there from other nodes of the component. It is natural to term such components 'non-progressive'.

The components for the statistical-mechanical models of Chapters 5 and 6 were non-progressive in that all their transitions, both internal and external, were reversible. The archetype of a progressive component is a linear system of depôts. That is, units arriving at node 1 pass independently and successively through nodes 1, 2, . . . , m at the rates indicated in Fig. 9.6.1. The equilibrium distribution is easily found to be

$$\pi(n) \propto \frac{\nu^N \mu^{-n_m}}{n_m!} \prod_{j=1}^{m-1} \frac{\lambda_j^{-n_j}}{n_j!} \tag{7}$$

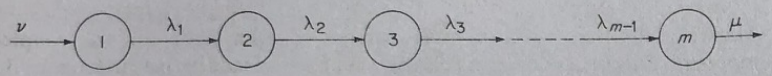

Fig. 9.6.1 A one-way linear system; the archetype of a progressive component.

The pair (N,n_m) is thus a sufficient statistic for the coupling parameters, but N itself is not.

Exercises and comments

1. Generalize the last example by considering a composite component with a single input node ($j = 1$, say) and a single and distinct output node ($j = m$, say) which is such that the output node cannot feed back into the component, but only to the outside. Show then that the coupling parameters appear in $\pi(n)$ only through a factor $\nu^N \mu^{-n_m}$, so that N and n_m are sufficient.

2. We shall see in the next chapter that a 'Jackson component' seems not to be able to represent a first-come first-served queue. The linear progressive system of Fig. 9.6.1 does represent an approximation to such a queue, however. While one unit can certainly overtake another in the system, the order of emission will be close to the order of arrival, the more so as the number of stages increase for a given expected transit time. If the transition $n \to n - e_j + e_{j+1}$ is given rate $\lambda_j n_j / N$ rather than $\lambda_j n_j$ (cf. Exercise (4.4)) then expression (7) is multiplied by $N!$ and the distribution of total number in the queue is geometric. However, the distribution of waiting time is not that of the single-server first-come first-served queue, and individuals are indeed sharing service at all stages.

7. THE OPTIMAL DESIGN OF A JACKSON NETWORK

Sections 7–9 are concerned with optimization and constitute something of a diversion from the main theme. However, it may be of some interest that the very features which give a Jackson network a particular character in equilibrium also

give its optimization rules a special structure. This fact may strengthen the hope, earlier expressed, that the Jackson network is the first in a natural sequence of progressively more 'intelligent' networks.

We shall consider the choice of routing rules for an open Jackson network which will induce it to clear the given input traffic, specified by the ν_j, as efficiently as possible. Specifically, we shall suppose that a network in state n incurs a cost $a(n)$ per unit time, and that a cost b_{jk} is also incurred whenever there is a direct $j \to k$ migration. One might, for example, assume $a(n)$ to have the form

$$a(n) = \sum_j a_j n_j \tag{1}$$

if waiting time at node j is costed at a_j per unit time. However, there is no need to specialize as yet. The b_{jk} component of cost reflects the cost of a $j \to k$ routing. For example, $b_{j0} = +\infty$ would imply that the system cannot be left directly from node j. We are assuming units of a single type, but one can choose the b_{jk} to constrain the path of a unit to ensure, if desired, that it follows some processing sequence in an acceptable order.

Appealing to the expressions for equilibrium distribution and flow rates derived in Theorems 3.1 and 3.3 we deduce that the expected rate of cost for a given network in equilibrium is

$$C = A(w) + \sum_j \sum_k b_{jk} \lambda_{jk} w_j \tag{2}$$

where

$$A(w) := \frac{\sum_n a(n) \Phi(n) \prod_j w_j^{n_j}}{\sum_n \Phi(n) \prod_j w_j^{n_j}}. \tag{3}$$

Certain aspects of the problem are prescribed; we assume these to be the set of nodes (but see Exercise 1), the input rates $\nu_j = \lambda_{0j}$ and the 'configuration' function $\Phi(n)$. We shall also assume that capacity constraints are placed on exit routing at nodes:

$$\lambda_j := \sum_k \lambda_{jk} \leqslant \bar{\lambda}_j \qquad (j = 1, 2, \ldots, m) \tag{4}$$

Here the bounds $\bar{\lambda}_j$ are prescribed. We regard the design problem as the task of minimizing expression (2) for C with respect to the routing matrix $\lambda = (\lambda_{jk})$ subject to the constraints listed and, of course, that of non-negativity: $\lambda_{jk} \geqslant 0$.

Let us rather regard it as the task of minimizing C with respect to the λ_{jk} and w_j $(j \neq 0)$ subject to the constraints (4) and the determining constraints

$$\sum_{k=0}^{m} (w_k \lambda_{kj} - w_j \lambda_{jk}) = 0 \qquad (j = 1, 2, \ldots, m) \tag{5}$$

plus non-negativity.

The values $w_0 = 1$ and $\lambda_{0k} = \nu_k$ are of course prescribed. We apply constraints (4), (5) only for $j \neq 0$; there is no constraint (4) for $j = 0$ and the constraint (5) for $j = 0$ is redundant. We have then the Lagrangian form

$$L = C + \sum_j \sum_k \xi_j (w_k \lambda_{kj} - w_j \lambda_{jk}) + \sum_j \eta_j \left(\sum_k \lambda_{jk} - \bar{\lambda}_j \right) \tag{6}$$

where ξ_j, η_j are Lagrangian multipliers associated with the constraints (5), (4). If we adopt the convention $\xi_0 = \eta_0 = 0$ then all summations in (6) can be taken over the range $(0, 1, \ldots, m)$. These multipliers have the interpretations

$$\xi_j = \frac{\partial \bar{C}}{\partial \nu_j}$$

$$\eta_j = -\frac{\partial \bar{C}}{\partial \bar{\lambda}_j}, \tag{7}$$

where $\bar{C}$ is the minimal constrained value of C (see Appendix 3). That is, ξ_j is the marginal cost of additional input at node j and η_j the marginal benefit of increased capacity at node j.

Now, the full formalism of convex programming described in Appendix 3 is not available, because the function $A(w)$ defined by (3) is not necessarily convex and the constraints (5) are not linear in (λ, w). One can nevertheless appeal to the Kuhn–Tucker optimality conditions (see Appendix 3) to deduce the necessary conditions asserted in the following theorem.

Theorem 7.1

(i) *The marginal costs of input ξ_j satisfy $\xi_0 = 0$ and*

$$\xi_j \leqslant c_j + \min_k (b_{jk} + \xi_k) \qquad (j \neq 0) \tag{8}$$

with equality for those nodes j which are used in the optimal design. Here

$$c_j := \frac{1}{\lambda_j} \frac{\partial A(w)}{\partial w_j} \tag{9}$$

(ii) *If the routing $j \to k$ is used in the optimal design then equality holds in* (8) *for the given j and the minimum in the right-hand member is attained at the given k.*

(iii) *If node j is not used in the optimal design then $w_j = 0$. If it is used but at less than full capacity then $c_j = 0$.*

Proof Applying the Kuhn–Tucker conditions in the region $(\lambda, w) \geqslant 0$ we deduce that

$$\lambda_j c_j + \sum_k \lambda_{jk} b_{jk} - \lambda_j \xi_j + \sum_k \lambda_{jk} \xi_k \geqslant 0 \tag{10}$$

with equality if $w_j > 0$, and

$$w_j (b_{jk} - \xi_j + \xi_k) + \eta_j \geqslant 0 \tag{11}$$

with equality if $\lambda_{jk} > 0$. We deduce from (11) that $b_{jk} + \xi_k$ will have the same value for all k such that $\lambda_{jk} > 0$; the value minimizing this expression over k. The second part of assertion (ii) is thus demonstrated. The remainder of assertions (i) and (ii) then follow by appeal to this and the condition associated with (10). The evaluation $\xi_0 = 0$ is a consistent convention, associated with the fact that there is no term for $j = 0$ in the first summation of expression (6). Finally, certainly $w_j = 0$ if node j is not used, and $\eta_j = 0$ if node j is used at less than full capacity. If the node is used but at less than full capacity then we deduce from (11) that

$$\min_k (b_{jk} - \xi_j + \xi_k) = 0 \tag{12}$$

equality following from the fact that $\lambda_{jk} > 0$ for some k. Comparing (11) and the equality version of (8) we deduce that $c_j = 0$.■

If we reduce the set of nodes to those which are actually used in the optimal design then inequality (8) takes the equality form

$$\xi_j = c_j + \min (b_{jk} + \xi_k) \tag{13}$$

with boundary condition $\xi_0 = 0$. We recognize in (13) a *dynamic programming equation* of the type that turns up in time- and path-optimization problems (see, e.g. Whittle, 1982c). The reason for this is clear, once one thinks about it. Recall the interpretation of ξ_j as the marginal cost of accepting new additional input at node j, for the optimal design. Equation (3) represents this cost as the sum of the marginal cost c_j of passing this input through node j, the transit cost b_{jk} of passing it to node k and the marginal cost ξ_k of accepting it at node k. The only following nodes k which are employed are those for which the sum of these cost components is minimal.

If the c_j were prescribed, as are the b_{jk}, then it would be exceptional for the minimum in (13) to be attained at more than one value of k, and so for the output from node j to be split. However, c_j is a function of w. The fact that some nodes can handle traffic only at a limited rate (if they are queue-like) will mean that traffic from a given node will often have to be split. That is, w, and so the c_j, will adjust so that the minimum is attained in (13) at several values of k. These multiple equalities give extra equations which determine the actual λ_{jk}. To fully determine the solution one must couple the w-determining relations (5) with the network-determining relations (4), (8) and (9).

The solution is plainly far from complete, but one can deduce a number of features already.

Corollary 7.1 *Suppose $c_j + b_{jk} > 0$ for $j \neq 0$ and all k. Then the optimal network has no cycles.*

Proof The assumption and relation (13) imply that ξ_j decreases strictly as j follows a possible path through the optimal network to the termination point $j = 0$. Cycles are thus excluded. ■

Note also

Theorem 7.2 *The marginal cost c_j of passage through node j has the interpretation*

$$c_j = \lambda_j^{-1} \operatorname{cov}(a(n), n_j) \tag{14}$$

where the covariance is calculated under the equilibrium statistics of the network, whether optimal or not. If $a(n)$ and $\Phi(n)$ are such that these covariances are necessarily strictly positive then all nodes which are used in the optimal design are used at full capacity.

Proof Relation (14) follows immediately from (3), (9). The conclusion $c_j > 0$ would imply the second assertion, by Theorem 7.1(iii). ■

Exercises and comments

1. The set of nodes is supposed given. However, nodes can be dropped from the optimal design, so if one begins with a dense set of nodes then there is virtually no constraint on the set of nodes actually to be employed. Suppose, for example, that any point in $\mathbb{R}^d$ could be a node, with the functions a, Φ and b all appropriately defined. Then, depending upon the convexity/concavity properties of these functions, the nodes of the optimal network will form either a continuous or a discrete set.

8. SOCIAL, INDIVIDUAL AND BUREAUCRATIC OPTIMA

We could well describe the optimization of the last section as a *social* optimization, since it is understood as the attempt of a planner to minimize collective costs. Let us assume in this section, for simplicity of argument, that all nodes are used and at full capacity in the optimal design.

Suppose now that

$$\Phi(n) = \prod_j (n_j!)^{-1} \tag{1}$$

so that all units move independently through the network. Suppose also that $a(n)$ takes the linear form (7.1). Then one finds that c_j, defined by (7.9), has the simple evaluation,

$$c_j = a_j / \bar{\lambda}_j \tag{2}$$

independent of w. The dynamic programming equation (7.13) will then yield an evaluation of the marginal input costs ξ_j and of the optimal routing which are also independent of w, i.e. independent of expected traffic conditions.

These costs are just those which would apply to an individual unit if it were understood that the individual himself bore a cost a_j per unit time while waiting at node j (when (2) would give the expected total waiting cost at node j) and himself bore the cost b_{jk} of passage between nodes. The route recommended by the minimizing option in (7.13) (with c given by (2)) is then exactly the route that an individual should take on leaving node j, in order to minimize his individual costs, and the w-independent solution $\xi_j = F_j$ of (7.13) is then exactly the minimal future cost faced by an individual entering node j.

Let us refer to this optimization as an *individual* optimization. Our conclusion may then be expressed.

Theorem 8.1 *Suppose that units move independently and that the cost function $a(n)$ has the linear form* (7.1). *Then social and individual optimizations agree, in that the optimal routing is the same in both cases, and the marginal costs ξ_j of the social case are exactly the minimal individual costs F_j.*

Suppose now one allows $\Phi(n)$ to be general and chooses

$$a(n) = \sum_j a_j \Phi(n - e_j)/\Phi(n) \tag{3}$$

Then we see by appeal to Theorem 3.3 that $A(w) = \sum_j a_j w_j$ and evaluation (2) still holds. The optimal social routing will then again be that recommended by the individual optimization. This may seem strange, because individuals now interact in general, and nodes may congest.

The optimization with choice (3) might be termed a *bureaucratic* optimization. The ratio $\Phi(n - e_j)/\Phi(n)$ is proportional to the rate of exit (i.e. of 'service') at node j. If cost (3) plus the transit costs b_{jk} are regarded as costs borne by the *operators* of the system then one can say that the operators are trying to choose a routing which gives them least work, consistent with full capacity ($\lambda_j = \bar{\lambda}_j$) working. That is, the routing is one that suits the bureaucracy best, a bureaucracy concerned by its own work-rate rather than by customer waiting times, etc.

With this understanding we have established

Theorem 8.2 *The bureaucratic optimal routing is that determined by the individual optimization rule, whatever $\Phi(n)$.*

9. ADAPTIVE ROUTING RULES

It was said in section 2 that a Jackson network is the least intelligent of all networks. To bring it to a higher level one must make the routing responsive to the general state n of the system. In other words, one must use a routing rule which is *adaptive*, in that it depends upon n.