Novel Approaches to Airplane Boarding

Qianwei Li
Arnav Mehta
Aaron Wise
Duke University
Durham, NC

Advisor: Owen Astrachan

Summary

Prolonged boarding not only degrades customers' perceptions of quality but also affects total airplane turnaround time and therefore airline efficiency [Van Landeghem 2002].

The typical airline uses a zone system, where passengers board the plane from back to front in several groups. The efficiency of the zone system has come into question with the introduction and success of the open-seating policy of Southwest Airlines.

We use a stochastic agent-based simulation of boarding to explore novel boarding techniques. Our model organizes the aircraft into discrete units called "processors." Each processor corresponds to a physical row of the aircraft. Passengers enter the plane and are moved through the aircraft based on the functionality of these processors. During each cycle of the simulation, each row (processor) can execute a single operation. Operations accomplish functions such as moving passengers to the next row, stowing luggage, or seating passengers. The processor model tells us, from an initial ordering of passengers in a queue, how long the plane will take to board, and produces a grid detailing the chronology of passenger seating.

We extend our processor model with a genetic algorithm to search the space of passenger configurations for innovative and effective patterns. This algorithm employs the biological techniques of mutation and crossover to seek locally optimal solutions.

We also integrate a Markov-chain model of passenger preference into our processor model, to produce a simulation of Southwest-style boarding, where

The UMAP Journal 28 (3) (2007) 353–370.

seats are not assigned but are chosen by individuals based on environmental constraints (such as seat availability).

We validate our model using tests for rigor in both robustness and sensitivity. Our model makes predictions that correlate well with empirical evidence.

We simulate many different a priori configurations, such as back-to-front, window-to-aisle, and alternate half-rows. When normalized to a random boarding sequence, window-to-aisle—the best-performing pattern—improves efficiency by 36% on average. Even more surprising, the most common technique, zone boarding, performs even worse than random.

We recommend a hybrid boarding process: a combination of window-to-aisle and alternate half-rows. This technique is a three-zone process, like window-to-aisle, but it allows family units to board first, simultaneously with window-seat passengers.

Survey of Previous Research

Discrete Random Process

Bachmat et al. [2006] propose a discrete boarding process in which passengers are assigned seats before boarding. The inputs to the process are an index for the position of each passenger in the queue and a seat assignment for each passenger. Additionally, the researchers define the aisle space that each passenger occupies, the time it takes to clear the aisle once the designated row is reached, and the distance between consecutive rows. The first two parameters are sampled from distributions defined by the researchers.

The model considers the travel path of each passenger. The passenger moves down the aisle until reaching an obstacle, which is either the back of a queue or a person who is preparing to sit in their row. Passengers who arrive at their row clear the aisle after a delay time; after that, the passengers behind continue down the aisle.

The researchers define an ordering relation between passengers. Each passenger is assigned a pointer to the last passenger who blocked their path. By following the trail of passengers, the longest chain in the ordering ending at any particular passenger can be identified. This chain specifies the number of rounds needed for the simulation.

Other Simulation Studies

Van Landeghem [2002] simulates different patterns of boarding sequences, based on a plane with 132 seats divided into 23 rows, with Row 1 and 23 having 3 seats and the others having 6. The first objective is to reduce total boarding time; the second objective is to augment the quality perception of the passengers by evaluating the average and maximum individual boarding times as seen by the passengers. Van Landeghem simulates calling passengers to board at random or by block (contiguous full rows), half-block (contiguous rows, port or starboard halves only), row, half-row, or individual seat. The shortest boarding time is by seat (in a particular order).

Model Overview

We present a simulation model that can be considered a stochastic agent-based approach.

We treat the plane as a line, with destinations (seats) at regular distances along the line. Each passenger, modeled as an agent, moves along the line until reaching the assigned seat. Each agent has a speed constrained by the slowest person in front.

This model takes into account the topology of the airplane. Each row is a discrete unit. We call these units *processors*, since they determine the rate that an individual moves through the system. Each processor has a queue, a list of people waiting to be processed by it (and hence moved to the next node of the system). Each agent has a destination processor, the row of the assigned seat.

We consider two major collision parameters. A scenario where a passenger is waiting for another passenger to stow baggage is a *baggage collision*. We also model *seat collisions*: when a passenger is sitting between another passenger and that passenger's seat (for example, the passenger with an assigned window seat must move around a passenger in an aisle seat).

We attempt to optimize boarding time based on the order in which passengers enter the plane, via a genetic algorithm over the search space of all possible orderings. Crossovers and mutations are defined so that no seats are "lost."

Our final model includes a Markov chain to model passenger preferences in an open seating environment. This model simulates a boarding process like the one used by Southwest Airlines.

Details of the Model

Basic Model

We use a compartmental model, calling the compartments "processors," each physically analogous to the space of one row. Differing layouts of processors can model varied plane topologies.

Each passenger is randomly assigned a seat. These seats are not necessarily unique; they are uniformly drawn from all seats on the plane. A seat is represented as a coordinate pair (c, r), where r is a row and c is a seat number.

Passengers move based on the function of the processors. The processors are in series, with each processor having the next as an output (**Figure 1**).

Since movement is performed by processors pushing passengers from one row to the next, each passenger stores only that passenger's destination. A passenger who reaches a processor waits in a first-in/first-out queue to be

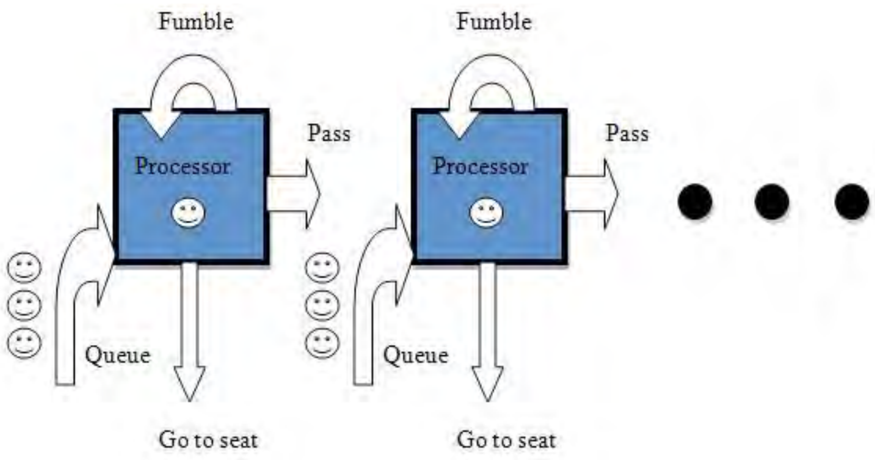

Figure 1. Processor-based model.

processed (people cannot move around one another while in the aisle). The initial state of the plane is that all passengers are queued at the first processor.

In each iteration, each processor looks at the destination of the passenger and performs one of the functions:

- **Pass.** A passenger who passes moves from the current processor to the end of the queue of the next processor.
- **Fumble.** With a certain small probability, the processor will do nothing this cycle (a bag gets caught in the aisle, a passenger trips, or some other time-wasting random event occurs). A fumble is not equivalent to time spent stowing baggage or rearranging passengers; our basic model accounts only for random time-wasting events.
- **Sit Down.** If this row contains the assigned seat for the passenger currently in the processor, the passenger leaves the aisle and is seated.
- **Idle.** If there is no passenger in the processor (and the queue is empty), the processor does nothing.

The processors run sequentially from back to front until every passenger is seated.

Assumptions

- The initial configuration is that all passengers are queued at the first row. In actuality, all passengers are initially queued at the ticket counter, where their boarding passes are scanned and they walk a short distance to the plane. Hence, a more realistic alternative would be a Poisson arrival process from the ticket counter to the queue for the first row. However, this additional process is unnecessary because of the high speed at which boarding passes are scanned, which approximates the speed of normal walking. Hence, passengers reach the queue at a much higher rate than they are moved forward through the plane; the queue at the first processor forms instantly when the first passengers walk into the plane.

- There is no idle time between the first passenger entering the first queue and the last passenger doing so. The airline could wait until there is no queue left before calling additional passengers to board. However, doing this is never to the airline's advantage.
- Special-needs and business-class passengers have already boarded; airlines have an obligation to these customers for early boarding. We start our simulation clock after these special classes of passenger have already boarded and deal only with the bulk passenger class.
- Every passenger functions individually. We expect improved efficiency when passengers travel in groups, since groups are self-organizing (the individuals in a group do not collide with one another).

Extended Model

Seat assignment

The initial model assigns seats randomly and without uniqueness. We remedy this to a one-to-one correspondence between passengers and seats.

Assumptions

- The plane is fully booked and every seat is occupied. This assumption allows us to optimize over the worst-case scenario.

Seat collisions

A common occurrence is a passenger needing to cross over a seated passenger. To account for such a *seat collision*, we implement a new processor function:

- **Rearrange.** This cycle is spent waiting for the aisle to clear after the seat collision. This operation reduces the seat collision counter by 1.

A seat collision has an associated time penalty that depends on the type of collision. When there is a seat collision, the processor for that row spends a number of cycles equal to the time-cost sorting out the collision. During that time, no other passengers can enter the processor (though they can enter the processor queue).

We determined values for the seat collision costs by physical experimentation involving multiple trials over a simulated plane row. All seat collisions have the same time cost, except that the penalty for crossing over two passengers is about 50% more than for crossing over a single passenger. We expect the variation among passengers to be small.

Luggage

A major factor in boarding times is stowing hand luggage. As the overhead bins fill, it takes longer to stow a bag. Hence, we developed a statistical model of luggage stowing. Luggage stowing is performed by the processor at a given row using the command:

- **Stow.** This cycle is spent by a passenger stowing a bag in the overhead bin. The stowing counter is decreased by 1.

For luggage stowage times, we use a Weibull distribution because of its flexibility in shape and scale. The probability density function is

$$f(x,\kappa,\lambda)=\frac{\kappa}{\lambda}\left(\frac{x}{\lambda}\right)^{\kappa-1}e^{-\left(\frac{x}{\lambda}\right)^{\kappa}},$$

where λ is a scaling parameter, κ is a shape parameter, and x is the number of people who have entered the plane. Its cumulative distribution function

$$F\left(x,\kappa,\lambda\right)=1-e^{-\left(\frac{x}{\lambda}\right)^{\kappa}}.$$

is a measure of the additional time to stow hand luggage as the plane fills up.

The waiting time of passenger x is

$$\lceil cF\left(x,\kappa,\lambda\right)+N\rceil,$$

where c is a measure of the additional time to store baggage when the plane is full and N is a Gaussian noise parameter that accounts for the nonuniformity of the boarding process.

Queue Size

The initial model assumes that each processor has an unlimited queue. This makes sense for the initial processor, whose queue consists of all passengers lined up along the loading ramp. However, for a processor inside the aircraft, the queue actually takes up physical space. We cap all processor queues but the first at a length of 2, which corresponds well with the ratio of aisle length to passenger size.

Multiple Aisles

We model multi-aisle planes as processor sets with multiple pipelines. Using this technique, planes of arbitrary sizes, topologies, and entrance points can be modeled. We describe here the technique for the modeling of a double-aisle plane, such as the Boeing 777.

As in the single-aisle model, all passengers are initially queued at a single processor. For a double-aisle plane, this processor represents the junction point at the entry of the plane. No passengers are assigned seats in this row. From the first processor, a passed passenger may move to either of two different processors. Each of these two processors begins a serial chain of processors akin to a single-aisle plane. Each passenger chooses an aisle based on seat assignment. As in real aircraft, certain rows of the plane are widened so that a passenger can move from one aisle to the other.

Some passengers (for example, those in the middle of a row) have seats equidistant to two aisles; they take the first available aisle and can switch aisles at junction points.

This procedure can be generalized to four-aisle aircraft as well, such as the forthcoming Airbus A380. In that aircraft, not all aisles connect: A passenger cannot move across from an upstairs aisle to a downstairs one.

We can also simulate a plane with the gate in the middle, or with two gates or more, by changing the configuration of processors. Thus, our procedure can be used to simulate any plane.

Assumptions

- All passengers choose the correct aisle, as usually happens, since a steward is positioned at the junction point (i.e., the first processor) to direct traffic. To make this choice easier, the airline could have color-coded boarding passes.
- Only passengers with middle-seat assignments switch aisles.

Deplaning

Our processor-based model can handle deplaning. The processors are reversed: They push passengers from the back of the plane towards the front. Time spent retrieving baggage follows an opposite distribution from the base model: The first passengers must spend more time retrieving luggage than later passengers. Furthermore, there are no seat collisions; everybody clears out of the plane in order. The destination of all passengers during deplaning is the front of the plane.

Assumptions

- Deplaning is uncoordinated. Though some variant of aisle-to-window deplaning is likely the fastest, we believe that any coordinated deplaning method would greatly decrease customer satisfaction. For example, an aisle-to-window deplaning process would cause window seat passengers near the front of the plane to have to wait for virtually the entire plane to disembark. In any case, it is impossible to control the movement of the passengers.

Genetic Algorithm

We used our model above to find the average times taken by various boarding techniques, including back-to-front and window-to-aisle. However, such known orderings may not be optimal.

Since the set of all possible orderings is vast, we need a heuristic to explore parts of the space that interest us the most. This heuristic, if it converges, gives an optimum that—while unlikely to be a global optimum—will be a strong local optimum.

To perform this search, we implement a genetic algorithm, a type of search algorithm that derives the principles of its functioning from evolutionary biology. A genetic algorithm models a solution as a set of "organisms." In our case, an organism is one possible arrangement of passengers in line waiting to board the plane. The algorithm begins with a set of organisms called the *population*. Each organism in the population is run through our processor model, and, based on the time that it takes for all passengers to be seated, given a "fitness" score.

Based on the scores, some organisms are selected to survive, while others die. Organisms with the highest score have the highest survival probability. Organisms that survive are kept in the population, and the others are deleted. The population is replenished by the addition of new organisms. New organisms are either offspring of two surviving organisms from the previous round or randomly generated. The algorithm runs for a set number of generations, at which point it returns the best organism remaining in the pool.

The core of a genetic algorithm is the evolution of the population over time. Over a significant number of generations (for our model, around 60), the algorithm converges. The convergence is a local maximum; the point of convergence is dependent on the initial random population of individuals. The point of convergence is reached using the properties of mutation and crossover.

Mutation and Crossover

Mutation is the process by which an organism changes from one generation to the next. A crossover is the genetic offspring of two individual organisms. We account for both types of evolution in our model.

We first must consider what the genome or "DNA" of our organisms looks like. An organism is a listing of passengers and seats in order (see **Table 1**).

Mutations are relatively simple. During a mutation, a random, sequential section of the DNA is chosen and moved to a different location. A mutation of the initial DNA could look like the bottom row of **Table 1**, which permutes the seat assignments among the passengers.

Crossovers are more complicated. A special property of our solution space is the one-to-one correspondence between passengers and seats. This means that the order of seat numbers in the DNA can be switched, but the seat numbers must stay the same. In normal DNA, a sequential piece of one organism's DNA is exchanged with the corresponding sequence of the other organism. Due to

Table 1.
Example of mutation.

Original organism

Passenger	1	2	3	4	5
Seat number	22A	23C	7A	30F	2B

Mutated organism

Passenger	1	2	3	4	5
Seat number	7A	30F	22A	23C	2B

the one-to-one correspondence property of our data, we cannot use this type of crossover: If the two sequences chosen did not have the same set of seats, our offspring would not have a valid genetic code.

Hence, we formulate a new form of crossover that preserves the elements of a DNA code but changes its order (**Figure 2**).

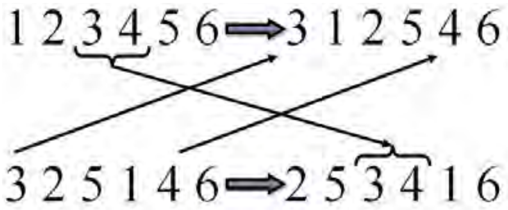

Figure 2. Processor-based model.

The crossover algorithm first chooses a sequence of seats from the genome of the first organism. It then identifies the indices of these seats in the second organism. The genomes of the two organisms are rearranged such that the ordering of the selected seats is switched between the two organisms, while all other seat assignments remain the same. In the example in the figure, the seat sequence (3. . . 4) is selected as the crossover. The indices of (3. . . 4) are 3 and 4 in the first organism and 1 and 5 in the second. After the crossover, the indices of (3. . . 4) are 1 and 5 in the first organism and 3 and 4 in the second. The order of all the other seats remains the same, but their indices are shifted due to the change in location of 3 and 4.

Population Seeding

We ran the genetic algorithm in two configurations, in the first determining the initial population randomly and in the second "seeding" it (adding nonrandom organisms). For seeding, we added two examples of each of the tested types of boarding configuration (e.g., window-to-aisle and back-to-front). Seeding helps the algorithm approach the global maximum, since the beginning population then contains individuals that have high fitness.

The Southwest Model

Model Overview

In the Southwest system, passengers board in order of arrival with no assigned seats.

In our model, seat preferences are encapsulated in a matrix

$$\mathbf{B} = \begin{bmatrix} b_1 & b_2 & \dots & b_6 \end{bmatrix},$$

which represents the spatial arrangement of seats in each row: Elements b_1 and b_6 represent the relative preferences for window seats while elements b_2 and b_5 represent those for middle seats.

The passenger's desire to sit at a given row, to move forward, or to move backward is encapsulated in a transition matrix,

$$\mathbf{P} = \begin{bmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,6} \\ a_{2,1} & a_{2,2} & \dots & a_{2,6} \\ \vdots & \vdots & \ddots & \vdots \\ a_{30,1} & a_{30,2} & \dots & a_{30,6} \end{bmatrix},$$

where **P** satisfies

$$0 \le \mathbf{P}_{i,j} \le 1, \qquad 1 \le i, j \le N;$$

$$\sum_{j=1}^{N} \mathbf{P}_{i,j} = 1, \qquad 1 \le i \le N.$$

Element $a_{i,i}$ represents the passenger's desire to sit in row i. Element $a_{i,i+1}$ represents the passenger's desire to move forward to row $(i+1)$. Element $a_{i,i-1}$ represents the passenger's desire to move back to row $(i-1)$. Although a passenger may prefer to move back a row, that is not possible in our model.

Since **P** is an irreducible aperiodic transition matrix, Markov-chain theory tells us that there is a stationary distribution $\bar{\pi}$, which gives the probability that a passenger will end up at a particular row.

The model incorporates each passenger's decision-making, the passenger's location within the plane, and environmental constraints. In deciding whether to move forward or to sit at the current row, a passenger first considers the current location. If at the end of the plane, there is no option but to sit in the last row. If the number of available seats in front of the passenger's current position exceeds the number of people ahead of the passenger, then the passenger can move forward to the next row; if not, the passenger has to sit in the current row. A passenger cannot move backwards.

As the passenger progresses, preferences need to be adjusted:

- As the passenger moves forward, there are fewer rows to consider.

- As the plane fills, certain rows no longer have available seats to consider.

In both cases, preferences are redistributed so that the relative preferences between all remaining available rows remain the same. Similarly, when seats in a particular row are occupied, the passenger's preference for a particular seat in that row is readjusted so that the relative preferences for available seats remain the same and the sum of seat preferences across the row is 1. Therefore, the preferences for each standing passenger are recomputed each time a passenger finds a seat.

When a passenger gets to a row, the decision of *whether* to sit is governed by a random process that favors the row according to the relative preference that passenger has for that particular row.

After a passenger decides to sit at a given row, if the row contains more than one available seat, his choice of *where* to sit is governed by a random process that favors each seat according to the passenger's relative preference.

From a macro perspective, each passenger makes the decision of where to sit autonomously. This decision is driven, however, by certain preferences and their corresponding probabilities that lend order to the seating sequence in the plane. In each cycle, the model recomputes the preferences of each passenger for each particular row and each seat.

Assumptions

- The movement of passengers along the aisle of the plane is unidirectional. Additionally, passengers are aware of the number of people and available seats in front of them. They will not move forward unless the number of available seats exceeds the number of people in front of them.

- All passengers share a common propensity to sit at any given row or to move forward along the aisle. Because passengers prefer seats closer to the front of the aircraft, the desire to sit at any given row is greater than the desire to move forward.

- All passengers share a common preference for seats, favoring window over aisle and aisle over middle. Having a wall or empty space on one side does not seem terribly unappealing; the window is most preferable because it offers a view and the benefit of resting your head.

- The decision to sit in a particular row is independent of the decision to sit at a particular seat in that row. In most cases, passengers first decide on row preference and then decide which seat they prefer.

- When a row of seats is filled, the probability that a passenger sits in that particular row becomes zero. The probability previously attributed to that row is then redistributed proportionally among the unfilled rows according to the preference probability already attributed to them. This process ensures that the relative preferences of the unfilled rows remain the same.

Boarding Patterns

Although our algorithm may be used to model planes of any size, we focus on a standard 180-person plane with 30 rows and 6 seats in every row.

Random Boarding

This boarding process is used as a baseline for comparison to other models. The process involves the random assignment of seats to passengers in the boarding queue followed by the boarding simulation.

Window-to-Aisle Boarding

Window-to-aisle boarding involves filling up all the window seats, followed by the middle seats, and then the aisle seats. In **Figure 3**, black tiles represent the earliest passengers to enter the plane and white tiles represent later passengers. The darkness of each tile decreases with increasing passenger numbers in the boarding queue.

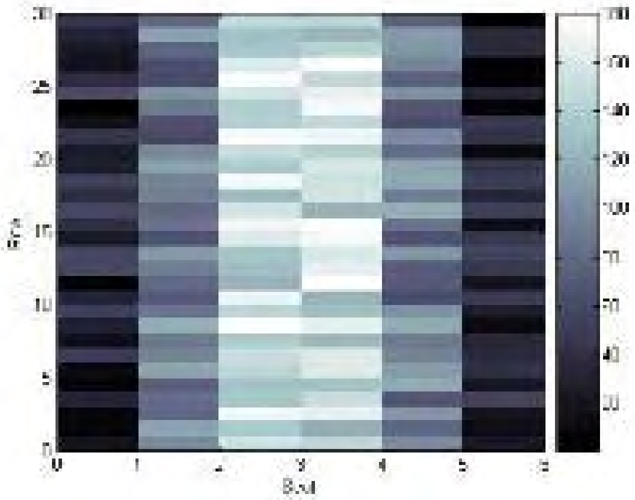

Figure 3. Window-to-aisle boarding.

This "outside-in" method eliminates all seat collisions. The sequence of window seat passengers is random; likewise, the orders of passengers with middle and aisle seats are each independently random. Thus, this boarding pattern still demonstrates significant baggage collisions from passengers interfering with one another's passage to their seat row.

Alternating Half-Rows Boarding

The plane is split into two halves along the aisle and one half is filled before the other half starts boarding. Each half is filled by loading every third row starting from the back. Once we reach the front, the process is repeated from the second-to-last row followed by the third-to-last row (**Figure 4**). The rows are filled in a random order, so there may be seat collisions. Each row must be filled before the next row can start loading. Once passengers in one half of the plane have all boarded, the second half begins boarding with the same process.

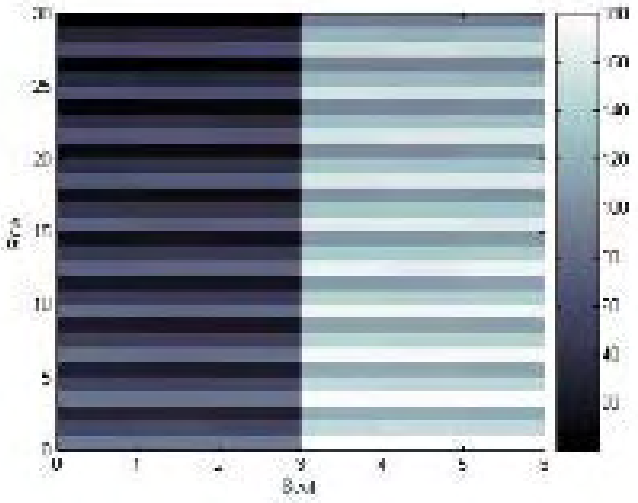

Figure 4. Alternating half-rows boarding.

Zone Boarding

In this boarding pattern, the plane is split into contiguous and evenly divided zones based on row number. The passengers in each zone are then randomly assigned to a seat in each zone. The zone farthest back in the plane boards first, followed by the next furthest, and so on till we reach the front of the plane (**Figure 5**). Passengers in a particular zone must board the plane before passengers in the next zone can begin boarding.

Rotating-Zone Boarding

Rows are filled from back to front in an alternating fashion. Thus, the seats in the back row are filled first, followed by the seats in then front row, the seats in the second to last row, and so on till we reach the middle rows of the plane (**Figure 6**). The seats in each row are assigned randomly and the passengers of a row must board before the passengers for the next row board.

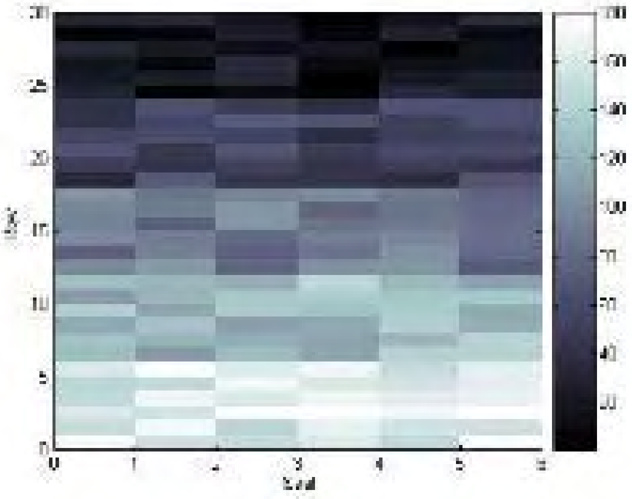

Figure 5. Zone boarding.

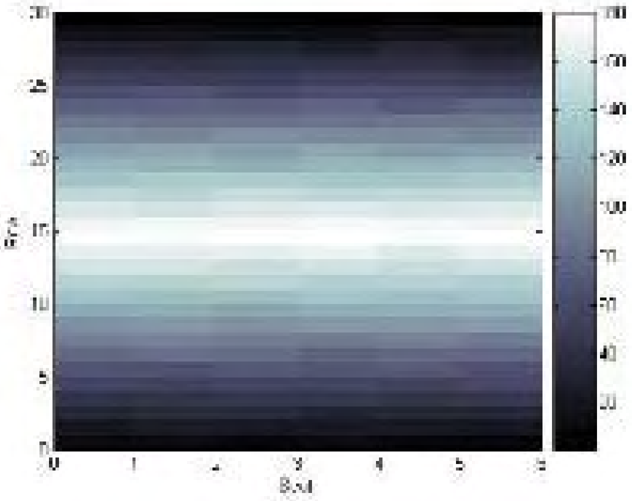

Figure 6. Rotating-zone boarding.

Results

We evaluated the efficiency of each seating pattern by averaging 30 runs of each simulation with 35 trials per simulation on a 180-person plane (30 by 6). Each run used a randomly-generated seating arrangement within the constraints of the pattern. The waiting times were normalized to the average waiting time of a randomly-generated seating arrangement. The normalization value was derived from analysis of 50 different random patterns.

Results are shown in **Table 2**.

Window-to-aisle is the most efficient; it eliminates seat collisions but its randomized column arrangement allows baggage collisions.

Table 2.
Average normalized times for boarding patterns.

Boarding pattern	Time
Deplaning	0.48
Window-to-aisle	0.64
Seeded genetic algorithm	0.67
Alternate half-rows	0.73
Genetic algorithm	0.81
Random	1
Southwest model	1.09
Back-to-front	1.10
Rotating-zone	1.71

Alternate half-rows minimizes spatial overlap between alternating groups of 3 passengers; any seat collision is not large enough (in a spatial sense) to extend to the half-row following. However, this localized congestion also explains why alternate half-rows is slower than window-to-aisle. It is possible that the time for this scenario is overstated, since three passengers walking to a half-row may self organize.

Back-to-front, the most common boarding technique, performs surprisingly poorly—worse than random, due to local congestion propagating to waiting passengers.

Rotating-zone presents collisions of the same sort as alternate half-rows. However, while in half-row boarding there are potentially 6 collisions among 3 passengers, rotating-zone boarding allows for 15 collisions among 6 people.

Southwest boarding suffers because passengers share a preference for seats closest to the exit, which can increase queuing early in the plane, and for aisle seats over middle seats, which also increases seat collisions.

The genetic algorithm applied to a random seating arrangement reached a steady-state solution that is most likely a local minimum for that problem instance. We ran the simulation multiple times, and the results displayed similar properties.

The seeded genetic algorithm resulted in a hybrid between window-to-aisle and alternating half-row boarding. Window seats fill first, followed by the middle and then aisle seats. However, on one side of **Figure 7** there are distinct alternating bands every third row. The algorithm shows a distinct window-to-aisle and alternating half-row hybrid boarding process (**Figure 7**), demonstrating that this hybrid forms a strong local optimum. We observe that the minimum obtained by this hybrid is quasistable. We do not notice any influence from the rotating-zone or back-to-front boarding patterns, indicating that these populations were not as "fit" as the former two and were eliminated from the gene pool. This boarding pattern allows for small families to board together.

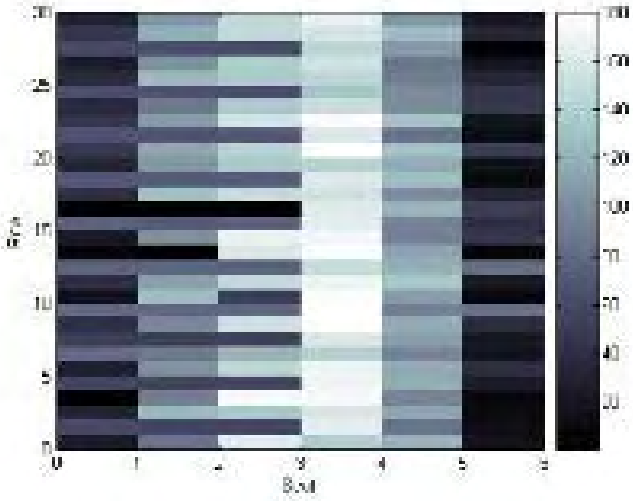

Figure 7. Seeded genetic algorithm.

Deplaning is 25% faster than window-to-aisle boarding and is consequently less useful to optimize.

Sensitivity and Robustness Testing

The robustness of our model is a measure of how it performs in extreme cases; a robust model is one that does not break down in such cases. The sensitivity of our model is a measure of the effect of small parameter changes; a good model should show small changes in response to small parameter changes.

Our model is well behaved; that is, it does not exhibit chaotic behavior. Small changes in parameters demonstrate small changes in results, demonstrating good sensitivity.

Baggage

We eliminated the delay from stowing luggage, a key factor responsible for aisle collisions. We expected that the window-to-aisle boarding would benefit more than alternate half-rows boarding. We observe a 26% decrease in time for window-to-aisle and a 16% decrease for alternating half-rows, consistent with our prediction.

Seat collisions

We eliminated time delays due to seat collisions. We expected that random boarding would perform as well as window-to-aisle, since the primary contribution to delay time will be aisle collisions. Our simulation performs as expected, with only a 2.3% difference in times.

Queuing

We allowed an unlimited queue for each processor. We expected elimination of local congestion and increases in efficiency for zone boarding. Our simulation confirms that zone boarding is 25% more efficient than random boarding.

Discussion and Conclusions

Results

To identify better boarding techniques, we employed a simulation model based on a stochastic agent-based approach. We simulated boarding sequences with embedded stochastic variability, including aisle and row congestion.

We find through simulation that window-to-aisle boarding is the most efficient. However, aisle congestion remains significant due to the random sequencing of passengers within the same boarding group. This in turn contributes to substantial delays due to the stowing of luggage.

Alternate half-row boarding is the next most efficient. Its speed derives from minimization of aisle congestion, despite seat collisions in each half-row.

We could both eliminate seat collisions and minimize aisle congestion by specifying the sequence of each passenger in the boarding queue; but such a method would not be practical, since it would require all passengers to arrive at the gate punctually and gate agents to spend time organizing passengers.

In general, seat collisions have relatively less impact near the end of boarding, because the time to stow luggage increases.

Optimal Recommendation

We found a hybrid between alternate half-rows and window-to-aisle to be a local optimal solution. We recommend hybrid boarding because it offers the versatility of both group and individual boarding. In this solution, the first boarding call is for families and window passengers. Since families self-organize, minimizing collisions, we expect hybrid boarding to be more efficient in practice than predicted by simulation.

Strengths and Weaknesses

Strengths

- Processor-based model has few input parameters, leading to good robustness and sensitivity.
- Genetic algorithm explores and optimizes known configurations.
- Variety of boarding patterns explored, including planned layouts, genetic optimization, and passenger preference

- Accounts for all major factors involved in plane boarding.
- Simulates both boarding and deplaning processes.
- Uses a variety of modeling techniques in an integrated holistic model.

Weaknesses

- Parameters have to be derived from physical occurrences.
- Genetic algorithm has high computational requirements and cannot identify global optimum.
- Does not account for non-uniform preferences among passengers.

Future Work

- Identify at which rows bottlenecks occur for any given time point.
- Investigate efficient deplaning algorithms.
- Better quantify passenger seating preferences

References

Bachmat, Eitan, Daniel Berend, Luba Sapir, Steven Skiena, and Natan Stolyarov. 2005. Analysis of airplane boarding times. http://www.cs.bgu.ac.il/~ebachmat/ORrevisionfinal.pdf .

________. 2006. Analysis of airplane boarding via space-time geometry and random matrix theory. *Journal of Physics A: Math and General* 39 (29): L453–L459. http://www.cs.bgu.ac.il/~ebachmat/jpafinal-new.pdf .

Disney, R., and P. Kiessler, P. 1987. *Traffic Processes in Queueing Networks: A Markov Renewal Approach*. Baltimore, MD: John Hopkins University Press.

Finney, Paul Burnham. 2006. Loading an airliner is rocket science. *New York Times* (14 November 2006). http://travel2.nytimes.com/2006/11/14/business/14boarding.html .

Lawler, Gregory F. 2006. *Introduction to Stochastic Processes*. 2nd ed. Boca Raton, FL: Chapman and Hall / CRC.

van dan Briel, Menkes H.L., J. Rene Villalobos, and Gary L. Hogg. 2003. The aircraft boarding problem. *Proceedings of the 12th Industrial Engineering Research Conference (IERC)*, article number 2153. http://www.public.asu.edu/~dbvan1/papers/IERC2003MvandenBriel.pdf .

Van Landeghem, H. 2002. A simulation study of passenger boarding times in airplanes. http://citeseer.ist.psu.edu/535105.html .

STAR:
(Saving Time, Adding Revenues)
Boarding/Deboarding Strategy

Bo Yuan
Jianfei Yin
Mafa Wang
National University of Defense Technology
Changsha, China

Advisor: Yi Wu

Summary

Our goal is a strategy to minimize boarding / deboarding time.

- We develop a theoretical model to give a rough estimate of airplane boarding time considering the main factors that may cause boarding delay.
- We formulate a simulation model based on cellular automata and apply it to different sizes of aircraft. We conclude that outside-in is optimal among current boarding strategies in both minimizing boarding time (23–27 min) and simplicity to operate. Our simulation results agree well with theoretical estimates.
- We design a luggage distribution control strategy that assigns row numbers to passengers according to the amount of luggage that they carry onto the plane. Our simulation results show that the strategy can save about 3 min.
- We build a flexible deboarding simulation model and fashion a new inside-out deboarding strategy.
- A 95% confidence interval for boarding time under our strategy has a half-width of 1 min.

We also do sensitivity analyses of the occupancy of the plane and of passengers taking the wrong seats, which show that our model is robust.

The UMAP Journal 28 (3) (2007) 371–384.

Introduction

Airline boarding and deboarding has been studied extensively in operations research literature. U.S. domestic carriers lose $220 million per year in revenue for take-off delays [Funk 2003].

We examine strategies for boarding and deboarding planes with varying numbers of passengers, trying to minimize the boarding and deboarding time.

Literature Review

Marelli et al. [1998] designed a computer program called PEDS (Passenger Enplaning / Deplaning Simulation) that used a probabilistic discrete-event simulation to simulate boarding methods. PEDS predicted that it would take 22 min to board a Boeing 747-200. However, the paper did not lay out the boarding procedure.

Van Landeghem [2000] stated that the fastest boarding strategy is individually boarding by seat and row number, and the second fastest is a back-to-front "alternate half-row" boarding system, which was cited to take 15.8 min. He also proposed strategies with small numbers of boarding groups that are both faster and more robust against disturbances. A problem with the data is that only five replications were done for each boarding procedure tested [Pan 2006].

Later, van den Briel et al. [2003] showed that a reverse-pyramid boarding strategy could reduce airplane's turn time by 3-5 min compared to a traditional back-to-front boarding approach. The boarding time depends on events called "interferences."

Unfortunately, all of these researchers used simulation based on small or mid-size airplanes that do not extend to the much larger aircraft under development today. Our approach and results can be applied in all sizes of airplanes.

Basic Assumptions

- **First-class passengers board first.** Hence, our simulation considers only economy-class passengers.

- **Passengers do not try to pass other passengers in the aisle.** The aisles are narrow, so passengers have to wait to move until there are no "obstacles" in front of them.

- **A "call-off" system is used.** Passengers board in ordered groups; gate agents announce which group is to board.

- **A passenger does not take the wrong seat and does not walk past the row of the right seat.** Such mistakes inevitably delay boarding.

Reasons for Boarding Delay

Normal Delay

"Interference" is the main reason for boarding delay. Van den Briel et al. [2003; 2005] divide boarding interferences into two types:

- **Aisle interference:** Since the aisle is narrow enough to allow only one passenger to proceed forward, aisle interference occurs when a passenger stows luggage. To do this, the passenger must stand in the aisle for a moment, thereby acting as an "obstacle" for passengers behind.

- **Seat interference:** This kind of interference occur when a passenger is stalled by another one or two passengers sitting in the same half-row. Because of the limited space between contiguous rows, this passenger must ask these passenger already sitting in their seats to stand up and move into the aisle.

Abnormal Delay

Passengers take the wrong seats, or are late. These behaviors can hardly be avoided. Because of their complexity and variety, we don't take them into consideration. Our main objective is to reduce seat and aisle interference.

Theoretical Estimate Model

We consider boarding time as made up of two parts:

- Free boarding time t_{free}, the total time if all passengers board without any interference or delay.

- Interference time t_{inter}, the total interference time including aisle interference and seat interference.

So the total boarding time is

$$T_{\text{tota l}} = t_{\text{free}} + t_{\text{inter}}, \tag{1}$$

Free Boarding Time

We consider the passengers as a steady flow that pours into the plane at a rate of v_{flow} passengers per minute. So the free boarding time is

$$t_{\text{free}} = \frac{n}{v_{\text{flow}}}, \tag{2}$$

where n is the number of passengers.

Interference Time

Seat Interference

We assume that the times to get from the seat to the aisle and get back are the same, both denoted by t_S. Suppose that three passengers on the same side of a row are assigned to the same boarding group, passengers sitting in positions A (window), B (middle), and C (aisle). There are six equally likely kinds of seat interferences, corresponding to the boarding orders ABC, ACB, BAC, BCA, CAB, CBA. We calculate the interference time for each case. Take ACB as an example: The window-seat passenger boards first, followed by the aisle seat passenger; then the middle-seat passenger needs the aisle-seat passenger to get up and move to the aisle, the middle-seat passenger moves from the aisle to the seat, and the aisle-seat passenger and sits back down again. So the interference time is $3t_S$. The results are shown in **Table 1**.

Table 1.
Seat interference time by boarding order.

Boarding order	ABC	ACB	BAC	BCA	CAB	CBA
Seat interference time	0	$3t_S$	$3t_S$	$5t_S$	$6t_S$	$8t_S$

The average seat interference time for 3 passengers in the same half-row is

$$\bar{t}_S = \frac{25}{6}\, t_S.$$

With n passengers boarding, the total seat interference time is

$$t_{S\text{:inter}} = \bar{t}_S \cdot \frac{n}{3} = \frac{25}{6}\, t_S \frac{n}{3}. \tag{3}$$

Aisle Interference

Let $P_1, \ldots, P_n$ be the passengers in order in the queue, with corresponding row numbers $r_1, \ldots, r_n$. We say P_i *blocks* P_j if $r_i < r_j$. We use the number of blocking times as the number of aisle interference times, that is, when we calculate total interference times, we don't consider the situation that two or more blockings happen together. For example, for passengers $P_1, \ldots, P_5$ in rows 1, 4, 5, 2 and 3, P_1 blocks P_2, P_2 blocks P_3, and P_4 blocks P_5. But actually, after P_1 is seated, P_2 and P_4 can stow luggage simultaneously, and only P_3 and P_5 need to wait (two intervals of interference) to stow luggage. To simplify the calculations, we think of this as a total of three intervals of interference.

As a result, to calculate the aisle interference times, we need calculate only the number of instances of $r_i < r_{i+1}$. Since the order of passengers is random, the number i of aisle interference times is a random variable. We assume that every permutation is equally likely, so the average aisle interference time is

$$I = \frac{1}{n!} \sum i(r_1, \ldots, r_n),$$

where we sum over all permutations. The permutations can be divided into $n!/2$ pairs, each of which is the reverse of the other; together, each pair will have $(n-1)$ instances of $r_i < r_{i+1}$. Hence

$$I = \frac{n-1}{2}.$$

With t_L for the average time to stow luggage, the total aisle interference time is

$$t_{A:\text{inter}} = \frac{n-1}{2} \cdot t_L. \qquad \textbf{(4)}$$

From **(1)–(4)**, we get the total boarding time as

$$T = \frac{n}{v_{\text{flow}}} + \frac{25}{6} t_s \frac{n}{3} + \frac{n-1}{2} t_L.$$

Data Collection

Aircraft of Different Sizes

We base our computer simulations on three types of airplane of different sizes: Airbus A320 (small—124 seats), Airbus A300 (midsize—266 seats), Airbus A380 (large—555 seats).

Experimental Data

We could not collect the needed by experimenting or by interviewing airline executives. Fortunately, this work has already been done by van den Briel et al. [2003] as cited by Pan [2006]. They found the following average times:

- Get-on time (time between gate agent and gate—assuming one gate agent): 9.0 s.
- To advance one row: 0.95 s.
- Stowage: 7.1 s.
- Seat interference time: 9.7 s.

Cellular Automata Simulation Algorithm

In the cellular automata model of boarding analysis, each cell is designated as a passenger, a barrier, a road or a seat. The model restricts individual movements on the plane and computes total boarding time. Time, position, and passenger behavior are each discrete quantities. The passenger compartment is specified as a grid of rectangular cells, while time is incremented using a convenient time step. During one simulation time step (STS), a passenger can move only one cell / row, and all cells representing passengers are processed once and in random order. The simulate iterates time steps and update passengers' state and position until all passengers sit down.

Call-off Function

Before passengers board the plane, they are usually divided by a gate agent into groups, often by consecutive rows, for boarding efficiency. We develop our call-off function with three steps:

1. Divide different seats into groups according to a specific strategy. For example, in implementing outside-in, we put seats in one column into a group.
2. Generate a random order number in each group.
3. Queue the groups consecutively.

Enplane Simulation Function

Simulation of the Next Passenger Boarding

The get-on time has an exponential distribution with mean that we estimate to be 10 STS.

Individual Behavior Judgments

What do passengers do in each time step?

- Stand still when there is an obstacle.
- Move forward by one cell toward the seat when there is free space in front.
- Stow luggage. This behavior needs a counter to record its STS because it requires more than one step.
- Seat interference when the passenger already seated must stand up and let other passengers move in. It also needs a counter.

Simulation Results and Analysis

We simulate common boarding strategies, including random, back-to-front, rotating-zone, outside-in, and reverse-pyramid [Finney 2006]. Back-to-front and rotating-zone allow us to choose the number of rows per group; we try 4, 6, and 8 to see how variation affects the strategies. Similarly, reverse-pyramid can also vary in layers, and we choose 2, 3, and 4 layers to analyze.

Simulation Results

We simulate each boarding strategy 100 times; the results are in **Table 3**.

Table 3.
Simulation results for strategies.

Strategy	Rows (or layers)	Average interference	Seat interference	Aisle
Random		24	72	52
Rows		32	72	55
Back-to-front #1	4	25	72	51
Back-to-front #2	6	25	72	52
Back-to-Front #3	8	25	73	53
Rotating #1	4	25	72	53
Rotating #2	6	25	73	54
Rotating #3	8	25	72	54
Outside-in		23	0	42
Reverse-pyramid #1	2	23	0	43
Reverse-pyramid #2	3	23	0	42
Reverse-pyramid #3	4	23	0	42

Analysis of the Simulation Results

- **The more rows in a group, the shorter the boarding time.** This is really unexpected! Usually, we think that if we divide the passengers into more groups before boarding in accordance with a boarding strategy, the passengers will be better organized and board the plane more efficiently. But to our surprise, our simulations run in the opposite direction. Take back-to-front as an example. When a group contains 8 rows, the boarding time is 24.6 min; but when there are 4 rows per group, the boarding time increases to 25.0 min. With the two extremes (i.e., one row per group vs. all the passenger as a group), the contrast is even more obvious: 32 min vs. 24 min.

  How could this happen? Through analysis of the simulation processes, we find that two or more interferences can happen at the same moment (**Figure 1**) without influencing the boarding process adversely. With more rows in a group, multi-interferences increase but boarding time decreases.

- **Dividing passenger groups according to their columns such as outside-in way and reverse-pyramid way avoids seat interference and reduces aisle interference.** This is easy to understand. If we divide the groups by rows, passengers in the same row get on the plane together, and try to stow their luggage at the same time. However, dividing the group by columns staggers the time when passengers stow luggage into the same overhead bin, which lead to a reduced number of aisle interference.

Optimal Strategy

Based on the above analysis, we draw the conclusion that dividing passenger groups by columns is more efficient than by rows. The two optimal strategies are outside-in (23.0 min) and reverse-pyramid (22.7 min). Although R-P takes a little less time, outside-in is easier to operate both for gate agents and also passengers. Considering this, *we choose outside-in as our boarding strategy*.

Cross-Validation between Theoretical and Simulation Models

We compare the results from the simulation with the results of our analytical mode, where we had total boarding time as

$$T = \frac{n}{v_{\text{flow}}} + \frac{25}{6}\, t_s\, \frac{n}{3} + \frac{n-1}{2}\, t_L.$$

Using parameter value estimates from van den Briel et al. [2003], we have

$$\bar{t_S} = \frac{25}{6}\, t_S = 9.7 \text{ s}, \qquad t_L = 7.1 \text{ s}.$$

We also estimate

$$\frac{1}{v_{\text{flow}}} = 4.5 \text{ s}^{-1}.$$

For the A320, we have $n = 126$, for which we calculate the total boarding time to be 23.2 min, a value that agrees closely with our simulation time.

Mid-size Planes

We extend our simulation model and boarding strategies to midsize aircraft such as the A300; outside-in takes 24.6 min, reverse-pyramid takes 24.4 min.

The A300 has two aisles in economy class, with most (although not all) rows in a 2–4–2 seat configuration. Correspondingly, we adjust our simulation algorithm. Since there are two aisles but only one boarding gate, we divide the passengers into two lines and assume that they don't get into the wrong aisle.

The two strategies are again comparable in average boarding time; again, considering simplicity, we recommend outside-in.

Large Planes

We extend our simulation model and boarding strategies to large aircraft such as the Airbus A380, with two decks and 555 seats in three classes.

Usually, the A380 opens two gates in front of the plane to let passengers board, one of which leads directly to the upper deck (where all business seats are located and a small portion of the economy seats) and the other goes to the main deck (where most economy seats are located).

Since seats in the upper deck are more spread out, it takes less time to board than the main deck. So we consider only the boarding process on the main deck, which is similar to that of a midsize plane, with two aisles and most rows with a 3–4–3 seat configuration. Both outside-in and reverse-pyramid take 26.8 min. We still recommend outside-in.

Luggage Distribution Control (LDC)

A Creative New Boarding Strategy

We offer a brand-new idea to reduce boarding time. During ticket-check time, the passengers are assigned numbers according to how many pieces of luggage they will take onto the plane. Although we do not completely control the order in which passengers board, we can control the distribution of passengers with different amounts of luggage.

A passenger in the last row of the plane blocks nobody when stowing luggage; a passenger in the front row blocks all other passengers behind. Let $P(r)$ denote the probability that a passenger in row r blocks other passengers behind; $P(r)$ is a decreasing function of r. The expected aisle interference time that this passenger causes is

$$t_{A:I} = P(r)t_L,$$

where t_l is the time to stow the luggage.. As for seat interference, it has no direct connection with the row number. We simply define the average seat interference time as $t_{S:I}$. So the total expected interference time is

$$T_{\text{total}:I} = \sum_{r=1}^{n}(t_{A:I} + t_{S:I}) = \sum_{r=1}^{n} P(r)t_L + T_{S:I},$$

where $T_{S:I} = \sum_{r=1}^{n} t_{S:I}$ is a constant.

A passenger with more luggage increases the total. To reduce the effect on interference time, we want to put this passenger as far back as possible.

Simulation Results of LDC

Through simulation, we compare outside-in and reverse-pyramid strategies with our LDC strategy. With our LDC strategy, boarding times for all sizes of aircraft can be reduced by 2–3 min. That is because we send passengers with much luggage to the back of the plane, which reduces the number of interference times.

How to Implement LDC?

Before passengers board, they exchange their ticket for a boarding card with their seat number. Our strategy is to assign seat numbers according to the amount of carry-on luggage. For the distribution of number of pieces of luggage, we use 60% have one piece, 30% have two pieces, and 10% have three.

We divide the seats from back to front in these proportions. We assign to passengers a seat in the group according to number of pieces of luggage; if seats in that group are exhausted, we still follow our basic principle: the more luggage a passenger takes, the farther back the seat.

Orderly Deboarding

Deboarding Strategies

Most airlines conduct deboarding without any organization. As a result, passengers in the front rows can easily get off first, stalling those behind, much like an inverse back-to-front procedure. This process is still faster than boarding. However, if we could adopt a strategy like outside-in, that is, let aisle passengers all get their luggage and get off the plane, then the middle passengers, and finally window passengers, we could fully use the aisle space without interference, leading to higher efficiency.

We put forward the deboarding strategies reversed from boarding strategies: random, front-to-back, inside-out, and V (the strategy derived from the reverse-pyramid boarding strategy).

Deboarding Simulation Model

We develop a simulation model to compare deboarding strategies. Differing from the boarding process, deboarding has its own characteristics, as follows:

- All passengers start in different positions (“their own seat”) and go to the same destination (“outside”).
- There is no seat interference, since in most cases passengers in the same row will leave from aisle seat to window seat.
- In the boarding simulation model, passengers enter the plane one by one, forming a queue. During deboarding, the passengers are a crowd and everyone tries to get out of the plane first.

Rush to One Goal: Object Position

During deboarding, passengers occupy the aisle. We cannot move the passengers according to a certain order, as in the boarding process, but have to consider the conflict that one position is wanted by several passengers. Therefore, we define the concept of *object position*, the position that a passenger wants to get into in the next time step. Our simulation program allows passengers to move forward by one cell in one time step; it can find out passengers' object positions before moving them, determine which passengers want to move to the same object position, determine which passengers cannot move because of obstacles, and then confirm which passengers can move forward and which must stay still. If an object position is wanted by several passengers, we randomly choose one to move and the others have to wait.

Applicability

Our deboarding strategies are to divide passengers into several groups, and then let the groups deboard in order. We define a *PAD (Passengers Allowed to Deboard) set*, a set of passengers allowed to deboard together.

Simulation Results and Analysis for Small Planes

We simulate each proposed deboarding strategy 100 times. Inside-out took 9.9 min, V 10.25, random 12.6, and front-to-back 14.0.

Compared with random and front-to-back, inside-out is better because it makes full use of the whole aisle, while the other two strategies only partly use the aisle. The main reason that we think the V-strategy is no better is that it needs to have more groups and it doesn't make full use of the aisle at the beginning and end of deboarding.

Is there any better strategy? Can inside-out be improved? During deboarding, passengers in the plane can still get their luggage as long as the aisle near their seats is empty. But during boarding, passengers who haven't boarded can do nothing but wait. Considering this, we find that there is no need to let the next group of passengers wait to deboard until the previous group is completely off the plane. We modify our model by changing it to when proportion α of the previous group still remains on board, we allow the next group to start to deboard—our *advanced inside-out strategy*. We find that $\alpha = 15\%$ to 20% yields best results, a deboarding time of about 9.4 min instead of 9.9. There is no need to get an exact optimal value of α, since it will be almost impossible for the flight crew to implement an optimal strategy exactly.

Deboarding with Luggage Distribution Control

If the airline is using our LDC boarding strategy, we already know the distribution of luggage. In this case, our simulation program does not need to judge if a passenger has to get luggage and how long it takes. We simulate the inside-out strategy with different values of α under the luggage distribution given by our LDC boarding strategy. Again, $\alpha = 15\%$ to 20% gives best results. The deboarding time too is reduced by 2–3 min; our LDC strategy can reduce not only boarding time but also deboarding time, because we put the passengers who need less time to get their luggage in the front of the plane. (The optimal value of α is not sensitive to the distribution of luggage.)

Results for Midsize and Large Planes

When we apply the advanced inside-out strategy in midsize and large planes:

- The optimal value of α increases to 20–30%. The reason for this is possibly the increased number of rows in the deck.

Testing of Simulation Models

Are our simulation results reliable? We apply probability theory.

We ran each simulation model 100 times. The times are independent trials from the same distribution. According to the Central Limit Theorem, the sample mean has approximately a normal distribution. As a result, we can make an interval estimate [Rozanov 1969]:

$$T = \overline{X} \pm \frac{s}{\sqrt{n}}\, t_{\alpha/2,n-1},$$

where s is the sample standard deviation and $n = 100$. We choose 95% confidence. We find for each boarding strategy an interval of ±1 min, meaning that our simulation results are reliably consistent.

Sensitivity Analyses

In reality, the boarding and deboarding times are influenced by various random events. Will these factors influence our simulation results?

- Occupancy level below 100%, that is, there will be empty seats. To show how occupancy affects our simulation result, we resimulate the strategies under occupancies from 20% to 90%. Result: If occupancy is more than 90%, there are no distinguishable changes in results with variation in time step size. If it is below 90%, the boarding time will be quite short and therefore affect boarding strategies very little.

- Passengers (especially those flying for the first time) may get into the wrong aisle in a midsize or large plane, which has more than one aisle. So we test strategies under a wrong-aisle possibility of 5%. Result: The boarding time increases by an average of 3 min. That is a long time! Proper guidance from the cabin crew is essential on midsize and large planes.

- Our boarding strategies can be implemented on all kinds of aircraft, because the outside-in strategy divides passengers by columns, so small variability in seat numbers won't affect our boarding strategy much.

Further Discussion

Passing

Our simulation models assume that passengers do not try to pass other passengers in the aisle. But in reality, research indicates that on average, one person in 10 does this.

Boarding Stairs

We assume a boarding bridge, but in reality a boarding stairs may be used (e.g., on the Airbus A380). The difference is that the airport must send a bus to take the passengers from the waiting room to the boarding stairs. Airports want to make full use of the bus and take as many passengers as possible. As a result, boarding in groups according to our strategy is hard to implement. But if the number of passengers in the bus equals the number in each group, we can still adopt our boarding strategy. When they are not equal, we adopt the following boarding strategy: Let R be the number of rows in the deck, with $R = pm + q$, where m is the half-capacity of the bus, p and q are integers, and $q < m$. We implement outside-in for pm rows in front; the other passengers are in one group and get on the plane randomly.

Disobedient Deboarders

Some passengers do not follow directions. We introduce an obedience factor β, the proportion of obedient passengers, picked at random. Disobedient passengers get off the plane if they get the chance, regardless of whether it is their turn. When obedient passengers are less than 40%, any strategy is useless.

Strengths and Weaknesses

Strengths

- We develop a simple theoretical model that gives a rough estimate of airplane boarding time, considering the main factors that may cause boarding delay.

- We design a new boarding strategy that assigns seats according to amount of luggage, which could save about 3 min in boarding.

- With 95% confidence, our simulation results fluctuate by only 1 min.

Weaknesses

- We don't consider the weight balance of a plane. Usually, the passenger and luggage distribution on the plane should be as uniform as possible.
- There are differences in seat configuration between our model and some actual planes.

References

CAAC Resource. 2006. First flight of Airbus A380. `http://www.carnoc.com/css/carnoc2` .

Finney, Paul Burnham. 2006. Loading an airliner is rocket science. *New York Times* (14 November 2006). `http://travel2.nytimes.com/2006/11/14/business/14boarding.html` .

Funk, M. 2003. The visualization of the quantification of the commodification of air travel or: Why flying makes you feel like a rat in a lab cage. *Popular Science* 263 (5) (November 2003): 67–74.

Marelli, Scott, Gregory Mattocks, and Remick Merry. 1998. The role of computer simulation in reducing airplane turn time. *Aero Magazine* (4th Quarter 1998). `http://www.boeing.com/commercial/aeromagazine/aero_01/textonly/t01txt.html` .

Pan, Matthew. 2006. [No title.] `http://www.public.asu.edu/~dbvan1/papers/MatthewPanEssay.pdf` .

Rozanov, Y.A. 1969. *Probability Theory*. New York: Dover.

van dan Briel, Menkes H.L., J. René Villalobos, and Gary L. Hogg. 2003. The aircraft boarding problem. *Proceedings of the 12th Industrial Engineering Research Conference (IERC)*, article number 2153. `http://www.public.asu.edu/~dbvan1/papers/IERC2003MvandenBriel.pdf` .

________, Tim Lindemann, and Anthony V. Mulé. 2005. America West Airlines develops efficient boarding strategies. *Interfaces* 35 (3) (May-June 2005): 191–200.

Van Landeghem, H. 2000. A simulation study of passenger boarding times in airplanes. `http://citeseer.ist.psu.edu/535105.html` .