DESIGN AND CONTROL OF LIFT SYSTEMS USING EXPERT SYSTEMS AND TRAFFIC SENSING

R.W.Prowse, T.Thomson, D.Howells
Brunel University, T. Dunwoody & Partners, Lloyds of London

ABSTRACT

We describe an approach to lift traffic analysis and control based on traffic sensing and rule-based expert systems. The system is implemented using standard packages, based on a spreadsheet in the first instance. Simulated input traffic is generated and dynamically linked to the simulator, showing car movements. An expert system linked to a traffic sensing system continuously calculates optimum car movements. Practical aspects of the implementation are discussed.

1 INTRODUCTION

The traditional lift controller is unable to 'see' how many people are waiting in a lobby, and can only estimate how many are in the car, typically by weighing. Optimal strategies for lift control are therefore heavily dependent on statistics, aiming to give satisfactory service at the most demanding periods, and better service at other times. Work on pattern recognition has been proceeding for many years at Brunel, as well as elsewhere, and it is becoming feasible to introduce quantitative estimates of passenger demand at the floors, through a combination of 'people sensing' and voice-based lift call methods.

Given data on passenger demand, it becomes possible to define rules for the assignment of cars to calls, and these rules can be listed as a 'Rule Base'. A 'Rule Interpreter' uses the current states of cars and calls, together with the rules, to determine the optimum car movements. The combination of Rule Base and Interpreter is called a 'Knowledge-Based System'(KBS), since knowledge about what to do i.e. the rules, is separated from the interpreter. This KBS in fact comprises rules that reflect expert experience (in this case in the control of lifts), and the rules may develop as expertise develops. The system is therefore called an 'Expert System'.

Overall then, we can introduce actual passenger numbers into the control strategy, and apply rules which are open to inspection and development. Thus we aim to take some of the 'fuzziness' out of the fuzzy logic that is inherent in any system where situations change rapidly, and where precise data is not easy to obtain, and may not even exist.

2 TRAFFIC SENSING

2.1 People Counting

Automatic counting of people, which is passive, and does not demand any intervention or action from those to be counted, can produce data of profound significance to the optimisation of lift control strategies. Much work is already in progress both in the UK and the USA aimed at monitoring people in other environments, notably for television audience analysis, and in remote monitoring of passenger densities on railway and underground platforms.

The concept of counting people is straightforward when carried out by humans in a stress-free environment. However, attempting to perform recognition and counting of people by computer, presents insuperable problems when executed through the medium of conventional software, since we cannot define the human processes sufficiently precisely to be programmed. People-counting in a crowded lift lobby is even more difficult, and taxes even the human's pattern recognition skills: more powerful assistance is required.

A technology which offers a potential solution to these requirements is the **logical neural network**. The property of a neural network which makes it attractive for people counting is that it can evolve its own functionality or set of primitive rules, when trained by showing it examples of typical inputs, and their corresponding output classification. The network is then able to classify inputs which are perceived as similar to, but not identical to, to those seen during the training. This is known as the **generalisation** property. In the case of a railway platform density detector, the categories might be broad levels of population, e.g. empty, low, medium, high, over-crowded. Interpolation between primary categories enables higher resolution to be achieved.

In applying people-counting to lift controllers, the total numbers of persons will be smaller, and categories of recognition could be the exact numbers of persons in the field of view.

Indoor environments are less variable than outdoor locations, but they cannot be regarded as constant. Variability will always be present in the form of changes in lighting, shadows, camera position, sensitivity due to ageing of electronic components, and other non-deterministic effects. The 'fuzzy' matching performed by neural networks in this environment, is superior to classical image processing techniques. Logical neural networks can be implemented in hardware at low cost, and workers at Brunel have demonstrated neural networks operating on television resolution images in real-time (forty milliseconds to process one television frame).

2.2 Speech Recognition

The other aspect of traffic sensing is that of determining passengers' destinations. Several methods have been proposed, but spoken requests, complemented by people sensing as above, provides the opportunity to achieve close to accurate estimates of passenger destinations before arrival of the lift, hence allowing optimal control of the set of cars.

Speech recognition has been the subject of much research in the past two decades. Early attempts used templates for matching speech objects with rather poor results. 'Dynamic Time Warping' allows for increased variability in the input data, whilst 'Hidden Markov Models' introduce probabilistic functions for the features of speech. Both these methods are computationally expensive. Neural network based speech recogntion however, maintains recognition rates comparable to established methods, but offers improved tolerance to noise and background. This, together with their ability to generalise, allows them to be more speaker independent, a property which is highly attractive for voice operated lift control.

Brunel's current work therefore matches well to the two aspects of people counting and call recognition which are required for effective use of rule-based expert system control.

3 Control and Modelling

3.1 Background

The objective of this phase of the work was to design a lift controller capable of showing the fastest response to the traffic as perceived by the above techniques. A prototype based on simulated movement, with graphical output was considered essential, however, and the software is therefore embedded in a lift model, running on a PC. Nevertheless, with suitable interfacing, it would be possible to achieve actual lift movement control in real time. The work so far relates to use as a model.

The design philosophy was that a 'rule based' system should be used since such systems have the ability to identify quickly essential information from masses of data. The software to be used initially should employ only readily available application packages running on a PC, with the simplest possible rules.

To this end, the system is defined using the 'Excel'^{R} spreadsheet package, and currently the rules are also embedded in the package, but may readily be extracted and incorporated into a separate expert system shell, for example 'Nexpert'^{R}. If required, optimisation can be achieved after prototyping on this model, by transferring the rules into conventional programming language code, for running on a stand-alone controller.

3.2 Two-Car Four-Floor Model

This was chosen as a simple model that would demonstrate principle without incurring the penalties of duplication, since nearly all the rules would have to be developed at an early stage. There are four parts to the complete model, as shown in the Dataflow diagram of Fig.1.

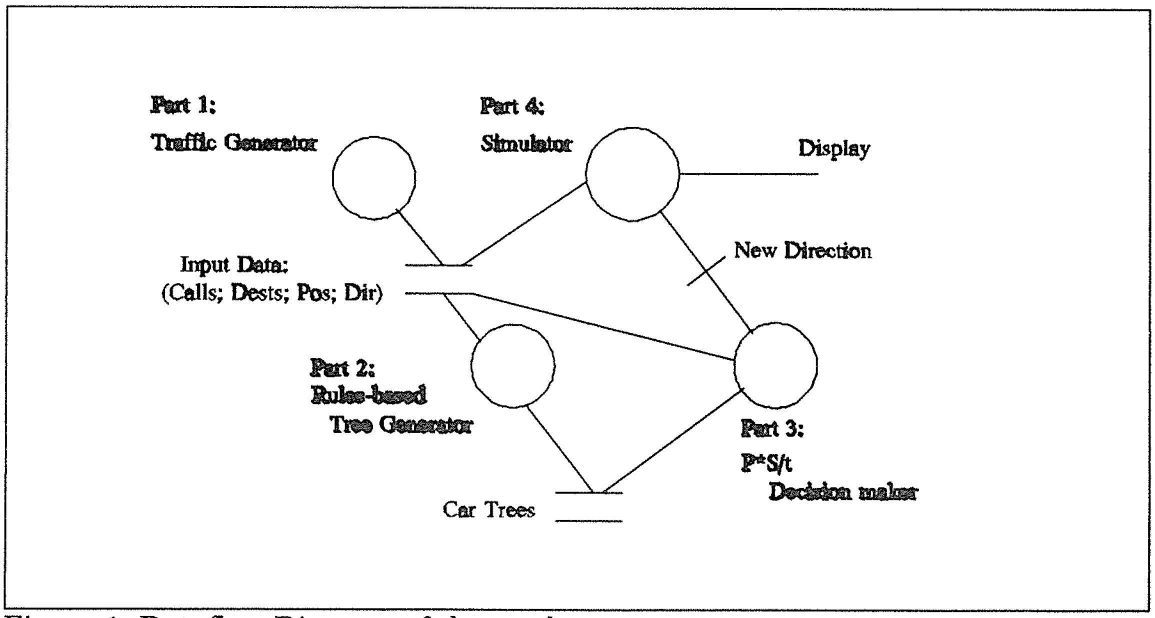


Figure 1: Dataflow Diagram of the total system.

Three of the parts shown are implemented on Excel spreadsheets and the fourth, the display, on Microsoft 'Toolbook'^{R}. To simplify presentation, nine levels are used, with two intermediate levels between each floor. The lifts move one level per sec, no allowance being made for acceleration and deceleration. Time per passenger entering or leaving is assumed to be 1s, with no allowance for door opening and closing times. The method of dealing with these additional parameters is dealt with later.

Part 1 is the input data section, Fig.2.

Floors: Calls						Cars: Calls, Positions & Directions					
						(------Car A------)			(------Car B------)		
	For 4	For 3	For 2	For 1	Level	Call	Dest	Position	Call	Dest	Position
From Level 4				1	9	y	1	.			.
					8			.			.
					7			.			.
From Level 3	1			1	6			.			.
					5			.			. S[2]
					4			.			.
From Level 2	2			1	3	y	4	.	y	2	.
					2			.			.
					1			.			.
From Level 1					0			. S[5]			.

Figure 2: The Input Data Spreadsheet.

This shows people on each floor and their destinations, number of passengers in each lift and lift position and direction at start. Car calls are also recorded. The information would be inserted by the start-up macro and up-dated at appropriate intervals as described below in accordance with the known traffic pattern.

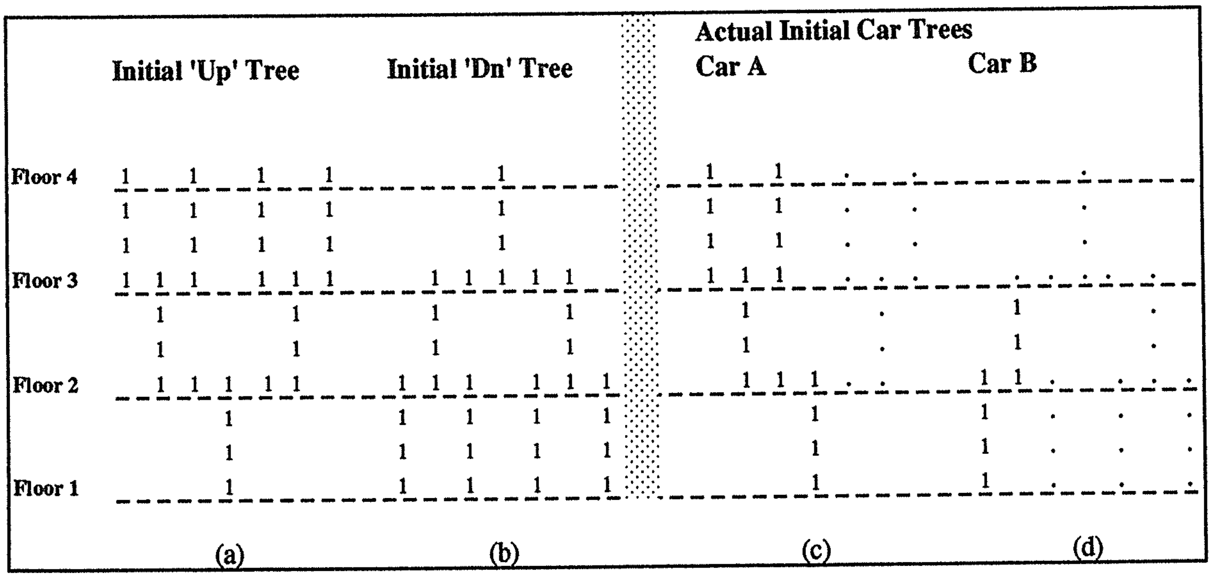


Figure 3: Trees generated by the Rules.

Part 2 comprises a set of rules which converts this data into a tree structure for each lift, as shown in Fig.3. At each intermediate floor the lift may stop, or not stop. A stop is indicated by a left branch in the tree; no stop is indicated by a right hand branch.

The maximum number of ways each lift can move is, therefore, the number of limbs at the end of the trees, shown in Figs 3a and 3b for up and down directions. For a four floor system, starting from floor **1** and moving to **4**, the lift may stop at **2**, **3** and **4**, or **3** and **4**, or **2** and **4** or go straight to **4**, i.e, four possibilities. Figs 3c and 3d show the number of possible journeys with the constraints of the car and floor calls registered in the input matrix. The rules for drawing the tree are contained in the spread sheet cells as follows:-

```
(1)   IF the cell is a left going branch at a junction,
      AND IF the position is a Start
              THEN Enter a marker
      OR
      (IF there is a car call OR there is a floor call),
      AND IF the cell is in the run sequence
              THEN Enter a marker

(2)   IF the cell is a right going branch at a junction, AND
      IF there is no car call, AND
      IF the cell is in the run sequence,
              THEN Enter a marker
```

These are the cells which determine the shape of the trees. The rules entered in other cells simply instruct them to duplicate the preceding cell if in a run sequence.

TIME	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
LEVEL	0	1	2	3	3	3	3	3	3	3	4	5	6	6	7	8	9
Passengers	5	5	5	5	4	3	2	1	2	3	3	3	3	4	4	4	4
P*S		5	10	15	15	15	15	15	15	15	18	21	24	24	28	32	$\boxed{36}$

TIME	12	13	14	15
LEVEL	6	7	8	9
Passengers	3	3	3	3
P*S	24	27	30	$\boxed{33}$

TIME	0	1	2	3	4	5	6	7	8	9
LEVEL	5	4	3	3	3	3	2	1	0	0
Passengers	2	2	2	1	0	1	1	1	1	0
P*S		2	4	4	4	4	5	6	7	$\boxed{7}$

TIME	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
LEVEL										0	1	2	3	4	5	6	6	7	8	9
Passengers										0	0	0	0	0	0	0	1	1	1	1
P*S										7	7	7	7	7	7	7	7	8	9	$\boxed{10}$

Figure 4: **P*S/t** calculations.

Part 3 of the system is the spreadsheet calculation to select the best combination of routes from the trees to move the traffic shown in the input matrix of part 1. The optimising algorithm used is **P*S/t** where **P** is number of passengers, **S** represents stages moved by each passenger and **t** is time elapsed. The structure of the calculation is shown in Fig.4.

The tables in the figure show the time interval in seconds for movement between levels, and for passengers to enter and leave. Car **A** would complete its run in the UP direction in 16s if it stops at level 6 (floor 3) to allow one passenger to enter, or in 15s if it does not stop at level 6. The alternative calculation is triggered by an IF statement taking effect when level 6 is reached in the main table:

> IF the right going branch cell in the tree of possible paths has a marker,
> THEN begin a new table in which the lift does not stop.

Meanwhile a similar, effectively simultaneous, calculation is made for Car **B**, shown in the lower two parts of the table. This shows that Car **B** completes its DOWN run in 8s and is available to move again in 1 further second. The original input floor call information, however, is no longer valid for **B**'s new movement, having been altered by **A**'s stop at level 3. A new input matrix is required, and a new set of trees. Note that the original matrix and tree set must still be maintained as Car **A** is still working to this pattern in advance of Car **B**. The relevant rules are:

> IF a Car, say **B**, is moving in the same direction as **A**,
> AND IF **B** is behind **A**
> THEN Generate for **B** a new calls matrix when any affected level is reached.

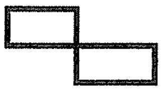
Floors: Calls	For 4	For 3	For 2	For 1	Car A	Car B
From Level 4						
From Level 3	1					
From Level 2						
From Level 1						

Figure 5: Call Matrix for matching call to car.

The new matrix is shown in Fig.5. The associated tree shows two possibilities for dealing with the call:

> Car **A** can stop at floor 3 (level 6).
>
> This is of course part of the original calculation and need not be repeated.
>
> The alternative is for Car **B** to be assigned the call, generating the **P*S** value shown in the lower part of Fig.4.

The new value for Car **B** must be added to the value taken by **A** in the complementary route, where **A** did not stop at floor 3. The total is 10 + 33 in 19s, to be compared with 36 + 7 in 16s. The better choice is now clear and the selected paths are indicated to the fourth part of the model which is a Toolbook simulation of movement. To make the model respond to a continuously changing traffic pattern, it is necessary to input the new data to the input matrix, so that a new set of paths may be chosen. This may be done at every stop made by either lift. In this way, lift movement is continuously optimised.

3.2 Practical Implementation of the Model

To allow realistic figures to be used, it is necessary a) to improve the simulation, and b) to allow for more floors and cars.

3.2.1 The Simulation

The simulation can be improved simply by increasing the number of interfloor levels in Fig.4. In addition, the time axis before and after a floor will be given greater definition, e.g. after 11s, the next column could read 11.2s, then 11.4s, etc. The altered time and distance scales would be initiated by IF statements at an appropriate level, and would allow settings of acceleration and deceleration, and door opening and closing times. The actual scale values could be selected automatically depending on the input information.

3.2.2 The Controller

The ability of the system to perform as a real time controller depends crucially on the time taken to calculate the **P*S/t** values. For the simple model shown, the worst case calculation time (both lifts stopping at all floors, 6 passengers leaving, 6 entering at the intermediate floors) is less than 1s on a PC/386. This will allow real-time simulations to be undertaken for a number of applications of interest, since the tree calculations need only be repeated on calling at floors. The lift movement performance is dependent on the number of levels between floors. Fast simulation is better performed on optimised software, where there is less emphasis on graphical output, and more on overall analysis.

The main development from this simple system, of course, is the modelling of lift systems with many floors, and many cars. A 20-floor building, with upwards of 4 cars would generate enormous trees. However, the behaviour of people, in the future, knowing their state in the present, rapidly becomes unpredictable. There is therefore no point in attempting to predict optimal car movement beyond a few calls beyond the present one. The result is that trees can be drastically pruned, and are not allowed to grow beyond a very limited number of branches.

Each car in practice has a tree which comprises only those paths off the 'no-stop' branch which involve up to four stops, as selected by the designer. The Controller therefore looks ahead to determine which of the possible car patterns will give optimal traffic flow for that limited time. There is no chance that lone passengers may be kept waiting at distant floors, because any car on its way in that direction will eventually have that call in its tree as it services the calls in front. Whether a car travelling in the opposite direction will change direction to service the call, depends on its completing all car calls, and being ready to reverse. If so, then the action is determined as usual by the relative **P*S/t** values.

4. CONCLUSIONS

The principle on which this work is based, is that of using advanced pattern recognition networks, already under development for similar applications. This allows the control strategy to be based on complete and up-to-date information about passenger requests. Since the information is updated at each decision point, and new trees are formed, the decision making process constantly reviews its previous decisions and thereby retains an optimal overall performance.

The team is now undertaking research to evaluate 'people-counting' methods, and alternative lookahead strategies, and to investigate further parameters which may be taken into account, since the information is available. The most obvious factor is current waiting time. Given the knowledge of passengers' arrival times in the lobby, the rules can be revised to give a weight to different calls depending on their waiting time so far.

5. ACKNOWLEDGEMENTS

The assistance and interest of Professor T.J. Stonham and Dr B. Wilkie of the Pattern Recognition Group at Brunel University is gratefully acknowledged.