Order picking in a parallel-aisle warehouse with turn penalties
(Online Supplement)

M. Çelik* and H. Süral

Department of Industrial Engineering, Middle East Technical University, Ankara, Turkey

(Received 00 Month 20XX; final version received 00 Month 20XX)

The online supplement is organized as follows: In three subsequent sections, we give results on single objective turn minimization problem as well as the travel time minimization problem with turn penalties, biobjective travel time and turn minimization, and the triobjective problem with U-turn minimization, respectively. Each section includes proofs of relevant statements and extensions of the results to the cross-depot and pick-depot cases.

Supplement 1 for Section 3: Single-Objective Turn Minimization and Time Minimization Problems

A. Proofs of statements

We present a more generalized version of Proposition 3.1, considering all possible P&D point positions.

Proposition 3.1. Connection type (5) in Figure 3 in the main text is not needed for solving the 2-OPP. Furthermore, it is not needed for solving the turn minimization problem, except possibly when the P&D point is on the corner of a cross aisle and a pick aisle.

Proof. Since the case with $n = 1$ is trivial, assume $n \geq 2$. In a given order picking tour, suppose there exists a connection of type (5) in aisle $j$. By enumerating the possible connection types in aisle $j + 1$ (in the case where $j = n$, aisle $n - 1$) and the possible connection types between these aisles, an appropriate transformation of the tour depending on the connection types in aisles $j$ and $j + 1$ and the inter-aisle connection type between the two aisles results in a total length and number of turns no more than those of the original one. To maintain feasibility and leave the rest of the tour unaffected in terms of the total travel time and number of turns, degree parities of the corner nodes and connectivity of the tour should be preserved. Figures 1 and 2 show the subgraphs induced by aisles $j$ and $j + 1$ for varying connection types in aisle $j + 1$ when inter-aisle connection is of types (i) and (ii), respectively. For inter-aisle connection types (iii) and (iv), the transformation is almost identical to that applied on the subgraphs on Figure 2. The graphs for the subtours after transformation are given in Figure 3 and details of how the subgraphs in Figures 1 and 2 are transformed to the ones in Figure 3 are given below. As opposed to the case in Figures 1 and 2, where the items on the aisles in the figures represent the two consecutive items with maximum pairwise travel time in the aisle, the items in Figure 3 are the ones closest to the back and front cross aisles, respectively.

*Corresponding author. Email: cmelih@metu.edu.tr
Figure 1. Possible connection types in aisle $j+1$ when inter-aisle connection is of type (i) and the resulting subgraphs induced by aisles $j$ and $j+1$

Figure 2. Possible connection types in aisle $j+1$ when inter-aisle connection is of type (ii) and the resulting subgraphs induced by aisles $j$ and $j+1$

The first subgraphs in Figures 1 and 2 incur six turns (two to enter and two to exit aisle $j$, one to enter and one to exit $j+1$). Similarly, the second subgraphs result in eight turns (two to enter and two to exit aisle $j$, one to enter and one to exit $j+1$, and two for the U-turn in $j+1$), the third subgraphs incur 12 turns (two to enter and two to exit each of aisles $j$ and $j+1$, four for the two U-turns in $j+1$), and the last subgraphs incur eight turns (two to enter and two to exit each of aisles $j$ and $j+1$).

After the transformation of the first subgraph in Figure 1 to the one in Figure 3(a) by using a type (2) connection in aisle $j$ instead of type (5), the number of turns incurred by the subgraph remains at six (one to enter and one to exit each of aisles $j$ and $j+1$, two for the U-turn in aisle $j$), whereas transforming the remaining subgraphs in Figure 1 to the one in Figure 3(b) by using a type (1) connection in both of aisles $j$ and $j+1$ instead of type (5) in $j$ and (2), (3), or (4) in $j+1$ and a type (ii) connection between the aisles instead of type (i) results in a reduction of turns in the subgraph to four (one to enter and one to exit each of aisles $j$ and $j+1$). The transformation of the first subgraph in Figure 2 to the one in Figure 3(c) by using type (1) and (2) connections in aisles $j$ and $j+1$ respectively and an inter-aisle connection of type (i) instead of (ii) keeps the number of turns in the subgraph at six (one to enter and one to exit each of aisles $j$ and $j+1$, two for the U-turn in aisle $j+1$). A transformation of the remaining subgraphs in Figure 2 to the one in Figure 3(d) by using a type (1) connection in both of aisles $j$ and $j+1$ instead of type (5) in $j$ and (2), (3), or (4) in $j+1$ and a type (i) connection between the aisles instead of type (ii) reduces the number of turns in each subgraph to four (one to enter and one to exit each of aisles $j$ and $j+1$). Furthermore, in each case, tour feasibility is maintained (as degree parities of the nodes in the original and transformed graphs are the same) and tour time is shorter.

The transformation improves the travel time minimization objective regardless of the position of the P&D point. The turn minimization objective is also improved as long as the P&D point is not on the corner of a cross aisle and a pick aisle. To see why connection type (5) might need to be used in turn minimization in this case, consider the example with $n = 3$ and 6 items in Figure 4. Without using connection type (5) in aisle 1, the turn minimizing tour (which can be found
by enumeration over all possible connection types) is given in Figure 4(a), and has seven turns. However, the turn minimizing tour when connection type (5) is allowed, shown in Figure 4(b), has six turns. It should be noted here that this move type is useful only when the aisle is entered and exited from the opposite corner of the P&D point. Otherwise, it can be replaced by an appropriate (2) or (3) connection type, incurring the same number of turns.

Proposition 3.2. Given $p$, $q$, and $n \geq 2$ on corner-depot graphs, the total number of turns in the tour is given by $2n + 2p + 6q + r - 1$, where $r = 1$ if the connection type in the pick aisle incident to the P&D point is (5), and $r = 0$ otherwise. $p + q + r$ has to be odd for odd $n$ and even for even $n$.

Proof. Each nonempty pick aisle requires one turn for entrance and one for exit. The U-turns introduced by connection types (2) and (3) require two extra turns. The two U-turns resulting from connection type (4) require four extra turns and since this connection type enters and exits the aisle twice, two more extra turns are incurred for entering and exiting. In the corner-depot case, entering the pick aisle incident to the P&D point does not incur a turn, except when move type (5) is used in the aisle. In this case, the picker has to re-enter the aisle before completing the tour.

Connection type (1) is the only one that traverses a pick aisle completely. Since the tour must start and end on the same side of the warehouse, there has to be an even number of this connection type. This leaves the total number of remaining connection types, $p + q$, as odd for odd $n$ and even for even $n$. If connection type (5) is used, it enters and exits the aisle from the same side. Therefore, there still has to be an even number of connection type (1). These imply $p + q + r$ odd for odd $n$ and even for even $n$.

Proposition 3.3. For the corner-depot case, the minimum number of turns is $2n - 1$ for even $n$ and $2n$ for odd $n$. 

Proof. For the corner-depot case, \( p = q = r = 0 \) for even \( n \) and \( p = q = 0, r = 1 \) for odd \( n \) are feasible and minimize \( 2n + 2p + 6q + r - 1 \).

**Theorem 3.4.** Algorithm 1 finds the turn minimizing tour on corner-depot graphs in \( O(n) \) time.

Proof. The correctness of the algorithm can easily be checked on a case-by-case basis and comparing to the minimum number of turns given in Proposition 3.3. Determining the intra-aisle connection types takes \( O(n) \) time. There are at most \( \left\lceil \frac{n}{2} \right\rceil \) connection type (1) pairs, therefore subtour formation and joining of subtours each take \( O(n) \) time, resulting in an overall complexity of \( O(n) \).

We also provide more generalized versions of Propositions 3.5 and 3.6, where all possible P&D point positions are considered.

**Proposition 3.5.** When edges connecting the same two pick aisles are of equal length, the problem of minimizing the number of turns on a 2-OPP graph is equivalent to one of solving the RPP with required edges as the within-aisle connection types determined by Algorithm 1.

Proof. For the travel time minimization problem, connection type (iv) in Figure 3 of the main text is only needed when connection type (4) is used at least once. By construction of the intra-aisle connection types in Algorithm 1, this is never the case. This implies that in a feasible tour, each neighboring nonempty pick aisle has to have exactly two edges connecting them. Therefore, once the required edges are determined, the corresponding RPP can be trivially solved by forming a feasible tour from these required edges by connecting each neighboring nonempty pick aisle using exactly two edges.

**Proposition 3.6.** (Complete traversal property) There exists a turn minimizing tour on 2-OPP corner-depot graphs that traverses each nonempty pick aisle completely.

Proof. When Algorithm 1 is used, all of these cases result in a turn minimizing tour that completely traverses each nonempty pick aisle.

**B. Extensions of the single objective turn minimization results to cross-depot and pick-depot graphs**

The results and algorithms for the corner-depot case can be mostly extended to the cross-depot and pick-depot cases as well, with a number of exceptions.

**Proposition 3.7.** Given \( p, q, \) and \( n \geq 2 \) on 2-OPP graphs, the total number of turns in the tour is given by

1. \( 2n + 2p + 6q \) for the cross-depot case, where \( p + q \) has to be odd for odd \( n \) and even for even \( n \).
2. \( 2n + 2p + 6q - 2s \) for the pick-depot case. Here, \( s = 1 \) if the connection type in the pick aisle containing the P&D depot is (2), (3), or (4) with one end of the double edges being the P&D point, and \( s = 0 \) otherwise.

For both cases, \( p + q \) has to be odd for odd \( n \) and even for even \( n \).

Here, the proof for the cross-depot case is identical to that for the corner-depot case, except that connection type (5) is never needed. In the pick-depot case, a connection type (2), (3), or (4) with one of the double edges starting from the P&D point saves a U-turn, as no turns are required to enter and exit the P&D point. Furthermore, if the pick aisle containing the P&D point is traversed partially, then it is entered and exited from the same side, which requires an even number of connection type (1). If the aisle is completely traversed, it is entered and exited from different corners. This requires the remaining connection type (1) moves to be odd, implying the total to be even. Hence the remaining connection types \( p + q \) must then be odd for odd \( n \) and even for even \( n \).
The following proposition allows to find the minimum number of turns for these cases.

**Proposition 3.8.**  
(1) For the cross-depot case, the minimum number of turns is \(2n\) for even \(n\) and \(2n+2\) for odd \(n\).

(2) For the pick-depot case where the P&Di point is on aisle \(j\), the minimum number of turns is \(2n\) for even \(n\). For odd \(n\), if there are no items between the P&Di point and \(a_j\) or the P&Di point and \(b_j\), the minimum number of turns is \(2n\). Otherwise, the minimum number of turns is \(2n+2\).

These are due to the fact that for the cross-depot case, \(p = q = 0\) for even \(n\) and \(p = 1, q = 0\) for odd \(n\) are feasible and minimize \(2n + 2p + 6q\). For the pick-depot case, \(p = q = s = 0\) is feasible and minimizes \(2n + 2p + 6q - 2s\) for even \(n\). For odd \(n\), \(s = 1\) in Proposition 3.7 is allowed only if there are no items between the P&D point and \(a_j\), or the P&D point and \(b_j\), in which case \(p = s = 1, q = 0\) is feasible and minimizes \(2n + 2p + 6q - 2s\). Otherwise, the number of turns is minimized by \(p = 1, q = s = 0\).

Based on the results in Propositions 3.7 and 3.8, the algorithms that find the turn minimizing tours for the cross-depot and corner-depot cases are given in Algorithms 1 and 2. For both cases, the algorithms run in \(O(n)\) time. Lastly, we state the cases for which the complete traversal property holds.

**Algorithm 1** Turn minimization algorithm for cross-depot graphs

```plaintext
if \(n\) is even then
    Apply connection type (1) for every aisle
else
    Apply connection type (2) or (3) on an arbitrary aisle; and type (1) on the remaining aisles
end if
```

Starting with the left-most one, join neighboring connection type (1) pairs using connection type (i)

Join subtours using connection types (ii) and (iii) in a way such that \(v_0\) is included

**Algorithm 2** Turn minimization algorithm for pick-depot graphs

```plaintext
if \(n\) is even then
    Apply connection type (1) for every aisle
else
    if \(\exists\) items both between \(v_0\) and \(a_j\); and \(v_0\) and \(b_j\) then
        Apply connection type (2) or (3) on an arbitrary aisle; and type (1) on the remaining aisles
    else if \(\exists\) items between \(v_0\) and \(a_j\); and \(\nexists\) items between \(v_0\) and \(b_j\) then
        Apply connection type (2) to the aisle containing \(v_0\); and type (1) on the remaining aisles
    else
        Apply connection type (3) to the aisle containing \(v_0\); and type (1) on the remaining aisles
    end if
end if
```

Join subtours using connection types (ii) and (iii) in a way such that the edge joining \(v_0\) to \(a_j\) or \(b_j\) is connected to the tour

**Proposition 3.9.** The complete traversal property directly extends to the cross-depot case, where the S-shape heuristic finds the turn minimizing tour. This also holds for the pick-depot case where there are no items between \(a_j\) and \(v_0\) and/or \(v_0\) and \(b_j\), in which case the only partially traversed pick aisle is the one containing \(v_0\).
Table 1. Number of calls to the turn constrained travel time minimization algorithm

| Graph type          | Odd n    | Even n    |
|---------------------|----------|-----------|
| Corner-depot        | 3n − 6   | n = 2     |
|                     |          | n ≥ 4     |
| Cross-depot         | (3n − 7)/2 | n/2      |
|                     |          | (3n − 6)/2 |
| Pick-depot          | 3n − 6 or 3n − 7 | n/2 |
|                     |          | 3n − 6    |

*a*When there are no items between the P&D point and \( a_i \) or \( b_j \).

Table 2. Solution of the example problem in Figure 7 of the main text under turn penalties, with the optimal solution shown in bold

| Equivalence class | Aisle 1 | Aisle 2 | Aisle 3 | Aisle 4 |
|-------------------|---------|---------|---------|---------|
|                   | \( L_1 \) | \( L_2 \) | \( L_3 \) | \( L_4 \) |
| 1. \((U, U, 1C)\) | 32, -1(1) | 36.1(1) | 58.1(2) | 62.1(1) |
| 2. \((E, 0, 1C)\) | 66, -1(2) | 70.2(2) | 92.2(2) | 74.4(3) |
| 3. \((0, E, 1C)\) | 54, -1(3) | 58.3(3) | 112.3(3) | 74.4(3) |
| 4. \((E, E, 1C)\) | - | - | 70.1(1) | 78.4(1) |
| 5. \((E, E, 2C)\) | 44, -1(4) | 52.5(4) | 74.5(2) | 82.5(2) |
| 6. \((0, 0, 0C)\) | - | 54.3(5) | - | 70.4(3) |
| 7. \((0, 0, 1IC)\) | - | - | - | 96.4(6) |

| Aisle 5 | Aisle 6 | Aisle 7 | Aisle 8 |
|---------|---------|---------|---------|
| \( L_5 \) | \( L_6 \) | \( L_7 \) | \( L_8 \) |
| 1. \((U, U, 1C)\) | 126.1(1) | 160.1(2) | 164.1(4) | 164.1(4) |
| 2. \((E, 0, 1C)\) | 128.4(4) | 182.3(3) | 164.1(4) | 164.1(4) |
| 3. \((0, E, 1C)\) | 128.4(4) | 182.3(3) | 224.2(2) | 224.2(2) |
| 4. \((E, E, 1C)\) | 132.4(4) | 160.1(1) | 164.1(4) | 198.1(1) |
| 5. \((E, E, 2C)\) | 108.2(4) | 182.3(2) | 170.2(4) | 194.2(4) |
| 6. \((0, 0, 0C)\) | 124.4(4) | - | 160.4(4) | - |
| 7. \((0, 0, 1IC)\) | 124.4(4) | - | - | 198.4(4) |

Figure 5. The resulting minimum-time tour for the example in Figure 7 of the main text with 266 units of travel time

C. Application of the modified Ratliff and Rosenthal algorithm for travel time minimization with turn penalties

In this part, we use the modified Ratliff and Rosenthal algorithm described in §3.2 of the main text to solve the example instance in Figure 7 of the main text. Here, we assume that each right and left turn takes 2 units of time, whereas a U-turn incurs 4 units of time (hence \( k = 2 \)).

Table 2 presents the detailed solution procedure. The resulting solution of the example is provided in Figure 5, where the total travel time is 266 units, and the tour consists of 35 turns.
Supplement 2 for Section 4: Biobjective Travel Time and Turn Minimization

A. Proofs of statements

Theorem 4.1. The travel time minimization problem under a fixed number of turns on corner-depot graphs can be solved in $O(n^6)$ time.

Proof. The number of possible states in the aforementioned travel time minimizing algorithm for fixed $p, q, r$, and $s$ is in $O(n^2)$ and the possible number of stages in the algorithm is $O(n)$, implying an overall complexity of $O(n^3)$ for a fixed set of these parameters. Given a fixed number of turns $k$, using Proposition 3.2, the number of possible combinations of $p$ and $q$ (and possibly $r$ or $s$) is in $O(n)$. Hence the overall travel time minimization procedure can be completed in $O(n^6)$ time.

Theorem 4.2. The biobjective turn and travel time minimization problem on corner-depot graphs can be solved in $O(n^7)$ time.

Proof. By Theorem 4.1, we can solve the problem of minimizing travel time subject to a fixed number of turns in polynomial time. Therefore, it suffices to show that Algorithm 2 requires the solution of this problem a polynomial number of times. For the corner-depot case, the maximum number of turns is given by $8n - 13$ for $n \geq 3$ and $4n - 1$ for $n = 2$, whereas from Proposition 3.3, the minimum number of turns are given as $2n$ for odd $n$ and $2n - 1$ for even $n$. This leads to $3n - 6$ calls to the turn constrained travel time minimization algorithm for odd $n \geq 3$, $3n - 7$ calls for even $n \geq 4$, and $n$ calls for $n = 2$. Since the number of calls is in $O(n)$ in all cases, overall complexity of Algorithm 2 is $O(n^7)$.

B. Extensions of the results to cross-depot and pick-depot graphs

For the cross-depot case, the algorithm under a fixed number of turns is identical to that for the corner-depot case, except the need to include $r$ as a parameter. Since $r$ is not included, the algorithm requires a total number of $7(p + 1)(q + 1) - 1$ possible equivalence classes, which implies that the number of such classes is still in $O(n^2)$.

For pick-depot graphs with the P&D point on aisle $j$, the parameter $s$ needs to be considered whenever there are no items between either $a_j$ and $v_0$, or between $v_0$ and $b_j$. As in cross-depot graphs, the parameter $r$ need not be considered. The equivalence class definition includes the remaining number of connection types (2), (3), or (4) with the P&D point as an end node, denoted by $\hat{s}$, whose value is initialized at $s$. If $s = 1$, connection types (2), (3), and (4) are the only ones that can be used in the aisle incident to the P&D point, and the latter must be at one end of the connection type used. The optimal solution requires that $\hat{s} = 0$ in the resulting equivalence class, in addition to $\hat{p} = \hat{q} = 0$. Similar to corner-depot graphs, the maximum number of possible equivalence classes is $7(p + 1)(q + 1)(s + 1) - 1$, which is in $O(n^2)$ as $s \in \{0, 1\}$.

The result in Theorem 4.2 extends to the cross-depot and pick-depot cases with the same worst-case complexity. The algorithms for these cases are provided in Algorithms 3 and 4, and the number of calls to the turn constrained travel time minimization algorithm for each case is given in Table 1.
Algorithm 3 Biobjective travel time and turn minimization for cross-depot graphs

\[ STOP = 0 \]
Find a minimum time solution. In case of alternative optima, find the optimal solution with minimum number of turns
\[ k \leftarrow \text{number of turns in the minimum time solution} \]
while \( STOP = 0 \) do
  if \( n \) is odd and \( k = 2n + 2 \), or \( n \) is even and \( k = 2n \) then
    \( STOP = 1 \)
  else
    \( k \leftarrow k - 4 \)
  end if
Find all \((p, q)\) pairs such that \( 2p + 6q = k - 2n \); and \( p + q \) odd for odd \( n \) and even for even \( n \)
Solve the restricted travel time minimization algorithm for each possible combination of the given parameters
If the new solution is dominated by an existing nondominated solution, then discard it
If the new solution dominates an existing nondominated solution, then discard the latter and add the former to the list of nondominated solutions
Otherwise add the new solution to the list of nondominated solutions
end while

Algorithm 4 Biobjective travel time and turn minimization algorithm for pick-depot graphs

\[ STOP = 0 \]
Find a minimum time solution. In case of alternative optima, find the optimal solution with minimum number of turns
\[ k \leftarrow \text{number of turns in the minimum time solution} \]
while \( STOP = 0 \) do
  if \( n \) is even, or \( n \) is odd and \( \exists \) items between the depot and a corner node; and \( k = 2n \) then
    \( STOP = 1 \)
  else if \( n \) is odd, \( k = 2n + 2 \) then
    \( STOP = 1 \)
  else
    \( k \leftarrow k - 2 \)
  end if
Find all \((p, q, s)\) triplets such that \( 2p + 6q - 2s = k - 2n \); and \( p + q \) odd for odd \( n \) and even for even \( n \), \( s \in \{0, 1\} \), \( s \leq p + q \)
Solve the restricted travel time minimization algorithm for each possible combination of the given parameters
If the new solution is dominated by an existing nondominated solution, then discard it
If the new solution dominates an existing nondominated solution, then discard the latter and add the former to the list of nondominated solutions
Otherwise add the new solution to the list of nondominated solutions
end while

Supplement 3 for Section 4.3: Triobjective Problem with U-turn Minimization

It can be observed from the example in Figure 11 of the main text that when U-turns are considered as a separate objective, connection type (5) in Figure 3 of the main text may be used to decrease the number of U-turns in the tour. A straightforward observation is that in order for connection type (5) to reduce the number of U-turns, the pick aisle must be entered and exited from opposite ends in each of the two visits to the aisle. In terms of the travel time minimization objective, this translates to the fact that if an inter-aisle connection type (ii) or (iii) precedes the aisle with connection type (5), the inter-aisle connection succeeding the aisle cannot be of the same type. It
is clear that a tour that uses more than one connection type (5) can be transformed into one that does not use any without affecting the number of U-turns.

For the remainder of this section, we use the same notation in the paper, with the addition that \( v \) denotes the number of connection type (5) used in a given tour, where \( v \in \{0,1\} \). The following proposition determines the number of turns for given parameters.

**Proposition 4.3.1.** Given \( p, q, r, v, \) and \( n \geq 2 \) on 2-OPP corner-depot graphs, the total number of turns in the tour is given by
\[
2n + 2p + 6q + r + 2v - 1,
\]
where \( p + q + r + v \) has to be odd for odd \( n \) and even for even \( n \).

*Proof.* From Proposition 3.2, each nonempty pick aisle is entered and exited at least once, hence requiring at least two turns. Furthermore, connection types (2) and (3) require two extra turns, and connection type (4) requires six extra turns. If connection type (5) is used in the aisle incident to the P&D point, one turn is saved as there is no turn needed to enter the aisle except when connection type (5) is used. A connection type (5) used in a pick aisle (not incident to the P&D in the corner-depot case) requires two entrances and two exits, resulting in two extra turns. By Proposition 3.2, there exist an even number of connection type (1), and therefore \( p + q + r + v \) has to have the same parity as \( n \).

A similar proposition determines the number of U-turns in the tour based on the parameters.

**Proposition 4.3.2.** Given \( p, q, r, v, \) and \( n \geq 2 \) on 2-OPP graphs, the total number of U-turns in the tour is given by
\[
p + 2q,
\]
where \( p + q + r + v \) has to have the same parity as \( n \).

*Proof.* Connection types (2) and (3) require a single U-turn, whereas connection type (4) requires two U-turns. Connection type (5) does not require a U-turn as long as the aisle is entered and exited twice. In the case where it is entered and exited only once, it can be replaced by a connection type (2) or (3), in which case the number of turns and U-turns does not change, and the total travel time is decreased.

Proposition 4.3.2 immediately leads to an approach that finds the tour with the minimum number of U-turns.

**Proposition 4.3.3.** For each 2-OPP instance, there exists a tour with no U-turns.

*Proof.* By Proposition 4.3.2, if \( n \) is even, setting \( p = q = r = v = 0 \) gives a feasible tour with no U-turns. This also implies that the S-shape heuristic minimizes the number of U-turns for even \( n \). If \( n \) is odd, setting \( p = q = r = 0 \) and \( v = 1 \) gives a feasible tour with no U-turns.

An algorithm to solve the triobjective problem would make use of the fact that once the number of U-turns is fixed, the remaining biobjective problem can be solved in polynomial time. To do so, the first step is to modify the travel time minimization algorithm under fixed \( p \) and \( q \) (as well as \( r \) or \( s \) when appropriate) to include the parameter \( v \). The algorithm would be identical to the one described in §4 of the main text, except that the equivalence class definition would include \( \hat{v} \), which denotes the remaining number of connection type (5) for each PTS. Furthermore, if an \( L_j^+ \) PTS uses connection type (5) and is preceded by a \( L_j^- \) PTS that uses connection type (ii) or (iii), we prohibit usage of the same inter-aisle connection type for resulting \( L_{j+1}^- \) PTSs. The resulting algorithm uses \( 7(p + 1)(q + 1)(r + 1)(v + 1) - 1 \) equivalence classes.

**Theorem 4.3.4.** The biobjective problem of travel time and turn minimization under a fixed number of U-turns on 2-OPP graphs can be solved in \( O(n^6) \) time.

*Proof.* Since \( v \in \{0,1\} \), the number of possible equivalence classes (states) in the travel time minimizing algorithm with fixed parameters is in \( O(n^2) \), whereas the number of stages is \( O(n) \), resulting in an overall complexity of \( O(n^5) \).

Given a fixed number of U-turns, the number of possible combinations of \( p \) and \( q \) is in \( O(n) \). Once these parameters are fixed, we need to enumerate over all feasible combinations of \( v \) and \( r \),
the number of which does not depend on \( n \). Hence solving the biobjective problem of travel time and turn minimization under a fixed number of U-turns requires making \( O(n) \) calls to an algorithm that finds each possible nondominated solution in \( O(n^5) \) time. After each solution is found, it is compared to the currently nondominated solutions, which needs at most \( O(n) \) comparisons. Since these comparisons do not increase the complexity, overall complexity is \( O(n^6) \).

**Algorithm 5** Triobjective turn, travel time, and U-turn minimization algorithm for 2-OPP corner-depot graphs

1: Solve the travel time minimization problem, \( t \leftarrow \) number of turns in the solution, \( u \leftarrow \) number of U-turns in the solution
2: \( l \leftarrow 2(n - 1) \)
3: while \( l \geq 0 \) do
4: Find all feasible \((p, q)\) pairs such that \( p + 2q = l, p + q \leq n - 1 \) if \( q > 0, \) and \( p \leq n \)
5: for each feasible \((p, q)\) pair do
6: if \( p + q \) and \( n \) have the same parity then
7: \( r_1 = v_1 = 0, r_2 = v_2 = \infty \)
8: else
9: \( r_1 = 1, v_1 = 0, r_2 = 0, v_2 = 1 \)
10: end if
11: if \((t, u) \not\succ (2n + 2p + 6q + r_1 + 2v_1, p + 2q)\) then
12: Solve the travel time minimization under \((p, q, r_1, v_1)\) constraints
13: if \((t, u) \not\succ (2n + 2p + 6q + r_2 + 2v_2, p + 2q)\) then
14: Solve the travel time minimization under \((p, q, r_2, v_2)\) constraints
15: end if
16: Discard new solution(s) if dominated. Else add to nondominated list. If any currently nondominated solution(s) is dominated by a new solution, then the latter is discarded.
17: end if
18: end for
19: \( l \leftarrow l - 1 \)
20: end while

We can extend the approach for the biobjective problem to one for the triobjective problem that considers U-turn minimization as a separate objective using an \( \epsilon \)-constraint approach which is given in Algorithm 5. Here, if \((a, b)\) and \((c, d)\) are objective values for two biobjective problems, \((a, b) \succ (c, d)\) implies that either \( a < c \) and \( b \leq d \), and/or \( a \leq c \) and \( b < d \).

Algorithm 5 starts by solving the travel time minimization problem and recording the number of turns and U-turns in the solution. Then, it fixes the number of U-turns (given by \( l \)) and finds all feasible sets of \( p \) and \( q \). For each feasible set, it finds the feasible values of \( v \) and checks if the given set of parameters would result in a solution that would be dominated by the one that minimizes the travel time. If not, it calls the travel time minimization algorithm under the given parameters and checks the solution for dominance, updating the set of currently nondominated solutions if necessary.

**Theorem 5.5.** The triobjective problem of travel time, turn, and U-turn minimization on 2-OPP graphs can be solved in \( O(n^7) \) time using Algorithm 5.

**Proof.** By Theorem 4.3.4, given a fixed number of U-turns, we can solve the biobjective problem of travel time and turn minimization and check for dominance in \( O(n^6) \) time. Therefore, it remains to show that we need a polynomial number calls to this algorithm to show that the overall procedure is polynomial. We need to call this algorithm whenever the number of allowed U-turns (which is controlled by \( l \)) changes. Since \( l \) is decreased from \( 2(n - 1) \) to 0, the number of calls is in \( O(n) \), implying an overall complexity of \( O(n^7) \) to find the complete Pareto front. Hence the triobjective problem can be solved in polynomial time.

The algorithm is applied to an example problem given in Figure 6, which consists of seven nonempty pick aisles and 15 pick items. The set of solutions is given in Figure 7, along with the
Figure 6. A corner-depot example for the triobjective 2-OPP with seven nonempty pick aisles and 15 pick items

Figure 7. Solutions found by Algorithm 3 for the example problem in Figure 6, with the Pareto front shown in squares

Pareto front, which includes six solutions, as opposed to the biobjective counterpart of the example with four nondominated solutions.

A. Extensions of the results to cross-depot and pick-depot graphs

**Proposition 4.3.6.** Given $p$, $q$, $s$, $v$, and $n \geq 2$ on 2-OPP graphs, the total number of turns in the tour is given by:

1. $2n + 2p + 6q + 2v$ turns and $p + 2q$ U-turns for the cross-depot case,
2. $2n + 2p + 6q - 2s + 2v$ and $p + 2q - s$ U-turns for the pick-depot case,

where $p + q + v$ has to be odd for odd $n$ and even for even $n$.

The proof for the cross-depot case is identical to that for the corner-depot graphs. For the pick-depot case, when $s = 1$, the picker exits the aisle with connection type (5) in the second visit from the same side as that of the entrance in the first visit. This leaves an even number of connection type (1), resulting in $p + q + v$ with the same parity as $n$. Furthermore, if the P&D point is entered and exited at one end of a connection type (2), (3), or (4), a potential U-turn is avoided as there
are no turns required to enter and exit the P&D point.

Using Proposition 4.3.6, the result in Proposition 5.3 of the main text can be extended to the cross-depot and pick-depot cases. For the former, setting $p = q = 0$, $v = 0$ for even $n$ and $v = 1$ for odd $n$ achieves the desired result. Similarly, for the latter, setting $p = q = s = 0$, $v = 0$ for even $n$ and $v = 1$ for odd $n$ leads to the same conclusion. By using an identical proof to that for Theorem 5.4 in the main text, it can be shown that the algorithms to solve the triobjective problem on cross-depot and pick-depot graphs (given in Algorithms 5 and 6) have the same worst-case complexity as Algorithm 5.

**Algorithm 6** Triobjective turn, travel time, and U-turn minimization algorithm for 2-OPP cross-depot graphs

Solve the travel time minimization problem, $t \leftarrow$ number of turns in the solution, $u \leftarrow$ number of U-turns in the solution

$l \leftarrow 2(n - 1)$

while $l \geq 0$

Find all feasible $(p, q)$ pairs such that $p + 2q = l$, $p + q \leq n - 1$ if $q > 0$, and $p \leq n$

for each feasible $(p, q)$ pair do

if $p + q$ and $n$ have the same parity then

$v = 0$

else

$v = 1$

end if

if $(t, u) \not\prec (2n + 2p + 6q + 2v, p + 2q)$ then

Solve the travel time minimization under $(p, q, v)$ constraints. If the new solution is dominated, it is discarded. Else add to nondominated list. If it dominates any currently nondominated solution(s), then the latter is discarded.

end if

end for

$l \leftarrow l - 1$

end while
Algorithm 7 Triobjective turn, travel time, and U-turn minimization algorithm for 2-OPP pick-depot graphs

Solve the travel time minimization problem, \( t \leftarrow \text{number of turns in the solution}, \ u \leftarrow \text{number of U-turns in the solution} \)
\( l \leftarrow 2(n-1) \)
while \( l \geq 0 \) do
Find all feasible \((p, q, s)\) triplets such that \( p + 2q - s = l \), \( p + q \leq n - 1 \) if \( q > 0 \), \( p \leq n \), and \( s \leq p + q \)
for each feasible \((p, q, s)\) triplet do
if \( p + q \) and \( n \) have the same parity then
\( v = 0 \)
else
\( v = 1 \)
end if
if \((t, u) \not\succ (2n + 2p + 6q - 2s + 2v, p + 2q - s)\) then
Solve the travel time minimization under \((p, q, v, s)\) constraints. If the new solution is dominated, it is discarded. Else add to nondominated list. If it dominates any currently nondominated solution(s), then the latter is discarded.
end if
end for
\( l \leftarrow l - 1 \)
end while