A Branch-Price-and-Cut Algorithm for the Joint Order Batching and Picker Routing Problem with Scattered Storage

Column Generation 2023: Montréal

Katrin Heßler and Stefan Irnich

katrin.hessler@dbschenker.com

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

Order Picking

Person-to-goods order picking:

https://www.schoeler-gabelstapler.de/media/Global-Content/03_Solutions_Loesungen/Applications/Order_picker-N20-Series_Moving-Warehouse-4540_8304_16x9w1920.jpg

Overall Idea

- SPRP: Single Picker Routing Problem
  - Dynamic program of Ratliff and Rosenthal (1983)

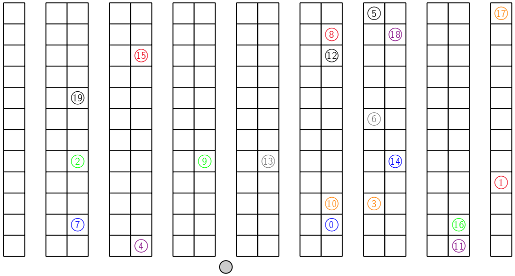
Overall Idea

- SPRP: Single Picker Routing Problem
  - Dynamic program of Ratliff and Rosenthal (1983)
- SPRP-SS: Single Picker Routing Problem with Scattered Storage
  - Extend the state space of the dynamic program of Ratliff and Rosenthal (1983)
  - Add additional variables and constraints for aspects not covered by the extended state space
  - Solve resulting formulation via MIP solver

- SPRP: Single Picker Routing Problem
  - Dynamic program of Ratliff and Rosenthal (1983)
- SPRP-SS: Single Picker Routing Problem with Scattered Storage
  - Extend the state space of the dynamic program of Ratliff and Rosenthal (1983)
  - Add additional variables and constraints for aspects not covered by the extended state space
  - Solve resulting formulation via MIP solver
- JOBPRP-SS: Joint Order Batching and Picker Routing Problem with Scattered Storage
  - Solution of JOBPRP by branch-price-and-cut algorithm, column generation/pricing via MIP solver
  - Pricing problem is Profitable Single Picker Routing Problem with Scattered Storage (PSPRP-SS)

Given: Set P of picking positions in the warehouse
Task: Find a minimum length picking tour that starts and ends at the given I/O point 0 and traverses all positions P (at least once)

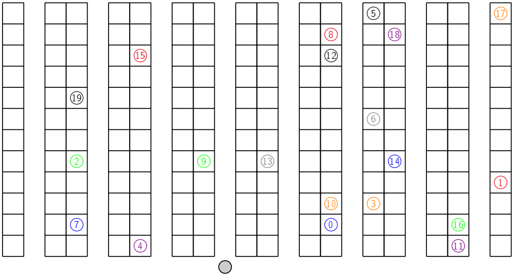
Example: Standard one-block warehouse



Given: Set P of picking positions in the warehouse
Task: Find a minimum length picking tour that starts and ends at the given I/O point 0 and traverses all positions P (at least once)

Can be modeled and solved as a TSP!

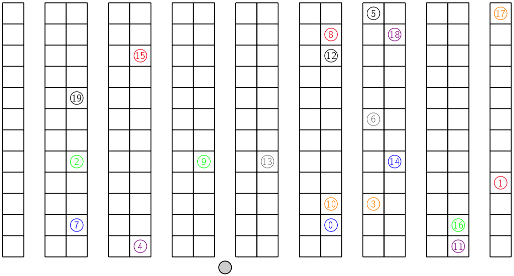
Example: Standard one-block warehouse



Given: Set P of picking positions in the warehouse
Task: Find a minimum length picking tour that starts and ends at the given I/O point 0 and traverses all positions P (at least once)

Can be modeled and solved as a TSP! But there is more structure in it...

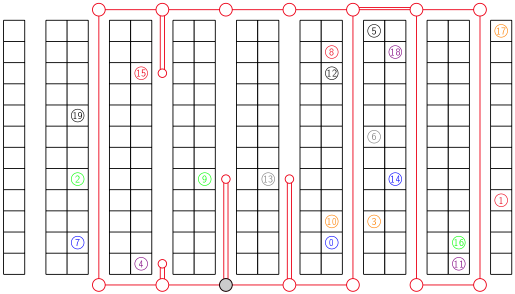
Example: Standard one-block warehouse



Given: Set P of picking positions in the warehouse
Task: Find a minimum length picking tour that starts and ends at the given I/O point 0 and traverses all positions P (at least once)

Can be modeled and solved as a TSP! But there is more structure in it. . .

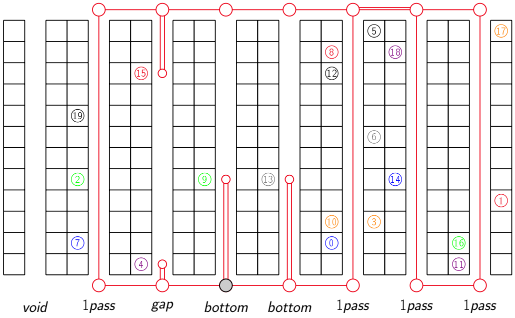
Example: Standard one-block warehouse



Given: Set P of picking positions in the warehouse
Task: Find a minimum length picking tour that starts and ends at the given I/O point 0 and traverses all positions P (at least once)

Can be modeled and solved as a TSP! But there is more structure in it. . .

Example: Standard one-block warehouse



- *aisle traversal*

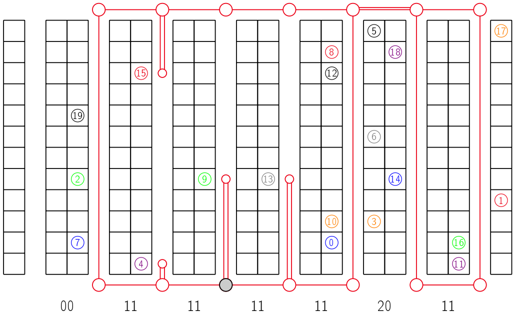
$$E^{aisle} = \{1pass, 2pass, top, bottom, gap, void\},$$

Single Picker Routing Problem

Given: Set P of picking positions in the warehouse
Task: Find a minimum length picking tour that starts and ends at the given I/O point 0 and traverses all positions P (at least once)

Can be modeled and solved as a TSP! But there is more structure in it...

Example: Standard one-block warehouse



- *aisle traversal*

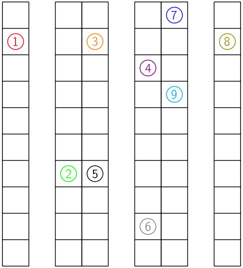
$$E^{aisle} = \{1pass, 2pass, top, bottom, gap, void\},$$

- *cross-aisle traversal*

$$E^{cross} = \{00, 11, 20, 02, 22\}$$

Let $J = \{1, 2, \ldots, m\}$ denotes the *aisles set*.

Idea of the DP:

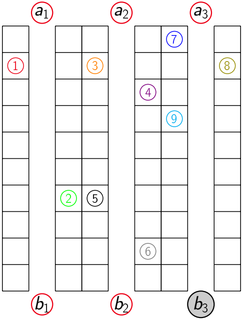


Dynamic Program of Ratliff and Rosenthal (1983)

Let $J = \{1, 2, \ldots, m\}$ denotes the *aisles set*.

Idea of the DP:

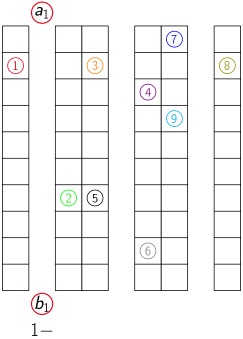
- The DP uses *partial tour subgraphs* (PTSs) with vertices a_j and b_j located at the top and bottom of each aisle $j \in J$, respectively.



Let $J = \{1, 2, \ldots, m\}$ denotes the *aisles set*.

Idea of the DP:

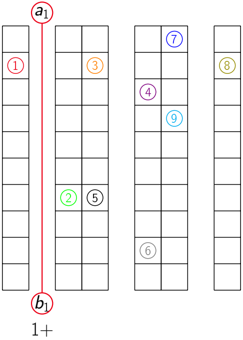
- The DP uses *partial tour subgraphs* (PTSs) with vertices a_j and b_j located at the top and bottom of each aisle $j \in J$, respectively.
- The PTSs comprise those parts of the picking tour that belong to the aisles 1 to j, either before the traversal of aisle j is included (stage j^-) or after its inclusion (stage j^+).



Let $J = \{1, 2, \ldots, m\}$ denotes the *aisles set*.

Idea of the DP:

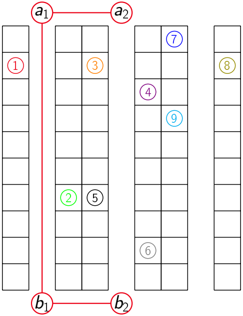
- The DP uses *partial tour subgraphs* (PTSs) with vertices a_j and b_j located at the top and bottom of each aisle $j \in J$, respectively.
- The PTSs comprise those parts of the picking tour that belong to the aisles 1 to j, either before the traversal of aisle j is included (stage j^-) or after its inclusion (stage j^+).



Let $J = \{1, 2, \ldots, m\}$ denotes the *aisles set*.

Idea of the DP:

- The DP uses *partial tour subgraphs* (PTSs) with vertices a_j and b_j located at the top and bottom of each aisle $j \in J$, respectively.
- The PTSs comprise those parts of the picking tour that belong to the aisles 1 to j, either before the traversal of aisle j is included (stage j^-) or after its inclusion (stage j^+).

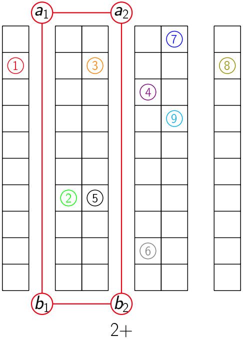


2−

Let $J = \{1, 2, \ldots, m\}$ denotes the *aisles set*.

Idea of the DP:

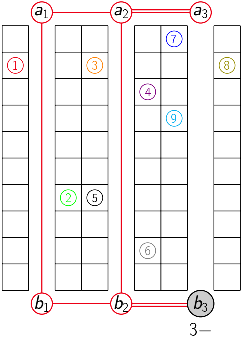
- The DP uses *partial tour subgraphs* (PTSs) with vertices a_j and b_j located at the top and bottom of each aisle $j \in J$, respectively.
- The PTSs comprise those parts of the picking tour that belong to the aisles 1 to j, either before the traversal of aisle j is included (stage j^-) or after its inclusion (stage j^+).



Let $J = \{1, 2, \ldots, m\}$ denotes the *aisles set*.

Idea of the DP:

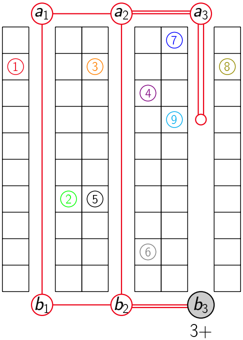
- The DP uses *partial tour subgraphs* (PTSs) with vertices a_j and b_j located at the top and bottom of each aisle $j \in J$, respectively.
- The PTSs comprise those parts of the picking tour that belong to the aisles 1 to j, either before the traversal of aisle j is included (stage j^-) or after its inclusion (stage j^+).



Let $J = \{1, 2, \ldots, m\}$ denotes the *aisles set*.

Idea of the DP:

- The DP uses *partial tour subgraphs* (PTSs) with vertices a_j and b_j located at the top and bottom of each aisle $j \in J$, respectively.
- The PTSs comprise those parts of the picking tour that belong to the aisles 1 to j, either before the traversal of aisle j is included (stage j^-) or after its inclusion (stage j^+).



Dynamic Program of Ratliff and Rosenthal (1983)

Ratliff and Rosenthal (1983) have shown that only *seven states* are possible for optimal picking tours, namely

$$\mathcal{S} = \{\mathtt{UU1c}, \mathtt{0E1c}, \mathtt{E01c}, \mathtt{EE1c}, \mathtt{EE2c}, \mathtt{000c}, \mathtt{001c}\}$$

with

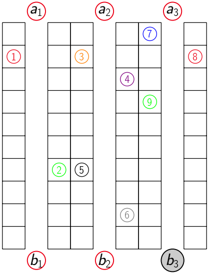
- 0=disconnected, U=odd (=uneven), and E=even degree of a_j and b_j, resp.;
- 0c=empty graph, 1c and 2c=one (two) connected component(s).

Ratliff and Rosenthal (1983) have shown that only *seven states* are possible for optimal picking tours, namely

$$\mathcal{S} = \{\mathtt{UU1c}, \mathtt{0E1c}, \mathtt{E01c}, \mathtt{EE1c}, \mathtt{EE2c}, \mathtt{000c}, \mathtt{001c}\}$$

with

- 0=disconnected, U=odd (=uneven), and E=even degree of a_j and b_j, resp.;
- 0c=empty graph, 1c and 2c=one (two) connected component(s).



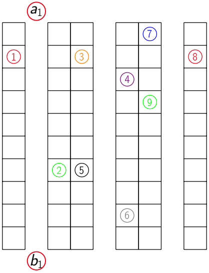
Dynamic Program of Ratliff and Rosenthal (1983)

Ratliff and Rosenthal (1983) have shown that only *seven states* are possible for optimal picking tours, namely

$$\mathcal{S} = \{\mathtt{UU1c}, \mathtt{0E1c}, \mathtt{E01c}, \mathtt{EE1c}, \mathtt{EE2c}, \mathtt{000c}, \mathtt{001c}\}$$

with

- 0=disconnected, U=odd (=uneven), and E=even degree of a_j and b_j, resp.;
- 0c=empty graph, 1c and 2c=one (two) connected component(s).

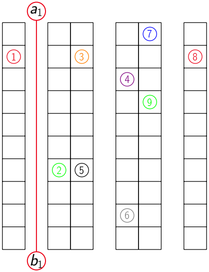


Ratliff and Rosenthal (1983) have shown that only *seven states* are possible for optimal picking tours, namely

$$\mathcal{S} = \{\mathtt{UU1c}, \mathtt{0E1c}, \mathtt{E01c}, \mathtt{EE1c}, \mathtt{EE2c}, \mathtt{000c}, \mathtt{001c}\}$$

with

- `0`=disconnected, `U`=odd (=uneven), and `E`=even degree of a_j and b_j, resp.;
- `0c`=empty graph, `1c` and `2c`=one (two) connected component(s).



`UU1c`

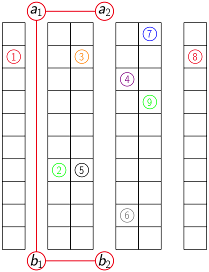
Dynamic Program of Ratliff and Rosenthal (1983)

Ratliff and Rosenthal (1983) have shown that only *seven states* are possible for optimal picking tours, namely

$$\mathcal{S} = \{\texttt{UU1c}, \texttt{0E1c}, \texttt{E01c}, \texttt{EE1c}, \texttt{EE2c}, \texttt{000c}, \texttt{001c}\}$$

with

- 0=disconnected, U=odd (=uneven), and E=even degree of a_j and b_j, resp.;
- 0c=empty graph, 1c and 2c=one (two) connected component(s).



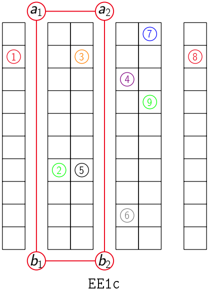
UU1c

Ratliff and Rosenthal (1983) have shown that only *seven states* are possible for optimal picking tours, namely

$$\mathcal{S} = \{\texttt{UU1c}, \texttt{0E1c}, \texttt{E01c}, \texttt{EE1c}, \texttt{EE2c}, \texttt{000c}, \texttt{001c}\}$$

with

- `0`=disconnected, `U`=odd (=uneven), and `E`=even degree of a_j and b_j, resp.;
- `0c`=empty graph, `1c` and `2c`=one (two) connected component(s).

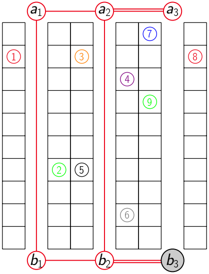


Ratliff and Rosenthal (1983) have shown that only *seven states* are possible for optimal picking tours, namely

$$\mathcal{S} = \{\mathtt{UU1c}, \mathtt{0E1c}, \mathtt{E01c}, \mathtt{EE1c}, \mathtt{EE2c}, \mathtt{000c}, \mathtt{001c}\}$$

with

- 0=disconnected, U=odd (=uneven), and E=even degree of a_j and b_j, resp.;
- 0c=empty graph, 1c and 2c=one (two) connected component(s).



EE1c

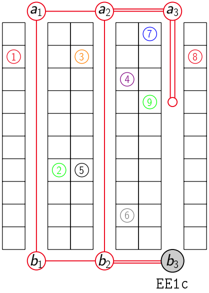
Dynamic Program of Ratliff and Rosenthal (1983)

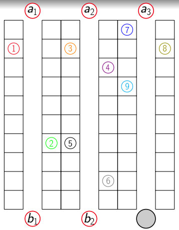
Ratliff and Rosenthal (1983) have shown that only *seven states* are possible for optimal picking tours, namely

$$\mathcal{S} = \{\texttt{UU1c}, \texttt{0E1c}, \texttt{E01c}, \texttt{EE1c}, \texttt{EE2c}, \texttt{000c}, \texttt{001c}\}$$

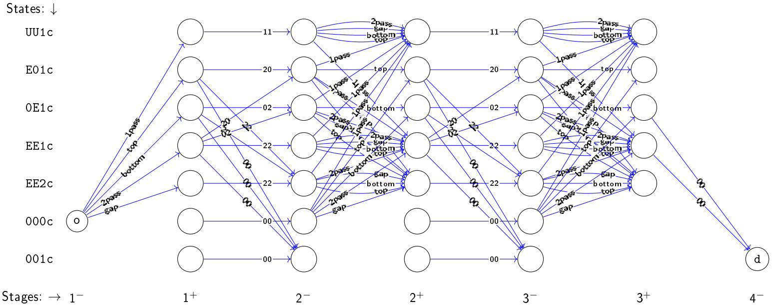
with

- 0=disconnected, U=odd (=uneven), and E=even degree of a_j and b_j, resp.;
- 0c=empty graph, 1c and 2c=one (two) connected component(s).



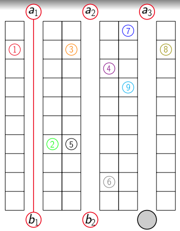


State Space:

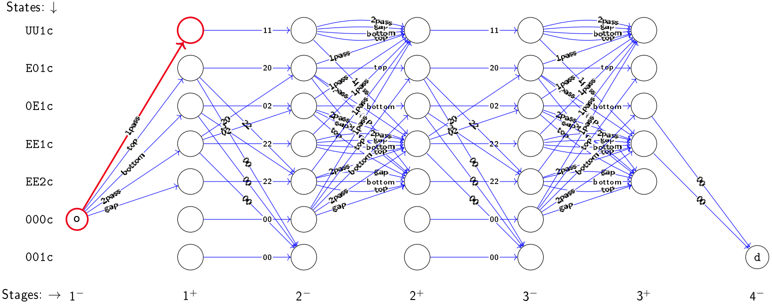


Sequence of states: $(\mathtt{o} = \mathtt{001c}, \mathtt{UU1c}, \mathtt{UU1c}, \mathtt{EE1c}, \mathtt{EE1c}, \mathtt{001c} = \mathtt{d})$

Sequence of transitions: $(1\textit{pass}, 11, 1\textit{pass}, 22, \textit{top}, 00)$

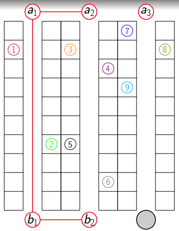


State Space:

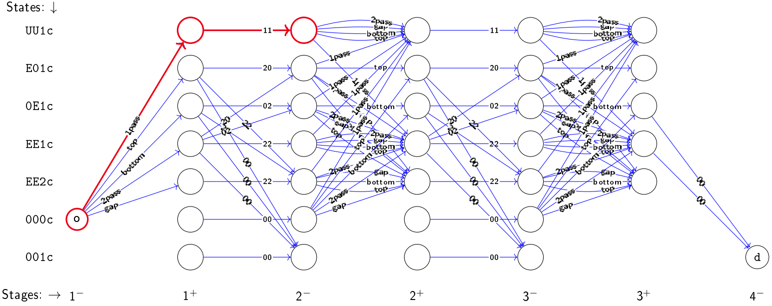


Sequence of states: $(\mathrm{o} = \mathtt{001c}, \mathtt{UU1c}, \mathtt{UU1c}, \mathtt{EE1c}, \mathtt{EE1c}, \mathtt{001c} = \mathrm{d})$

Sequence of transitions: $(1\textit{pass}, 11, 1\textit{pass}, 22, \textit{top}, 00)$

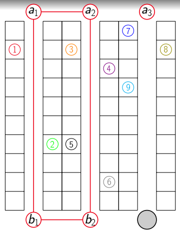


State Space:

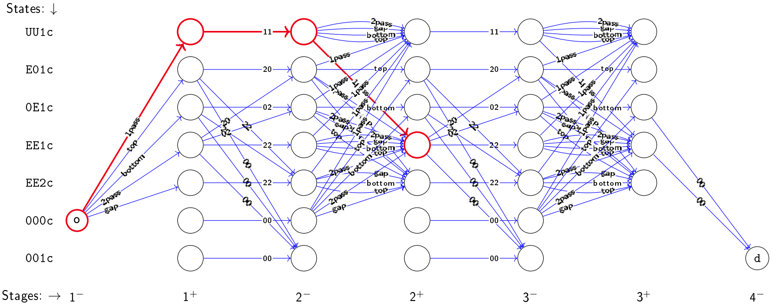


Sequence of states: $(\mathtt{o} = \mathtt{001c}, \mathtt{UU1c}, \mathtt{UU1c}, \mathtt{EE1c}, \mathtt{EE1c}, \mathtt{001c} = \mathtt{d})$

Sequence of transitions: $(1pass, 11, 1pass, 22, top, 00)$

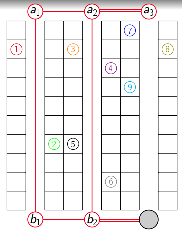


State Space:

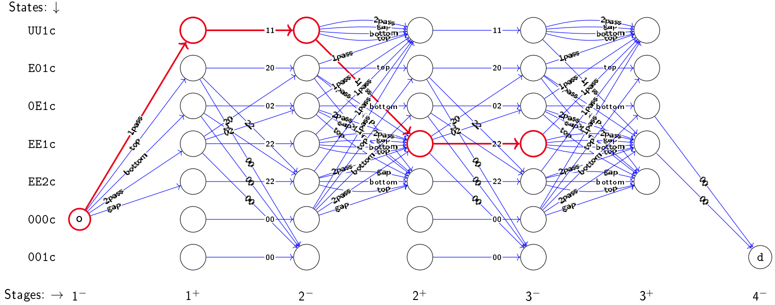


Sequence of states: $(\mathtt{o} = \mathtt{001c}, \mathtt{UU1c}, \mathtt{UU1c}, \mathtt{EE1c}, \mathtt{EE1c}, \mathtt{001c} = \mathtt{d})$

Sequence of transitions: $(1pass, 11, 1pass, 22, top, 00)$

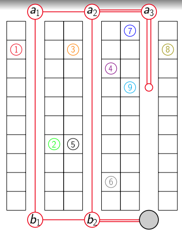


State Space:

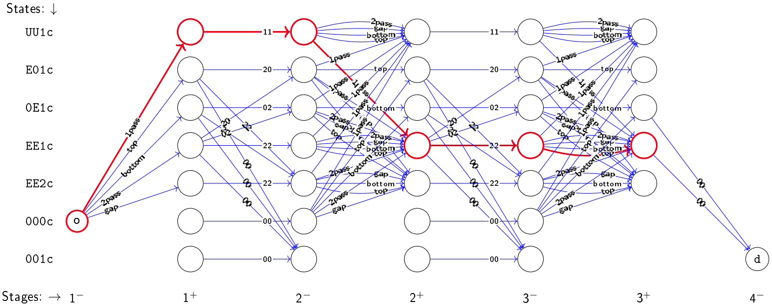


Sequence of states: $(\mathtt{o} = \mathtt{001c}, \mathtt{UU1c}, \mathtt{UU1c}, \mathtt{EE1c}, \mathtt{EE1c}, \mathtt{001c} = \mathtt{d})$

Sequence of transitions: $(1\mathit{pass}, 11, 1\mathit{pass}, 22, \mathit{top}, 00)$



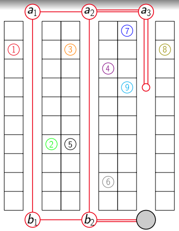
State Space:



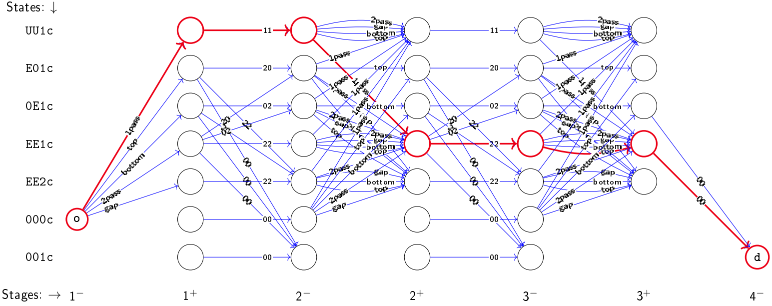
Sequence of states: $(\mathtt{o} = \mathtt{001c}, \mathtt{UU1c}, \mathtt{UU1c}, \mathtt{EE1c}, \mathtt{EE1c}, \mathtt{001c} = \mathtt{d})$

Sequence of transitions: $(1\mathit{pass}, 11, 1\mathit{pass}, 22, \mathit{top}, 00)$

Dynamic Program of Ratliff and Rosenthal (1983)



State Space:



Sequence of states: $(\mathtt{o} = \mathtt{001c}, \mathtt{UU1c}, \mathtt{UU1c}, \mathtt{EE1c}, \mathtt{EE1c}, \mathtt{001c} = \mathtt{d})$

Sequence of transitions: $(1pass, 11, 1pass, 22, top, 00)$

Scattered Storage

When one or several articles are pickable from more than one picking position, the warehouse is operated as a *scattered storage* warehouse a.k.a. *mixed shelves* warehouse (Weidinger and Boysen, 2018).

Scattered Storage

When one or several articles are pickable from more than one picking position, the warehouse is operated as a *scattered storage* warehouse a.k.a. *mixed shelves* warehouse (Weidinger and Boysen, 2018).

- Scattered storage is predominant in modern e-commerce warehouses of companies like Amazon or Zalando (Weidinger, 2018; Boysen *et al.*, 2019; Weidinger *et al.*, 2019).
- Main advantage: "items of demanded articles are found close by irrespective of the position within the warehouse [so that] distance [...] for order picking is reduced" (Weidinger, 2018, p. 140).

Scattered Storage

When one or several articles are pickable from more than one picking position, the warehouse is operated as a *scattered storage* warehouse a.k.a. *mixed shelves* warehouse (Weidinger and Boysen, 2018).

- Scattered storage is predominant in modern e-commerce warehouses of companies like Amazon or Zalando (Weidinger, 2018; Boysen *et al.*, 2019; Weidinger *et al.*, 2019).
- Main advantage: "items of demanded articles are found close by irrespective of the position within the warehouse [so that] distance [. . .] for order picking is reduced" (Weidinger, 2018, p. 140).

Theoretical and computational results:

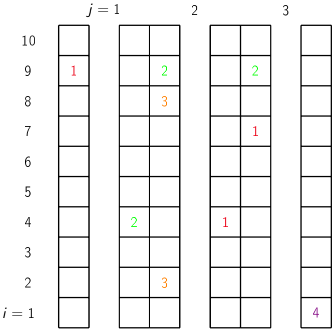
- NP-hard (Weidinger, 2018, Theorem 1)
- Unit-demand case can be modeled and solved as a generalized TSP (GTSP)
- All exact approaches are MIP-based (model solved with MIP solver) (Weidinger, 2018; Weidinger *et al.*, 2019; Goeke and Schneider, 2021)
- Best performing approaches by Goeke and Schneider (2021) (GS-Model) and Heßler and Irnich (2023) (NF-Model)

Scattered Storage

Different stock keeping units (=articles): $S = \{1, 2, 3, 4\}$

Two tasks:

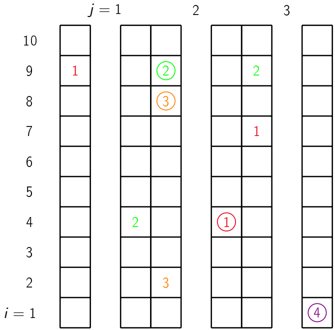
1. Select picking position(s) for each $s \in S$



Different stock keeping units (=articles): $S = \{1, 2, 3, 4\}$

Two tasks:

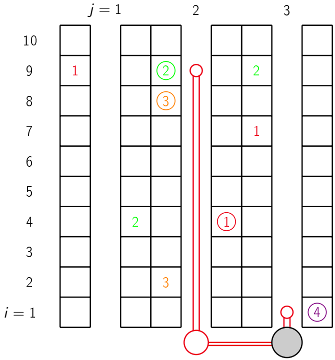
1. Select picking position(s) for each $s \in S$



Different stock keeping units (=articles): $S = \{1, 2, 3, 4\}$

Two tasks:

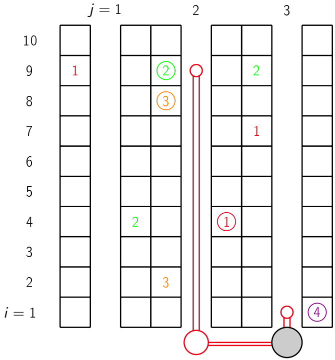
1. Select picking position(s) for each $s \in S$
2. Find minimal length picker route



Different stock keeping units (=articles): $S = \{1, 2, 3, 4\}$

Two tasks:

1. Select picking position(s) for each $s \in S$
2. Find minimal length picker route



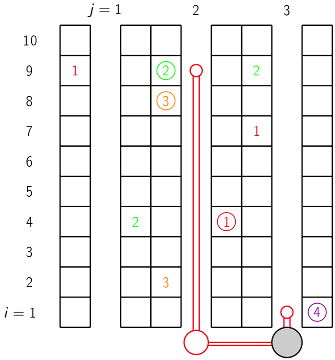
Not directly solvable with dynamic programming! But...

1. Reuse and extend state space

Different stock keeping units (=articles): $S = \{1, 2, 3, 4\}$

Two tasks:

1. Select picking position(s) for each $s \in S$
2. Find minimal length picker route



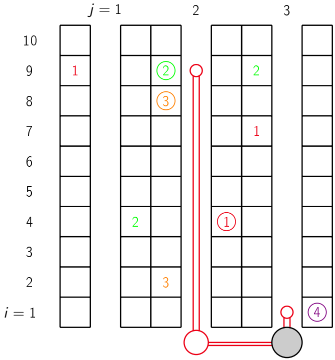
Not directly solvable with dynamic programming! But. . .

1. Reuse and extend state space
2. Formulate an IP model:
   → Shortest path with additional covering conditions

Different stock keeping units (=articles): $S = \{1, 2, 3, 4\}$

Two tasks:

1. Select picking position(s) for each $s \in S$
2. Find minimal length picker route



Not directly solvable with dynamic programming! But...

1. Reuse and extend state space
2. Formulate an IP model:
   → Shortest path with additional covering conditions

Aisle	Type of	additional Transitions
$j = 1$	top(i)	Cell $i = 9$
	bottom(i)	Cell $i = 4$
	void	
$j = 2$	top(i)	Cells $i \in \{4, 8\}$
	bottom(i)	Cells $i \in \{2, 4, 8\}$
	gap(i, k)	Cells $(i, k) \in \{(2, 8), (2, 9), (4, 9)\}$
$j = 3$	bottom(i)	Cell $i \in \{1, 7\}$
	gap(i, k)	Cells $(i, k) = (1, 9)$

New Network-Flow Formulation

Notation:

- Extended state space (V, E)
- Cost c_e of a transition $e \in E$ is length of the associated part of the tour
- Demand d_s for all stock keeping units (SKUs) $s \in S$
- Supply b_{se}, i.e., quantity of SKU s that can be picked with transition $e \in E$

Notation:

- Extended state space (V, E)
- Cost c_e of a transition $e \in E$ is length of the associated part of the tour
- Demand d_s for all stock keeping units (SKUs) $s \in S$
- Supply b_{se}, i.e., quantity of SKU s that can be picked with transition $e \in E$

IP formulation (network flow, NF-Model):

$$\min \sum_{e \in E} c_e x_e \tag{1a}$$

$$\text{subject to} \quad \sum_{e \in \delta^+(\sigma)} x_e - \sum_{e \in \delta^-(\sigma)} x_e = \begin{cases} +1, & \text{if } \sigma = \mathtt{o} \\ -1, & \text{if } \sigma = \mathtt{d} \\ 0, & \text{otherwise} \end{cases} \qquad \forall \sigma \in V \tag{1b}$$

$$\sum_{e \in E} b_{se} x_e \geq d_s \qquad \forall s \in S \tag{1c}$$

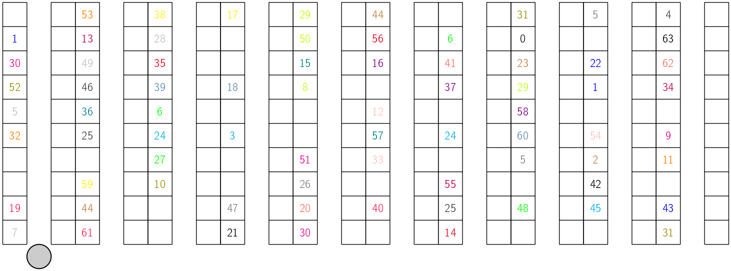
$$x_e \in \{0, 1\} \qquad \forall e \in E \tag{1d}$$

(1a), (1b), and (1d): Shortest path problem
where (1b) can be rewritten as $\mathcal{N}\boldsymbol{x} = \boldsymbol{u}_{\mathtt{o}} - \boldsymbol{u}_{\mathtt{d}}$

(1c): Additional covering constraints

Given: Set O of orders with:

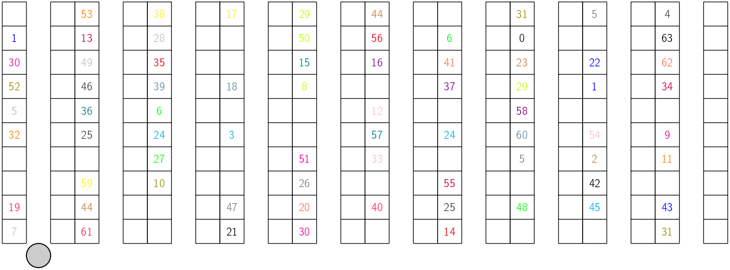
- Subset $S_o \subset S$ of SKUs requested in an order $o \in O$
- Profit $\pi_o > 0$
- Weight $w_o > 0$ (in kg, liter, or the number of compartments)
- Picker capacity Q



Given: Set O of orders with:

- Subset $S_o \subset S$ of SKUs requested in an order $o \in O$
- Profit $\pi_o > 0$
- Weight $w_o > 0$ (in kg, liter, or the number of compartments)
- Picker capacity Q

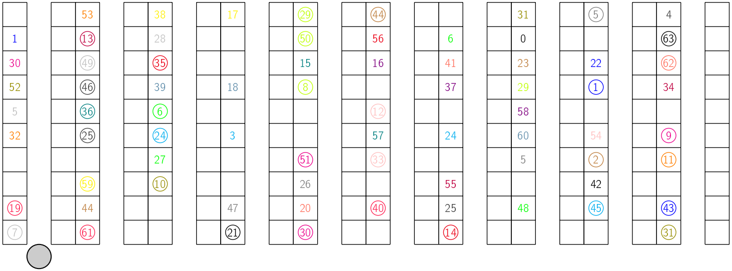
Task: Select a capacity-feasible subset of the orders and find a picking tour that collects the requested SKUs of these orders to minimize the length of the picker tour minus the collected profit.



Given: Set O of orders with:

- Subset $S_o \subset S$ of SKUs requested in an order $o \in O$
- Profit $\pi_o > 0$
- Weight $w_o > 0$ (in kg, liter, or the number of compartments)
- Picker capacity Q

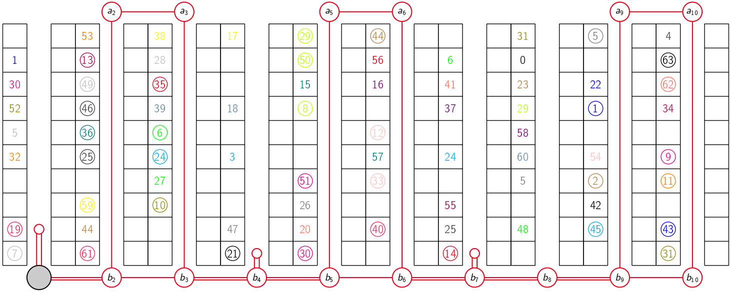
Task: Select a capacity-feasible subset of the orders and find a picking tour that collects the requested SKUs of these orders to minimize the length of the picker tour minus the collected profit.



Given: Set O of orders with:

- Subset $S_o \subset S$ of SKUs requested in an order $o \in O$
- Profit $\pi_o > 0$
- Weight $w_o > 0$ (in kg, liter, or the number of compartments)
- Picker capacity Q

Task: Select a capacity-feasible subset of the orders and find a picking tour that collects the requested SKUs of these orders to minimize the length of the picker tour minus the collected profit.



Additional Variables:

- $z_o \in \{0, 1\}$ selection of order $o \in O$
- $y_s \in \{0, 1\}$ indicator whether SKU $s \in S$ must be collected

Additional Variables:

- $z_o \in \{0,1\}$ selection of order $o \in O$
- $y_s \in \{0,1\}$ indicator whether SKU $s \in S$ must be collected

$$c(\pi) = \min \sum_{e \in E} c_e x_e - \sum_{o \in O} \pi_o z_o \tag{2a}$$

$$\text{subject to} \quad \mathcal{N} \boldsymbol{x} = \boldsymbol{u}_{\mathrm{o}} - \boldsymbol{u}_{\mathrm{d}} \tag{2b}$$

$$\sum_{e \in E_s} x_e \geq y_s \qquad \forall s \in S \tag{2c}$$

$$y_s \geq z_o \qquad \forall o \in O, \forall s \in S_o \tag{2d}$$

$$\sum_{o \in O} w_o z_o \leq Q \tag{2e}$$

$$x_e \in \{0,1\} \qquad \forall e \in E \tag{2f}$$

$$y_s \in \{0,1\} \qquad \forall s \in S \tag{2g}$$

$$z_o \in \{0,1\} \qquad \forall o \in O \tag{2h}$$

Given: Set O of orders with:

- Subset $S_o \subset S$ of SKUs requested in an order $o \in O$
- Weight $w_o > 0$ (in kg, liter, or the number of compartments)
- Picker capacity Q

Given: Set O of orders with:

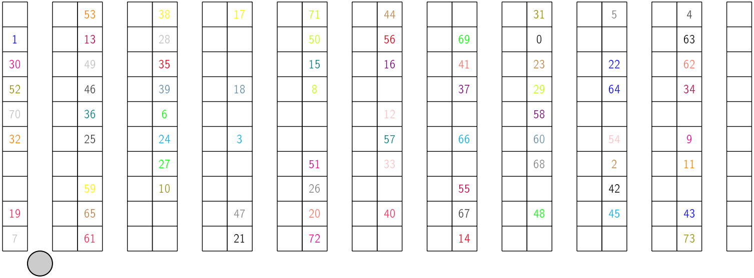
- Subset $S_o \subset S$ of SKUs requested in an order $o \in O$
- Weight $w_o > 0$ (in kg, liter, or the number of compartments)
- Picker capacity Q

Task: Group/partition the orders into capacity-feasible batches and find for each batch a picking tour that collects the requested SKUs of the respective batch so that the total length of all picker tours is minimized.

Given: Set O of orders with:

- Subset $S_o \subset S$ of SKUs requested in an order $o \in O$
- Weight $w_o > 0$ (in kg, liter, or the number of compartments)
- Picker capacity Q

Task: Group/partition the orders into capacity-feasible batches and find for each batch a picking tour that collects the requested SKUs of the respective batch so that the total length of all picker tours is minimized.

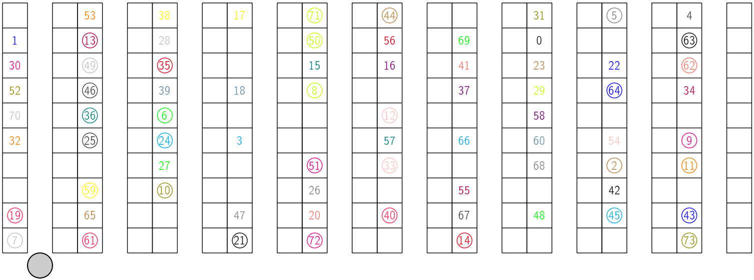


Given: Set O of orders with:

- Subset $S_o \subset S$ of SKUs requested in an order $o \in O$
- Weight $w_o > 0$ (in kg, liter, or the number of compartments)
- Picker capacity Q

Task: Group/partition the orders into capacity-feasible batches and find for each batch a picking tour that collects the requested SKUs of the respective batch so that the total length of all picker tours is minimized.

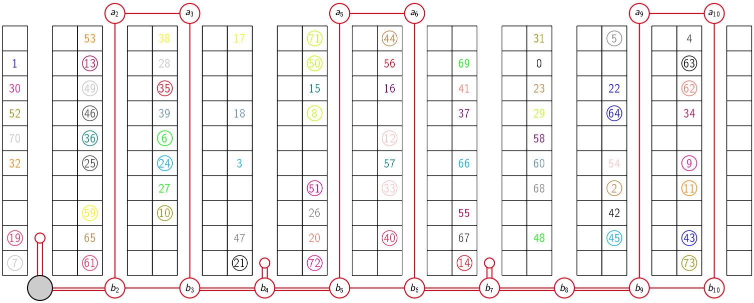
Tour 1:



Given: Set O of orders with:

- Subset $S_o \subset S$ of SKUs requested in an order $o \in O$
- Weight $w_o > 0$ (in kg, liter, or the number of compartments)
- Picker capacity Q

Task: Group/partition the orders into capacity-feasible batches and find for each batch a picking tour that collects the requested SKUs of the respective batch so that the total length of all picker tours is minimized.

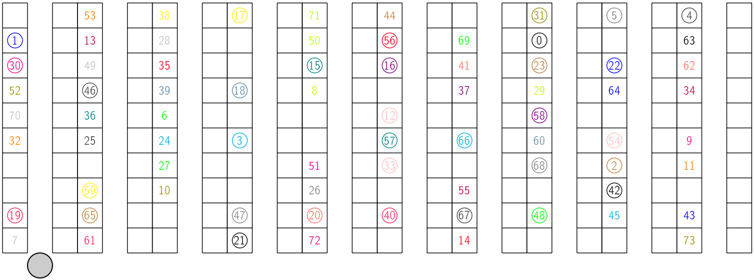


Given: Set O of orders with:

- Subset $S_o \subset S$ of SKUs requested in an order $o \in O$
- Weight $w_o > 0$ (in kg, liter, or the number of compartments)
- Picker capacity Q

Task: Group/partition the orders into capacity-feasible batches and find for each batch a picking tour that collects the requested SKUs of the respective batch so that the total length of all picker tours is minimized.

Tour 2:

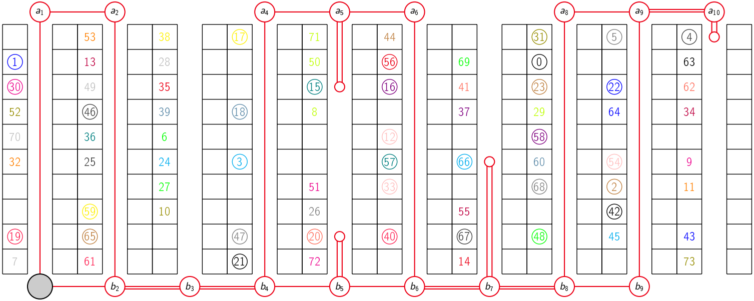


Given: Set O of orders with:

- Subset $S_o \subset S$ of SKUs requested in an order $o \in O$
- Weight $w_o > 0$ (in kg, liter, or the number of compartments)
- Picker capacity Q

Task: Group/partition the orders into capacity-feasible batches and find for each batch a picking tour that collects the requested SKUs of the respective batch so that the total length of all picker tours is minimized.

Tour 2:



JOBPRP (without SS)

- Two-level problem
- Can be modeled and solved as Soft-Clustered VRP
- Recent BPC approach of Wahlen and Gschwind (2023) is state-of-the-art
  - Pricing problem modeled as SPPRC and solved by a labeling algorithm that relies on strong completion bounds

JOBPRP (without SS)

- Two-level problem
- Can be modeled and solved as Soft-Clustered VRP
- Recent BPC approach of Wahlen and Gschwind (2023) is state-of-the-art
  - Pricing problem modeled as SPPRC and solved by a labeling algorithm that relies on strong completion bounds

JOBPRP-SS

- Three-level problem
- Is a 'combination' of the Soft-Clustered VRP and Generalized VRP
- To the best of our knowledge not tackled in the literature yet
- Only known solution for the pricing problem is MIP-based

Pure binary model:

$$\min \sum_{b \in B} \sum_{e \in E} c_e x_e^b \tag{3a}$$

$$\text{subject to} \quad \sum_{b \in B} z_o^b = 1 \qquad \forall o \in O \tag{3b}$$

$$\mathcal{N} \boldsymbol{x}^b = \boldsymbol{u}_{\mathrm{o}} - \boldsymbol{u}_{\mathrm{d}} \qquad \forall b \in B \tag{3c}$$

$$\sum_{e \in E_s} x_e^b \geq y_s^b \qquad \forall b \in B, \forall s \in S \tag{3d}$$

$$y_s^b \geq z_o^b \qquad \forall b \in B, \forall o \in O, \forall s \in S_o \tag{3e}$$

$$\sum_{o \in O} w_o z_o^b \leq Q \qquad \forall b \in B \tag{3f}$$

$$x_e^b \in \{0,1\} \qquad \forall b \in B, \forall e \in E \tag{3g}$$

$$y_s^b \in \{0,1\} \qquad \forall b \in B, \forall s \in S \tag{3h}$$

$$z_o^b \in \{0,1\} \qquad \forall b \in B, \forall o \in O \tag{3i}$$

Remark: Constr. (3c)–(3i) are $|B|$-times those of the profitable SPRP-SS.

Dantzig-Wolfe decomposition according to the order partitioning conditions (3b) and subsequent aggregation leads to a *b*-index-free formulation, which has the advantage of eliminating the inherent symmetry. Let

$$(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}, \bar{\boldsymbol{z}}) \in \{0,1\}^{|E|+|S|+|O|}$$

be an extreme point of a block. Since all variables are binary, the set of these extreme points is

$$\mathcal{W} = \{(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}, \bar{\boldsymbol{z}}) \in \{0,1\}^{|E|+|S|+|O|} : \text{fulfills (3c)–(3f)}\}.$$

Dantzig-Wolfe decomposition according to the order partitioning conditions (3b) and subsequent aggregation leads to a *b*-index-free formulation, which has the advantage of eliminating the inherent symmetry. Let

$$(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}, \bar{\boldsymbol{z}}) \in \{0,1\}^{|E|+|S|+|O|}$$

be an extreme point of a block. Since all variables are binary, the set of these extreme points is

$$\mathcal{W} = \{(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}, \bar{\boldsymbol{z}}) \in \{0,1\}^{|E|+|S|+|O|} : \text{fulfills (3c)–(3f)}\}.$$

Extensive (=set partitioning, batch-based) formulation:

$$\min \sum_{w=(\bar{x},\bar{y},\bar{z})\in\mathcal{W}} (\boldsymbol{c}^T \bar{\boldsymbol{x}})\lambda_w \tag{4a}$$

$$\text{subject to} \sum_{w=(\bar{x},\bar{y},\bar{z})\in\mathcal{W}} (\bar{z}_o)\lambda_w = 1 \quad \text{dual: } [\pi_o] \quad \forall o \in O \tag{4b}$$

$$\lambda_w \in \{0,1\} \quad \forall w \in \mathcal{W} \tag{4c}$$

BPC Algorithm for JOBPRP-SS

Components of the BPC algorithm:

- Column Generation:
  - Pricing problem is the PSPRP-SS solved by a MIP solver
  - Partial pricing hierarchy: (1) Hash table, (2) VND-based heuristic, and (3) MIP solver on reduced extended state space

BPC Algorithm for JOBPRP-SS

Components of the BPC algorithm:

- Column Generation:
  - Pricing problem is the PSPRP-SS solved by a MIP solver
  - Partial pricing hierarchy: (1) Hash table, (2) VND-based heuristic, and (3) MIP solver on reduced extended state space
- Branching:
  1. Number of batches (a priori computation of $\underline{b}$)
  2. Ryan/Foster ($z_{o_1} = z_{o_2}$ or $z_{o_1} + z_{o_2} \leq 1$; prioritize branching on large orders)

BPC Algorithm for JOBPRP-SS

Components of the BPC algorithm:

- Column Generation:
  - Pricing problem is the PSPRP-SS solved by a MIP solver
  - Partial pricing hierarchy: (1) Hash table, (2) VND-based heuristic, and (3) MIP solver on reduced extended state space
- Branching:
  1. Number of batches (a priori computation of $\underline{b}$)
  2. Ryan/Foster ($z_{o_1} = z_{o_2}$ or $z_{o_1} + z_{o_2} \leq 1$; prioritize branching on large orders)
- MIP Solver Heuristic: Solve RMP as an integer program with the MIP solver in a limited number of branch-and-bound nodes

BPC Algorithm for JOBPRP-SS

Components of the BPC algorithm:

- Column Generation:
  - Pricing problem is the PSPRP-SS solved by a MIP solver
  - Partial pricing hierarchy: (1) Hash table, (2) VND-based heuristic, and (3) MIP solver on reduced extended state space
- Branching:
  1. Number of batches (a priori computation of $\underline{b}$)
  2. Ryan/Foster ($z_{o_1} = z_{o_2}$ or $z_{o_1} + z_{o_2} \leq 1$; prioritize branching on large orders)
- MIP Solver Heuristic: Solve RMP as an integer program with the MIP solver in a limited number of branch-and-bound nodes
- Cutting: Subset-row inequalities (Jepsen *et al.*, 2008) for subsets $|R| = 3$ and 4, capacity cuts (Baldacci *et al.*, 2008)

New Benchmark Set for JOBPRP-SS

- Picker capacity Q: 20, 50
- Number of orders $|O|$: 10, 20, 50
- Order size s: uniformly distributed on $[3, 7]$, $[10, 20]$
- Class-based storage policies

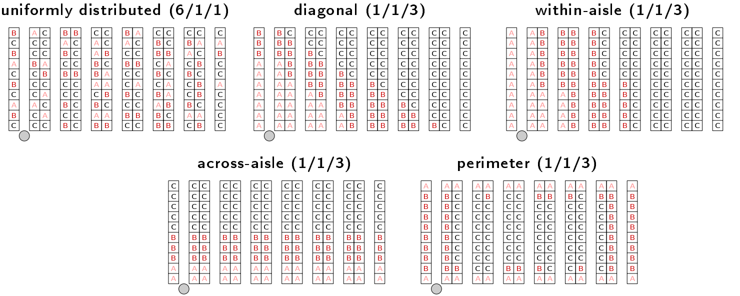
class A:	20% of articles	$\rightarrow$	80% of sales
class B:	30% of articles	$\rightarrow$	15% of sales
class C:	50% of articles	$\rightarrow$	5% of sales

New Benchmark Set for JOBPRP-SS

- Picker capacity Q: 20, 50
- Number of orders $|O|$: 10, 20, 50
- Order size s: uniformly distributed on $[3,7]$, $[10,20]$
- Class-based storage policies

class A:	20% of articles	$\rightarrow$	80% of sales
class B:	30% of articles	$\rightarrow$	15% of sales
class C:	50% of articles	$\rightarrow$	5% of sales

- Scatter factor $\alpha = 2$, scattering of (A/B/C) dependent on storage policy (Korbacher *et al.*, 2022)



Preliminary Results for JOBPRP-SS

				across-aisle		diagonal		perimeter		uniform		within-aisle	
Q	$\bar{s}$	$\|O\|$	#inst	#opt	time $\bar{t}$	#opt	time $\bar{t}$	#opt	time $\bar{t}$	#opt	time $\bar{t}$	#opt	time $\bar{t}$
20	5	10	10	10	**4.2**	10	5.2	10	33.3	10	59.0	10	17.2
		20	10	10	268.2	10	**139.1**	5	1903.7	7	1385.9	8	1281.9
		50	10	2	**3039.0**	0	*TL*	0	*TL*	0	*TL*	0	*TL*
	15	10	10	10	1.4	10	1.4	10	1.7	10	29.5	10	**1.1**
		20	10	10	5.1	10	5.2	10	5.0	10	86.0	10	**4.6**
		50	10	10	**77.8**	10	93.5	10	85.4	10	207.4	10	80.6
50	5	10	10	10	8.7	10	7.4	10	6.3	10	115.0	10	**4.9**
		20	10	6	2222.3	7	**1643.2**	5	2439.6	2	3297.9	4	2315.6
		50	10	0	*TL*	0	*TL*	0	*TL*	0	*TL*	0	*TL*
	15	10	10	10	6.7	10	7.3	10	**5.3**	10	132.9	10	13.9
		20	10	10	**91.5**	9	701.0	6	1508.2	9	1676.5	8	874.4
		50	10	1	3548.8	1	**3359.8**	1	3550.5	0	*TL*	0	*TL*
Total			120	**89**		87		77		78		80	
Average					**1052.0**		1096.9		1394.9		1482.5		1282.8

- Across-aisle and diagonal are easiest to solve
- Instances with many orders $|O|$ and many orders per tour $Q/\bar{s}$ are difficult to solve

Average Costs for JOBPRP-SS

Q	$\bar{s}$	$\|O\|$	within-aisle	diagonal	across-aisle	perimeter	uniform
20	5	10	**332.4**	359.0	378.6	440.4	447.6
		20	**521.8**	596.8	643.6	751.2	780.3
	15	10	**934.0**	1142.0	1242.4	1466.0	1648.2
		20	**1828.8**	2397.8	2508.4	2830.4	3492.8
50	5	10	**221.6**	241.0	247.4	235.8	303.8
		20	**331.0**	370.0	386.7	382.0	502.0
	15	10	**471.0**	544.6	603.0	602.6	822.4
		20	**769.3**	1037.8	1117.4	1090.3	1576.0
Average			**1325.8**	1664.9	1736.9	1880.8	2628.2

- Within-aisle has on average lowest cost
- Uniformly distributed has on average highest cost

Conclusions and Outlook for JOBPRP-SS

BPC algorithm for JOBPRP-SS:

- To the best of our knowledge first solution approach
- Instances of medium size can be solved to proven optimality
- Cost comparison between different storage policies

Conclusions and Outlook for JOBPRP-SS

BPC algorithm for JOBPRP-SS:

- To the best of our knowledge first solution approach
- Instances of medium size can be solved to proven optimality
- Cost comparison between different storage policies

Outlook:

- Refinement of the BPC (strong branching, heuristic pricing, number of SRIs/CCs, etc.)
- State-space and extended state-space can be modified to restrict solution to routing policies `traversal`, `midpoint`, `return`, `largest gap`, `composite` (Korbacher *et al.*, 2022)

Thank you for listening!

Questions?!

Contact:
Katrin Heßler
Operations Research Specialist

katrin.hessler@dbschenker.com

Baldacci, R., Christofides, N., and Mingozzi, A. (2008). An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming*, **115**(2), 351–385.

Boysen, N., de Koster, R., and Weidinger, F. (2019). Warehousing in the e-commerce era: A survey. *European Journal of Operational Research*, **277**(2), 396–411.

Goeke, D. and Schneider, M. (2021). Modeling single-picker routing problems in classical and modern warehouses. *INFORMS Journal on Computing*, **33**(2), 436–451.

Heßler, K. and Irnich, S. (2023). Exact solution of the single picker routing problem with scattered storage. Technical Report LM-2023-02, Chair of Logistics Management, Gutenberg School of Management and Economics, Johannes Gutenberg University Mainz, Mainz, Germany.

Hintsch, T., Irnich, S., and Kiilerich, L. (2021). Branch-price-and-cut for the soft-clustered capacitated arc-routing problem. *Transportation Science*, **55**(3), 687–705.

Jepsen, M., Petersen, B., Spoorendonk, S., and Pisinger, D. (2008). Subset-row inequalities applied to the vehicle-routing problem with time windows. *Operations Research*, **56**(2), 497–511.

Korbacher, L., Heßler, K., and Irnich, S. (2022). An evaluation of several heuristic routing policies for the single picker routing problem with scattered storage. Technical Report LM-2022-0x, Chair of Logistics Management, Gutenberg School of Management and Economics, Johannes Gutenberg University Mainz, Mainz, Germany. In preparation.

Ratliff, H. D. and Rosenthal, A. S. (1983). Order-picking in a rectangular warehouse: A solvable case of the traveling salesman problem. *Operations Research*, **31**(3), 507–521.

Wahlen, J. and Gschwind, T. (2023). Branch-price-and-cut-based solution of order batching problems. *Transportation Science*.

Weidinger, F. (2018). Picker routing in rectangular mixed shelves warehouses. *Computers & Operations Research*, **95**, 139–150.

Weidinger, F. and Boysen, N. (2018). Scattered storage: How to distribute stock keeping units all around a mixed-shelves warehouse. *Transportation Science*, **52**(6), 1412–1427.

Weidinger, F., Boysen, N., and Schneider, M. (2019). Picker routing in the mixed-shelves warehouses of e-commerce retailers. *European Journal of Operational Research*, **274**(2), 501–515.

Subset-Row Inequalities in MIP-based Pricing

Master problem: A SRI is defined by a subset $R = \{o_1, o_2, \ldots, o_q\} \subseteq O$ of $q \geq 3$ different rows and weights $\boldsymbol{u} = (u_1, u_2, \ldots, u_q)$ as

$$\sum_{w=(\bar{x},\bar{y},\bar{z})\in\mathcal{W}} \left\lfloor \sum_{j=1}^{q} \bar{z}_{o_j} u_{o_j} \right\rfloor \lambda_w \leq \left\lfloor \sum_{j=1}^{q} u_j \right\rfloor . \qquad \text{dual:} \quad [\tau_{(R,u)}]$$

Subset-Row Inequalities in MIP-based Pricing

Master problem: A SRI is defined by a subset $R = \{o_1, o_2, \ldots, o_q\} \subseteq O$ of $q \geq 3$ different rows and weights $\boldsymbol{u} = (u_1, u_2, \ldots, u_q)$ as

$$\sum_{w=(\bar{x},\bar{y},\bar{z})\in\mathcal{W}} \left\lfloor \sum_{j=1}^{q} \bar{z}_{o_j} u_{o_j} \right\rfloor \lambda_w \leq \left\lfloor \sum_{j=1}^{q} u_j \right\rfloor . \qquad \text{dual:} \quad [\tau_{(R,u)}]$$

Reduced cost of a variable λ_w for $w = (\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}, \bar{\boldsymbol{z}}) \in \mathcal{W}$ is:

$$\begin{aligned}\tilde{c}_w(\boldsymbol{\pi}, \tau) = \sum_{e\in E} c_e \bar{x}_e - \sum_{o\in O} \bar{z}_o \pi_o \\ - \sum_{\substack{(R=\{o_1,o_2,\ldots,o_q\},\\ u=(u_1,u_2,\ldots,u_q))}} \left\lfloor \sum_{j=1}^{q} \bar{z}_{o_j} u_{o_j} \right\rfloor \tau_{(R,u)}\end{aligned}$$

Subset-Row Inequalities in MIP-based Pricing

Master problem: A SRI is defined by a subset $R = \{o_1, o_2, \dots, o_q\} \subseteq O$ of $q \geq 3$ different rows and weights $\boldsymbol{u} = (u_1, u_2, \dots, u_q)$ as

$$\sum_{w=(\bar{x},\bar{y},\bar{z})\in\mathcal{W}} \left\lfloor \sum_{j=1}^{q} \bar{z}_{o_j} u_{o_j} \right\rfloor \lambda_w \leq \left\lfloor \sum_{j=1}^{q} u_j \right\rfloor . \qquad \text{dual:} \quad [\tau_{(R,u)}]$$

Reduced cost of a variable λ_w for $w = (\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}, \bar{\boldsymbol{z}}) \in \mathcal{W}$ is:

$$\begin{aligned}\tilde{c}_w(\boldsymbol{\pi}, \tau) = \sum_{e\in E} c_e \bar{x}_e - \sum_{o\in O} \bar{z}_o \pi_o \\ - \sum_{\substack{(R=\{o_1,o_2,\dots,o_q\},\\ u=(u_1,u_2,\dots,u_q))}} \left\lfloor \sum_{j=1}^{q} \bar{z}_{o_j} u_{o_j} \right\rfloor \tau_{(R,u)}\end{aligned}$$

Pricing problem: For each active SRI defined by (R, u), a non-negative integer variable $t_{R,u}$ must be introduced. It models the coefficient of $\tau_{(R,u)}$ in the last sum.

For $R = \{o_1, o_2, o_3\}$ and the unique undominated weights $(u_1, u_2, u_3) = (1/2, 1/2, 1/2)$, the coupling between the z- and the t-variable can be accomplished via

$$z_{o_1} + z_{o_2} + z_{o_3} - 2t_{R,u} \leq 1 \qquad \text{or} \qquad \begin{array}{rrrrl} z_{o_1} & + z_{o_2} & & - & t_{R,u} \leq 1 \\ z_{o_1} & & + z_{o_3} & - & t_{R,u} \leq 1 \\ & + z_{o_2} & + z_{o_3} & - & t_{R,u} \leq 1 \end{array}$$

For $R = \{o_1, o_2, o_3\}$ and the unique undominated weights $(u_1, u_2, u_3) = (1/2, 1/2, 1/2)$, the coupling between the z- and the t-variable can be accomplished via

$$z_{o_1} + z_{o_2} + z_{o_3} - 2t_{R,u} \leq 1 \quad \text{or} \quad \begin{array}{rrrrl} z_{o_1} & + z_{o_2} & & - & t_{R,u} \leq 1 \\ z_{o_1} & & + z_{o_3} & - & t_{R,u} \leq 1 \\ & + z_{o_2} & + z_{o_3} & - & t_{R,u} \leq 1 \end{array}$$

For $R = \{o_1, o_2, o_3, o_4\}$ and the unique undominated weights $(u_1, u_2, u_3, u_4) = (2/3, 1/3, 1/3, 1/3)$,

$$2z_{o_1} + z_{o_2} + z_{o_3} + z_{o_4} - 3t_{R,u} \leq 2 \quad \text{or} \quad \begin{array}{rrrrrl} z_{o_1} & + z_{o_2} & & & - & t_{R,u} \leq 1 \\ z_{o_1} & & z_{o_3} & & - & t_{R,u} \leq 1 \\ z_{o_1} & & & z_{o_4} & - & t_{R,u} \leq 1 \\ & z_{o_2} & + z_{o_3} & + z_{o_4} & - & t_{R,u} \leq 1 \end{array}$$

For $R = \{o_1, o_2, o_3\}$ and the unique undominated weights $(u_1, u_2, u_3) = (1/2, 1/2, 1/2)$, the coupling between the z- and the t-variable can be accomplished via

$$z_{o_1} + z_{o_2} + z_{o_3} - 2t_{R,u} \leq 1 \quad \text{or} \quad \begin{array}{rrrrl} z_{o_1} & + z_{o_2} & & - & t_{R,u} \leq 1 \\ z_{o_1} & & + z_{o_3} & - & t_{R,u} \leq 1 \\ & + z_{o_2} & + z_{o_3} & - & t_{R,u} \leq 1 \end{array}$$

For $R = \{o_1, o_2, o_3, o_4\}$ and the unique undominated weights $(u_1, u_2, u_3, u_4) = (2/3, 1/3, 1/3, 1/3)$,

$$2z_{o_1} + z_{o_2} + z_{o_3} + z_{o_4} - 3t_{R,u} \leq 2 \quad \text{or} \quad \begin{array}{rrrrrl} z_{o_1} & + z_{o_2} & & & - & t_{R,u} \leq 1 \\ z_{o_1} & & z_{o_3} & & - & t_{R,u} \leq 1 \\ z_{o_1} & & & z_{o_4} & - & t_{R,u} \leq 1 \\ & z_{o_2} & + z_{o_3} & + z_{o_4} & - & t_{R,u} \leq 1 \end{array}$$

Neither of the two formulations is dominating the other. Use both formulations together (Hintsch *et al.*, 2021).