Column Generation for Outbound Baggage Handling at Airports

Frey Markus
TUM School of Management, Technische Universität München, Munich, Germany, markus.frey@tum.de

Kolisch Rainer
TUM School of Management, Technische Universität München, Munich, Germany, rainer.kolisch@tum.de
Edward P. Fitts Department of Industrial and Systems Engineering, North Carolina State University, Raleigh, USA

Artigues Christian
LAAS–CNRS/Université de Toulouse, Toulouse, France, artigues@laas.fr

The planning of outbound baggage handling at international airports is challenging. Outgoing flights have to be assigned and scheduled to handling facilities at which the outgoing baggage is loaded into containers. To avoid disruptions of the system the objective is to minimize workload peaks over the entire system. The resource demand of the jobs which have to be scheduled depends on the arrival process of the baggage. In this paper we present a time-indexed mathematical programming formulation for planning the outbound baggage. We propose an innovative decomposition procedure in combination with a column generation scheme to solve practical problem instances. The decomposition significantly reduces the symmetry effect in the time-indexed formulation and also speeds up the computational time of the corresponding Dantzig-Wolfe formulation. To further improve our column generation algorithm we propose state-of-the-art acceleration techniques for the primal problem and the pricing problem. Computational results based on real data from a major European Airport show that the proposed procedure reduces the maximal workloads by more than 60% in comparison to the current assignment procedure used.

Key words: outbound baggage handling; binary programming; column generation; cutting planes
Subject classifications: freight/materials handling; applications; Benders' decomposition
History: Received February, 2015; received first revision November, 2015; received second revision July, 2016;

Appendix A: Proof of Theorem 1

For the proof it is sufficient to consider flights assigned to the same carousel. Given a solution of the TIF, providing a feasible assignment of flights to carousels in terms of parking positions and working stations as well as a feasible schedule for flights' baggage handling, we have to show that that there is an assignment procedure which can polynomially transform the TIF's solution into a solution satisfying the assignment

regulation described in §2. The regulation requires that all containers of the flights are assigned to the carousel such that all assigned working stations for the flights are reached and there is no overlapping of containers or working stations with other flight handled simultaneously. For the polynomial assignment procedure, we order the flights by increasing baggage handling ending times. A flight's container row is assigned clockwise from 1 to K_r^{pp} beginning with the parking position immediately following the container row of the previously assigned flight such that all assigned working stations are reached. Figure 3 shows the result of the assignment procedure for the flights i_1, i_2 and i_3 requiring $P_{i_1} = 7$, $P_{i_2} = 4$ and $P_{i_3} = 5$ parking positions, respectively, to a carousel with 4 work stations and 20 parking positions. Let us assume that flight i_1's to i_{k-1}'s container rows are assigned with the above assignment procedure and assume, w.l.o.g., that flight i_k has to be assigned to the last available working station on that carousel. If flight i_k's container row has to be split on both sides of the carousel such that it overlaps with flight i_1's container row by $p \geq 1$ parking positions, we right-shift the container row of flight i_1 for p parking positions clockwise until the overlapping conflict is solved. If it is not possible to right-shift flight i_1's container row as it is blocked by flight i_2's container row, see e.g. Figure 3, we sequentially right-shift the container rows of the succeeding flights i_2 to i_{k-2}; a right-shift of flight i_{k-1}'s container rows would directly lead to a reduction of available parking positions for flight i_k. If it is not possible to right-shift the container rows of flight i_1 to i_{k-2} such that p parking positions for flight i_k become available in the working station segment of flight i_1, then there are either not enough parking positions available for flight i_k, or a right-shift of flight i_1's container row would contradict the requirement that flight i_1's container row reaches the assigned working stations. In the first case, we violate the assumption that it is possible to handle the flights i_1 to i_k simultaneously due to a violation of the parking position capacity. In the latter case, flight i_k would require at least $p \geq K_r^{\mathrm{ppws}}$ additional parking positions, and hence at least one additional working station, which violates the working station capacity and, thus, also contradicts the assumption that the flights can be handled simultaneously.

Appendix B: Preprocessing for the TIF

Let $\rho > 0$ be the number of bags transferred from the storage system to any carousel. If start time tuple $\tau = \langle s_\tau^{\mathrm{h}}, s_\tau^{\mathrm{d}} \rangle \in \mathcal{S}_i$ is selected for flight $i \in \mathcal{F}$, the number of bags stored up to period t is given by the expected amount of baggage arrived up to time t minus the amount of baggage already sent to the carousel, i.e.

$$\Phi_{t,\tau}^{i} = \left(\sum_{t' < \min\left\{t, s_\tau^{\mathrm{h}}\right\}} \mathbb{E}(A_{i,t'}) - \mathbb{1}_{\left\{t \in \mathcal{T} \mid s_\tau^{\mathrm{d}} \leq t\right\}}(t) \cdot \left(t - s_\tau^{\mathrm{d}} + 1\right) \cdot \rho \right)^{+}.$$

Let $\alpha > 0$ be the number of bags loaded into a container per period at a working station. Then, if w working stations are assigned to flight i, the number of bags on carousel c in period $t \in \mathcal{T} \setminus \{0\}$ is derived by the recursive formula

$$\Gamma_{t,\tau}^{i,w} = \left(\Gamma_{\tau,t-1}^{i,w} + \mathbb{1}_{\left\{s_\tau^{\mathrm{h}} \leq t < S_i^{\mathrm{e}}\right\}}(t) \cdot A_{i,t} + \mathbb{1}_{\left\{t \in \mathcal{T} \mid s_\tau^{\mathrm{d}} \leq t < S_i^{\mathrm{e}}\right\}} \min\left\{\rho, \Phi_{t,\tau}^{t}\right\} - \alpha \cdot w \right)^{+}$$

with $\Gamma_{\tau,0}^{i,w} = 0$. $\Gamma_{\tau,t-1}^{i,w}$ yields the number of bags on the carousel's conveyor belt from the previous period. The indicator function indicates whether the baggage handling has started such that the baggage amount $A_{i,t}$ arriving in period t is directed to the carousel. The second term adds the amount of baggage arriving at the assigned carousel from the storage system. Finally, the third term subtracts the bags taken from the conveyor belt and loaded into the containers.

Appendix C: Proof of Theorem 3

Trivially, the problem lies in NP. We proof the NP-hardness of the problem by a reduction from the Bin-Packing Problem, which is known to be NP-complete (see Garey and Johnson (1979)). Let us consider a single carousel with F working stations, i.e. all flights $\mathcal{F} = \{1, ..., F\}$ (items) can be handled simultaneously. Assume that the flights' baggage handling are scheduled from periods t_1 to t_T (bins) and that for each flight $i \in \mathcal{F}$ all bags have arrived in period t_0, i.e. we only have to decide about flights' start times of the storage depletion between periods t_1 to t_T. Once the baggage is released from the storage system at time $t_0 < t \leq t_T$, a workload of $s_i \leq B$ bags (item size) is immediately transferred to the carousel's conveyor belt were it arrives in period t. A solution for the scheduling of flights' storage depletion such that a workload lower or equal to B is obtained yields also a solution for the Bin Packing Problem in which items with a given size have to be packed in T bins with capacity B.

Appendix D: Proof of Proposition 2

If start time tuple τ_2 never leads to a violation of the storage capacity, and if the workload imposed by τ_2 is less then or equal to the workload imposed by τ_1 in all periods, then tuple τ_2 dominates tuple τ_1 as the flight is assigned to the carousel for a shorter time period with no drawback in terms of the objective function and storage capacity. The storage capacity is never violated for a start time tuple τ_2, if $\left(\sum_{j \in \mathcal{F}_i^{\text{parF}}} \Phi^j_{t,\tau_j^{\max}}\right) + \Phi^i_{t,\tau_2} \leq K^{\text{s}}_t$ is satisfied for all $t \in \left[S^{\text{es}}_i, s^{\text{d}}_{\tau_2} + p^{\text{d}}_i(s^{\text{h}}_{\tau_2}) - 1\right]$, where $\tau_j^{\max} = \max\{\tau \in \mathcal{S}_j \mid \tau \succeq \tau' \text{ for all } \tau' \in \mathcal{S}_j\}$ and $p^{\text{d}}_i(s^{\text{h}}_\tau) \geq 0$ is the time to send all stored bags to the assigned carousel if the baggage handling start at s^{h}.

Appendix E: Set-covering heuristic

To build the search tree for the primal heuristic, we use the greedy-based set-covering heuristic consisting of the following three steps:

`selection`($\mathcal{D}^{\text{pos}}$): $\mathcal{D}^{\text{pos}}$ denotes the set of duties which can be selected. The procedure first chooses those duties corresponding to an integer solution, i.e. $z_d = 1$. Then, these duties are selected containing at least one priority flight, i.e. a flight which could not be assigned during the last run of the set-covering heuristic. Last, we select duties containing the maximal number of flights not assigned so far in the partial solution (see Johnson (1974) and Korte and Vygen (2012)).

`storageCapacityCheck`($\mathcal{D}^{\text{sel}}$): After a duty d is added to partial solution $\mathcal{D}^{\text{sel}}$, the heuristic checks violations of the central storage capacity. Let $\mathcal{F}_t^{\text{parT}}$ be the subset of flights handled during period $t \in \mathcal{T}$. If the storage capacity during some period t is violated, flight

$$i^*_d = \arg \max_{i \in \mathcal{F}_t^{\text{parT}}} \left\{ \frac{\Phi^i_{t,\bar{\tau}_i} - \Phi^i_{t,\tau_i}}{P_i \cdot \left(s^{\text{h}}_{\bar{\tau}_i} - s^{\text{h}}_{\tau_i}\right)} \mid \{\tau_i, w_i, P_i\} \text{ is feasible for duty } d \right\}$$

is left-shifted to start time tuple $\tau_i \preceq \bar{\tau}_i$; the number of working stations assigned to flight i is equal to the maximal possible number. We continue left-shifting flights as long as the storage conflict is not solved or there is no flight which can be left-shifted. In the latter case, we delete the flight from a duty with the greatest number of stored bags relative to the consumed parking positions capacity, i.e. we remove flight

$$i^* = \arg \max_{i \in \mathcal{F}_t^{\text{parT}}} \left\{ \frac{\Phi^i_{t,\bar{\tau}_i}}{P_i \cdot \left(S^E_i - s^{\text{h}}_{\bar{\tau}_i}\right)} \right\}.$$

`add`($\mathcal{D}^{\text{sel}}$,$\mathcal{F}^{\text{nA}}$): Flights $\mathcal{F}^{\text{nA}}$, not added at the end of the heuristic, are assigned to 1 of the selected duties $\mathcal{D}^{\text{sel}}$ in decreasing order of the number of required parking positions. Flights not added are collected in a priority set.

Acknowledgments

The authors gratefully thank the associate editor and the two referees for their fruitful comments which helped to improve the paper significantly.

References

Garey, M., D. S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York.

Johnson, D.S. 1974. Approximation algorithms for combinatorial problems. *Journal of Computer and System Science* **9**(3) 156 – 278.

Korte, B., J. Vygen. 2012. *Combinatorial Optimization*, vol. 5. Springer.