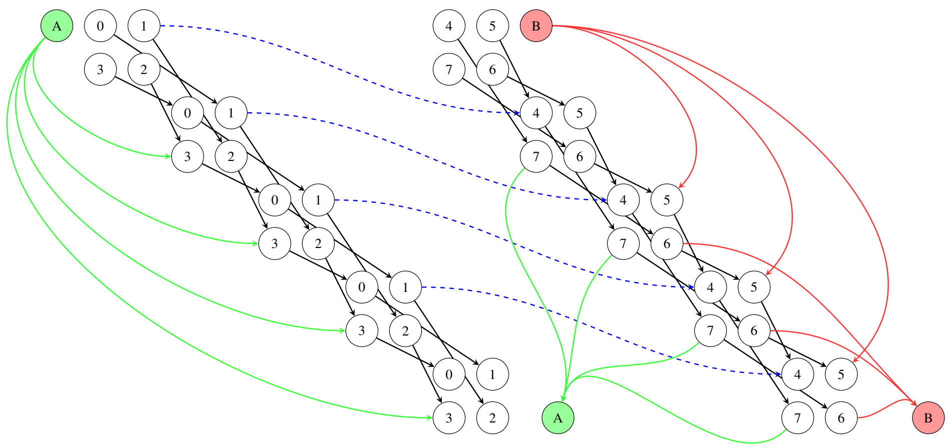


Figure 5: Example of a time-expanded network $\overline{G}$.

network flow problem is NP-hard already for two commodities even when restricted to networks with unit capacities (Garey and Johnson, 1979). Therefore, even if there would be an efficient way of accommodating vertex precedences into the ordinary integer multi-commodity flow problem, a substantial complexity is already introduced by the formalism itself. Nevertheless, the practical experience with mixed-integer linear programming (MILP) formulations of the integer multi-commodity flow problem indicates that even large instances can be solved in a reasonable time.

4.3 MILP Model

Let $T = \{0, 1, \ldots, H-1\}$ be a set of all time instants within the horizon H. The main decision variable is $f_t^{(m)}(i,j) \in \{0,1\}$ with the meaning whether the commodity m is transported along the edge $(t,i,j) \in \overline{E}$ in time-expanded network $\overline{G}$. Furthermore, we use a binary indicator $x_{t,i}^{(m)}$, which enforces that the commodity m leaves vertex i at layer t. This variable is used to enforce the vertex precedence constraints. With the above, we state the full model as:

$$\min \sum_{t\in T} \sum_{m\in M} \sum_{i\in \overline{V}} (t - r^{(m)}) \cdot f_t^{(m)}(i, O^{(m)}) \tag{4.1}$$

subject to

$$\sum_{(t,v,j)\in\overline{E}} f_t^{(m)}(v,j) - \sum_{(t-1,i,v)\in\overline{E}} f_{t-1}^{(m)}(i,v) = b^{(m)}(v)$$

$$\forall v \in \overline{V}, \forall m \in M \tag{4.2}$$

$$\sum_{m\in M} \sum_{(t,i,v)\in\overline{E}} f_t^{(m)}(i,v) \le 1 \quad \forall v \in \overline{V} \tag{4.3}$$

$$\sum_{j\in\overline{V}} f_t^{(m)}(i,j) \ge x_{t,i}^{(m)} \quad \forall t \in T, \forall i \in \pi^{(m)} \tag{4.4}$$

$$\sum_{t\in T} x_{t,i}^{(m)} = 1 \quad \forall m \in M, \forall i \in \pi^{(m)} \tag{4.5}$$

$$\sum_{t\in T} x_{t,i}^{(m)} \cdot t + 1 \le \sum_{t\in T} x_{t,j}^{(m)} \cdot t \quad \forall m \in M, \forall (i \to j) \in \pi^{(m)} \tag{4.6}$$

$$f_t^{(m)}(i,j) \in \{0,1\} \quad \forall m \in M, \forall (t,i,j) \in \overline{E} \tag{4.7}$$

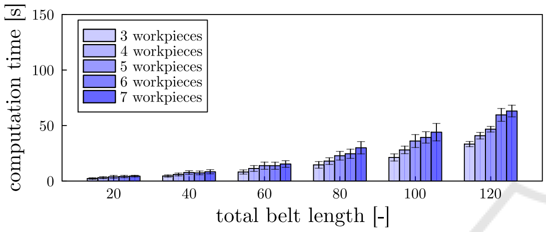
$$x_{t,i}^{(m)} \in \{0,1\} \quad \forall m \in M, \forall t \in T, \forall i \in \pi^{(m)}. \tag{4.8}$$

The objective (4.1) represents the minimization of the total flow time. Flow conservation constraint is expressed by (4.2). Since the value of $b^{(m)}(v)$ is set as described in Section 4.2, the constraint (4.2) also enforces that workpiece m appears at inbound location $I^{(m)}$ and eventually reaches its outbound $O^{(m)}$. The constraint (4.3) models vertex capacities to enforce that, at most, one workpiece occupies a belt position at any time.

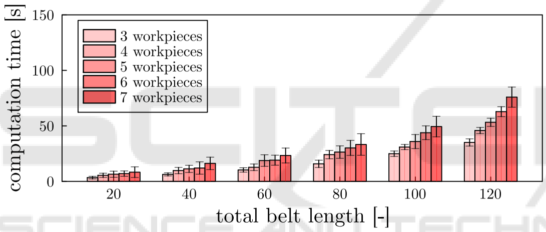
Finally, the constraints (4.4)–(4.6) are used to model vertex precedence constraints. If variable $x_{t,i}^{(m)}$ is set to 1, then we interpret it such that an operation is performed on the workpiece m at time t. Therefore, we require that such the operation $i \in \pi^{(m)}$ is performed exactly once by constraint (4.5), and the corresponding commodity must enter (and leave) the specific vertex at the required time by constraint (4.4). The correct ordering of vertices in $\pi^{(m)}$ is enforced by (4.6). For any two consecutive elements of the station sequence $(i \to j) \in \pi^{(m)}$ the model computes the times when the operations are performed by terms $\sum_{t\in T} x_{t,i}^{(m)} \cdot t$ and $\sum_{t\in T} x_{t,j}^{(m)} \cdot t$. Then, it is enforced that the time of the operation in the vertex i is smaller than in j.

5 EXPERIMENTS

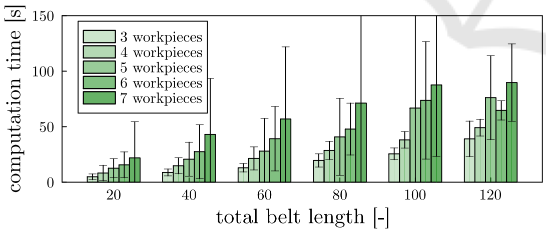
The performance of the proposed multi-commodity flow model has been assessed by the following set of experiments. The key parameters that influence the complexity of a problem instance are (i) the number of workpieces, (ii) the number of stations to visit, (iii) the number of belts and their total length, and (iv) the length of the time horizon. The specific range of the values used for these parameters is described in each experiment. The instances were generated such that the resulting system of conveyor belts is a strongly connected graph.



(a) Instances with two stations.



(b) Instances with three stations.



(c) Instances with four stations.

Figure 6: Effect of the number of required stations on computational time.

The time expansion of the graph was performed in Python 3. The resulting MILP formulation was solved with the Gurobi 9.1.1 solver utilizing at most eight threads of Intel Xeon E5-2690 CPU. The measured CPU times reflect both the time spent by time expansion as well as the computation time of the solver.

5.1 Effect of the Number of Stations

To assess how the computational time scales with respect to the number of required stations, we have performed the following set of experiments with the varying numbers of stations required. In total, we have generated 3360 instances with $|M| \in \{3,\dots,7\}$ workpieces, total belt length was generated in interval $[20,120]$ and the number of required stations by all workpieces $m \in M$ was $|\pi^{(m)}| \in \{2,3,4\}$. The length of time horizon H was set to 180.

The results are displayed in Figure 6. Each graph displays mean computational times with standard deviations grouped by the total belt length and the number of workpieces M. As expected, the complexity of an instance depends largely on the number of required stations as it introduces additional variables (4.8) and deteriorates the structure of the problem further from the ordinary multi-commodity flow. The main challenges for the model appear with instances with four stations. There, we can see that the computational times start to fluctuate under the presence of outliers represented by the occasional long running time of the solver. The practical experience with the solver behavior has revealed that the optimal solution is often attained soon after the root node is solved. However, this follows after quite a long preprocessing step, which greatly reduces the size of the model. Therefore, it seems that improvements in the optimization model are possible.

5.2 Effect of the Horizon Length

The experiments in Section 5.1 were run with the fixed length of the time horizon H. To test its influence on the computation time, we have fixed the number of stations to 3 and generated a total of 1120 instances varying in the total belt length that was set to be contained within $[20,120]$. This set of instances was solved with time horizon lengths $H \in \{120,140,\dots,220\}$. The results are displayed in Figure 7.

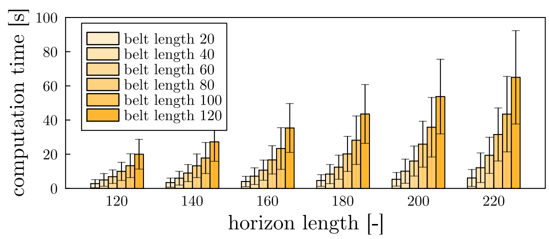


Figure 7: Scaling with respect to the length of the horizon H.

As suggested by our preliminary experiments, the length of the time horizon has only a moderate effect on the overall computation time.

6 CONCLUSION

We studied the problem of optimal routing for the set of workpieces in a system of circular conveyor belts where the movement of workpieces cannot be directly affected, but they can be controlled indirectly via the set of gates connecting different carousels. Our main idea used in the solution is to discretize the time and positions on the belts and to model the system with a graph consisting of circular components. Then we formulate it as an integer multi-commodity flow problem for a time-expanded system graph with the vertex precedence constraint.

For future work, we suggest considering the energy consumed by the system. At certain moments, the belts might be switched to a power-saving mode, e.g., by reducing the speed of the movement or shutting down completely.

ACKNOWLEDGEMENTS

This work was supported by the EU and the Ministry of Industry and Trade of the Czech Republic under the Project OP PIK CZ.01.1.02/0.0/0.0/20_321/0024399.

REFERENCES

Ahuja, R. K., Magnanti, T. L., and Orlin, J. B. (1988). Network flows.

Barták, R., Švancara, J., and Vlk, M. (2018). A scheduling-based approach to multi-agent path finding with weighted and capacitated arcs. In *AAMAS'18*, pages 748–756.

Bock, F. and Bruhn, H. (2021). Case study on scheduling cyclic conveyor belts. *Omega*, 102:102339.

Chen, T.-L. et al. (2021). Solving the layout design problem by simulation-optimization approach–a case study on a sortation conveyor system. *Simulation Modelling Practice and Theory*, 106:102192.

Fatemi-Anaraki, S. et al. (2022). Scheduling of multi-robot job shop systems in dynamic environments: Mixed-integer linear programming and constraint programming approaches. *Omega*, page 102770.

Garey, M. R. and Johnson, D. S. (1979). Computers and intractability.

Novák, P. and Vyskočil, J. (2022). Digitalized automation engineering of Industry 4.0 production systems and their tight cooperation with digital twins. *Processes*, 10(2).

Pugliese, L. D. P. and Guerriero, F. (2013). A survey of resource constrained shortest path problems: Exact solution approaches. *Networks*, 62(3):183–200.

Qiu, L., Hsu, W.-J., Huang, S.-Y., and Wang, H. (2002). Scheduling and routing algorithms for AGVs: a survey. *International Journal of Production Research*, 40(3):745–760.

Stern, R. et al. (2019). Multi-agent pathfinding: Definitions, variants, and benchmarks. In *SoCS'19*.