Nested and Interlaced Ticketing for Multiple Travelers

Dongyu Lv* Yizhi Song† Chao Xu‡

Abstract

Nested ticketing represents a prevalent practice in airline bookings, employed to bypass certain airline ticketing regulations with the aim of reducing costs on multiple round-trip tickets. We consider the computational complexity of nested ticketing, and it is 'covert' version, which we call interleaved ticketing. Consider multiple agents living in different locations and a single client who demands that one agent be present each week. The objective for the company is to schedule these agents and arrange their flight itineraries to minimize the cost. We show that when nested ticketing is allowed, the problem is NP-hard when there are at least two agents. We also show there exists a polynomial-time algorithm if only interleaved ticketing is allowed, and the number of airlines is bounded.

1 Introduction

Within commercial air transport, travelers may adopt a wide range of strategies to save on travel costs. Airline booking ploys are ways to circumvent airline ticket rules in order to spend less on the ticket [9]. One common case is nested ticketing, also known as back-to-back ticketing, bracketing and cross-ticketing.

Airlines often charge more for round-trip tickets that only stay the weekdays (mid-week) to target business travelers who prefer to return before the weekend. Conversely, tickets that include a weekend stay are usually cheaper, aiming at leisure travelers. Nested ticketing allows one to potentially save money on multiple round trips between two locations. Such an opportunity arises when a mid-week round-trip ticket is more expensive than the combination of two round-trip tickets involving a stay over the weekend. By buying two round-trip tickets and only using one flight leg per mid-week round-trip ticket, the mid-week round-trip ticket can be simulated at a lower fare.

Although these days airlines use much more sophisticated methods for ticket pricing [6, 12, 17], the strategy still holds: nested ticketing expands the possible combination of round trip tickets, and allow potential savings.

Previous studies for nested ticketing focus more on economic impact, e.g., price fairness for customers [3], or the legal and moral issues [4]. None of these studies the computational difficulty of employing such strategy.

Other common booking ploys like hidden city ticketing was considered extensively in [4, 5, 13], discussing their fairness and issues. We recommend the reader the report by U.S. Government Accountability Office for detailed information [10]. The pricing's impact on revenue was studied in [11, 16].

For many airlines, nested ticketing that circumventing fare rules is explicitly forbidden, this includes all three major US airlines American Airlines [2], Delta [7] and United [15].

In order to be compliant, one can execute nested ticketing *across different airlines*. That is, for a single passenger, for each individual airline, the stay time of the round trips must not overlap. We call

*University of Electronic Science and Technology of China, Chengdu, China, lvdongyu0214@gmail.com
†City University of Hong Kong, Hong Kong, China, yizhisong2-c@my.cityu.edu.hk
‡University of Electronic Science and Technology of China, Chengdu, China, the.chao.xu@gmail.com

this *interleaved ticketing*. Interleaved ticketing has been known to the frequent flyer community at large (e.g. [1]), but no academic studies we are aware of. See Figure 1.1 for an example of nested ticketing and interleaved ticketing with their money savings.

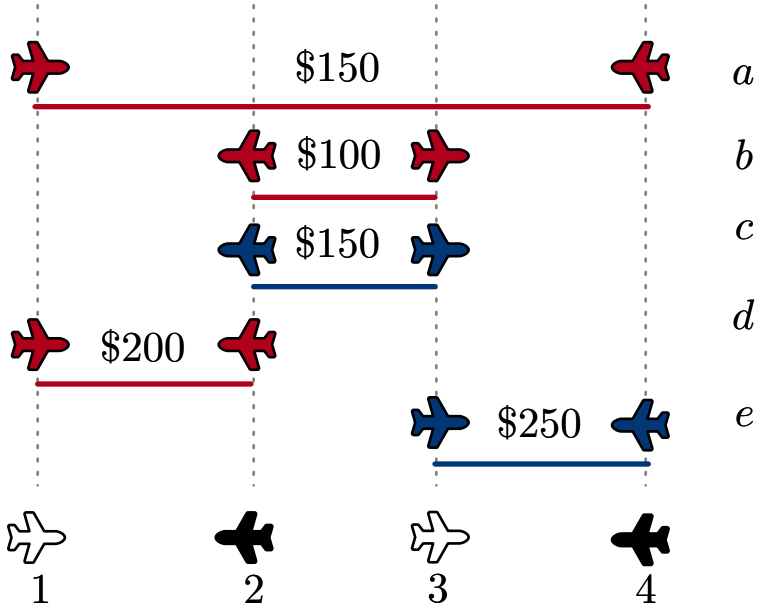


Figure 1.1: Consider a two-week travel schedule where an individual based at A needs to travel to B and return on Monday and Friday, respectively. In this context, ✈ represents a flight from A to B, and ✈ symbolizes a flight from B to A. There are five round-trip tickets available: tickets a, b, and d are operated by airline 1, while tickets c and e are by airline 2. Tickets a, b, and c are relatively more affordable, as they include a weekend stay. For a standard round-trip covering the two weeks, one would typically choose tickets d and e, totaling a cost of \$450. However, a traveler can utilize nested ticketing strategies by purchasing round-trip tickets a and b instead, at a total cost of \$250. It is important to note that tickets a and b are nested within airline 1. To avoid nesting within any single airline, employing interleaved ticketing, the most cost-effective solution is to purchase round-trip tickets a and c, resulting in a total expense of \$300.

In this paper, we begin by examining an empirical scenario and subsequently model optimizing the cost of buying some round trip tickets as matching problems. We frame the problem as identifying the minimum weight matching satisfying some label and color constraints. Moreover, we provide a detailed discussion of various specific scenarios arising from two key conditions: (1) whether there is a single traveler or multiple travelers, and (2) whether nested ticketing within a single airline is allowed or forbidden.

Our contribution We show that the optimizing the round-trip itinerary of two people allowing nested ticketing is NP-hard. In contrast, for fixed number of airlines, one obtain polynomial time solvability if we only allow interleaved ticketing.

2 Preliminaries

Let $[n] = \{1, \ldots, n\}$ to denote all positive integers up to n. We consider multigraphs where there can be parallel edges and self-loops.

Travelers	1	2	$p \geq 3$
Nested ticketing	$n^{2+o(1)}$	NP-hard	NP-hard
Interleaved ticketing	$O(n^k)$	$O(n^{2k})$	$O(n^{pk})$

Figure 2.1: The computational complexity of finding the optimum cost solution to the general scenario if nested ticketing or interleaved ticketing is allowed, for k airlines, where k is a constant.

A *matching* is a set of disjoint edges. The matching is *perfect* if every vertex is contained in some edge of the matching. The bipartite perfect matching (BPM) problem is to find a perfect matching in a bipartite (multi)graph.

2.1 Problem Setup

We also establish three progressively more general scenarios.

The first scenario, referred to as the *solo scenario*, involves determining the optimal set of tickets for an individual who requires multiple round trips to the same city, a situation commonly faced by business travelers.

In the next scenario, we introduce the *dual scenario*, which considers a couple in a long-distance relationship, residing in separate home cities. They aim to visit each other every weekend, with one partner flying to the other's city each Friday and returning on Sunday. This scenario is an extension of the solo scenario, as it can be reduced to the solo case by assigning an infinite cost to the second person's ticket. Additionally, within the dual scenario, we define a *fair dual scenario* where each individual is required to fly an equal number of times, assuming their meetings are an even number.

The *general scenario*, accommodating multiple travelers. Imagine a company with a client in city C_0 and p agents based in cities $C_1, \ldots, C_p$. The client requires the presence of some agent each week for the next n weeks, who can be different each week. Every Monday, an agent travels to the client's city, returning on Friday. The company's objective is to strategically schedule agents' visits and purchase tickets in a way that minimizes the total cost of travel.

One can observe when $p = 1$, the general scenario is the solo scenario.

It is somewhat less apparent, but when $p = 2$, the general scenario is the dual scenario. In the dual scenario, two individuals travel to each other's cities, rather than to a separate third location. Nonetheless, in the general scenario, the specific destination of each agent's travel is not the primary concern. The crucial aspect is that if an agent does travel, their departure and return city remain consistent, aligning with the conditions of the dual scenario.

The general scenario where only the simplest ticket buying strategy is allowed (buy a round trip ticket for each week) is a simple assignment problem. The objective is to assign each week to an agent such that the cost, defined as the round-trip expense for that agent in a given week, is minimized.

This is a trivial assignment problem. A simple greedy algorithm works: for each week i, we determine which agent incurs the lowest cost to fly and assign that agent to the week. This process is repeated for each subsequent week.

When considering nested ticketing, the problem can still be framed as an assignment problem, albeit with an altered cost structure. The task remains to assign specific weeks to agents. However, in this context, the cost of each assignment is contingent upon the *specific set of weeks* allocated to an agent. Once an assignment is established, the cost for each agent can be calculated using an algorithm designed to solve the solo scenario.

In the next section, we model our scenarios as graph matching problems with extra constraints. As a preview, we show the computational complexity of solving different scenarios in Figure 2.1.

2.2 Model

Consider the general scenario. For a schedule spanning n weeks, we build a multigraph comprising $2n$ vertices. The edges of this graph are labeled by a function $\ell : E \to [p]$, where each label corresponds to a specific agent. In this representation, the $(2i-1)$-th vertex signifies the outbound flight (departing from the agent's location) in the i-th week, while the $(2i)$-th vertex denotes the inbound flight (returning to the agent's location) in the same week. Here, i ranges from 1 to n, inclusively.

In a labelled graph where the vertices are $[2n]$, a matching is *homogeneous* if $(2i-1)$ and $(2i)$ are covered by edges of the same label, for each i. A round-trip ticket for agent q, originating in the i-th week and returning in the j-th week with a cost of c, is modeled as an undirected edge connecting vertices $(2i-1)$ and $(2j)$, bearing the label q and assigned the weight c. In a similar vein, a one-way ticket is represented by a self-loop. Note the graph is a bipartite graph. A homogeneous matching make sure the in and out flight of a week is taken by the same agent. Perfect matching forces each flight has to be taken. In order to minimize the total cost of the tickets, it translates to finding a homogeneous perfect matching of minimum weight.

Therefore, we are interested in solving the following problem

Problem 1 (Minimum weight homogeneous BPM (HM(k)))
Input: *A bipartite graph $G = ([2n], E)$ where the bipartition is all the even vertices and all the odd vertices. There is a label function on the edges $\ell : E \to [k]$, and there is a weight function $w : E \to \mathbb{R}$.*
Output: *A homogeneous perfect matching of minimum weight.*

Now, if we impose a restriction to avoid nested ticketing for each airline, this can be modeled by introducing a color, to each edge. Specifically, for each ticket, we assign a color corresponding to its respective airline.

We say a matching M of $G = ([2n], E)$ is *linear*, if for any two edges of the same color $\{a, b\}, \{c, d\} \in M$, we have $[a, b] \cap [c, d] = \emptyset$. Here $[a, b]$ means the real interval from a to b.

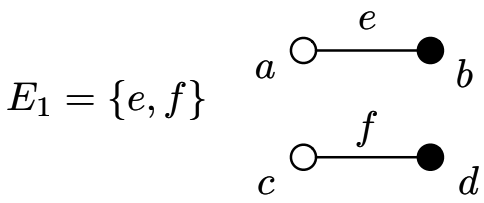
To minimize the total cost of tickets in the general scenario that also does not allow nested ticketing (but allows interleaved ticketing), is equivalent to the following algorithmic problem.

Problem 2 (Minimum weight linear homogenous BPM (LM(k, p))
Input: *A bipartite graph $G = ([2n], E)$, an edge weight function $w : E \to \mathbb{R}$, edge label function $\ell : E \to [p]$ and edge color function $c : E \to [k]$.*
Output: *A linear homogenous perfect matching M of minimum weight.*

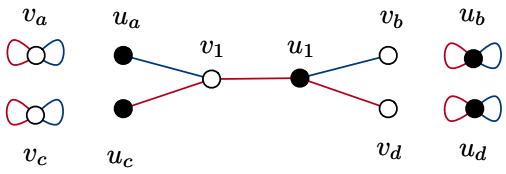
Finally, we want to add an additional problem just to handle the fair dual scenario, allowing only interleaved ticketing.

Problem 3 (Minimum weight fair linear homogenous BPM (FM(k))
Input: *A bipartite graph $G = ([2n], E)$ where n is an even number, an edge weight function $w : E \to \mathbb{R}$, edge label function $\ell : E \to [2]$ and edge color function $c : E \to [k]$.*
Output: *A linear homogenous perfect matching M, so either label 0 or label 1 vertices appears exactly $n/2$ times. Under previous constraints, the weight is minimized.*

In a solution, we say a vertex has label (color) k, if it is contained in an edge of label (color) k. In a solution, a vertex is the start vertex, if it is not the larger numbered vertex. Otherwise, it is the end vertex. Note in particular, this means a vertex with a self-loop is an end vertex.



(a) The color class E_1 for the rainbow matching problem.



(b) The transformed gadget.

Figure 3.1: The reduction gadget.

3 Minimum weight homogeneous BPM

We set up the problem with a bipartite graph $G = ([2n], E)$. For simplicity, it is more helpful to consider the vertices partitions to be $U = \{u_1, \ldots, u_n\}$ and $V = \{v_1, \ldots, v_n\}$, where u_i and v_i are supposed to be contained in edges of the same label. We use $G[U]$ to mean the *induced subgraph* of G on the vertex set U.

HM(1) is the standard minimum weight bipartite perfect matching problem, which can be solved in polynomial time [14]. Next, we will show **HM**(2) is a NP-hard problem. Since **HM**(i) is no harder than **HM**(j) for $i \leq j$, it proves **HM**(k) is NP-hard for all $k \geq 2$.

For the reduction, we need a few definitions. A matching is called a *rainbow matching*, if each edge in the matching has a different color.

Problem 4 (At-most-2 Rainbow BPM (A2RBPM))
Input: *A bipartite graph $G = (V, E)$ with vertex bipartition A and B. The edges are partitioned into color classes $E_1, \ldots, E_m$, such that $|E_i| = 2$ for each i. Also, the edges in each E_i are disjoint.*
Output: *A perfect matching M that is a rainbow matching.*

A2RBPM is known to be NP-complete [8]. We will reduce **A2RBPM** to **HM**(2).

Theorem 3.1 *HM*(2) *is NP-Hard.*

Proof: We begin the description of the reduction to **A2RBPM**. Let $G = (A \cup B, E)$, $E_1, \ldots, E_m$ be an instance of **A2RBPM**. We want to construct an instance to **HM**(2), with input $H = (U \cup V, F)$, label function ℓ and weight function w, such that G has a rainbow perfect matching if and only if the H has a homogenous matching. Note our weight function will just be the all 0 weight function.

The idea is each color class in G transforms into two vertices in H.

For each vertex $a \in A$, we construct two vertices $u_a \in U$ and $v_a \in V$. v_a is there only for formality. We add self-loops of each label to v_a. Similarly, for each vertex $b \in B$, we construct $u_b \in U$ and $v_b \in V$ where u_b has a self-loop of each label. We call v_a and u_b dummy vertices.

For each color class $E_i = \{e, f\}$, we create vertices u_i and v_i and add to U and V, respectively. We add an edge between u_i and v_i for each label. Assuming that $e = \{a, b\}$ and $f = \{c, d\}$. We add edges $u_a v_i$ and $v_b u_i$ of label 0. We also add edges $u_c v_i$ and $v_d u_i$ of label 1. This forms a single gadget. This reduction takes polynomial time.

We show that a rainbow matching in G exists if and only if a homogeneous perfect matching in H exists.

Without loss of generality, assume the rainbow matching in G using edges $a_i b_i \in E_i$ for i from 1 to n. This would imply a homogeneous perfect matching in H as follows: u_i and v_i is matched to u_{a_i}, and v_{b_i}, and has the same label (either 0 or 1 depending on construction). By our construction, u_a and v_b

are matched for $a \in A$ and $b \in B$. For all color classes E_j, we add $u_j v_j$ into the matching. Finally, for each dummy vertex, add the self-loop with the corresponding vertex's label.

On the other hand, suppose H has a homogeneous perfect matching M. We construct a rainbow perfect matching M' in G. We look at each i where $u_i v_i \notin M$. Because M is a perfect matching, then u_i matched with v_a and v_i matched with u_b for some a and b. We add the edge $e = ab$ to M'. M' is a rainbow matching since the edges do not have the same color. M' is a perfect matching, since if u_a is matched in M for $a \in A$, then a is also matched in M'. It holds true for v_b where $b \in B$.

Hence, this shows **A2RBPM** reduces to **HM**(2), and therefore **HM**(2) is NP-hard. □

4 Minimum weight linear homogeneous BPM

In this section, we show a polynomial time algorithm for **LM**(k, p) for fixed k and p.

4.1 Single label

For ease of understanding, we start our discussion with the scenario involving only a single label. In this case, the primary focus is on colors rather than labels. We build a solution through dynamic programming.

Imagine building the solution incrementally by sequentially inspecting vertices from 1 to n. At the i-th step, the task is to determine the color of the vertex and decide whether this vertex represents the start or end of an edge of that color. We then move to the next vertex and continue this process until a complete solution is formed. Every solution can be constructed in this manner: one simply needs to take any given solution and follow its pattern, performing the construction sequentially from vertices 1 to n. What information is required for each decision? As an example, if we chose the vertex to have the color red, if the last red vertex is not an end vertex of an edge, then the current vertex has to be an end vertex of a red edge.

Consequently, when assigning a specific color to a vertex, it is imperative to know the location of the previous vertex of the same color and whether it is an end vertex.

Therefore, we are able to derive the required dynamic programming formulation. This information can be effectively encoded using two vectors, A and B. Here, A_i represents the index of the preceding vertex with color i, while B_i is assigned the value of 1 if the preceding vertex of color i is a start vertex, and 0 if otherwise.

Formally, let $A \in ([2n] \cup \{\infty\})^k$. We say A is *valid*, if all the non-infinity elements in A are distinct. Let $\mathcal{A}$ be all the valid vectors. We let $\mathcal{A}_m$ to be all valid vectors where the maximum element is m. $B \in \{0,1\}^k$. Define $\mathcal{M}(A,B)$ to be the set of feasible solutions such that the last vertex before m of color i is A_i, and there is an edge $\{A_i, t\}$ for some $t > i$ if and only if $B_i = 0$. Note if there is no vertex before m of color i, A_i would be ∞. Define $D[A,B,m] = \min_{M \in \mathcal{M}(A,B)} w(M[V_m])$, where V_m are the first m vertices.

The solution to our problem is precisely $\min\{D[A,B,2n] \mid A \in \mathcal{A}_{2n}, B \in \{0,1\}^k\}$. We describe the recurrence relation, where A is a valid vector with $\max(A) = m$, and $A_i = m$.

First, in order to construct a solution captured by $D[A,B,m]$, we can look at what is happening before, at $D[A',B',m-1]$. Observe here $A'_j = A_j$ for all $j \neq i$, and $B'_j = B_j$ for all $j \neq i$. Therefore, we just have to know about A'_i and B'_i. We consider two different cases. If $B_i = 1$, then this implies m is an end vertex of color i, and we can ask where is the start vertex. Again, there are two cases: either the start vertex is m, or the start vertex is before m. In the first case, we can see A'_i to be anything other than the positive elements in A'_j and $B'_i = 1$. In the second case, we would mark A'_i to be any valid number, and set $B'_i = 0$. Subsequently, if $B_i = 0$, this situation becomes straightforward. We simply choose A'_i to be any valid number and then set $B'_i = 1$.

```
D[A, B, m]:
  if m = 0 :
      return 0
  i ← the index where A_i = m
  A' ← A
  B' ← B
  X ← ∅
  N ← [m − 1] \ {A_j | j ∈ [k]} ∪ {∞}
  for n' in N:
      A'_i ← n'
      if B_i = 0:
          B'_i ← 1
          X ← X ∪ {D[A', B', m − 1]}
      if B_i = 1:
          B'_i ← 0
          X ← X ∪ {D[A', B', m − 1] + w(e(A'_i, m, i))}
          B'_i ← 1
          X ← X ∪ {D[A', B', m − 1] + w(e(m, m, i))}
  return min(X)
```

Figure 4.1: Pseudocode to find the optimal value. Assume $e(a, b, i)$ returns the edge between a and b with color i.

The base case is when $m = 0$, which we would define it to be 0. The pseudocode can be seen in Figure 4.1.

Observe that the number of states in the dynamic program initially appears to be $O(n^k 2^k n)$. However, the last parameter is actually determined by the first vector, reducing the number of states to $O(n^k 2^k)$. To compute a single state, we require $O(n)$ time. Thus, the total running time would be $O(n^{k+1} 2^k)$. A more refined analysis, however, allows us to eliminate a factor of n.

Consider a graph representing the states in the dynamic program. There is an edge from state $(A', B', m-1)$ to state (A, B, m) if $D[A, B, m]$ calls $D[A', B', m-1]$. We assert that this graph has at most $O(kn^k 2^k)$ edges. Indeed, for a given $(A', B', m-1)$, there are at most $2k$ different (A, B, m) that depend on it. This is because A can differ from A' in only one coordinate, and if different, that coordinate must be m. Thus, the number of possible A is at most k. Similarly, B and B' can differ in at most one position, which must align with the same position of A, allowing at most 2 choices. As such, the number of outgoing edges from $(A', B', m-1)$ is at most $2k$. Consequently, the dynamic program operates in $O(kn^k 2^k)$ time. For a fixed k, the runtime is $O(n^k)$, leading us to the following theorem.

Theorem 4.1 ***LM***$(k, 1)$ *can be solved in* $O(n^k)$ *time for a graph on* n *vertices for fixed* k.

4.2 Multiple labels

The idea is basically identical to the single label version. However, we now need to concurrently track both the color and label of each vertex. We define the ordered pair comprising the color and label of a vertex as the vertex's signature. For instance, a vertex with label i and color j possesses the signature (i, j).

In place of the vectors A and B, we now require matrices A and B. Here, $A_{i,j}$ represents the last vertex that holds the signature (i, j), while matrix B indicates whether the last vertex with signature (i, j) is an end vertex. The complete pseudocode for this dynamic programming approach is provided in Figure 4.2. The primary modification from the single label version is the enforcement that the label

of vertex $(2i-1)$ must match that of vertex $(2i)$. Utilizing a similar process, we arrive at our main theorem.

```
D[A, B, m]:
  if m = 0 :
      return 0
  i, q ← the index where A_{i,q} = m
  A' ← A
  B' ← B
  X ← ∅
  ⟨⟨Make sure the labels are the same.⟩⟩
  if i%2 = 0 and m − 1 ∉ {A_{j,q} | j ∈ [k]}:
      N ← {m − 1}
  else:
      N ← [m − 1] \ {A_{j,q'} | j ∈ [k], q' ∈ [p]} ∪ {∞}
  for n' in N:
      A'_{i,q} ← n'
      if B_{i,q} = 0:
          B'_{i,q} ← 1
          X ← X ∪ {D[A', B', m − 1]}
      if B_{i,q} = 1:
          B'_{i,q} ← 0
          X ← X ∪ {D[A', B', m − 1] + w(e(A'_{i,q}, m, i, q))}
          B'_{i,q} ← 1
          X ← X ∪ {D[A', B', m − 1] + w(e(m, m, i, q))}
  return min(X)
```

Figure 4.2: Pseudocode to find the optimal value for multiple labels. Assuming $e(a, b, i, q)$ returns the edge between a and b with color i and label q.

Theorem 4.2 ***LM***(k, p) *can be solved in* $O(n^{kp})$ *time for a graph on* n *vertices when* k *and* p *are fixed.*

Remark If we define k_q to be the number of possible colors when fixing the label to be q, then the running time would be $O(n^{\sum_{q=1}^{p} k_q})$.

Furthermore, it is straightforward to observe that **FM**(k) can be resolved by additionally recording the number of occurrences of label 0 in the solution. This modification results in an increase in the running time by an additional factor of $O(n)$ compared to **LM**$(k, 2)$. Consequently, we establish the following theorem.

Theorem 4.3 ***FM***(k) *can be solved in* $O(n^{2k+1})$ *time for fixed* k.

5 Experiments

We acquired a dataset of airfare prices from Google Flights, which includes the ticket information from New York to Los Angeles for ten weeks starting from March 11, 2024 (Monday). During the data collection, we limited the airlines to three commonly seen carriers on this route, namely American, United, and Delta, and the tickets were specifically for flights departing on Monday and returning on Friday.

For comparative analysis, we conducted three distinct experimental sets. In the first strategy, termed 'simple round trip', we purchased round-trip tickets on a weekly basis without allowing for any nesting.

In the second strategy, we permitted interleaved ticketing. The third and final strategy allowed for nested ticketing. The outcomes of these different strategies are detailed in Table 1.

Price \ Week Count	4 weeks	5 weeks	6 weeks	7 weeks	8 weeks
Simple Round Trip	1352	1687	2022	2366	2701
Interleaved Ticketing	1221	1501	1836	2137	2424
Nested Ticketing	1193	1464	1798	2098	2387

Table 1: The cost for solo scenario, in US dollars.

Price \ Week Count	4 weeks	5 weeks	6 weeks	7 weeks	8 weeks
Simple Round Trip	1297	1632	1967	2311	2646
Interleaved Ticketing	1183	1463	1753	2043	2378
Nested Ticketing	1183	1463	1753	2043	2378

Table 2: The cost for dual scenario, in US dollars.

As indicated in Table 1, for purchasing tickets across consecutive weeks of future travel, interleaved ticketing typically offers a savings of about 10% compared to the simple round trip strategy. While nested ticketing may be more cost-effective, this method is prohibited by airlines. With only a marginal price difference compared to nested ticketing, interleaved ticketing emerges as the more advantageous ticket-purchasing strategy for compliance with airline policies.

In the dual scenario, further experiments were conducted and comparatively analyzed. The results of these experiments are displayed in Table 2.

The findings align with our expectations. Similar to the solo scenario, interleaved ticketing in the dual scenario yields savings of over 10% compared to simple round trips. Interestingly, for our specific data, nested ticketing is not more cost-effective than interleaved ticketing, despite differences in flight selections under varying constraints. This suggests that the optimal solution might inherently exclude any nesting within airlines.

When considering fairness between two individuals, we documented the results under a specific scenario where both parties fly an equal number of times over a six-week period. The results are as follows: the cost for a simple round trip totaled $1967, while both interleaved and nested ticketing strategies resulted in a cost of $1764, approximately 10% less expensive.

Acknowledgements. Chao like to thank Lingyu Xu for the inspiration of the problem, Naonori Kakimura for helpful pointers related to constrained matchings, and 美卡论坛, where he learned about airline booking ploys.

References

[1] Back-to-back tickets. to do or not to do? https://www.flyertalk.com/forum/milesbuzz/938-back-back-tickets-do-not-do.html, 1999. Accessed on 01.14.2024.

[2] Agent International Passenger Rules Airline Tariff Publishing Company and Fares. Tariff no. aa1. https://www.aa.com/content/images/tariff/american-airlines-general-rules-of-the-international-tariff.pdf, 2023. Accessed on 01.14.2024.

[3] Shirin Aslani, Mohammad Modarres, and Soheil Sibdari. On the fairness of airlines' ticket pricing as a result of revenue management techniques. *Journal of Air Transport Management*, 40:56–64, 2014.

[4] Gregor Bischoff, Sven Maertens, and Wolfgang Grimme. Airline Pricing Strategies Versus Consumer Rights. *Transportation Journal*, 50(3):232–250, 07 2011.

[5] Gerald N. Cook and Bruce G. Billig. *Airline operations and management: A management textbook*. Routledge, London, England, 2 edition, 2023.

[6] Christine SM Currie, Russell CH Cheng, and Honora K Smith. Dynamic pricing of airline tickets with competition. *Journal of the Operational Research Society*, 59(8):1026–1037, 2008.

[7] Delta. Delta domestic general rules tariff. https://www.delta.com/content/dam/delta-www/pdfs/dl-dgr-master-22jun23.pdf, 2023. Accessed on 01.14.2024.

[8] Van Bang Le and Florian Pfender. Complexity results for rainbow matchings. *Theoretical Computer Science*, 524:27–33, March 2014.

[9] Sarah Meire and Ben Derudder. Pirating the skies? a review of airline booking ploys. *Research in Transportation Business & Management*, 43:100721, 2022.

[10] U.S. Government Accountability Office. Aviation competition: Restricting airline ticketing rules unlikely to help consumers, July 2001. Report to Congressional Committees.

[11] Jaelynn Oh and Woonghee Huh. Hidden city travel and its impact on airfare: The case with competing airlines. *Transportation Research Part B: Methodological*, 156:101–109, 02 2022.

[12] Daniel F Otero and Raha Akhavan-Tabatabaei. A stochastic dynamic pricing model for the multiclass problems in the airline industry. *European Journal of Operational Research*, 242(1):188–200, 2015.

[13] James J. Rakowski. Does the consumer have an obligation to cooperate with price discrimination? *Business Ethics Quarterly*, 14(2):263–274, 2004.

[14] Alexander Schrijver. *Combinatorial Optimization*. Algorithms and Combinatorics. Springer, Berlin, Germany, 2003 edition, December 2002.

[15] Inc. United Airlines. Contract of carriage document. https://www.united.com/en/us/fly/contract-of-carriage.html, 2023. Accessed on 01.14.2024.

[16] Zizhuo Wang and Yinyu Ye. Hidden-city ticketing: The cause and impact. *Transportation Science*, 50(1):288–305, 2016.

[17] Kevin Williams. Dynamic airline pricing and seat availability. *Cowles Foundation discussion paper*, 2020.