Strategic Resource Planning for Optimum Service Quality Provided by Gail Lochtie, British Telecom

John Perram, Dave Wood, David Allwright, Marit Schou,
Malwina Luczak, Alan Zinober, Tom Felici, Andy Waugh,
Daniel Scrase, Robert Kretzscher
Plus occasional contributions from
Colin Please (non-linear regression), Sean McKee (queueing theory).

April 1999

1 Outline of the Problem

British Telecommunications Plc would like to optimise the performance of their procedure for scheduling installation and repair of telephone lines and equipment for both residential and business customers.

The company sets goals, in terms of percentage completion of different types of services within a specified time period for each job type. For the purposes of the problem presented here, there were six job types, four of which involved appointment times, and two of which had to be completed within a specified time delay from the job being first reported. BT wants to achieve these goals by assigning priorities to the different types of service. Incoming jobs are then scheduled for completion by available engineers based on these priorities using a central piece of software. The priorities p_i for job type i are normalised so that $p_1 + \ldots + p_k = 1$, where in our case $k = 6$.

The mathematical problem is thus quite simple: find the "best" values of these priorities to achieve the specified completion goals.

2 More detailed Analysis.

Requests for installation or repair of telephones comes from either business or residential customers according to a complicated distribution which is approximately periodic over a week, with the heaviest load on Mondays, declining steadily through the course of the week and weekend. In the case of installations, an appointment time is agreed with the customer; other services are provided on a deadline basis (i.e. the service has to be completed by a certain time).

Scheduling is currently carried out centrally (within a given region) using a proprietary piece of software called Works Manager. This seeks to assign current jobs to available engineers using a simulated annealing algorithm using some objective function which depends on the values of the priorities.

Roughly speaking this function takes the distance that each free engineer is from the queued jobs and scales this distance according to the priority of those jobs. The engineer then attends that job which is closest to him in terms of this new 'distance' function. The scheduler is run about every half an hour as new jobs arrive and current jobs are completed.

Throughout the scheduling, travel time is considered to be negligible.

In order to simulate job allocation to check scheduling algorithms and the effects of changing job priority values, BT use a simulation code which takes A set of data gathered over a three week period and computes goal compliance from input values of the priorities. A number of strategies to attempt to optimise the performance of this code over priority values, but the major constraint was one of time since to run the code for a virtual three week period took several hours of real computer time which severely limited the number of potential vectors in priority space which we could test.

3 Notation and Analysis

We are therefore in the situation where we are attempting to optimise an extremely complicated function which maps priorities for each of the jobs to the goals achieved (percentage of jobs of each time completed within the specified time). This function will contain hundreds of iterates over the virtual three week run, and the final result is a statistical measure of the data sets thus produced. In particular there is no way that any form of derivative of this function can be derived and so we are forced to try alternative approaches to those that one may hope to apply.

The first approach was to investigate if there was a linear relation between the goals achieved and the priorities which were used to achieve them. It was hoped that as a first approximation this would give quite a reasonable fit on which to base further investigation.

This can be done by least square fitting of a relation of the form:

$$A\mathbf{p} = \mathbf{m}$$

where $\mathbf{p}$ is a vector containing the priorities and $\mathbf{m}$ is a vector containing the goal compliances. When we have six, $k = 6$, job types, the matrix A is found from $k = 6$ representative values (found from numerical simulation) of the priorities and goals as the solution of the equations

$$AU = V$$

Where U and V are the matrices with column vectors given by the representative priority vectors and their respective goal compliance. Provided that the priority vectors are suitably chosen, U is inevitable and our first Choice for A is simply given by

$$A = U^{-1}V.$$

If the relation between priorities and goal compliance can indeed be well represented by a linear relationship, then a better set of priorities will now be given by the solution of

$$\hat{\mathbf{p}} = A^{-1}\hat{\mathbf{m}}$$

where $\hat{\mathbf{m}}$ contains the desired goals. These new priorities are then fed into the simulator. As new values of the goal compliance are obtained, they are used to update the matrix A.

This algorithm seems to produce better values, but has not been pursued further because it was felt that convergence would be slow. In fact, some preliminary Mathematica experiments seemed to indicate that the algorithm doesn't converge at all, and even generates negative priorities. In addition, the components of the improved priority vectors **p** seldom summed to one and an ad-hoc normalisation step had to be added to the algorithm. Note that these equations are still valid if the **p** vector contains other basis functions than linear ones.

Another approach which was applied to the problem, with more encouraging results, was to apply some multivariable optimisation technique using as an objective function the sum of the squares of the negative deviations from the goals.

A candidates here was the simplex algorithm, which does not require gradients, due to Nelder and Meade (see for example [4]). In addition the simplex algorithm has the advantage that if the six input priorities are a probability measure (elements sum to unity), then so is the next iteration.

The basic simplex algorithm can be summarised by considering the optimisation of points in R^3. An initial four points are chosen which, for sake of argument, geometrically represent the corners of a tetrahedron (see Figure 1. The values of the function in question are then found for each of these four points. One of these points will have a higher value (error) associated with it, and this becomes the point to be moved for the next iteration. There are several ways of choosing a new point to replace this one, but the simplest is to take the reflection of this point in the plane defined by the other three points of the tetrahedron. The next iteration then starts with this modified set of four numbers.

Numerical results found during the Study Group show a slow improvement in performance.

Preliminary calculations were also performed on non-linear regression. Because of the few data points, only a few correlations could be tried in the quadratic model. Correlations were found by analysing the covariance matrix of the data. Those chosen appeared to reflect the prejudices of the group as to which jobs should be correlated due to the multi-skilling of the engineers. Optimisation of the regression model led us back close to the original estimate in which all the priorities were equal, and which is still the best value found in terms of its objective function.

In addition to these numerical approaches, some attempt was made to analyse the Situation more rigorously. In particular an analysis of a two job scenaio was derived.

4 Theoretical Analysis for a Reduced Model.

We assume that customers (requests for repair) arrive as a Poisson process to a single exponential channel and that upon arrival to the system each unit will be designated to be a member of one of two priority classes (corresponding to two types of requests). Further, Poisson arrivals of the first class have mean rate λ_1 and those of the second class have mean rate λ_2, such that $\lambda = \lambda_1 + \lambda_2$. When queues of customers from both classes are nonempty, then the server picks a customer from class 1 with probability p_1 and a customer from class 2 with probability p_2 ($p_1 + p_2 = 1$). The service time of each customer is exponential with rate μ (it is possible to generalise to class-dependent mean, i.e., μ_1, μ_2) and independent of service times of all other customers. The special case when $p_1 = 1$

is the case when the first-priority items have the right to be served ahead of others, but there is no pre-emption.

A system steady state balance equations may be established for

$$\pi_{m,n,r} = \mathbf{Pr}\{\text{in steady state, } m \text{ units of priority 1 and } n \text{ units of priority 2 are in the system, and a unit of priority } r = 1 \text{ or } 2 \text{ is in service}\}.$$

These take the following form:

$$\pi_{m,n,1}(\lambda+\mu) = \lambda_1\pi_{m-1,n,1} + \lambda_2\pi_{m,n-1,1} + \mu p_1(\pi_{m+1,n,1} + \pi_{m,n+1,2}) \quad m > 1, n > 0$$

$$\pi_{m,n,2}(\lambda+\mu) = \lambda_1\pi_{m-1,n,2} + \lambda_2\pi_{m,n-1,2} + \mu p_2(\pi_{m+1,n,1} + \pi_{m,n+1,2}) \quad m > 0, n > 1$$

$$\pi_{1,n,1}(\lambda+\mu) = \lambda_2\pi_{1,n-1,1} + \mu p_1(\pi_{2,n,1} + \pi_{1,n+1,2}) \quad n > 0$$

$$\pi_{m,1,2}(\lambda+\mu) = \lambda_1\pi_{m-1,1,2} + \mu p_2(\pi_{m+1,1,1} + \pi_{m,2,2}) \quad m > 0$$

$$\pi_{m,0,1}(\lambda+\mu) = \lambda_1\pi_{m-1,0,1} + \mu(\pi_{m+1,0,1} + \pi_{m,1,2}) \quad m > 1$$

$$\pi_{1,0,1}(\lambda+\mu) = \lambda_1\pi_0 + \mu(\pi_{2,0,1} + \pi_{1,1,2})$$

$$\pi_{0,n,2}(\lambda+\mu) = \lambda_2\pi_{0,n-1,2} + \mu(\pi_{0,n+1,2} + \pi_{1,n,1})$$

$$\pi_{0,1,2}(\lambda+\mu) = \lambda_2\pi_0 + \mu(\pi_{0,2,2} + \pi_{1,1,1})$$

$$\pi_0\lambda = \mu(\pi_{1,0,1} + \pi_{0,1,2})$$

It should be clear that $\pi_0 = 1-\rho$, where $\rho = \frac{\lambda}{\mu} = \rho_1 + \rho_2$, and that

$$\pi_n = \sum_{m=0}^{n-1}(\pi_{n-m,m,1} + \pi_{m,n-m,2}) = (1-\rho)\rho^n \quad n > 0.$$

Also, since the percentage of time the system is busy is ρ, the percentage of time it is busy with a type-r job will be $\rho\lambda_r/\lambda$, so that

$$\sum_{m=1}^{\infty}\sum_{n=0}^{\infty}\pi_{m,n,1} = \frac{\lambda_1}{\mu} = \rho_1 \quad \text{and} \quad \sum_{m=0}^{\infty}\sum_{n=1}^{\infty}\pi_{m,n,2} = \frac{\lambda_2}{\mu} = \rho_2.$$

However, obtaining a reasonable solution to these stationary equations is a very difficult matter, even in the special case $p_1 = 1$. The most one can do comfortably is obtain expected values via two-dimensional generating functions.

We define

$$H_1(y,z) = \sum_{m=1}^{\infty}\sum_{n=0}^{\infty} y^m z^n \pi_{m,n,1} \quad H_2(y,z) = \sum_{m=0}^{\infty}\sum_{n=1}^{\infty} y^m z^n \pi_{m,n,2}$$

$$H(y,z) = H_1(y,z) + H_2(y,z) + \pi_0.$$

Note that $H(1,1) = 1, H_1(1,1) = \rho_1, H_2(1,1) = \rho_2, \pi_0 = 1-\rho = 1-\rho_1-\rho_2$. Then $H(y,z)$ is the joint generating function for the two classes, regardless of which type is in service. Note that $H(y,y) = \pi_0/(1-\rho y)$, since $H(y,z)$ collapses to the generating function of the M/M/1 queue when $z = y$ and thus no class distinction is made. Hence if L_1 and

L_2 are used to denote the mean number of customers present in the system for each of the two priority classes, then

$$\frac{\delta H(y,z)}{\delta y}|_{y=z=1} = L_1 = L_{q1} + \frac{\lambda_1}{\mu} = \lambda_1 W_1$$

and
$$\frac{\delta H(y,z)}{\delta z}|_{y=z=1} = L_2 = L_{q2} + \frac{\lambda_2}{\mu} = \lambda_2 W_2,$$

where L_{q1} and L_{q2} are the respective mean queue lengths, and W_1 and W_2 are the respective mean waiting times.

If we multiply the balance equations by appropriate powers of y and z and sum accordingly, we get

$$(1+\rho-\rho_1 y-\rho_2 z-\frac{p_1}{y})H_1(y,z) = \frac{p_1}{z}H_2(y,z) + \pi_0\rho_1 y - p_1\sum_{n=0}^{\infty}\pi_{1,n,1}z^n$$

$$-\frac{p_1}{z}\sum_{n=0}^{\infty}z^{n+1}\pi_{0,n+1,2} + \frac{p_2}{y}\sum_{m=0}^{\infty}\pi_{m+1,0,1}y^{m+1} + p_2\sum_{m=0}^{\infty}\pi_{m,1,2}y^m - p_2(\pi_{1,0,1}+\pi_{0,1,2}),$$

$$(1+\rho-\rho_1 y-\rho_2 z-\frac{p_2}{z})H_2(y,z) = \frac{p_2}{y}H_1(y,z) + \pi_0\rho_2 z - p_2\sum_{m=0}^{\infty}\pi_{m,1,2}y^m$$

$$-\frac{p_2}{y}\sum_{m=0}^{\infty}y^{m+1}\pi_{m+1,0,1} + \frac{p_1}{z}\sum_{n=0}^{\infty}\pi_{0,n+1,2}z^{n+1} + p_1\sum_{n=0}^{\infty}\pi_{1,n,1}z^n - p_1(\pi_{1,0,1}+\pi_{0,1,2}).$$

Summing over n the equations involving $\pi_{0,n,2}$ and summing over m equations involving $\pi_{m,0,1}$, we obtain

$$\sum_{n=0}^{\infty}\pi_{1,n,1}z^n = \sum_{n=1}\pi_{0,n,2}z^n(1+\rho-\rho_2 z-\frac{1}{z}) + \pi_0(\rho-\rho_2 z),$$

$$\sum_{m=0}^{\infty}\pi_{m,1,2}y^m = \sum_{m=1}\pi_{m,0,1}y^m(1+\rho-\rho_1 y-\frac{1}{y}) + \pi_0(\rho-\rho_1 y).$$

Hence

$$(1+\rho-\rho_1 y-\rho_2 z-\frac{p_1}{y})H_1(y,z) = \frac{p_1}{z}H_2(y,z) + p_1\pi_0(-\rho+\rho_1 y+\rho_2 z)$$

$$-\sum_{n=1}^{\infty}\pi_{0,n,2}z^n(1+\rho-\rho_2 z)p_1 + \sum_{m=1}^{\infty}\pi_{m,0,1}y^m(1+\rho-\rho_1 y)p_2,$$

and
$$(1+\rho-\rho_1 y-\rho_2 z-\frac{p_2}{z})H_2(y,z) = \frac{p_2}{y}H_1(y,z) + p_2\pi_0(-\rho+\rho_1 y+\rho_2 z)$$

$$+\sum_{n=1}^{\infty}\pi_{0,n,2}z^n(1+\rho-\rho_2 z)p_1 - \sum_{m=1}^{\infty}\pi_{m,0,1}y^m(1+\rho-\rho_1 y)p_2.$$

It can be calculated from these (by putting $y = z = 1$) that

$$\sum_{m=1}^{\infty} \pi_{m,0,1} = \frac{\rho_1(1+\rho p_1)}{1+\rho_1 p_1+\rho_2 p_2}$$

$$\sum_{n=1}^{\infty} \pi_{0,n,2} = \frac{\rho_2(1+\rho p_2)}{1+\rho_1 p_1+\rho_2 p_2}.$$

Hence we finally get

$$H_1(y,z)\left(\frac{p_2}{y} - \frac{z}{p_1}(1+\rho-\rho_1 y-\rho_2 z-\frac{p_1}{y})(1+\rho-\rho_1 y-\rho_2 z-\frac{p_2}{z})\right)$$

$$= \pi_0(\rho-\rho_1 y-\rho_2 z)(p_2+z(1+\rho-\rho_1 y-\rho_2 z-\frac{p_2}{z})$$

$$+\sum_{m=1}^{\infty} \pi_{m,0,1} y^m (1+\rho-\rho_1 y)\left(p_2 - \frac{p_2}{p_1} z(1+\rho-\rho_1 y-\rho_2 z-\frac{p_2}{z})\right)$$

$$+\sum_{n=1}^{\infty} \pi_{0,n,2} z^n (1+\rho-\rho_2 z)\left(-p_1 + z(1+\rho-\rho_1 y-\rho_2 z-\frac{p_2}{z})\right)$$

and

$$H_2(y,z)\left(\frac{p_1}{z} - \frac{y}{p_2}(1+\rho-\rho_1 y-\rho_2 z-\frac{p_1}{y})(1+\rho-\rho_1 y-\rho_2 z-\frac{p_2}{z})\right)$$

$$= \pi_0(\rho-\rho_1 y-\rho_2 z)(p_1+y(1+\rho-\rho_1 y-\rho_2 z-\frac{p_1}{y})$$

$$+\sum_{m=1}^{\infty} \pi_{m,0,1} y^m (1+\rho-\rho_1 y)\left(-p_2 + y(1+\rho-\rho_1 y-\rho_2 z-\frac{p_1}{y})\right)$$

$$+\sum_{n=1}^{\infty} \pi_{0,n,2} z^n (1+\rho-\rho_2 z)\left(p_1 - \frac{p_1}{p_2} y(1+\rho-\rho_1 y-\rho_2 z-\frac{p_1}{y})\right).$$

Thus we see that $H_1(y,z), H_2(y,z)$ satisfy relations of the form $H(y,z)f(y,z) = g(y,z)$, where $f(1,1) = 0$. Therefore to obtain the first derivatives of $H_1(y,z)$ and $H_2(y,z)$ at $y = z = 1$, we are forced to differentiate these relations twice, which gives at point $(1,1)$,

$$2H_y f_y + H f_{yy} = g_{yy}$$

$$H_y f_z + H_z f_y + H f_{yz} = g_{yz}$$

$$2H_z f_z + H f_{zz} = g_{zz}.$$

Here

$$f_1(y,z) = \frac{p_2}{y} - \frac{z}{p_1}(1+\rho-\rho_1 y-\rho_2 z-\frac{p_1}{y})(1+\rho-\rho_1 y-\rho-2z-\frac{p_2}{z})$$

$$f_2(y,z) = \frac{p_1}{z} - \frac{y}{p_2}(1+\rho-\rho_1 y-\rho_2 z-\frac{p_1}{y})(1+\rho-\rho_1 y-\rho-2z-\frac{p_2}{z})$$

$$g_1(y,z) = \pi_0(\rho-\rho_1 y-\rho_2 z)\left(p_2+z(1+\rho-\rho_1 y-\rho_2 z-\frac{p_2}{z})\right)$$

$$+A(y)(1+\rho-\rho_1 y)\left(p_2-\frac{zp_2}{p_1}(1+\rho-\rho_1 y-\rho_2 z-\frac{p_2}{z})\right)$$

$$+B(z)(1+\rho-\rho_2 z)\left(-p_1+z(1+\rho-\rho_1 y-\rho_2 z-\frac{p_2}{z})\right)$$

$$g_2(y,z)=\pi_0(\rho-\rho_1 y-\rho_2 z)\left(p_1+y(1+\rho-\rho_1 y-\rho_2 z-\frac{p_1}{y})\right)$$

$$+A(y)(1+\rho-\rho_1 y)\left(-p_2+y(1+\rho-\rho_1 y-\rho_2 z-\frac{p_1}{y})\right)$$

$$+B(z)(1+\rho-\rho_2 z)\left(p_1-\frac{p_1 y}{p_2}(1+\rho-\rho_1 y-\rho_2 z-\frac{p_1}{y})\right),$$

where $A(y)=\sum_{m=1}^{\infty}\pi_{m,0,1}y^m$, $B(z)=\sum_{n=1}^{\infty}\pi_{0,n,2}z^n$.

This is a set of six linear equations in six unknowns. The unknowns are $\frac{\delta H_1}{\delta y}|_{y=z=1}$, $\frac{\delta H_1}{\delta z}|_{y=z=1}$, $\frac{\delta H_2}{\delta y}|_{y=z=1}$, $\frac{\delta H_2}{\delta z}|_{y=z=1}$, $\frac{dA}{dy}|_{y=1}$, $\frac{dB}{dz}|_{z=1}$. We can calculate the coefficients of the system from

$$f_{(1)y}|_{y=z=1}=\frac{\rho_1}{p_1}-1, f_{(2)z}|_{y=z=1}=\frac{\rho_2}{p_2}-1,$$

$$f_{(1)z}|_{y=z=1}=\frac{\rho_2-p_2}{p_1}, f_{(2)y}|_{y=z=1}=\frac{\rho_1-p_1}{p_2},$$

$$f_{(1)yy}|_{y=z=1}=2(1+\rho_1-\frac{\rho_1^2}{p_1}), f_{(2)zz}|_{y=z=1}=2(1+\rho_2-\frac{\rho_2^2}{p_2}),$$

$$f_{(1)yz}|_{y=z=1}=(1-\rho_2)(\frac{\rho_1}{p_1}-1), f_{(2)yz}|_{y=z=1}=(1-\rho_1)(\frac{\rho_2}{p_2}-1),$$

$$f_{(1)zz}|_{y=z=1}=\frac{2\rho_2}{p_1}(1+p_2-\rho_2), f_{(2)yy}|_{y=z=1}=\frac{2\rho_1}{p_2}(1+p_1-\rho_1),$$

$$g_{(1)yy}|_{y=z=1}=2\rho_1^2(1-\rho)+\frac{2\rho_1 p_2}{p_1}\left[(1+\rho_2)A'(1)-\rho_1 A(1)\right]$$

$$g_{(2)zz}|_{y=z=1}=2\rho_2^2(1-\rho)+\frac{2\rho_2 p_1}{p_2}\left[(1+\rho_1)B'(1)-\rho_2 B(1)\right]$$

$$g_{(1)yz}|_{y=z=1}=(1-\rho)\rho_1(2\rho_2-1)+\frac{p_2}{p_1}(1-\rho_2)\left[\rho_1 A(1)-(1+\rho_2)A'(1)\right]$$

$$+\rho_1\rho_2 B(1)-\rho_1(1+\rho_1)(B(1)+B'(1))$$

$$g_{(2)yz}|_{y=z=1}=(1-\rho)\rho_2(2\rho_1-1)+\frac{p_1}{p_2}(1-\rho_1)\left[\rho_2 B(1)-(1+\rho_1)B'(1)\right]$$

$$+\rho_1\rho_2 A(1)-\rho_2(1+\rho_2)(A(1)+A'(1))$$

$$g_{(1)zz}|_{y=z=1}=2(1-\rho)\rho_2(\rho_2-1)-A(1)(1+\rho_2)\frac{2p_2\rho_2}{p_1}$$

$$+2(1-\rho_2)\left[B'(1)(1+\rho_1)-\rho_2 B(1)\right]-2B(1)\rho_2(1+\rho_1)$$

$$g_{(2)yy}|_{y=z=1}=2(1-\rho)\rho_1(\rho_1-1)-B(1)(1+\rho_1)\frac{2p_1\rho_1}{p_2}$$

$$+2(1-\rho_1)\left[A'(1)(1+\rho_2)-\rho_1 A(1)\right]-2A(1)\rho_1(1+\rho_2)$$

Note that

$$A(1) = \frac{\rho_1(1+\rho p_1)}{1+\rho_1 p_1+\rho_2 p_2} \quad B(1) = \frac{\rho_2(1+\rho p_2)}{1+\rho_1 p_1+\rho_2 p_2}.$$

In the special case where $p_1 = 1$, one gets

$$H(y,z) = \frac{(1-y)\pi_0}{1-y-\rho y(1-z-\lambda_1 y/\lambda+\lambda_1 z/\lambda)}$$

$$+\frac{(1+\rho-\rho z+\rho_1 z)(z-y)B(z)}{z[1+\rho-\rho_1 y-\rho_2 z][1-y-\rho y(1-z-\lambda_1 y/\lambda+\lambda_1 z/\lambda)]};$$

$$B(1) = \frac{\rho_2}{1+\rho_1}.$$

Next one can take partial derivatives of H with respect to both y and z, and then evaluate at $(1,1)$ to find the means L_1 and L_2. It turns out that the exact functional relationship for $B(z)$ (or both $A(y)$ and $B(z)$ in the general case) is not needed: in the special case, it is enough to know $B(1)$ (generally, one calculates $A(1), B(1), A'(1), B'(1)$ from the equations). The final results when $p_1 = 1$ are [2]

$$L_1 = \frac{\rho_1(1+\rho_2)}{1-\rho_1}, L_{q1} = \frac{\rho\rho_1}{1-\rho_1}, W_{q1} = \frac{\rho}{\mu-\lambda_1},$$

$$L_2 = \frac{\rho_2(1+\rho\rho_1-\rho_1)}{(1-\rho)(1-\rho_1)}, L_{q2} = \frac{\rho\rho_2}{(1-\rho)(1-\rho_1)}, W_{q2} = \frac{\rho}{(1-\rho)(\mu-\lambda_1)}.$$

In fact, using the theory of multidimensional birth-death processes, Miller [2] has shown that the actual probabilities for priority-1 customers are

$$\pi_{n_1} = (1-\rho){\rho_1}^{n_1} + \frac{\lambda_2}{\lambda_1}{\rho_1}^{n_1}\left(1-\frac{\rho_1^{n-1}}{(1+\rho_1)^{n_1+1}}\right) \quad n_1 \geq 0.$$

As expected, when $p_1 = 1$ class 2 customers wait longer than class 1 customers. Also, as $\rho \to 1$, $L_2, L_{q2}, W_{q2} \to \infty$, while the corresponding means for the first priority approach finite limits. First priority expectations only go to infinity when $\rho_1 \to 1$. The average number in queue is $L_q = L_{q1} + L_{q2} = \rho^2/(1-\rho)$ (the same as for the nonpriority case), and the average unconditional queueing time $W_q = (\lambda_1/\lambda)W_{q1} + (\lambda_2/\lambda)W_{q2}$ is the same as that for the nonpriority case.

Similar results have been obtained for the generalisation of the model with $p_1 = 1$ to the case where the two classes are served at rates μ_1 and μ_2, respectively [3]. Here

$$L_{q1} = \rho_1\hat{\rho}\frac{\lambda_1/\lambda+\lambda_2/\lambda(\mu_1^2/\mu_2^2)}{1-\rho_1}$$

$$L_{q2} = \frac{(\lambda_2/\mu_1)\hat{\rho}}{1-\rho_1}\frac{\lambda_1/\lambda+(\lambda_2/\lambda)(\mu_1^2/\mu_2^2)}{1-\rho}, \quad \hat{\rho} = \lambda/\mu_1.$$

Again, Miller [2], using the theory of multidimensional birth-death processes, displayed the actual probabilities for priority-1 customers as

$$\pi_{n_1} = (1-\rho){\rho_1}^{n_1} + \frac{\lambda_2}{\lambda_1+\mu_2-\mu_1}\left[{\rho_1}^{n_1} - \frac{\mu_1{\lambda_1}^{n_1}}{(\lambda_1+\mu_2)^{n_1+1}}\right] \quad n_1 \geq 0,$$

where $\rho_i = \frac{\lambda_i}{\mu_i}$, $\rho = \rho_1 + \rho_2$. Unfortunately, his approach does not seem to generalise to the case when the probabilities of serving classes one and two, that is, p_1, p_2 $(p_1 + p_2 = 1)$ respectively, can be both non-zero.

4.1 Optimisation of the M/G/1 Queue with Postponable Priorities

The first part of this section follows [5]. Arrivals of each class form independent Poisson processes. There are k priority classes; customers in class i have arrival rate λ_i and service distribution G_i, where S_i denotes a draw from G_i and $\rho_i = \lambda_i E(S_i)$. Class 1 has highest priority, class 2 next highest, etc., where on each service completion, the next customer is drawn from those in queue having the highest priority. The overall arrival rate is $\lambda = \sum_{i=1}^k \lambda_i$, while S is a service distribution drawn from the overall service distribution $G = \sum_{i=1}^k \lambda_i G_i/\lambda$, and $\rho = \lambda E(S) = \sum_{i=1}^k \rho_i$. Service times are independent of each other and of the arrival process, customers within the same class are served FIFO, and, once service begins, each customer is served to completion without interruption (called the *postponable* case). We denote by d_i, Q_i, etc., the averages with respect to customers in class i.

Little's Law holds for the system [5], that is,

$$L = \lambda W,$$

where L is the stationary average number of customers in the system (in the queue plus being served), W is the average waiting time in the system (includes the service time). Also,

$$Q = \lambda d,$$

is a similar relation for the number in queue and queueing delay; and

$$L_s = \frac{\lambda}{\mu},$$

where L_s is the proportion of the time the server is busy.

The system is *conservative* (that is, no work is created or destroyed within). Let V be the total remaining work in the system. Then its average can be represented as

$$E(V) = \sum_{i=1}^{k} \rho_i d_i + \lambda E(S^2)/2 = \sum_i Q_i S_i + \lambda E(S^2)/2,$$

and we know $E(V) = const = \lambda E(S^2)/2(1-\rho)$ [5].

Since arrivals are Poisson, the average work found by an arrival is $E(V)$. Furthermore, since work is conserved, we have [5]

$$E(V) = \lambda E(S^2)/2(1-\rho).$$

For the highest class,

$$d_1 = \rho_1 d_1 + \lambda E(S^2)/2,$$

where the second term represents the remaining service of whoever may be in service. Hence $d_1 = \lambda E(S^2)/2(1-\rho)$. When $k = 2$, one can now solve for d_2.

For $k \geq 3$, for any class i, combining all classes of higher priority than i into a new "super" class 1 will not affect d_i. Thus as far as class k is concerned, there are only two classes, k and higher than k. Hence

$$d_k = \lambda E(S^2)/2(1 - \sum_{j<k} \rho_j)(1-\rho).$$

The general expression is

$$d_i = \lambda E(S^2)/2(1 - \sum_{j<i} \rho_j)(1 - \sum_{j\leq i} \rho_j).$$

One could now ask the following question: suppose customers arrive in identifiable classes with known service distributions. How should these classes be assigned priorities so as to minimise overall cost of delay, i.e., $C = \sum_{i=1}^{k} c_i \lambda_i d_i / \lambda$. It is easy to see from the above that the $c\mu$-rule is optimal: order classes by the product of the cost rate and service rate, with the highest being assigned class 1, and so forth. By the above, $\sum_i \rho_i d_i = const = \rho d_F$, independent of the priority rule, where d_F is the average delay under FIFO.

Thus we want to minimise $\sum_{i=1}^{k} c_i \lambda_i d_i / \lambda$ subject to $\sum_i \rho_i d_i = const = K$.

Suppose $c_1 \mu_1 \geq c_2 \mu_2 \geq \ldots$. In the case of two priority classes, we minimise

$$c_1 \lambda_1 d_1 + \frac{c_2 \lambda_2}{\rho_2}(K - \rho_1 d_1) = \frac{c_2 \lambda_2 K}{\rho_2} + \frac{d_1 \lambda_1}{\mu_1}(c_1 \mu_1 - c_2 \mu_2).$$

Since $d_1 \geq d_1 \rho_1 + \lambda E(S^2)/2$, the first class should be given priority.

However, one can see that the rule remains optimal with respect to overall delay cost also in our generalised model. That is, it is optimal in comparison with rules with general p_1, p_2 $(p_1 + p_2 = 1)$. To see this, consider the case when we have only two classes, and assume $c_1 \mu_1 \geq c_2 \mu_2$. Then we want d_1 as small as possible subject to $d_1 \geq \rho_1 d_1 + \lambda E(S^2)/2$ and $\rho_1 d_1 + \rho d_2 = const$. Hence $d_1 = \lambda E(S^2)/2(1 - \rho)$. When there are more classes, we proceed by induction. Classes $2, \ldots, k$ are combined into one and compared to the first class. Hence again the first class should have priority, and so forth.

We have only analysed optimisation with respect to overall delay cost. The case of deadlines is different and much harder to analyse. In fact its discrete, finite version, where n jobs are to be processed on a single machine so as to minimise the cost of failing to meet deadlines, is NP-complete (so-called "minimum tardiness", or more generally, "minimum weighted tardiness" problem). If d_i is the mean delay of class i customers, T_i is the allowed lapse time between the arrival and departure of a type i customer from the system, and c_i is the cost of the tardiness of class i per unit time, then the cost function becomes

$$\sum_{i=1}^{k} \frac{c_i \lambda_i}{\lambda} [d_i + (1/\mu_i) - T_i]_+,$$

where $[x]_+ = \max\{x, 0\}$. Then, for various sets of T_i, λ_i, c_i, rules with general probabilities $p_1, \ldots, p_k$ of picking customer of a given class may perform better than rules that put an absolute (but postponable) priority ordering on classes. When there are two classes, we have shown how to calculate d_i. However, further investigation is required to extend our analysis to a general number of priority classes.

5 Conclusions and Further Work

For the numerical approach two avenues of attack have been pursued with disappointing results and one with more promising ones. The hope that the data could be represented by a linear model in a sufficiently small region near the optimum foundered on the size of the sufficiently small region. Classical optimisation seems to converge rather slowly, so given the essential serial nature of optimisation and the time it takes to compute the objective function, is unlikely to be useful. Preliminary results involving optimising a non-linear regression model seem promising.

Although the above is more or less what could be accomplished with the knowledge, and data, available and the need not to duplicate the mathematics clearly contained in both the scheduler and the simulator, the question remains as to what else could be done to help decision making. It has been suggested that we could have designed a toy queuing system which contained the salient features of the problem domain and which would be subject to analytic treatment. This would greatly increase the speed of generating data points, the main bottleneck, given that it takes the simulator about 20 minutes to generate a point.

Another possibility is to try non-linear regression techniques. This would take much more data than could be generated at the study group, but would have the advantage that the optimisation could be carried out on the regression model rather than using the simulator. The input data could be generated rather quickly in parallel on a network of work stations.

The analytical treatment of the problem has lead to a promising start, although to extend the analysis so far carried out, without even considering the full problem with which BT are interested in, would be a major undertaking.

References

[1] D. Gross, and C. M. Harris. *Fundamentals of Queueing Theory.*[New York, Wiley, 1998].

[2] D. R. Miller. Computation of Steady State Probabilities for M/M/1 Priority Queues. *Operations Research,* 29:945-958, 1981

[3] P. M. Morse. *Queues, Inventories, and Maintenance.* [New York, Wiley, 1958].

[4] W. H. Press, B. P. Flannery, S. A. Teukolsky, W. T. Vetterling. *Numerical Recipes in C.* [Cambridge University Press, 1989].

[5] R. Wolff. *Stochastic Modelling and the Theory of Queues.* [Englewood Cliffs, Prentice Halls, 1989].

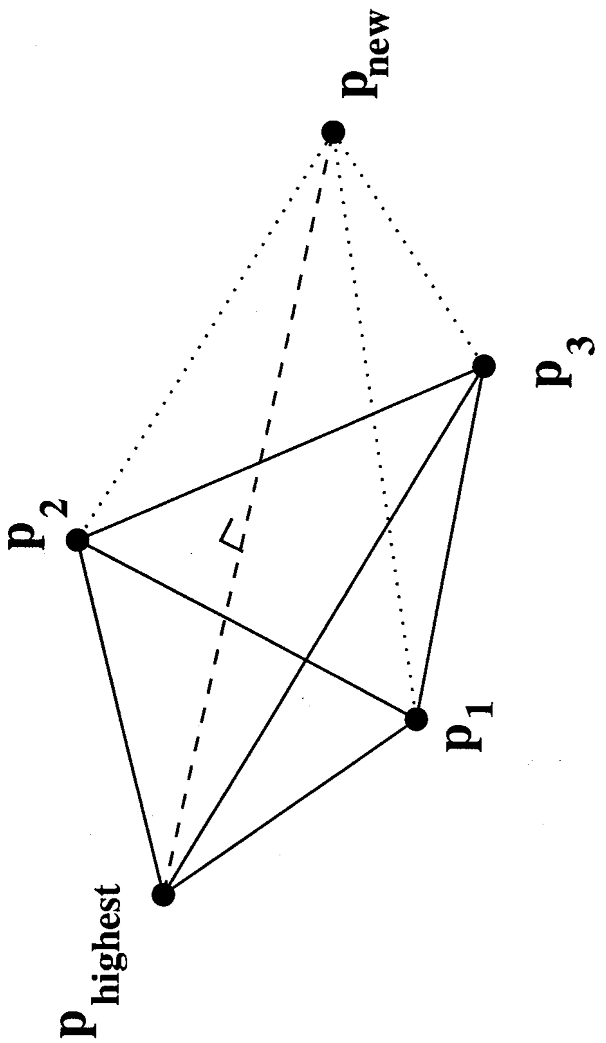

Figure 1: Simplex Algorithm: the lowest valued point $p_{highest}$ is reflected in the plane defined by the remaining three points p_1, p_2, p_3 to give it's replacement p_{new}. The value of the function of this point is then evaluated here and the algorithm repeated.