

# Solving Check-in Counter Constraints with ILOG SOLVER

Hon Wai Chun<sup>1</sup>

City University of Hong Kong  
Department of Electronic Engineering  
Tat Chee Avenue  
Kowloon, Hong Kong  
tel: (852)-2788-7194  
fax: (852)-2788-7791  
email: eehwchun@cityu.edu.hk

---

## Abstract

This paper describes our progress in developing a computerised check-in counter allocation system for the Hong Kong International Airport using a constraint-based approach. It provides an overview of the system architecture, the constraints used during allocation, and the current status of implementation. In Hong Kong, check-in counters are centrally managed and allocated by the Civil Aviation Department. This aviation authority must make daily assignments of check-in counters to airlines or handling agents to service departing passengers. The key problem is to find a satisfactory allocation, given limited check-in counter resources, that can adequately fulfil the requirements of each airline and at the same time meet all other organisational and physical constraints the airport may have. We have developed a system called *SPEEDCHECK* which solves this problem using ILOG SOLVER and ILOG VIEWS.

---

## 1. INTRODUCTION

This paper describes the overall architecture and development progress of a constraint-based check-in counter allocation system called *SPEEDCHECK*. It is being developed for the Hong Kong Kai Tak International Airport using ILOG SOLVER and ILOG VIEWS. This airport is the third busiest international airport in the world, with around 150,000 flights/year and servicing around 75,000 passengers/day. However, due to physical limitations of the airport terminal building, there is only a limited number of check-in counters available for each flight. These counters are managed centrally by the Civil Aviation Department which makes daily assignments of check-in counters to airlines or their designated handling agents. The aviation authority is faced with a daily problem of deciding how many and which counters should be assigned to each departing flight in order to adequately service all the passengers.

Currently, check-in counter allocation at Kai Tak Airport is performed daily by human schedulers. However, with the ever increasing traffic at this airport, the human schedulers find that it takes increasingly more time to develop a schedule that

---

<sup>1</sup> Dr. Chun is also a Principal Consultant of Resource Technologies Limited; a Hong Kong high-tech firm specialising in resource management systems.

can satisfy most of the airport constraints. In addition, there are other factors that are too computationally complex for a human to consider, such as variations in passenger arrival rates and check-in counter service rates which requires statistical analysis, queuing theory, and computer simulation.

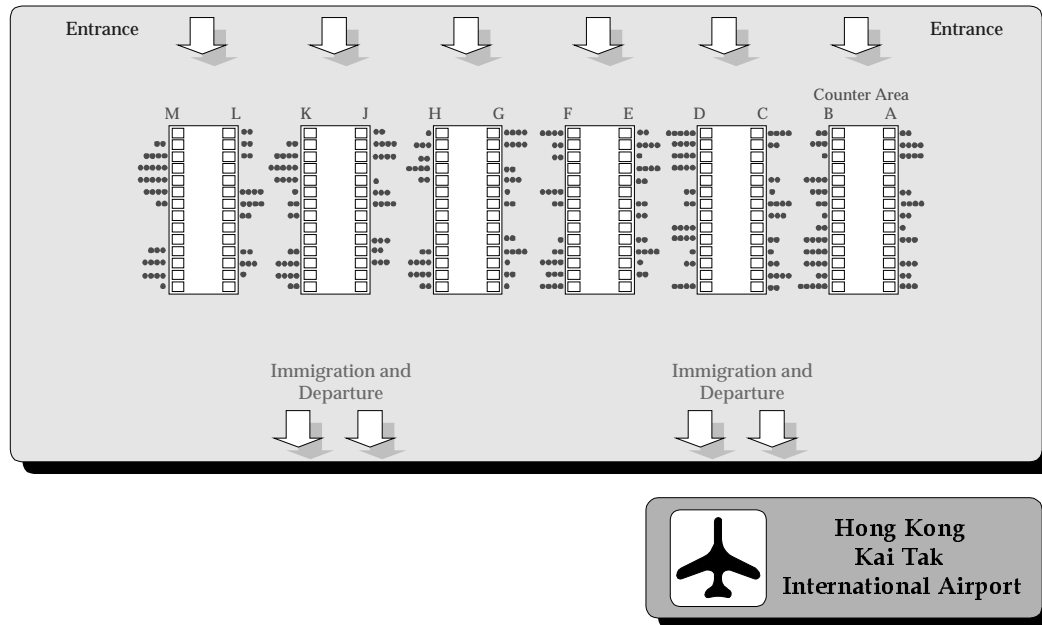


Fig. 1 The check-in counter area within the Kai Tak Airport terminal building.

Figure 1 is a simplified diagram of the airport terminal building. Passengers enter from one side of the terminal (top part of Fig.1) where there are display panels listing which check-in counters are servicing their flights. The check-in counters are organised into rows of check-in areas labelled 'A' to 'M'. The dots in front of the counters are used to illustrate the queuing condition. After check-in, passengers then proceed to the immigration counters through a set of doors on the other side of the building.

## 2. SYSTEM GOALS

The most important concern for the aviation authority is the ability to maintain a consistently high quality of schedule from day to day given the variations in the skill and experience of each human scheduler. The check-in counter allocation schedule has a very important impact to the overall image of the airport as perceived by the passengers. A good schedule will mean shorter queues, less congestion, and faster check-in; and consequently more happy passengers.

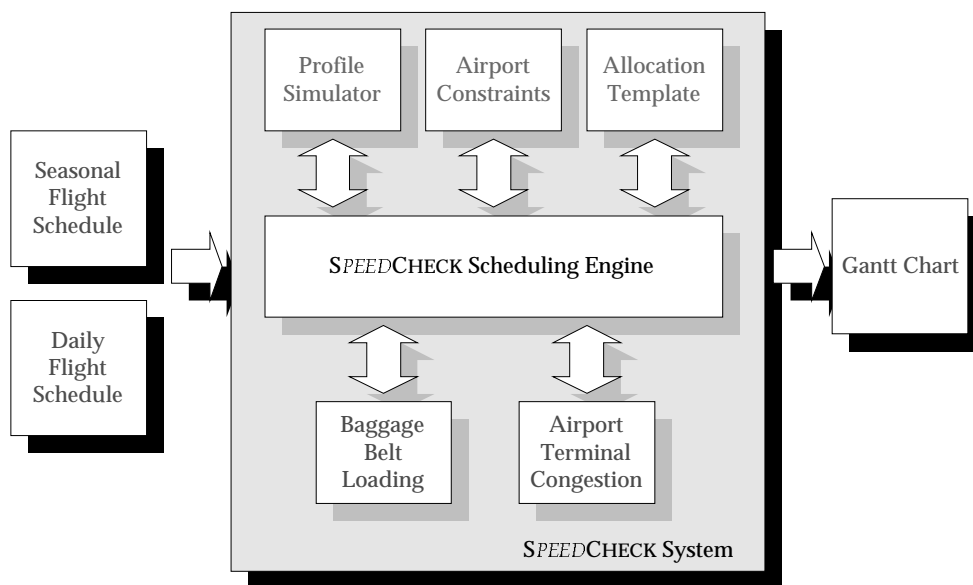
Another goal for this system is to provide better handling of irregular operations such as typhoon conditions where many flights will be delayed. Currently, the amount of decisions to make under these abnormal situations are too much for a human to handle. Reactive scheduling becomes a very important design requirement.

Reducing the amount of time needed in producing a seasonal plan is another major objective. Currently, it takes a human scheduler more than a week's time to produce a seasonal plan which might not even be optimal.

After check-in counters have been allocated by the system, airlines and handling agents must be notified. In addition, the system must also interface to a LED display system that displays messages on top of each check-in counter indicating which flight is being served, the destination, and the passenger class.

### 3. SYSTEM ARCHITECTURE

*SPEEDCHECK* generates a check-in counter allocation in the form of a Gantt chart. Given the daily flight schedule, *SPEEDCHECK* determines, for each departing flight, when counters should be opened, how many counters are needed, and how long counters should operate. It then makes allocation based upon airline preferences, historical statistics, physical restrictions, and other operational constraints. *SPEEDCHECK* was designed with several major components to handle different types of information and to satisfy different system requirements.



**Figure 2.** The overall system architecture of *SPEEDCHECK*.

The *SPEEDCHECK* system can be used to perform long-range planning on a seasonal basis as well as daily scheduling and reactive scheduling. Figure 2 is the overall system architecture of *SPEEDCHECK*. The input to *SPEEDCHECK* can either be a seasonal flight schedule or a daily flight schedule. The seasonal flight schedule is used for long-range planning while the daily flight schedule is used to generate an operational schedule.

There are three main components to the *SPEEDCHECK* system: the profile simulator, the airport constraints, and the allocation template. Two additional modules can be used to improve the allocation results: the baggage belt loading and

the airport terminal congestion modules.

- **The Profile Simulator** - Performs knowledge-based simulation using SIMAN simulation language to determine check-in counter requirements for each type of flight.
- **The Airport Constraints** - The various constraints that are considered during check-in counter allocation are encoded in this module.
- **The Allocation Template** - A master allocation schedule used to maintain allocation consistency from week to week. This template is generated by *SPEEDCHECK* from a seasonal schedule.
- **Baggage Belt Loading Module** - Adjusts check-in counter allocation to uniformly distribute baggage loading and to match the handling capacity of each baggage belt.
- **Airport Terminal Congestion Module** - Adjusts allocation based upon airport physical layout to avoid congestion “hotspots.”

#### 4. CONSTRAINTS

*SPEEDCHECK* represents the airport check-in counter allocation problem as a constraint-satisfaction problem (CSP) [KUMA92]. Although CSP or constraint-programming has a relatively long history [STEE80], with constraint language extensions found in Prolog [COLM90, VANH89], and Lisp [SISK93], it is only recently that constraint-programming became more popular with the availability of the ILOG’s C++ class libraries [PUGE94a]. This library provided a very efficient and clean implementation of constraint-based programming features in a conventional language. Airport constraints in *SPEEDCHECK* are being implemented using ILOG SOLVER.

The *SPEEDCHECK* constraints can be roughly categorised into the following key categories:

- **Proximity** - This constraint can be further divided into several types of proximity requirements:
  - Counters allocated to a flight must be in near proximity to each other; within the same counter area will be ideal.
  - Counters allocated to a flight must be in near proximity to other counters allocated to flights operated by the same airline.
  - Counters allocated to a flight must be in near proximity to other counters being handled by the same handling agent.
  - If possible counters should be allocated close to an airline’s information or ticketing counters.

This constraint makes it more convenient for passengers to locate their check-in counters and also more convenient for the staff members of the handling agent to service these passengers.

- **Preferences** - Airlines, particularly those with many regular flights, may have particular preferences to which area of the airport they would like to be allocated. This is a convenience for those who travel often, such as business persons taking regular flights. It also provides some consistency in working environment for handling agents' staff members.
- **Disfavours** - The opposite of preferences is airline disfavours. For various reasons, an airline might disfavour a particular part of the airport. Or for sensitive political or cultural reasons, an airline might disfavour being assigned check-in counters nearby a particular other airline.
- **Safety** - Due to security reasons, certain airlines may be allocated areas that are further away from the main part of the airport terminal building. For example, airlines that have been threaten by terrorists.
- **Regularity** - The system should make allocation with some regularity over the season. For example, if a flight flies Monday, Wednesday, and Friday, the same set of counters should be allocated to this flight on all three days if possible. Historical patterns should be followed.
- **Queuing Area** - Certain flights to particular destinations may require more queuing area since the party size of non-passengers will be large. These flights should be allocated to areas with more open space.

The *SPEEDCHECK* system was designed so that the user will have full control over the constraints. Each constraint may be turned “on” or “off.” For non-binary constraints, the user may also attach an “importance factor” to indicate how undesirable it is to violate that constraint.

## 5. BAGGAGE BELT LOADING

*SPEEDCHECK* has a module, integrated into the constraint handling system, to balance the baggage loading on the different baggage belts and to ensure that the belt capacity has no been exceeded.

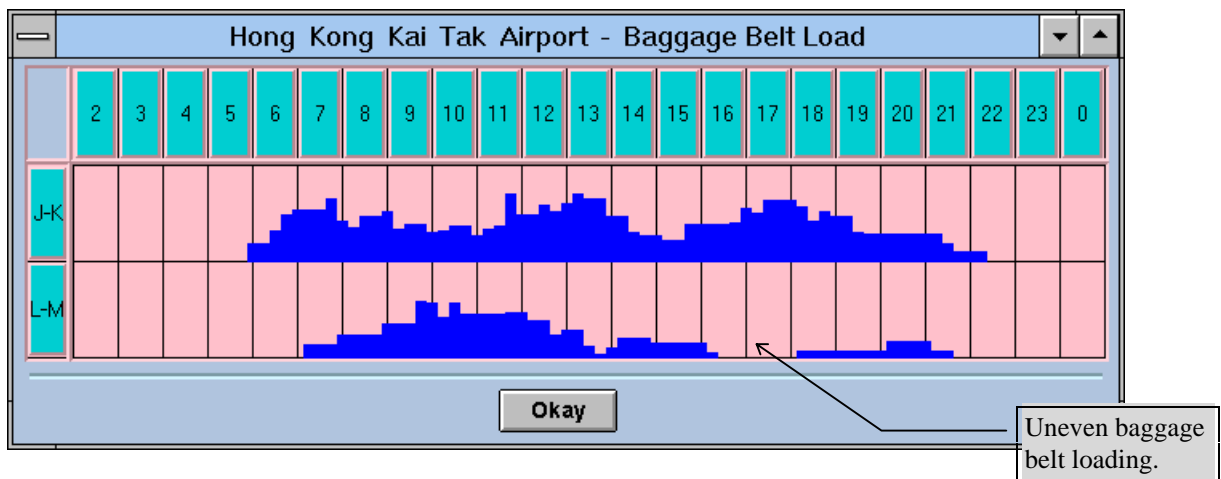


Figure 3. The baggage belt loading without loading control.

In Kai Tak Airport, there are a total of six different baggage belts. Each belt serves two adjacent check-in counter areas. The figures show the baggage belt loading on two of these belts. Figure 3 is the predicted baggage belt loading of an allocation without using the “Baggage Belt Loading” module. As the chart indicates, one of the belt is not being utilised for around 2 hours.

Figure 4 represents the new allocation with the use of the “Baggage Belt Loading” module. As the figure indicates, the previously unused belt is now utilised and the loading on the two belts are more uniformly distributed.

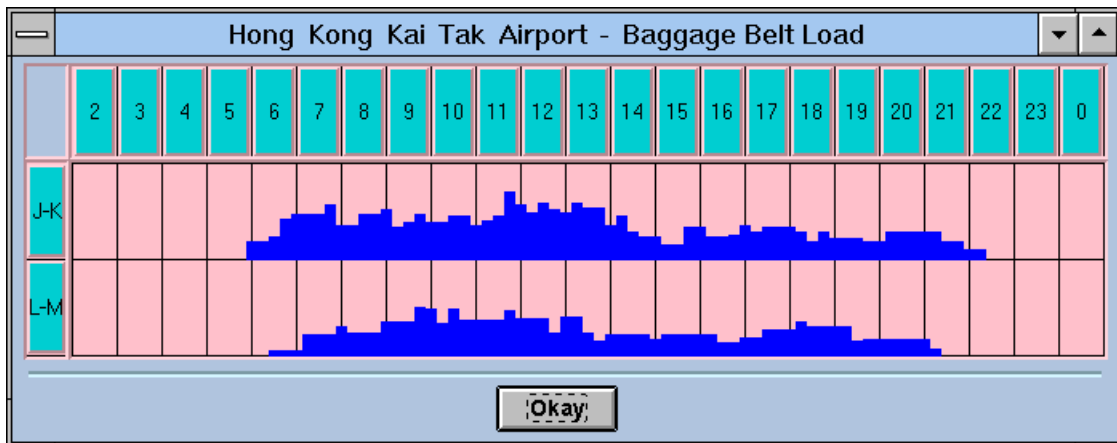


Figure 4. The baggage belt loading with loading control.

The “Airport Terminal Congestion” module offers related feature that tries to uniformly distribute passengers within the terminal building and checks for queuing capacity in different areas of the airport.

## 5. IMPLEMENTATION STATUS

The *SPEEDCHECK* system is scheduled to go into production at the Hong Kong International Airport by the end of 1995. The system is being implemented with ILOG SOLVER and ILOG VIEWS on a Pentium PC running Windows for Workgroup. Microsoft FoxPro provides the database, forms, and report generation facilities. The SIMAN simulation language provides discrete event simulation capabilities.

The current version of the system is being tested in scheduling the most difficult part of the airport - counters that service smaller airlines with irregular flights. So far, the performance has been very encouraging. To generate a whole day's allocation for these counters requires only a few seconds to less than half a minute depending on the complexity of that day's schedule, the number of constraints that were “switched on”, and the reasoning modules that were used. The reactive scheduling performance is roughly one second. On the other hand, it usually takes a human scheduler several hours to one day to draw an allocation chart by hand while considering less constraints than our system.

## 6. CONCLUSION

This paper documents our current progress in designing and developing a computerised check-in counter allocation system for one of the busiest international airports in the world. With the productivity gain in using the ILOG tools, we were able to develop a solution to a relatively complex problem with only a fraction of time normally required for this type of system.

## ACKNOWLEDGEMENTS

The author would like to thank ILOG for their valuable technical support and quick response time - especially when tight deadlines had to be met.

## REFERENCES

- [COLM90] A. Colmerauer, *An Introduction to Prolog III*, Communications of the ACM, 33(7), pp.69-90, 1990.
- [DUNC94] T. Duncan, "Intelligent Vehicle Scheduling: Experiences with a Constraint-based Approach," ILOG Technical Report 94-04.
- [FOX84] M.S. Fox and S.F. Smith, "ISIS: A Knowledge-Based System for Factory Scheduling," In *Expert Systems*, 1(1), pp.25-49, 1984.
- [KUMA92] V. Kumar, "Algorithms for Constraint Satisfaction Problems: A Survey," In *AI Magazine*, 13(1), pp.32-44, 1992.
- [LEPA93] C. Le Pape, "Using Object-Oriented Constraint Programming Tools to Implement Flexible "Easy-to-use" Scheduling Systems," In *Proceedings of the NSF Workshop on Intelligent, Dynamic Scheduling for Manufacturing*, Cocoa Beach, Florida, 1993.
- [MACK77] A.K. Mackworth, "Consistency in Networks of Relations," In *Artificial Intelligence*, 8, pp.99-118, 1977.
- [PUGE94a] J.-F. Puget, "A C++ Implementation of CLP," In *ILOG Solver Collected Papers*, ILOG SA, France, 1994.
- [PUGE94b] J.-F. Puget, "Object-Oriented Constraint Programming for Transportation Problems," In *ILOG Solver Collected Papers*, ILOG SA, France, 1994.
- [SISK93] J.M. Siskind and D.A. McAllester, "Nondeterministic Lisp as a Substrate for Constraint Logic Programming," In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, Washington, DC, pp.133-138, July, 1993.
- [STEE80] G.L. Steele Jr., *The Definition and Implementation of a Computer Programming Language Based on Constraints*, Ph.D. Thesis, MIT, 1980.
- [VANH89] P. Van Hentenryck, *Constraint Satisfaction in Logic Programming*, MIT Press, 1989.
- [WALT72] D.L. Waltz, "Understanding Line Drawings of Scenes with Shadows," In *The Psychology of Computer Vision*, McGraw-Hill, pp.19-91, 1975.